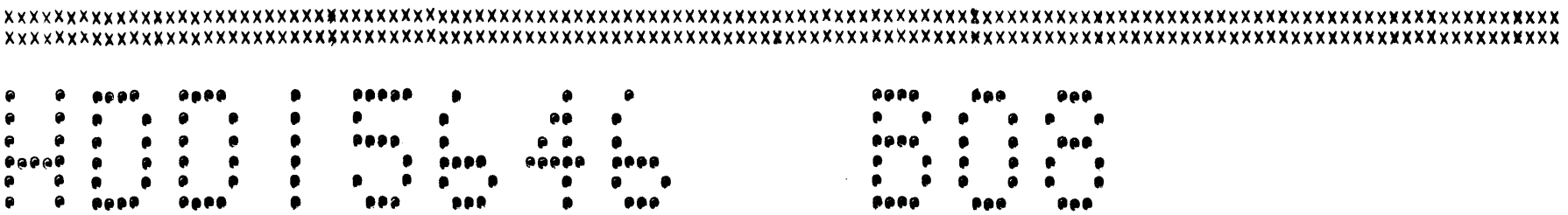

05/31/72 01:53:07

```
**********************************
**********************************
**                              **
**  PDP-9 MINI TIME-SHARING SYSTEM  **
**            DTSS:BASQ         **
**            MTSS:BQ6          **
**                              **
**********************************
**********************************
```

```
100             .TITLE  PDP-9 BASIC INTERPRETER
110             .NAME   BAS0
```

DEFINITIONS

```
                    120             .STITL  DEFINITIONS
                    130     *
                    140     *       DEFINITION -- PSEUDO OPS
                    150     *
                    160     RET     .OPDEF  JMP+20000       RETURN -- JMP **,X
                    170     *
                    180     *       DEFINITION -- PROGRAM CONSTANTS
                    190     *
                    200             .HEAD   S               S FOR STORAGE
000001              210     UNIT    .EQU    1               ARITHMETIC PRECISION
                    220     *
                    230     *       DEFINITION -- AUTO INDEX REGISTERS
                    240     *
                    250             .HEAD   T               T FOR TTY
000010              260     TTYX    .EQU    10              XR-0 USED FOR TTY I/O
000011              270     CHRX    .EQU    11              XR-1 GLOBAL INPUT POINTER
000017              280     SBUF    .EQU    17              XR-7 USED FOR SOURCE BUFFER POINTER
                    290             .HEAD   0,P,I,T
000012              300     X       .EQU    12              LOCAL
000013              310     Y       .EQU    13              LOCAL
000014              320     Z       .EQU    14              LOCAL
                    330     *
                    340     *       DEFINITION -- USEFUL MACROS
                    350     *
                    360     SOS     .DEFINT                 SUBTRACT ONE FROM STORAGE
                    370             CLC                     C(AC) := 777777
                    380             TAD     #1              C(AC) := C(MEM)-1
                    390             DAC     #1              C(MEM) := C(AC)
                    400             .ENDM   SOS
                    410     *
                    420     *       PROGRAM ORIGIN
                    430     *
                    440     *        .LOC 100
015777              450             .LOC    16000-1
015777 615716       460             JMP     SUP             A CONVENIENT RESTART LOCATION
011551              470             .LOC    16000-4227
                    480             .HEAD   T               T FOR TTY
```

T TTY PRINT ROUTINE -- PRINT

```
                    490             .STITL  TTY PRINT ROUTINE -- PRINT
                    500     *
                    510     *       THIS ROUTINE PRINTS A SINGLE CHARACTER TO THE TTY.
                    520     *
                    530     *       ENTER WITH CHARACTER IN R-AC.
                    540     *
                    550     *       DEFINITION -- 'PRINT' -- PRINT CHARACTER MACRO
                    560     *
                    570     PRINT   .DEFIN
                    580             JMS     TSPRINT         CALL THE PRINT ROUTINE
                    590             .ENDM   PRINT
                    600     *
                    610     *       TSPRINT
                    620     *
011551  000000      630     PRINT   0
011552  700401      640             TSF                     ***
011553  611552      650             JMP     .-1             * PRINT THE CHARACTER WHEN THE TTY IS READY
011554  700406      660             TLS                     ***
011555  631551      670             RET     PRINT           RETURN
```

```
T                       MESSAGE PRINTING ROUTINE -- MESS

                  680           .STITL  MESSAGE PRINTING ROUTINE -- MESS
                  690     *
                  700     *     THIS ROUTINE IS CALLED TO PRINT A MESSAGE ON THE TTY.
                  710     *
                  720     *     ENTER WITH THE MESSAGE FOLLOWING THE CALL.  THE END OF THE
                  730     *     MESSAGE IS FLAGGED WITH A WORD OF -1.
                  740     *
                  750     *     DEFINITION -- 'MESS' -- MESSAGE MACRO
                  760     *
                  770     MESS  .DEFIN                  <MESSAGE>
                  780           JMS     TSMESS          CALL THE MESSAGE ROUTINE
                  790           .TEXT   \#1\
                  800           .IFE    '#2','CRLF',1
                  810           .DATA   15,12           GIVE CARRIAGE RETURN/LINE-FEED IF WANTED
                  820           777777                  FLAG THE END OF THE MESSAGE
                  830           .ENDM   MESS
                  840     *
                  850     *     TSMESS
                  860     *
011556  000000  870     MESS  0
011557  211556  880           LAC     MESS            ***
011560  040010  890           DAC     TTYX            * SET UP POINTER TO MESSAGE
011561  231556  900           LAC     MESS,X          ***
011562          910     MSA   ...
011562  551717  920           SAD     MONE            CHECK FOR END OF MESSAGE (EOM) FLAG
011563  611567  930           JMP     MSX             EOM -- RETURN
011564          940           PRINT                   PRINT THE CHARACTER
011565  220010  950           LAC     TTYX,X          FETCH THE NEXT CHARACTER
011566  611562  960           JMP     MSA             AND LOOP
011567          970     MSX   ...
011567  211713  980           LAC     FILL            SEND A FILL
011570          990           PRINT                   TO DELAY A LITTLE
011571  620010  1000          RET     TTYX,X          THEN RETURN
```

T TTY READ ROUTINE -- READ

```
                    1010            .STITL  TTY READ ROUTINE -- READ
                    1020    *
                    1030    *       THIS ROUTINE IS CALLED TO READ A LINE FROM THE TTY.
                    1040    *
                    1050    *       IT PLACES THE LINE ONE CHARACTER TO A WORD IN TSIBUF, AFTER
                    1060    *       PERFORMING SEVERAL ELEMENTARY EDITING OPERATIONS:
                    1070    *
                    1080    *       CTRL-X    DELETE THE LINE
                    1090    *       BACKARROW DELETE THE PRECEEDING CHARACTER
                    1100    *
011572  000000      1110    READ    0
011573  211714      1120            LAC     IBLN            GET THE BUFFER LENGTH
011574  051716      1130            DAC     IBCNT           SAVE FOR COUNTING
011575  211715      1140            LAC     IBFPT           GET INITIAL POINTER
011576  040011      1150            DAC     CHRX            SAVE IN LOCAL AUTO INDEX REGISTER
                    1160    *
                    1170    *       READ THE ACTUAL CHARACTER
                    1180    *
    011577          1190    RDA     ...
011577  700301      1200            KSF                     ***
011600  611577      1210            JMP     .-1             * READ THE CHARACTER
011601  700312      1220            KRB                     ***
                    1230    *
                    1240    *       CHECK FOR AN EDITING CHARACTER
                    1250    *
011602  311713      1260            AND     FILL            MASK OFF PARITY BIT
011603  355750      1270            SAD     (30)            CHECK FOR CTRL-X
011604  611643      1280            JMP     CTRLX           YES -- DELETE THE LINE
011605  355751      1290            SAD     (137)           CHECK FOR BACKARROW
011606  611660      1300            JMP     BACK            YES -- BACK UP
011607  355752      1310            SAD     (5)             CHECK FOR CTRL-E
011610  611701      1320            JMP     WRU             TELL HIM WHO WE ARE
011611  060011      1330            DAC     CHRX,X          SAVE THE CHARACTER IN THE BUFFER
011612  355753      1340            SAD     (15)            SEE IF THE END OF LINE
011613  611671      1350            JMP     EOL             YES -- CLEAN US UP
011614  451716      1360            ISZ     IBCNT           COUNT THE NEW CHARACTER
011615  611577      1370            JMP     RDA             IF MORE ROOM -- GET ANOTHER CHARACTER
```

T TTY READ ROUTINE -- READ

```
                    1380            .EJECT
                    1390    *
                    1400    *       LINE TOO LONG -- GIVE MESSAGE AND DELETE IT
                    1410    *
011616  111556      1420            JMS     TSMESS          CAN'T USE THE MESSAGE MACRO BECAUSE OF CR/LF
011617  000015      1430            .DATA   15,12           CR/LF
011620  000012
011621  000314      1440            .TEXT   \LINE TOO LONG -- \
011622  000311
011623  000316
011624  000305
011625  000240
011626  000324
011627  000317
011630  000317
011631  000240
011632  000314
011633  000317
011634  000316
011635  000307
011636  000240
011637  000255
011640  000255
011641  000240
011642  777777      1450            777777                  END FLAG
```

T TTY READ ROUTINE -- READ

```
                    1460            .EJECT
                    1470    *
                    1480    *       DELETE LINE ROUTINE
                    1490    *
    011643          1500    CTRLX   ...
    011643          1510            MESS    < DELETED>,CRLF
011657  611573      1520            JMP     READ+1          TRY AGAIN
                    1530    *
                    1540    *       BACKARROW ROUTINE
                    1550    *
    011660          1560    BACK    ...
011660  211717      1570            LAC     MONE            GET -1
011661  340011      1580            TAD     CHRX            AND GET THE POINTER LESS ONE IN R-AC
011662  555754      1590            SAD     (IBUF-2)        SEE IF TOO FAR
011663  611643      1600            JMP     CTRLX           IF SO -- SAY DELETED
011664  040011      1610            DAC     CHRX            RESTORE POINTER
011665  211717      1620            LAC     MONE            ***
011666  351716      1630            TAD     IBCNT           * DECREMENT THE COUNTER BY ONE
011667  051716      1640            DAC     IBCNT           ***
011670  611577      1650            JMP     RDA             AND READ ANOTHER CHARACTER
                    1660    *
                    1670    *       END OF LINE ROUTINE
                    1680    *
    011671          1690    EOL     ...
011671  215755      1700            LAC     (12)            SEND A LINE-FEED
    011672          1710            PRINT                   FOR GOOD MEASURE
011673  060011      1720            DAC     CHRX,X          SAVE IN BUFFER
011674  211717      1730            LAC     MONE            GET AN EOM FLAG
011675  060011      1740            DAC     CHRX,X          SAVE IN BUFFER
011676  211715      1750            LAC     IBFPT           SET UP A POINTER
011677  040011      1760            DAC     CHRX            FOR THE CALLER
011700  631572      1770            RET     READ            AND RETURN
                    1780    *
                    1790    *       WHO ARE YOU? ROUTINE
                    1800    *
    011701          1810    WRU     ...
    011701          1820            MESS    <BASIC>,CRLF
011712  611577      1830            JMP     RDA
```

T TTY READ ROUTINE -- READ

```
                    1840            ,EJECT
                    1850    *
                    1860    *       INPUT BUFFER AND STORAGE
                    1870    *
011713  000177      1880    FILL    177                     ASCII FILL/PARITY MASK
    000100          1890    IBN     ,EQU    100             LENGTH OF VISIBLE BUFFER
011714  777677      1900    IBLN    ,DATA   -IBN-1          LENGTH TO INITIALIZE COUNTER
011715  011717      1910    IBPT    ,DATA   IBUF-1          INITIAL POINTER TO THE BUFFER
    011716          1920    IBCNT   ,BLOCK  1               COUNTER
011717  777777      1930    MONE    777777                  A MINUS ONE
    011720          1940    IBUF    ,BLOCK  IBN+2           INPUT BUFFER
```

T TTY READ -- CHAR

```
                    1950            .STITL  TTY READ -- CHAR
                    1960    *
                    1970    *       THIS ROUTINE IS CALLED TO PICK UP A CHARACTER FROM
                    1980    *       THE SOURCE LINE.
                    1990    *
                    2000    *       DEFINITION -- 'CHAR' -- GET CHARACTER MACRO
                    2010    *
                    2020    CHAR    .DEFIN
                    2030            JMS     TSCHAR          CALL THE SUBROUTINE
                    2040            .ENDM   CHAR
                    2050    *
                    2060    *       TSCHAR
                    2070    *
012022  000000      2080    CHAR    0
012023  220011      2090            LAC     CHRX,X          GET THE NEXT CHARACTER
012024  551717      2100            SAD     MONE            SEE IF EOM CHARACTER
012025  615551      2110            JMP     ESPARSE         SHOULDN'T GET THERE
012026  555756      2120            SAD     (040)           SEE IF BLANK
012027  612023      2130            JMP     CHAR+1          IGNORE IF SO
012030  052032      2140            DAC     LCHAR           SAVE IN CASE WE LOOK AGAIN
012031  632022      2150            RET     CHAR            AND RETURN
                    2160    *
                    2170    *       DEFINITION -- 'LCHAR' -- GET PREVIOUS CHARACTER MACRO
                    2180    *
                    2190    LCHAR   .DEFIN
                    2200            LAC     TSLCHAR         GET THE CHARACTER
                    2210            .ENDM   LCHAR
012032              2220    LCHAR   .BLOCK  1               LAST CHARACTER READ
```

```
                                        TTY READ -- GET LINE NUMBER

                    2230                ,STITL  TTY READ -- GET LINE NUMBER
                    2240    *
                    2250    *           THIS SUBROUTINE IS CALLED BY THE FILE BUILDING
                    2260    *           ROUTINE TO ASSEMBLE A LINE NUMBER
                    2270    *
012033  000000      2280    GNUM        0
    012034          2290                CHAR                        GET A CHARACTER
012035  052053      2300                DAC     TEM                 SAVE IT
012036  777720      2310                LAW     -60                 ***
012037  352053      2320                TAD     TEM                 * SEE IF LESS THAN A NUMBER
012040  741100      2330                SPA                         ***
012041  632033      2340                RET     GNUM                IF SO -- RETURN
012042  052053      2350                DAC     TEM                 SAVE IT
012043  777766      2360                LAW     -12                 ***
012044  352053      2370                TAD     TEM                 * SEE IF GREATER THAN A DIGIT
012045  740100      2380                SMA                         ***
012046  632033      2390                RET     GNUM                IF SO -- RETURN
012047  212033      2400                LAC     GNUM                FIX UP THE RETURN
012050  040012      2410                DAC     X                   TO RETURN TO CALLER+2
012051  212053      2420                LAC     TEM                 GET THE DIGIT
012052  620012      2430                RET     X                   AND RETURN
    012053          2440    TEM         ,BLOCK  1
    012054          2450    TEM1        ,BLOCK  1
```

```
T                                   TTY READ -- SOURCE FILE BUILDING ROUTINE

                    2460            .STITL  TTY READ -- SOURCE FILE BUILDING ROUTINE
                    2470    *
                    2480    *       THIS ROUTINE IS USED TO BUILD THE PACKED SOURCE FILE
                    2490    *       FROM INPUT FROM THE TERMINAL
                    2500    *
012055  000000      2510    LINE    0
012056  111572      2520            JMS     READ            READ IN THE LINE
012057  112033      2530            JMS     GNUM            GET THE FIRST DIGIT OF LINE NUMBER
012060  612126      2540            JMP     LINX            END IF DIDN'T BEGIN WITH DIGIT
012061  052054      2550            DAC     TEM1            SAVE THE FIRST DIGIT
    012062          2560    LINA    ...
012062  112033      2570            JMS     GNUM            GET THE NEXT DIGIT
012063  612074      2580            JMP     LINB            NOT A DIGIT -- WE HAVE THE NUMBER
012064  744000      2590            CLL                     ***
012065  212054      2600            LAC     TEM1            * MULTIPLY PREVIOUS NUMBER
012066  653122      2610            MUL                     * BY 10
012067  000012      2620            10.                     ***
012070  641002      2630            LACQ                    ***
012071  312053      2640            ADD     TEM             * AND ADD THE NEW DIGIT
012072  052054      2650            DAC     TEM1            ***
012073  612062      2660            JMP     LINA            LOOP
                    2670    *
                    2680    *       SET UP TO CONDENSE THE LINE
                    2690    *
    012074          2700    LINB    ...
012074  212054      2710            LAC     TEM1            GET THE LINE NUMBER
012075  060017      2720            DAC     SBUF,X          AND SAVE IN THE BUFFER
012076  220017      2730            LAC     SBUF,X          MAKE A PLACE FOR THE COUNT
012077  200017      2740            LAC     SBUF            GET POINTER TO COUNT
012100  052134      2750            DAC     BSBF            SAVE IT
012101  172134      2760            DZM     BSBF,X          ZERO IT OUT
012102  212032      2770            LAC     LCHAR           GET THE LAST CHARACTER READ
012103  612105      2780            JMP     LIND            AND ENTER LOOP
    012104          2790    LINC    ...
    012104          2800            CHAR                    GET THE NEXT CHARACTER
    012105          2810    LIND    ...
012105  640711      2820            ALS     11              PUT THE FIRST CHARACTER IN THE UPPERHALF
012106  052054      2830            DAC     TEM1            SAVE IT
012107  555757      2840            SAD     (015000)        CHECK FOR EOL
012110  612120      2850            JMP     LINF+1          YES
    012111          2860            CHAR                    GET THE NEXT CHARACTER
012112  555753      2870            SAD     (15)            SEE IF EOL
012113  612117      2880            JMP     LINF            YES
012114  312054      2890            ADD     TEM1            ADD IN FIRST CHARACTER
012115  060017      2900            DAC     SBUF,X          SAVE IT IN BUFFER
012116  612104      2910            JMP     LINC            LOOP
```

T TTY READ -- SOURCE FILE BUILDING ROUTINE

```
                    2920            ,EJECT
                    2930    *
                    2940    *       CLEAN UP AT END OF LINE
                    2950    *
        012117      2960    LINF    ...
012117  312054      2970            ADD     TEM1            ADD IN THE LAST CHARACTER
012120  060017      2980            DAC     SBUF,X          SAVE IN THE BUFFER
012121  212134      2990            LAC     BSBF            GET POINTER TO BEGINNING OF LINE
012122  740001      3000            CMA                     COMPUTE THE LENGTH OF THE LINE
012123  300017      3010            ADD     SBUF            FROM THE POINTERS
012124  072134      3020            DAC     BSBF,X          SAVE IT IN THE BUFFER
012125  612056      3030            JMP     LINE+1          AND LOOP
        012126      3040    LINX    ...
012126  200017      3050            LAC     SBUF            ***
012127  555760      3060            SAD     (SBFR-1)        * SEE IF JUST A COMMAND
012130  632055      3070            RET     LINE            ***
012131  160017      3080            DZM     SBUF,X          ***
012132  160017      3090            DZM     SBUF,X          * SET UP A NULL LINE
012133  632055      3100            RET     LINE            ***
012134  000000      3110    BSBF    ,DATA   0               POINTER TO BEGINNING OF LINE
```

T TTY READ -- RE-EXPAND A LINE

```
                    3120            .STITL  TTY READ -- RE-EXPAND A LINE
                    3130    *
                    3140    *       THIS ROUTINE IS CALLED BY THE PARSER TO RE-EXPAND
                    3150    *       A SOURCE LINE FROM THE BUFFER.
                    3160    *
012135  000000      3170    NLINE   0
012136  750004      3180            LAS                     ***
012137  741100      3190            SPA                     * SEE IF THE SWITCHES INDICATE A CHANGE OF HEART
012140  612173      3200            JMP     MON             ***
012141  220017      3210            LAC     SBUF,X          GET THE LINE NUMBER
012142  052240      3220            DAC     LNUM            SAVE IT
012143  220017      3230            LAC     SBUF,X          IGNORE THE LINE SIZE
012144  741200      3240            SNA                     SEE IF THERE IS ONE
012145  632135      3250            RET     NLINE           NO
012146  211715      3260            LAC     IBFPT           INITIALIZE THE CHARACTER BUFFER POINTER
012147  040011      3270            DAC     CHRX            FOR THE EXPANSION
                    3280    *
                    3290    *       EXPAND A LINE FROM THE SOURCE BUFFER INTO THE CHARACTER BUFFER
                    3300    *
    012150          3310    NLNA    ...
012150  220017      3320            LAC     SBUF,X          ***
012151  640511      3330            LRS     11              * GET THE FIRST CHARACTER OF THE WORD
012152  511713      3340            AND     FILL            * WITHOUT WHAT WAS IN THE LINK
012153  060011      3350            DAC     CHRX,X          ***
012154  555753      3360            SAD     (15)            SEE IF EOL
012155  612164      3370            JMP     NLNX            YES
012156  640611      3380            LLS     11              GET THE SECOND CHARACTER
012157  511713      3390            AND     FILL            MASK TO JUST THE CHARACTER
012160  060011      3400            DAC     CHRX,X          SAVE IT
012161  555753      3410            SAD     (15)            SEE IF EOL
012162  612164      3420            JMP     NLNX            YES
012163  612150      3430            JMP     NLNA
    012164          3440    NLNX    ...
012164  777777      3450            LAW     -1              FLAG THE EOL
012165  060011      3460            DAC     CHRX,X          FOR ALL TO KNOW
012166  211715      3470            LAC     IBFPT           REINITIALIZE THE POINTER
012167  040011      3480            DAC     CHRX            FOR THE SCAN TO USE
012170  212135      3490            LAC     NLINE           ***
012171  040012      3500            DAC     X               *NORMAL RETURN
012172  620012      3510            RET     X               ***
```

T MONITOR

```
                    3520            .STITL  MONITOR
                    3530    *
                    3540    *       THIS ROUTINE FUNCTIONS AS THE MONITOR FOR THE BASIC
                    3550    *       INTERPRETER,  IT EITHER BEGINS THE BUILDING OF
                    3560    *       A FILE OR ACCEPTS A COMMAND.
                    3570    *
    012173          3580    MON     ...
    012173          3590            MESS    <ENTER FILE OR COMMAND>,CRLF
012224  215760      3600            LAC     (SBFR-1)        POINTER TO THE SOURCE BUFFER
012225  040017      3610            DAC     SBUF            SAVE FOR SCAN
012226  112055      3620            JMS     LINE            BUILD THE PROGRAM
012227  215760      3630            LAC     (SBFR-1)
012230  040017      3640            DAC     SBUF            REINITIALIZE THE BUFFER POINTER
012231  212032      3650            LAC     LCHAR           GET THE COMMAND
012232  555761      3660            SAD     (122)
012233  614220      3670            JMP     PSPARSE         RUN
012234  555762      3680            SAD     (114)
012235  613241      3690            JMP     LIST            LIST
012236  705001      3700            705001                  ELSE HALT FOR NOW
012237  612173      3710            JMP     MON             BUT BE ABLE TO START UP AGAIN
    012240          3720    LNUM    .BLOCK  1               LINE NUMBER
    012241          3730    SBFR    .BLOCK  1000            SOURCE BUFFER
```

```
T                                       LIST A SOURCE FILE

                    3740                .STITL  LIST A SOURCE FILE
                    3750    *
                    3760    *           THIS ROUTINE IS USED TO LIST THE SOURCE FILE
                    3770    *
        013241      3780    LIST        ...
013241  112135      3790                JMS     NLINE           GET THE NEXT LINE
013242  612173      3800                JMP     MON             REACHED THE END
013243  212240      3810                LAC     LNUM            GET THE LINE NUMBER
013244  114375      3820                JMS     I$PRT           PRINT IT
013245  760040      3830                LAW     040             A SPACE
        013246      3840                PRINT                   PRINT IT
        013247      3850    LISA        ...
        013247      3860                CHAR                    GET THE NEXT CHARACTER
        013250      3870                PRINT                   PRINT IT
013251  555753      3880                SAD     (015)           SEE IF EOL
013252  613254      3890                JMP     LISX            YES
013253  613247      3900                JMP     LISA            NO -- LOOP
        013254      3910    LISX        ...
013254  760012      3920                LAW     012             GIVE THE LINE FEED
        013255      3930                PRINT                   TO CLEAN US UP
013256  613241      3940                JMP     LIST            AND LOOP
                    3950                .HEAD
```

STACK MANAGEMENT -- MACROS

```
3960                ,STITL  STACK MANAGEMENT -- MACROS
3970        *
3980        *       THIS MACRO IS USED TO PUSH THE CONTENTS OF
3990        *       R-AC ONTO ONE OF THE THREE STACKS IN THIS
4000        *       PROGRAM.  THE STACK ARE:
4010        *
4020        *       R     RECURSION CONTROL STACK
4030        *       O     OPERATOR STACK
4040        *       V     VARIABLE STACK
4050        *
4060        *       DEFINITION -- 'PUSH' -- PUSH MACRO
4070        *
4080        PUSH    ,DEFIN                      <LETTER OF STACK>
4090                JMS     #1$PUSH             CALL THE PROPER SUBROUTINE
4100                ,ENDM   PUSH
4110        *
4120        *       THIS MACRO IS USED TO POP THE TOP DATUM FROM
4130        *       ONE OF THE THREE STACKS INTO R-AC.  IT USES
4140        *       THE SAME SYMBOLS AS 'PUSH'.
4150        *
4160        *       DEFINITION -- 'POP' -- POP MACRO
4170        *
4180        POP     ,DEFIN                      <LETTER OF STACK>
4190                JMS     #1$POP              CALL THE PROPER SUBROUTINE
4200                ,ENDM   POP
4210        *
4220        *       DEFINITION -- 'POPV' -- POP VARIABLE AND DEREFERENCE
4230        *
4240        POPV    ,DEFIN
4250                POP     V                   GET THE POINTER TO THE DATUM
4260                DAC     TEM                 SAVE IT TEMPORARILY
4270                LAC     TEM,X               GET THE DATUM
4280                ,ENDM   POPV
```

STACK MANAGEMENT -- PUSH

```
                    4290            .STITL  STACK MANAGEMENT -- PUSH
                    4300    *
                    4310    *       THIS SUBROUTINE PUSHES THE DATUM SAVED IN $DATUM
                    4320    *       ONTO THE STACK WHOSE POINTERS ARE AT C(AC)+1 ON
                    4330    *       ENTRANCE.  THE FORMAT OF THIS POINTER BLOCK FOR
                    4340    *       EACH OF THE STACKS IS:
                    4350    *
                    4360    *       0     CURRENT STACK POINTER
                    4370    *       1     POINTER TO BOTTOM OF STACK
                    4380    *       2     -SIZE-1 (ONE'S COMPLEMENT OF SIZE)
                    4390    *
                    4400    *       $PUSH
                    4410    *
013257  000000      4420    PUSH    0
013260  040012      4430            DAC     X               SAVE POINTER TO STACK DESCRIPTION
013261  040013      4440            DAC     Y               ALSO FOR UPDATING POINTER
013262  220012      4450            LAC     X,X             ***
013263  040014      4460            DAC     Z               * SAVE STACK POINTER AND
013264  740001      4470            CMA                     * COMPUTE AMOUNT OF STACK IN USE
013265  360012      4480            TAD     X,X             ***
013266  560012      4490            SAD     X,X             COMPARE WITH THE SIZE
013267  615570      4500            JMP     ESSOVF          STACK OVERFLOW -- EXPRESSION TOO COMPLICATED
013270  213315      4510            LAC     DATUM           GET THE DATUM TO BE PUSHED
013271  060014      4520            DAC     Z,X             PUSH
013272  460013      4530            ISZ     Y,X             INCREMENT THE REAL POINTER
013273  633257      4540            RET     PUSH            RETURN
013274  740040      4550            HLT                     SHOULDN'T GET HERE
```

STACK MANAGEMENT -- POP

```
                    4560            ,STITL  STACK MANAGEMENT -- POP
                    4570    *
                    4580    *       THIS SUBROUTINE POPS THE DATUM FROM THE TOP OF THE
                    4590    *       STACK WHOSE DESCRIPTION BLOCK IS AT C(AC)+1 ON ENTRANCE.
                    4600    *       THIS DESCRIPTION BLOCK IS THE SAME AS THE ONE USED BY PUSH.
                    4610    *
                    4620    *       $POP
                    4630    *
013275  000000      4640    POP     0
013276  040012      4650            DAC     X               SAVE POINTER TO DESCRIPTION BLOCK
013277  040013      4660            DAC     Y               ALSO FOR UPDATING POINTER
013300  220012      4670            LAC     X,X             GET POINTER TO STACK
013301  053315      4680            DAC     DATUM           SAVE TEMPORARILY
013302  220012      4690            LAC     X,X             ***
013303  740001      4700            CMA                     * COMPUTE -AMOUNT IN USE
013304  353315      4710            TAD     DATUM           ***
013305  553314      4720            SAD     MTWO            SEE IF WE WERE AT THE BOTTOM
013306  615551      4730            JMP     $SPARSE         WE'VE BLOWN IT SOMEWHERE
013307  777777      4740            LAW     -1              ***
013310  353315      4750            TAD     DATUM           * DECREMENT REAL POINTER
013311  060013      4760            DAC     Y,X             ***
013312  233315      4770            LAC     DATUM,X         GET THE DATUM
013313  633275      4780            RET     POP             AND RETURN
                    4790    *
                    4800    *       STORAGE FOR $PUSH AND $POP
                    4810    *
013314  777776      4820    MTWO    ,DATA   -2              FOR CHECKING STACK UNDERFLOW
013315              4830    DATUM   ,BLOCK  1               FOR DATUM OR POINTER TO DATUM
                    4840            ,EOT    BAS1
                    4000            ,HEAD   R               R FOR RECURSION
```

R STACK MANAGEMENT -- RECURSION CONTROL STACK

```
                    4010            ,STITL  STACK MANAGEMENT -- RECURSION CONTROL STACK
                    4020    *
                    4030    *       THIS ROUTINE IS CALLED TO PUSH THE DATUM IN R-AC
                    4040    *       ONTO THE RECURSION CONTROL STACK, R$STACK.
                    4050    *
013316  000000      4060    PUSH    0
013317  053315      4070            DAC     $DATUM          SAVE DATUM FOR $PUSH
013320  773327      4080            LAW     SDB-1           POINTER TO STACK DESCRIPTION BLOCK
013321  113257      4090            JMS     $PUSH           PUSH
013322  633316      4100            RET     PUSH            AND RETURN
                    4110    *
                    4120    *       THIS ROUTINE IS USED TO POP THE DATUM ON THE
                    4130    *       TOP OF THE RECURSION CONTROL STACK INTO R-AC.
                    4140    *
013323  000000      4150    POP     0
013324  773327      4160            LAW     SDB-1           POINTER TO STACK DESCRIPTION BLOCK
013325  113275      4170            JMS     $POP            POP
013326  633323      4180            RET     POP             AND RETURN
                    4190    *
                    4200    *       STACK DESCRIPTION BLOCK
                    4210    *
000400              4220    SIZE    ,EQU    400             STACK SIZE
013327  013332      4230    IPTR    ,DATA   STACK-1         INITIAL POINTER
013330              4240    SDB     ,,,
013330  013332      4250            ,DATA   STACK-1         STACK POINTER
013331  013333      4260            ,DATA   STACK           BOTTOM POINTER
013332  777377      4270            ,DATA   -SIZE-1         ONE'S COMPLEMENT OF SIZE
013333              4280    STACK   ,BLOCK  SIZE            STACK
                    4290            ,HEAD   0               0 FOR OPERATOR
```

0 STACK MANAGEMENT -- OPERATOR STACK

```
                    4300            .STITL  STACK MANAGEMENT -- OPERATOR STACK
                    4310    *
                    4320    *       THIS ROUTINE IS CALLED TO PUSH THE DATUM IN R-AC
                    4330    *       ONTO THE OPERATOR STACK, OSSTACK.
                    4340    *
013733  000000      4350    PUSH    0
013734  053315      4360            DAC     $DATUM          SAVE DATUM FOR $PUSH
013735  773744      4370            LAW     SDB-1           POINTER TO STACK DESCRIPTION BLOCK
013736  113257      4380            JMS     $PUSH           PUSH
013737  633733      4390            RET     PUSH            AND RETURN
                    4400    *
                    4410    *       THIS ROUTINE IS USED TO POP THE DATUM ON THE
                    4420    *       TOP OF THE OPERATOR STACK INTO R-AC.
                    4430    *
013740  000000      4440    POP     0
013741  773744      4450            LAW     SDB-1           POINTER TO STACK DESCRIPTION BLOCK
013742  113275      4460            JMS     $POP            POP
013743  633740      4470            RET     POP             AND RETURN
                    4480    *
                    4490    *       STACK DESCRIPTION BLOCK
                    4500    *
000100              4510    SIZE    .EQU    100             STACK SIZE
013744  013747      4520    IPTR    .DATA   STACK-1         INITIAL POINTER
013745              4530    SDB     ...
013745  013747      4540            .DATA   STACK-1         STACK POINTER
013746  013750      4550            .DATA   STACK           BOTTOM POINTER
013747  777677      4560            .DATA   -SIZE-1         ONE'S COMPLEMENT OF SIZE
013750              4570    STACK   .BLOCK  SIZE            STACK
                    4580            .HEAD   V               V FOR VARIABLE
```

V STACK MANAGEMENT -- VARIABLE STACK

```
                    4590            .STITL  STACK MANAGEMENT -- VARIABLE STACK
                    4600    *
                    4610    *       THIS ROUTINE IS CALLED TO PUSH THE DATUM IN R-AC
                    4620    *       ONTO THE VARAIBLE STACK, VSSTACK.
                    4630    *
014050  000000      4640    PUSH    0
014051  053315      4650            DAC     $DATUM          SAVE DATUM FOR $PUSH
014052  774061      4660            LAW     SDB-1           POINTER TO STACK DESCRIPTION BLOCK
014053  113257      4670            JMS     $PUSH           PUSH
014054  634050      4680            RET     PUSH            AND RETURN
                    4690    *
                    4700    *       THIS ROUTINE IS USED TO POP THE DATUM ON THE
                    4710    *       TOP OF THE VARIABLE STACK INTO R-AC.
                    4720    *
014055  000000      4730    POP     0
014056  774061      4740            LAW     SDB-1           POINTER TO STACK DESCRIPTION BLOCK
014057  113275      4750            JMS     $POP            POP
014060  634055      4760            RET     POP             AND RETURN
                    4770    *
                    4780    *       STACK DESCRIPTION BLOCK
                    4790    *
000100              4800    SIZE    .EQU    100             STACK SIZE
014061  014064      4810    IPTR    .DATA   STACK-1         INITIAL POINTER
014062              4820    SDB     ...
014062  014064      4830            .DATA   STACK-1         STACK POINTER
014063  014065      4840            .DATA   STACK           BOTTOM POINTER
014064  777677      4850            .DATA   -SIZE-1         ONE'S COMPLEMENT OF SIZE
014065              4860    STACK   .BLOCK  SIZE            STACK
                    4870            .HEAD   S               S FOR STORAGE
```

S STORAGE MANAGEMENT -- TEMPORARY

```
                    4880            .STITL  STORAGE MANAGEMENT -- TEMPORARY
                    4890    *
                    4900    *       THIS ROUTINE ALLOCATES A TEMPORARY STORAGE CELL FOR
                    4910    *       THE RESULT OF A COMPUTATION.  IT RETURNS A POINTER IN
                    4920    *       R-AC TO THE BEGINNING OF THE ALLOCATED STORAGE.  IT
                    4930    *       ALLOCATES THE CURRENT LOGICAL (ARITHMETIC) WORD SIZE.
                    4940    *
014165  000000      4950    TEMP    0
014166  777777      4960            LAW     -UNIT           DECREMENT THE CURRENT
014167  354176      4970            TAD     TCNT            TEMPORARY POINTER
014170  741100      4980            SPA                     IF WE'VE RUN OUT
014171  214175      4990            LAC     TSIZE           START OVER AGAIN
014172  054176      5000            DAC     TCNT            SAVE NEW COUNT
014173  354177      5010            TAD     TPRT            RELOCATE BY BASE OF AREA
014174  634165      5020            RET     TEMP            RETURN
                    5030    *
                    5040    *       TEMPORARY STORAGE AREA
                    5050    *
000020              5060    TN      .EQU    20              NUMBER OF TEMPORARY CELLS
014175  000017      5070    TSIZE   .DATA   TN*UNIT-UNIT    SIZE OF TEMPORARY AREA
014176  000000      5080    TCNT    .DATA   0               CURRENT POINTER
014177  014200      5090    TPRT    .DATA   .+1             POINTER TO TEMPORARY AREA BASE
014200              5100            .BLOCK  TN*UNIT         TEMPORARY AREA
                    5110            .HEAD   P               P FOR PARSE
```

P PARSER -- INITIALIZATION

```
                    5120            .STITL  PARSER -- INITIALIZATION
                    5130    *
                    5140    *       THIS ROUTINE IS USED TO INITIALIZE THE SCAN OF
                    5150    *       A LINE.  IT:
                    5160    *
                    5170    *       READS THE LINE
                    5180    *       INITIALIZES THE STACKS
                    5190    *       ENTERS THE PARSER
                    5200    *
    014220          5210    PARSE   ..,
014220  112135      5220            JMS     T$NLINE         GET THE NEXT LINE
014221  615644      5230            JMP     E$NEND          NO END STATEMENT
                    5240    *
                    5250    *       INITIALIZE THE STACKS
                    5260    *
014222  213327      5270            LAC     R$IPTR          RECURSION STACK
014223  053330      5280            DAC     R$SDB           SAVE
014224  200011      5290            LAC     T$CHRX          GET THE CHARACTER POINTER
    014225          5300            PUSH    R               AND PUSH
014226  215763      5310            LAC     (S$STATE)       ***
014227  054263      5320            DAC     ALT             * GET FIRST ALTERNATIVE
    014230          5330            PUSH    R               ***
014231  215764      5340            LAC     (S$STATE+2)     ***
014232  054262      5350            DAC     PART            * GET FIRST PART
    014233          5360            PUSH    R               AND PUSH
014234  213744      5370            LAC     O$IPTR          ***
014235  053745      5380            DAC     O$SDB           * OPERATOR STACK
    014236          5390            PUSH    R               ***
014237  214061      5400            LAC     V$IPTR          ***
014240  054062      5410            DAC     V$SDB           * VARIABLE STACK
    014241          5420            PUSH    R               ***
014242  614266      5430            JMP     TEST            AND START IN THE MIDDLE
```

P PARSER -- PUSH ONTO THE CONTROL STACK

```
                    5440            .STITL  PARSER -- PUSH ONTO THE CONTROL STACK
                    5450    *
                    5460    *       PUSH THE CURRENT STATE ONTO THE STACK
                    5470    *
        014243      5480    PUSH    ...
    014243  200011  5490            LAC     T$CHRX          THE SOURCE POINTER
        014244      5500            PUSH    R               PUSH
    014245  214263  5510            LAC     ALT             GET THE CURRENT ALTERNATIVE
        014246      5520            PUSH    R               AND PUSH
    014247  214262  5530            LAC     PART            GET THE CURRENT PART
        014250      5540            PUSH    R               AND PUSH
    014251  213745  5550            LAC     O$SDB           OPERATOR STACK POINTER
        014252      5560            PUSH    R               PUSH
    014253  214062  5570            LAC     V$SDB           VARIABLE STACK POINTER
        014254      5580            PUSH    R               PUSH
    014255  234262  5590            LAC     PART,X          GET POINTER TO NEW TYPE
    014256  054263  5600            DAC     ALT             SAVE AS NEW ALTERNATIVE
    014257  355765  5610            TAD     (2)             AND MAKE POINTER TO NEW PART
    014260  054262  5620            DAC     PART            SAVE
    014261  614266  5630            JMP     TEST            AND TEST THE NEW ONE
                    5640    *
                    5650    *       ALTERNATIVE AND PART POINTERS FOR OUR USE
                    5660    *
        014262      5670    PART    .BLOCK  1               POINTER TO CURRENT PART
        014263      5680    ALT     .BLOCK  1               POINTER TO CURRENT ALTERNATIVE
        014264      5690    TEMP    .BLOCK  1               A LOCAL TEMPORARY
        014265      5700    LPART   .BLOCK  1               LAST PART RECOGNIZED
```

P PARSER -- TEST

```
                    5710            .STITL  PARSER -- TEST
                    5720    *
                    5730    *       THIS ROUTINE IS USED TO DECIDE WHETHER THE NEXT
                    5740    *       SYNTACTIC OBJECT IS A TYPE OR AN ATOM.
                    5750    *
                    5760    *       IF A TYPE -- PUSH THE CURRENT STATE ONTO THE
                    5770    *       RECURSION CONTROL STACK AND TRY THE NEW TYPE.
                    5780    *
                    5790    *       IF AN ATOM TRY TO MATCH -- ON SUCCESS GO TO TRY THE NEXT PART
                    5800    *       ON FAILURE -- GO TO PSFAIL
                    5810    *
        014266      5820    TEST    ...;
014266  234262      5830            LAC     PART,X          GET THE NEW PART
014267  741100      5840            SPA                     SEE IF A LITERAL
014270  614272      5850            JMP     MLIT            YES -- TRY TO MATCH LITERAL
014271  614243      5860            JMP     PUSH            IS A TYPE -- PUSH AND KEEP GOING
                    5870    *
                    5880    *       NEXT OBJECT IS AN ATOM -- TRY TO MATCH
                    5890    *
        014272      5900    MLIT    ...;
014272  555766      5910            SAD     (-1)            IF EMPTY
014273  614336      5920            JMP     TRUE            WE ALWAYS MATCH
014274  511713      5930            AND     TSFILL          MASK THE LITERAL FLAG
014275  054264      5940            DAC     TEMP            AND SAVE
        014276      5950            CHAR                    GET A CHARACTER FROM THE SOURCE STRING
014277  554264      5960            SAD     TEMP            SEE IF WE'VE MATCHED
014300  614336      5970            JMP     TRUE            YES -- SEE ABOUT THE NEXT THING
```

P PARSER -- FAIL

```
                    5980            .STITL  PARSER -- FAIL
                    5990    *
                    6000    *       THIS ROUTINE TRYS TO GET THE NEXT ALTERNATIVE WHEN A MATCH
                    6010    *       FAILS.  IF THERE ARE NO MORE ALTERNATIVES TO THE CURRENT DEFINITION,
                    6020    *       IT POPS THE CONTROL STACK AND TRYS FOR MORE IN THAT ONE.
                    6030    *       IF WE GET BACK TO THE BOTTOM OF THE STACK, WE'VE FAILED UTTERLY.
                    6040    *
                    6050    *       FALL THROUGH FROM THE PREVIOUS PAGE OR CALLS ITSELF.
                    6060    *
    014301          6070    FAIL    ...;
014301  234263      6080            LAC     ALT,X           GET THE NEXT ALTERNATIVE
014302  054263      6090            DAC     ALT             SAVE AS NEW ALTERNATIVE, MAYBE
014303  741200      6100            SNA                     SEE IF THERE WERE MORE
014304  614324      6110            JMP     FPOP            NO -- GO POP
                    6120    *
                    6130    *       BACK UP AND TRY AGAIN
                    6140    *
014305  777773      6150            LAW     -5              ***
014306  353330      6160            TAD     R$SDB           * FUDGE A POINTER INTO THE STACK
014307  040012      6170            DAC     X               ***
014310  220012      6180            LAC     X,X             GET THE OLD SOURCE POINTER
014311  040011      6190            DAC     T$CHRX          RESTORE IT
014312  220012      6200            LAC     X,X             IGNORE OLD ALTERNATIVE
014313  220012      6210            LAC     X,X             IGNORE OLD PART
014314  220012      6220            LAC     X,X             GET OLD OPERATOR STACK POINTER
014315  053745      6230            DAC     O$SDB           RESTORE IT
014316  220012      6240            LAC     X,X             GET OLD VARIABLE STACK POINTER
014317  054062      6250            DAC     V$SDB           RESTORE IT
014320  214263      6260            LAC     ALT             GET THE ONE WE JUST FIGURED OUT
014321  355765      6270            TAD     (2)             MAKE IT POINT TO THE FIRST PART
014322  054262      6280            DAC     PART            AND SAVE IN PART POINTER
014323  614266      6290            JMP     TEST            TRY AGAIN
                    6300    *
                    6310    *       TRY GETTING THE NEXT ALTERNATIVE FROM THE ONE ABOVE US
                    6320    *
    014324          6330    FPOP    ...;
    014324          6340            POP     R               GET VARIABLE STACK POINTER
    014325          6350            POP     R               GET OPERATOR STACK POINTER
    014326          6360            POP     R               GET PART POINTER
    014327          6370            POP     R               GET ALTERNATIVE POINTER
014330  054263      6380            DAC     ALT             AND USE IT
    014331          6390            POP     R               GET CHARACTER POINTER
014332  213330      6400            LAC     R$SDB           SEE IF WE'RE DONE
014333  353327      6410            SAD     R$IPTR          SEE IF AT THE BEGINNING
014334  615625      6420            JMP     E$FAIL          WE BLEW IT
014335  614301      6430            JMP     FAIL            TRY AGAIN
```

P PARSER -- SUCCESSFUL MATCH

```
                    6440            .STITL  PARSER -- SUCCESSFUL MATCH
                    6450    *
                    6460    *       COME HERE ON A SUCCESSFUL MATCH
                    6470    *
      014336        6480    TRUE    ...
014336  214262      6490            LAC     PART            SAVE WHATEVER WE JUST MATCHED
014337  054265      6500            DAC     LPART           IN CASE WE SHOULD WANT TO REFER TO IT
      014340        6510    TRUA    ...
014340  454262      6520            ISZ     PART            INCREMENT PART POINTER
014341  234262      6530            LAC     PART,X          GET THE POINTER TO THE SUCCESS ROUTINE (MAYBE)
014342  054264      6540            DAC     TEMP            SAVE IT IN CASE IT IS
014343  214263      6550            LAC     ALT             ***
014344  040012      6560            DAC     X               * GET THE PART COUNTER
014345  220012      6570            LAC     X,X             ***
014346  554262      6580            SAD     PART            SEE IF WE'RE DONE WITH THIS TYPE
014347  634264      6590            JMP     TEMP,X          DO WHAT IT SAYS
014350  614266      6600            JMP     TEST            ELSE TRY TO MATCH THIS PART
                    6610    *
                    6620    *       COME HERE AFTER DOING WHATEVER ACTION WAS NECESSARY
                    6630    *
                    6640    *       POP THE STACK AND GET THE NEXT PART OF THE CALLER
                    6650    *
      014351        6660    OK      ...
      014351        6670            POP     R               POP THE VARIABLE STACK
      014352        6680            POP     R               AND THE OPERATOR STACK
      014353        6690            POP     R               AND THE PART
014354  054262      6700            DAC     PART            SAVE AS CURRENT PART
      014355        6710            POP     R               GET THE ALTERNATIVE
014356  054263      6720            DAC     ALT             AND SAVE AS ALTERNATIVE
      014357        6730            POP     R               THROW AWAY THE SOURCE POINTER
014360  213330      6740            LAC     RSSDB           ***
014361  553327      6750            SAD     RSIPTR          * SEE IF WE'RE DONE
014362  614220      6760            JMP     PARSE           ***
014363  614340      6770            JMP     TRUA            TRY TO POP AGAIN
                    6780            .HEAD   I               I FOR INTERPRETER
```

```
                I                       INTERPRETER -- PRINT

                     6790               .STITL  INTERPRETER -- PRINT
                     6800       *
                     6810       *       THIS ROUTINE PRINTS THE VALUE OF THE THING
                     6820       *       POINTED TO BY THE TOP OF THE VARIABLE STACK
                     6830       *
          014364     6840       PRINT   ...
          014364     6850               POPV                    GET THE DATUM
     014367  114375  6860               JMS     PRT             CALL THE PRINT SUBROUTINE
     014370  111556  6870               JMS     T$MESS          PRINT CR/LF
     014371  000015  6880               .DATA   015,012,-1
     014372  000012
     014373  777777
     014374  614351  6890               JMP     P$OK
```

I INTERPRETER -- PRINT SUBROUTINE

```
                    6900            ,STITL  INTERPRETER -- PRINT SUBROUTINE
                    6910    *
                    6920    *       THIS ROUTINE PRINTS THE VALUE OF R-AC ON ENTRANCE
                    6930    *
014375  000000      6940    PRT     0
014376  054442      6950            DAC     NTEM            SAVE IT
014377  741200      6960            SNA                     SEE IF TO PRINT ZERO
014400  614437      6970            JMP     PRT0            YES
014401  740100      6980            SMA
014402  614411      6990            JMP     PRTA            POSITIVE -- PRINT NO SIGN
                    7000    *
                    7010    *       HANDLE NEGATIVE NUMBERS
                    7020    *
014403  740001      7030            CMA                     MAKE THE NUMBER POSITIVE
014404  054442      7040            DAC     NTEM            SAVE IT BACK
014405  741200      7050            SNA                     CHECK FOR THE OTHER ZERO
014406  614437      7060            JMP     PRT0            PRINT A ZERO IF SO
014407  760055      7070            LAW     55              PRINT A '-'
    014410          7080            PRINT                   PRINT IT
                    7090    *
                    7100    *       DIVIDE BY 10 AND STACK REMAINDER
                    7110    *       UNTIL A QUOTIENT OF 0
                    7120    *
    014411          7130    PRTA    ...
014411  777772      7140            LAW     -6              SET UP A COUNTER
014412  054264      7150            DAC     PSTEMP          THE NUMBER OF DIGITS
014413  214442      7160            LAC     NTEM            GET THE NUMBER
    014414          7170    PRTB    ...
014414  741200      7180            SNA                     SEE IF MORE TO DO
014415  614425      7190            JMP     PRTC            NO
014416  744000      7200            CLL                     CLEAR THE LINK
014417  657323      7210            IDIVS                   DIVIDE
014420  000012      7220            10.                     BY 10
    014421          7230            PUSH    V               PUT THE CHARACTER ON THE VARIABLE
014422  641002      7240            LACQ                    GET THE QUOTIENT INTO AC
014423  454264      7250            ISZ     PSTEMP          COUNT THIS DIGIT
014424  614414      7260            JMP     PRTB            IF MORE ROOM -- LOOP
```

I INTERPRETER -- PRINT SUBROUTINE

```
                    7270            .EJECT
                    7280    *
                    7290    *       PRINT THE DIGITS WE STACKED
                    7300    *
     014425         7310    PRTC    ..;
 014425  214264     7320            LAC     PSTEMP          ***
 014426  355752     7330            TAD     (5)             * SEE HOW MANY TO PRINT
 014427  740001     7340            CMA                     ***
 014430  054264     7350            DAC     PSTEMP          SAVE THE COUNT
     014431         7360    PRTD    ..;
     014431         7370            POP     V               ***
 014432  355767     7380            TAD     (60)            * PRINT THE DIGIT
     014433         7390            PRINT                   ***
 014434  454264     7400            ISZ     PSTEMP          SEE IF DONE
 014435  614431     7410            JMP     PRTD            NO -- GET ANOTHER
     014436         7420    PRTX    ..;
 014436  634375     7430            RET     PRT             AND RETURN
     014437         7440    PRTO    ..;
 014437  760060     7450            LAW     60              GET A ZERO
     014440         7460            PRINT                   PRINT IT
 014441  614436     7470            JMP     PRTX            AND EXIT
     014442         7480    NTEM    .BLOCK  1               TEMPORARY FOR NUMBERS
     014443         7490    TEM     .BLOCK  1               TEMPORARY FOR DEREFERENCING VARIABLES
```

```
I                       INTERPRETER -- GOTO

                7500            ,STITL  INTERPRETER -- GOTO
                7510    *
                7520    *       THIS ROUTINE SETS UP A TRANSFER OF CONTROL TO
                7530    *       THE LINE WHOSE NUMBER IS ON THE TOP OF THE VARIABLE STACK
                7540    *
    014444      7550    GOTO    ...
    014444      7560            POPV                    GET THE LINE NUMBER
014447  054442  7580            DAC     NTEM            SAVE IT
    014450      7582    GOTE    ...
014450  215760  7590            LAC     (TSSBFR-1)      GET THE BUFFER POINTER
    014451      7600    GOTA    ...
014451  040012  7610            DAC     X               SAVE IT
014452  214442  7620            LAC     NTEM            GET THE LINE NUMBER
014453  560012  7630            SAD     X,X             SEE IF WE MATCH
014454  614462  7640            JMP     GOTX            YES -- EXIT
014455  220012  7650            LAC     X,X             GET THE LENGTH OF THIS LINE
014456  741200  7660            SNA                     SEE IF WE'RE PAST THE END
014457  615666  7670            JMP     ESUND           UNDEFINED LINE NUMBER
014460  300012  7680            ADD     X               AND A POINTER TO THE NEXT ONE
014461  614451  7690            JMP     GOTA            AND LOOP
    014462      7700    GOTX    ...
014462  777777  7710            LAW     -1              DECREMENT THE POINTER BY ONE
014463  340012  7720            TAD     X               SO THAT IT IS PROPER FOR THIS LINE
014464  040017  7730            DAC     TSSBUF          SET THE NEW LINE
014465  614351  7740            JMP     PSOK            AND EXIT
014466  000000  7742    ITEM0   0
014467  000000  7744    ITEM1   0
```

```
I                                   INTERPRETER -- IF

                    7750            ,STITL  INTERPRETER -- IF
                    7760    *
                    7770    *       THIS ROUTINE DETERMINES THE TRUTH OR FALSITY OF A
                    7780    *       CONDITIONAL
                    7790    *
    014470          7800    IF      ...
    014470          7810            POPV                    GET THE LINE NUMBER
014473  054442      7820            DAC     NTEM            SAVE IT FOR THE TRANSFER ROUTINE
    014474          7830            POPV                    GET THE RIGHT RESULT
014477  054467      7840            DAC     ITEM1           SAVE IT
    014500          7850            POPV                    GET THE LEFT RESULT
014503  054466      7860            DAC     ITEM0           SAVE IT
    014504          7870            POP     0               ***
014505  054264      7880            DAC     PSTEMP          * BRANCH ON THE CONDITIONAL
014506  634264      7890            JMP     PSTEMP,X        ***
                    7920    *
                    7930    *       =
                    7940    *
    014507          7950    EQ      ...
014507  214466      7960            LAC     ITEM0           GET THE LEFT HALF
014510  554467      7970            SAD     ITEM1           COMPARE
014511  614450      7980            JMP     GOTE            EQUAL
014512  614351      7990            JMP     PSOK            NOT EQUAL
                    8000    *
                    8010    *       <
                    8020    *
    014513          8030    LESS    ...
014513  214467      8040            LAC     ITEM1           GET RIGHT RESULT
014514  740001      8050            CMA                     COMPLEMENT
014515  314466      8060            ADD     ITEM0           SUBTRACT FROM LEFT
014516  614522      8070            JMP     IFA             AND DO COMPARE
                    8080    *
                    8090    *       >
                    8100    *
    014517          8110    GTR     ...
014517  214466      8120            LAC     ITEM0           GET LEFT RESULT
014520  740001      8130            CMA                     COMPLEMENT
014521  314467      8140            ADD     ITEM1           SUBTRACT FROM RIGHT
    014522          8150    IFA     ...
014522  741100      8160            SPA                     COMPARE
014523  614450      8170            JMP     GOTE            TRUE
014524  614351      8180            JMP     PSOK            FALSE
                    8190    *
                    8200    *       <>
                    8210    *
    014525          8220    NEQ     ...
014525  214467      8230            LAC     ITEM1           GET RIGHT RESULT
014526  540000      8240            SAD                     COMPARE
014527  614351      8250            JMP     PSOK            FALSE
014530  614450      8260            JMP     GOTE            TRUE
```

```
                I                       INTERPRETER -- STOP/END

                          8270          ,STITL  INTERPRETER -- STOP/END
                          8280    *
                          8290    *     THIS ROUTINE HANDLES A STOP OR END INSTRUCTION
                          8300    *
                014531    8310    END   ...
014531  612173  8320                    JMP     TSMON           EXIT TO THE MONITOR
```

I INTERPRETER -- ASSIGN

```
                    8330            .STITL  INTERPRETER -- ASSIGN
                    8340    *
                    8350    *       THIS ROUTINE IS CALLED TO ASSIGN THE VALUE OF
                    8360    *       THE THING POINTED TO BY THE TOP ELEMENT OF THE
                    8370    *       VARIABLE STACK TO THE NEXT TO TOP ELEMENT OF THE STACK.
                    8380    *
     014532         8390    ASIGN   ..;
     014532         8400            POP     V               GET THE POINTER FROM WHICH TO ASSIGN
014533  054264      8410            DAC     PSTEMP          SAVE IT
     014534         8420            POP     V               GET THE POINTER TO ASSIGN
014535  054442      8430            DAC     NTEM            SAVE IT
014536  234264      8440            LAC     PSTEMP,X        GET THE VALUE
014537  074442      8450            DAC     NTEM,X          REPLACE IT
014540  614351      8460            JMP     PSOK            AND EXIT
```

```
                I                       INTERPRETER -- UNARY MINUS

                        8470            .STITL  INTERPRETER -- UNARY MINUS
                        8480    *
                        8490    *       THIS ROUTINE NEGATES THE TOP DATUM OF THE VARIABLE
                        8500    *       STACK.  IT ALSO MAPS -0 INTO PLUS 0
                        8510    *
            014541      8520    UNMIN   ..:
014541  114165  8530                    JMS     SSTEMP          GET A TEMPORARY FOR THE RESULT
014542  054442  8540                    DAC     NTEM            SAVE THE POINTER
            014543      8550            POPV                    POP THE VARIABLE
014546  740200  8560                    SZA                     IF TWO'S ZERO -- DON'T NEGATE
014547  740001  8570                    CMA                     NEGATE
014550  074442  8580                    DAC     NTEM,X          SAVE THE NEW VALUE
014551  214442  8590                    LAC     NTEM            GET THE REFERENCE
            014552      8600            PUSH    V               AND PUSH IT
014553  614351  8610                    JMP     PSOK            EXIT
```

```
I                                   INTERPRETER -- ADD

                    8620            .STITL  INTERPRETER -- ADD
                    8630    *
                    8640    *       THIS ROUTINE FORMS THE SUM OF THE TOP 1 OR TWO OBJECTS
                    8650    *       ON THE VARIABLE STACK.  IT DECIDES HOW MANY AND
                    8660    *       WHETHER TO ADD OR SUBTRACT BASED ON THE TOP OF THE OPERATOR
                    8670    *       STACK.
                    8680    *
    014554          8690    ADD     ...
014554  114755      8700            JMS     ITER            SEE IF WE'VE ITERATED
014555  015150      8710            .DATA   SSSTAG,
014556  114165      8720            JMS     SSTEMP          GET A TEMPORARY FOR THE RESULT
014557  054442      8730            DAC     NTEM            SAVE THE POINTER
    014560          8740            POP     0               GET THE OPERATION
014561  741200      8750            SNA                     SEE IF + OR -
014562  614571      8760            JMP     AD1             PLUS
    014563          8770            POPV                    GET THE SECOND TERM
014566  740200      8780            SZA                     DON'T NEGATE IF ZERO
014567  740001      8790            CMA                     NEGATE IT
014570  614574      8800            JMP     AD2
    014571          8810    AD1     ...
    014571          8820            POPV                    GET THE SECOND TERM
    014574          8830    AD2     ...
014574  074442      8840            DAC     NTEM,X          SAVE IT
    014575          8850            POPV                    GET FIRST TERM
014600  334442      8860            ADD     NTEM,X          ADD THEM
    014601          8870    ARTHX   ...
014601  074442      8880            DAC     NTEM,X          SAVE THE RESULT
014602  214442      8890            LAC     NTEM            GET POINTER
    014603          8900            PUSH    V               AND STACK IT
014604  614266      8910            JMP     PSTEST          AND EXIT
```

```
I                                INTERPRETER -- MULTIPLICATION

                        8920             .STITL  INTERPRETER -- MULTIPLICATION
                        8930     *
                        8940     *       THIS ROUTINE IS CALLED TO COMPUTE THE PRODUCT OF THE
                        8950     *       LAST N OBJECTS ON THE  VARIABLE STACK
                        8960     *
    014605              8970     MULT    ...,
014605  114755          8980             JMS     ITER            THIS MAY BE ITERATED
014606  015176          8990     MUL.    .DATA   SSPTAG.
014607  114165          9000             JMS     SSTEMP          ASSIGN A TEMPORARY
014610  054442          9010             DAC     NTEM            SAVE THE POINTER
    014611              9020             POP     0               GET  THE OPERATION
014612  315770          9030             ADD     (INST)          ***
014613  054264          9040             DAC     PSTEMP          * GET THE ACTUAL INSTRUCTION
014614  234264          9050             LAC     PSTEMP,X        ***
014615  054632          9060             DAC     MULY-1          SAVE TO EXECUTE
    014616              9070             POPV                    GET THE MULTIPLIER/DIVISOR
014621  054264          9080             DAC     PSTEMP          SAVE IT
    014622              9090             POPV                    GET THE MULTIPLICAND/DIVIDEND
014625  074442          9100             DAC     NTEM,X          SAVE IT
014626  214264          9110             LAC     PSTEMP          RESTORE MULTIPLIER/DIVISOR
014627  664000          9120             GSM                     GET ITS SIGN
014630  054633          9130             DAC     MULY            SAVE IT FOR THE OPERATION
014631  234442          9140             LAC     NTEM,X          RESTORE MULTIPLICAND/DIVIDENT
014632  657122          9150             MULS                    MULTIPLY/DIVIDE
014633  000000          9160     MULY    .DATA   0               MULTIPLIER/DIVISOR
014634  641002          9170             LACQ                    GET THE RESULT
014635  614601          9180             JMP     ARTHX           AND EXIT
014636  657122          9190     INST    MULS                    MULTIPLY INSTRUCTION
014637  657323          9200             IDIVS                   DIVIDE INSTRUCTION
```

```
                I                       INTERPRETER -- EXPONENTIATION

                        9210            .STITL  INTERPRETER -- EXPONENTIATION
                        9220    *
                        9230    *       THIS ROUTINE IS CALLED TO COMPUTE POWERS
                        9240    *
          014640        9250    EXP     ..,
014640  114755  9260                    JMS     ITER            THIS MAY BE ITERATED
014641  015224  9270    EXP.            .DATA   SSFTAG,
```

I INTERPRETER -- STACK AN OPERATOR

```
                    9280            .STITL  INTERPRETER -- STACK AN OPERATOR
                    9290    *
                    9300    *       THIS ROUTINE IS CALLED TO STACK AN OPERATOR ONTO
                    9310    *       THE OPERATOR STACK.
                    9320    *
    014642          9330    INSTK   ...
014642  454647      9340            ISZ     OP              SET FLAG FOR AN INVERSE OPERATION
    014643          9350    OPSTK   ...
014643  214647      9360            LAC     OP              GET THE OPERATION CODE
    014644          9370            PUSH    0               PUSH IT
014645  154647      9380            DZM     OP              CLEAR THE CODE BACK TO 0
014646  614351      9390            JMP     PSOK            AND EXIT
014647  000000      9400    OP      .DATA   0               OPERATOR CODE
```

```
                I                       INTERPRETER -- STACK A RELATIONAL

                        9410            .STITL  INTERPRETER -- STACK A RELATIONAL
                        9420    *
                        9430    *       THIS ROUTINE IS USED TO STACK THE RELATIONAL OPERATORS
                        9440    *
014650  454660  9450    RNEQ    ISZ     ROP             NOT EQUAL
014651  454660  9460    RGTR    ISZ     ROP             GREATER (>)
014652  454660  9470    RLES    ISZ     ROP             LESS (<)
    014653      9480    REQ     ...
014653  234660  9490            LAC     ROP,X           GET THE OPERATOR
    014654      9500            PUSH    0               PUSH ON THE OPERATOR STACK
014655  215771  9510            LAC     (RTAB)          REINITIALIZE THE POINTER
014656  054660  9520            DAC     ROP             REPLACE
014657  614351  9530            JMP     PSOK            AND EXIT
014660  014661  9540    ROP     .DATA   .+1             RELATIONAL OPERATOR
    014661      9550    RTAB    ...
014661  014507  9560            .DATA   EQ              EQUAL
014662  014513  9570            .DATA   LESS            LESS
014663  014517  9580            .DATA   GTR             GREATER
014664  014525  9590            .DATA   NEQ             NOT EQUAL
```

```
I                                   INTERPRETER -- STACK A VARIABLE

                    9600            .STITL  INTERPRETER -- STACK A VARIABLE
                    9610    *
                    9620    *       THIS ROUTINE IS CALLED TO PLACE A VARIABLE ON THE
                    9630    *       VARIABLE STACK, MAPPING IT APPROPRIATELY.
                    9640    *
    014665          9650    VSTK    ...
    014665          9660            LCHAR                   GET THE VARIABLE NAME
014666  355772      9670            TAD     (VTAB-101)      MAKE POINT TO THE TABLE
    014667          9680            PUSH    V               PUSH ONTO THE STACK
014670  614351      9690            JMP     P$OK            2AND EXIT
    014671          9700    VTAB    .BLOCK  26.             THE VARIABLE AREA FOR NOW
```

```
I                                   INTERPRETER -- STACK A DIGIT

                9710                .STITL  INTERPRETER -- STACK A DIGIT
                9720    *
                9730    *           THIS ROUTINE IS CALLED TO STACK A DIGIT ON THE VARIABLE
                9740    *           STACK,  IT DIFFERS FROM VSTK ONLY IN THAT IT DOESN'T MAP
                9750    *           THINGS
                9760    *
    014723      9770    DSTK        ,,,
    014723      9780                LCHAR                   GET THE DIGIT
    014724      9790                PUSH    V               STACK IT
014725  614351  9800                JMP     PSDK            AND EXIT
```

I INTERPRETER -- EVALUATE A NUMBER

```
                    9810            .STITL  INTERPRETER -- EVALUATE A NUMBER
                    9820    *
                    9830    *       THIS ROUTINE EVALUATES THE LAST N DIGITS ON THE
                    9840    *       VARIABLE STACK AS A NUMBER
                    9850    *
    014726          9860    EVAL3   ...
014726  114165      9870            JMS     S$TEMP          ***
014727  054264      9880            DAC     P$TEMP          * GET A TEMPORARY CELL AND STACK POINTER
    014730          9890            PUSH    V               ***
014731  214442      9900            LAC     NTEM            ***
014732  074264      9910            DAC     P$TEMP,X        * SAVE THE VALUE IN THE TEMPORARY
014733  614351      9920            JMP     PSOK            ***
                    9930    *
                    9940    *       THIS ROUTINE IS FOR THE FIRST DIGIT
                    9950    *
    014734          9960    EVAL1   ...
    014734          9970            POP     V               ***
014735  515773      9980            AND     (17)            * GET AND SAVE THE FIRST DIGIT
014736  054442      9990            DAC     NTEM            ***
014737  614351      10000           JMP     PSOK
                    10010   *
                    10020   *       THIS ROUTINE IS FOR ALL OTHER DIGITS
                    10030   *
    014740          10040   EVAL2   ...
014740  114755      10050           JMS     ITER            SEE IF WE ARE ITERATING
014741  015305      10060           .DATA   S$DTAG,
014742  744000      10070           CLL                     CLEAR THE LINK
014743  214442      10080           LAC     NTEM            ***
014744  653122      10090           MUL                     * SHIFT PREVIOUS TOTAL OVER FOR NEW ONE
014745  000012      10100           10.                     ***
    014746          10110           POP     V               GET THE DIGIT
014747  515773      10120           AND     (17)            MASK TO THE DIGIT
014750  054442      10130           DAC     NTEM            ***
014751  641002      10140           LACQ                    * ADD THE PREVIOUS TO THE NEW
014752  314442      10150           ADD     NTEM            ***
014753  054442      10160           DAC     NTEM            RESTORE
014754  614266      10170           JMP     P$TEST          AND TRY AGAIN
```

```
                I                       INTERPRETER -- SEE IF A FIELD IS ITERATED

                        10180           .STITL  INTERPRETER -- SEE IF A FIELD IS ITERATED
                        10190   *
                        10200   *       THIS ROUTINE CHECKS FOR AN ITERATED FIELD BY
                        10210   *       SEEING IF THE LAST MATCH WAS <EMPTY>
                        10220   *
    014755  000000      10230   ITER    0
    014756  214265      10240           LAC     PSLPART         GET THE LAST PART MATCHED
    014757  574755      10250           SAD     ITER,X          SEE IF WAS EMPTY OF THING WE TRIED FOR
    014760  614351      10260           JMP     PSOK            YES -- NO MORE
        014761          10270           SOS     PSPART          ELSE SUBTRACT ONE FROM PART
    014764  777774      10280           LAW     -4              ***
    014765  353330      10290           TAD     RSSDB           * GET  A POINTER TO THE SOURCE POINTER IN THE STACK
    014766  054264      10300           DAC     PSTEMP          ***
    014767  200011      10310           LAC     TSCHRX          GET THE CURRENT SOURCE POINTER
    014770  074264      10320           DAC     PSTEMP,X        FUDGE THE STACK
    014771  214755      10330           LAC     ITER            ***
    014772  040012      10340           DAC     X               * RETURN
    014773  620012      10350           RET     X               ***
```

I INTERPRETER -- EXIT

```
                    10360           .STITL  INTERPRETER -- EXIT
                    10370   *
                    10380   *       THIS ROUTINE IS A TEMPORARY COMMON EXIT FOR
                    10390   *       THE INTERPRETER,
                    10400   *
          014774    10410   EXIT    .,,
014774    111556    10420           JMS     T$MESS
014775    000015    10430           .DATA   15,12,-1
014776    000012
014777    777777
015000    614351    10440           JMP     P$OK            AND REALLY EXIT
                    10450           .HEAD   S               S FOR SYNTAX
```

S DEFINITION -- SYNTAX TABLE

```
                    10460           .STITL  DEFINITION -- SYNTAX TABLE
                    10470   *
                    10480   *       THIS TABLE IS DERIVED FROM A BNF SYNTAX FOR A SUBSET
                    10490   *       OF BASIC.  IT IS COMPOSED OF ENTRIES IN THE FOLLOWING
                    10500   *       FORMAT:
                    10510   *
                    10520   *       WORD 0    POINTER TO NEXT ALTERNATIVE DEFINITION
                    10530   *                 0 INDICATES THAT THIS IS THE LAST
                    10540   *       WORD 1    NUMBER OF PARTS TO THIS ALTERNATIVE
                    10550   *       WORD 2
                    10560   *         ,       PARTS -- POINTERS TO OTHER DEFINITIONS OR LITERAL
                    10570   *         ,       CHARACTERS.  LITERALS FLAGGED NEGATIVE.
                    10580   *       WORD N
                    10590   *       WORD N+1 POINTER TO ROUTINE ON SUCCESSFUL RECOGNITION.
                    10600   *
                    10610   *       DEFINITION -- SPECIAL SYNTAX SYMBOLS
                    10620   *
    400015          10630   EOL     .EQU    400015          END OF LINE -- LITERAL CR
    777777          10640   EMPTY   .EQU    777777          EMPTY -- A DISTINCTIVE LITERAL
                    10650           .EOT    BASSYN
    015001          9000    STATE   ...
015001  015005      9010            .DATA   .+4             POINTER TO NEXT ALTERNATIVE
015002  015004      9020            .DATA   .+2             NUMBER OF PARTS
015003  015025      9030            .DATA   LETS
015004  014351      9040            .DATA   PSOK
015005  015011      9050            .DATA   .+4             POINTER TO NEXT ALTERNATIVE
015006  015010      9060            .DATA   .+2             NUMBER OF PARTS
015007  015034      9070            .DATA   PRINT
015010  014351      9080            .DATA   PSOK
015011  015015      9090            .DATA   .+4             POINTER TO NEXT ALTERNATIVE
015012  015014      9100            .DATA   .+2             NUMBER OF PARTS
015013  015046      9110            .DATA   GOTO
015014  014351      9120            .DATA   PSOK
015015  015021      9130            .DATA   .+4             POINTER TO NEXT ALTERNATIVE
015016  015020      9140            .DATA   .+2             NUMBER OF PARTS
015017  015057      9150            .DATA   IF
015020  014351      9160            .DATA   PSOK
015021  000000      9170            .DATA   0               LAST ALTERNATIVE
015022  015024      9180            .DATA   .+2             NUMBER OF PARTS
015023  015116      9190            .DATA   END
015024  014351      9200            .DATA   PSOK
    015025          9210    LETS    ...
015025  000000      9220            .DATA   0               LAST ALTERNATIVE
015026  015033      9230            .DATA   .+5             NUMBER OF PARTS
015027  400114      9240            .DATA   400114
015030  400105      9250            .DATA   400105
015031  400124      9260            .DATA   400124
015032  015125      9270            .DATA   ASIGN
015033  014351      9280            .DATA   PSOK
    015034          9290    PRINT   ...
015034  000000      9300            .DATA   0               LAST ALTERNATIVE
015035  015045      9310            .DATA   .+8             NUMBER OF PARTS
```

S DEFINITION -- SYNTAX TABLE

```
015036  400120  9320            ,DATA   400120
015037  400122  9330            ,DATA   400122
015040  400111  9340            ,DATA   400111
015041  400116  9350            ,DATA   400116
015042  400124  9360            ,DATA   400124
015043  015134  9370            ,DATA   SUM
015044  400015  9380            ,DATA   EOL
015045  014364  9390            ,DATA   ISPRINT
    015046      9400    GOTO    ...
015046  000000  9410            ,DATA   0               LAST ALTERNATIVE
015047  015056  9420            ,DATA   .+7             NUMBER OF PARTS
015050  400107  9430            ,DATA   400107
015051  400117  9440            ,DATA   400117
015052  400124  9450            ,DATA   400124
015053  400117  9460            ,DATA   400117
015054  015266  9470            ,DATA   NUM
015055  400015  9480            ,DATA   EOL
015056  014444  9490            ,DATA   ISGOTO
    015057      9500    IF      ...
015057  000000  9510            ,DATA   0               LAST ALTERNATIVE
015060  015074  9520            ,DATA   .+14            NUMBER OF PARTS
015061  400111  9530            ,DATA   400111
015062  400106  9540            ,DATA   400106
015063  015134  9550            ,DATA   SUM
015064  015075  9560            ,DATA   RELOP
015065  015134  9570            ,DATA   SUM
015066  400107  9580            ,DATA   400107
015067  400117  9590            ,DATA   400117
015070  400124  9600            ,DATA   400124
015071  400117  9610            ,DATA   400117
015072  015266  9620            ,DATA   NUM
015073  400015  9630            ,DATA   EOL
015074  014470  9640            ,DATA   ISIF
    015075      9650    RELOP   ...
015075  015101  9660            ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015076  015100  9670            ,DATA   .+2             NUMBER OF PARTS
015077  400075  9680            ,DATA   400075
015100  014653  9690            ,DATA   ISREQ
015101  015106  9700            ,DATA   .+5             POINTER TO NEXT ALTERNATIVE
015102  015105  9710            ,DATA   .+3             NUMBER OF PARTS
015103  400074  9720            ,DATA   400074
015104  400076  9730            ,DATA   400076
015105  014650  9740            ,DATA   ISRNEQ
015106  015112  9750            ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015107  015111  9760            ,DATA   .+2             NUMBER OF PARTS
015110  400074  9770            ,DATA   400074
015111  014652  9780            ,DATA   ISRLES
015112  000000  9790            ,DATA   0               LAST ALTERNATIVE
015113  015115  9800            ,DATA   .+2             NUMBER OF PARTS
015114  400076  9810            ,DATA   400076
015115  014651  9820            ,DATA   ISRGTR
    015116      9830    END     ...
```

S DEFINITION -- SYNTAX TABLE

```
015116  000000  9840            .DATA   0               LAST ALTERNATIVE
015117  015124  9850            .DATA   .+5             NUMBER OF PARTS
015120  400105  9860            .DATA   400105
015121  400116  9870            .DATA   400116
015122  400104  9880            .DATA   400104
015123  400015  9890            .DATA   EOL
015124  014531  9900            .DATA   ISEND
    015125      9910    ASIGN   ...
015125  000000  9920            .DATA   0               LAST ALTERNATIVE
015126  015133  9930            .DATA   .+5             NUMBER OF PARTS
015127  015262  9940            .DATA   VAR
015130  400075  9950            .DATA   400075
015131  015134  9960            .DATA   SUM
015132  400015  9970            .DATA   EOL
015133  014532  9980            .DATA   ISASIGN
    015134      9990    SUM     ...
015134  000000  10000           .DATA   0               LAST ALTERNATIVE
015135  015140  10010           .DATA   .+3             NUMBER OF PARTS
015136  015162  10020           .DATA   PROD
015137  015141  10030           .DATA   STAG
015140  014554  10040           .DATA   ISADD
    015141      10050   STAG    ...
015141  015146  10060           .DATA   .+5             POINTER TO NEXT ALTERNATIVE
015142  015145  10070           .DATA   .+3             NUMBER OF PARTS
015143  015152  10080           .DATA   ADDOP
015144  015162  10090           .DATA   PROD
015145  014351  10100           .DATA   PSOK
015146  000000  10110           .DATA   0               LAST ALTERNATIVE
015147  015151  10120           .DATA   .+2             NUMBER OF PARTS
015150  777777  10130   STAG,   .DATA   EMPTY
015151  014351  10140           .DATA   PSOK
    015152      10150   ADDOP   ...
015152  015156  10160           .DATA   .+4             POINTER TO NEXT ALTERNATIVE
015153  015155  10170           .DATA   .+2             NUMBER OF PARTS
015154  400053  10180           .DATA   400053
015155  014643  10190           .DATA   ISOPSTK
015156  000000  10200           .DATA   0               LAST ALTERNATIVE
015157  015161  10210           .DATA   .+2             NUMBER OF PARTS
015160  400055  10220           .DATA   400055
015161  014642  10230           .DATA   ISINSTK
    015162      10240   PROD    ...
015162  000000  10250           .DATA   0               LAST ALTERNATIVE
015163  015166  10260           .DATA   .+3             NUMBER OF PARTS
015164  015210  10270           .DATA   FACT
015165  015167  10280           .DATA   PTAG
015166  014605  10290           .DATA   ISMULT
    015167      10300   PTAG    ...
015167  015174  10310           .DATA   .+5             POINTER TO NEXT ALTERNATIVE
015170  015173  10320           .DATA   .+3             NUMBER OF PARTS
015171  015200  10330           .DATA   MLOP
015172  015210  10340           .DATA   FACT
015173  014351  10350           .DATA   PSOK
```

S DEFINITION -- SYNTAX TABLE

```
015174  000000  10360          ,DATA   0               LAST ALTERNATIVE
015175  015177  10370          ,DATA   .+2             NUMBER OF PARTS
015176  777777  10380  FTAG,   ,DATA   EMPTY
015177  014351  10390          ,DATA   PSOK
    015200      10400  MLOP    ,,
015200  015204  10410          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015201  015203  10420          ,DATA   .+2             NUMBER OF PARTS
015202  400052  10430          ,DATA   400052
015203  014643  10440          ,DATA   ISOPSTK
015204  000000  10450          ,DATA   0               LAST ALTERNATIVE
015205  015207  10460          ,DATA   .+2             NUMBER OF PARTS
015206  400057  10470          ,DATA   400057
015207  014642  10480          ,DATA   ISINSTK
    015210      10490  FACT    ,,
015210  000000  10500          ,DATA   0               LAST ALTERNATIVE
015211  015214  10510          ,DATA   .+3             NUMBER OF PARTS
015212  015232  10520          ,DATA   SAE
015213  015215  10530          ,DATA   FTAG
015214  014640  10540          ,DATA   ISEXP
    015215      10550  FTAG    ,,
015215  015222  10560          ,DATA   .+5             POINTER TO NEXT ALTERNATIVE
015216  015221  10570          ,DATA   .+3             NUMBER OF PARTS
015217  015226  10580          ,DATA   EXPOP
015220  015232  10590          ,DATA   SAE
015221  014351  10600          ,DATA   PSOK
015222  000000  10610          ,DATA   0               LAST ALTERNATIVE
015223  015225  10620          ,DATA   .+2             NUMBER OF PARTS
015224  777777  10630  FTAG,   ,DATA   EMPTY
015225  014351  10640          ,DATA   PSOK
    015226      10650  EXPOP   ,,
015226  000000  10660          ,DATA   0               LAST ALTERNATIVE
015227  015231  10670          ,DATA   .+2             NUMBER OF PARTS
015230  400136  10680          ,DATA   400136
015231  014643  10690          ,DATA   ISOPSTK
    015232      10700  SAE     ,,
015232  015240  10710          ,DATA   .+6             POINTER TO NEXT ALTERNATIVE
015233  015237  10720          ,DATA   .+4             NUMBER OF PARTS
015234  400050  10730          ,DATA   400050
015235  015134  10740          ,DATA   SUM
015236  400051  10750          ,DATA   400051
015237  014351  10760          ,DATA   PSOK
015240  015244  10770          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015241  015243  10780          ,DATA   .+2             NUMBER OF PARTS
015242  015262  10790          ,DATA   VAR
015243  014351  10800          ,DATA   PSOK
015244  015250  10810          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015245  015247  10820          ,DATA   .+2             NUMBER OF PARTS
015246  015266  10830          ,DATA   NUM
015247  014351  10840          ,DATA   PSOK
015250  015255  10850          ,DATA   .+5             POINTER TO NEXT ALTERNATIVE
015251  015254  10860          ,DATA   .+3             NUMBER OF PARTS
015252  400055  10870          ,DATA   400055
```

S DEFINITION -- SYNTAX TABLE

```
015253  015232  10880           ,DATA   SAE
015254  014541  10890           ,DATA   ISUNMIN
015255  000000  10900           ,DATA   0               LAST ALTERNATIVE
015256  015261  10910           ,DATA   .+3             NUMBER OF PARTS
015257  400053  10920           ,DATA   400053
015260  015232  10930           ,DATA   SAE
015261  014351  10940           ,DATA   PSOK
    015262      10950   VAR     ,..
015262  000000  10960           ,DATA   0               LAST ALTERNATIVE
015263  015265  10970           ,DATA   .+2             NUMBER OF PARTS
015264  015357  10980           ,DATA   LET
015265  014665  10990           ,DATA   ISVSTK
    015266      11000   NUM     ,..
015266  000000  11010           ,DATA   0               LAST ALTERNATIVE
015267  015272  11020           ,DATA   .+3             NUMBER OF PARTS
015270  015273  11030           ,DATA   DHEAD
015271  015277  11040           ,DATA   DTAG
015272  014726  11050           ,DATA   ISEVAL3
    015273      11060   DHEAD   ,..
015273  000000  11070           ,DATA   0               LAST ALTERNATIVE
015274  015276  11080           ,DATA   .+2             NUMBER OF PARTS
015275  015307  11090           ,DATA   DIGIT
015276  014734  11100           ,DATA   ISEVAL1
    015277      11110   DTAG    ,..
015277  015303  11120           ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015300  015302  11130           ,DATA   .+2             NUMBER OF PARTS
015301  015307  11140           ,DATA   DIGIT
015302  014740  11150           ,DATA   ISEVAL2
015303  000000  11160           ,DATA   0               LAST ALTERNATIVE
015304  015306  11170           ,DATA   .+2             NUMBER OF PARTS
015305  777777  11180   DTAG.   ,DATA   EMPTY
015306  014351  11190           ,DATA   PSOK
    015307      11200   DIGIT   ,..
015307  015313  11210           ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015310  015312  11220           ,DATA   .+2             NUMBER OF PARTS
015311  400060  11230           ,DATA   400060
015312  014723  11240           ,DATA   ISDSTK
015313  015317  11250           ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015314  015316  11260           ,DATA   .+2             NUMBER OF PARTS
015315  400061  11270           ,DATA   400061
015316  014723  11280           ,DATA   ISDSTK
015317  015323  11290           ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015320  015322  11300           ,DATA   .+2             NUMBER OF PARTS
015321  400062  11310           ,DATA   400062
015322  014723  11320           ,DATA   ISDSTK
015323  015327  11330           ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015324  015326  11340           ,DATA   .+2             NUMBER OF PARTS
015325  400063  11350           ,DATA   400063
015326  014723  11360           ,DATA   ISDSTK
015327  015333  11370           ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015330  015332  11380           ,DATA   .+2             NUMBER OF PARTS
015331  400064  11390           ,DATA   400064
```

S DEFINITION -- SYNTAX TABLE

```
015332  014723  11400          ,DATA   I$DSTK
015333  015337  11410          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015334  015336  11420          ,DATA   .+2             NUMBER OF PARTS
015335  400065  11430          ,DATA   400065
015336  014723  11440          ,DATA   I$DSTK
015337  015343  11450          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015340  015342  11460          ,DATA   .+2             NUMBER OF PARTS
015341  400066  11470          ,DATA   400066
015342  014723  11480          ,DATA   I$DSTK
015343  015347  11490          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015344  015346  11500          ,DATA   .+2             NUMBER OF PARTS
015345  400067  11510          ,DATA   400067
015346  014723  11520          ,DATA   I$DSTK
015347  015353  11530          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015350  015352  11540          ,DATA   .+2             NUMBER OF PARTS
015351  400070  11550          ,DATA   400070
015352  014723  11560          ,DATA   I$DSTK
015353  000000  11570          ,DATA   0               LAST ALTERNATIVE
015354  015356  11580          ,DATA   .+2             NUMBER OF PARTS
015355  400071  11590          ,DATA   400071
015356  014723  11600          ,DATA   I$DSTK
    015357      11610  LET     .,,
015357  015363  11620          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015360  015362  11630          ,DATA   .+2             NUMBER OF PARTS
015361  400101  11640          ,DATA   400101
015362  014351  11650          ,DATA   P$OK
015363  015367  11660          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015364  015366  11670          ,DATA   .+2             NUMBER OF PARTS
015365  400102  11680          ,DATA   400102
015366  014351  11690          ,DATA   P$OK
015367  015373  11700          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015370  015372  11710          ,DATA   .+2             NUMBER OF PARTS
015371  400103  11720          ,DATA   400103
015372  014351  11730          ,DATA   P$OK
015373  015377  11740          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015374  015376  11750          ,DATA   .+2             NUMBER OF PARTS
015375  400104  11760          ,DATA   400104
015376  014351  11770          ,DATA   P$OK
015377  015403  11780          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015400  015402  11790          ,DATA   .+2             NUMBER OF PARTS
015401  400105  11800          ,DATA   400105
015402  014351  11810          ,DATA   P$OK
015403  015407  11820          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015404  015406  11830          ,DATA   .+2             NUMBER OF PARTS
015405  400106  11840          ,DATA   400106
015406  014351  11850          ,DATA   P$OK
015407  015413  11860          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015410  015412  11870          ,DATA   .+2             NUMBER OF PARTS
015411  400107  11880          ,DATA   400107
015412  014351  11890          ,DATA   P$OK
015413  015417  11900          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015414  015416  11910          ,DATA   .+2             NUMBER OF PARTS
```

S DEFINITION -- SYNTAX TABLE

```
015415  400110  11920          ,DATA   400110
015416  014351  11930          ,DATA   PSOK
015417  015423  11940          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015420  015422  11950          ,DATA   .+2             NUMBER OF PARTS
015421  400111  11960          ,DATA   400111
015422  014351  11970          ,DATA   PSOK
015423  015427  11980          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015424  015426  11990          ,DATA   .+2             NUMBER OF PARTS
015425  400112  12000          ,DATA   400112
015426  014351  12010          ,DATA   PSOK
015427  015433  12020          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015430  015432  12030          ,DATA   .+2             NUMBER OF PARTS
015431  400113  12040          ,DATA   400113
015432  014351  12050          ,DATA   PSOK
015433  015437  12060          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015434  015436  12070          ,DATA   .+2             NUMBER OF PARTS
015435  400114  12080          ,DATA   400114
015436  014351  12090          ,DATA   PSOK
015437  015443  12100          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015440  015442  12110          ,DATA   .+2             NUMBER OF PARTS
015441  400115  12120          ,DATA   400115
015442  014351  12130          ,DATA   PSOK
015443  015447  12140          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015444  015446  12150          ,DATA   .+2             NUMBER OF PARTS
015445  400116  12160          ,DATA   400116
015446  014351  12170          ,DATA   PSOK
015447  015453  12180          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015450  015452  12190          ,DATA   .+2             NUMBER OF PARTS
015451  400117  12200          ,DATA   400117
015452  014351  12210          ,DATA   PSOK
015453  015457  12220          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015454  015456  12230          ,DATA   .+2             NUMBER OF PARTS
015455  400120  12240          ,DATA   400120
015456  014351  12250          ,DATA   PSOK
015457  015463  12260          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015460  015462  12270          ,DATA   .+2             NUMBER OF PARTS
015461  400121  12280          ,DATA   400121
015462  014351  12290          ,DATA   PSOK
015463  015467  12300          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015464  015466  12310          ,DATA   .+2             NUMBER OF PARTS
015465  400122  12320          ,DATA   400122
015466  014351  12330          ,DATA   PSOK
015467  015473  12340          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015470  015472  12350          ,DATA   .+2             NUMBER OF PARTS
015471  400123  12360          ,DATA   400123
015472  014351  12370          ,DATA   PSOK
015473  015477  12380          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015474  015476  12390          ,DATA   .+2             NUMBER OF PARTS
015475  400124  12400          ,DATA   400124
015476  014351  12410          ,DATA   PSOK
015477  015503  12420          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015500  015502  12430          ,DATA   .+2             NUMBER OF PARTS
```

S DEFINITION -- SYNTAX TABLE

```
015501  400125  12440          ,DATA   400125
015502  014351  12450          ,DATA   PSOK
015503  015507  12460          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015504  015506  12470          ,DATA   .+2             NUMBER OF PARTS
015505  400126  12480          ,DATA   400126
015506  014351  12490          ,DATA   PSOK
015507  015513  12500          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015510  015512  12510          ,DATA   .+2             NUMBER OF PARTS
015511  400127  12520          ,DATA   400127
015512  014351  12530          ,DATA   PSOK
015513  015517  12540          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015514  015516  12550          ,DATA   .+2             NUMBER OF PARTS
015515  400130  12560          ,DATA   400130
015516  014351  12570          ,DATA   PSOK
015517  015523  12580          ,DATA   .+4             POINTER TO NEXT ALTERNATIVE
015520  015522  12590          ,DATA   .+2             NUMBER OF PARTS
015521  400131  12600          ,DATA   400131
015522  014351  12610          ,DATA   PSOK
015523  000000  12612          ,DATA   0               LAST ALTERNATIVE
015524  015526  12614          ,DATA   .+2             NUMBER OF PARTS
015525  400132  12616          ,DATA   400132
015526  014351  12618          ,DATA   PSOK
                12620          ,EOT    BAS2
                12000          ,HEAD   E               E FOR ERROR
                12010          ,PMC    SAVE,OFF
```

E ERROR MESSAGE ROUTINES

```
                    12020           .STITL  ERROR MESSAGE ROUTINES
                    12030   *
                    12040   *       THESE ROUTINES SEND ERROR MESSAGES FOR BOTH USER
                    12050   *       AND BASIC GENERATED ERRORS.
                    12060   *
    015527          12070   ERROR   ...
    015527          12071           MESS    < AT LINE >
015542  212240      12072           LAC     T$LNUM          GET THE LINE NUMBER
015543  114375      12073           JMS     I$PRT           PRINT IT
015544  111556      12080           JMS     T$MESS          ***
015545  000012      12090           .DATA   12,15           * SEND A CR/LF
015546  000015
015547  777777      12100           777777                  ***
015550  612173      12110           JMP     T$MON           AND EXIT TO THE MONITOR
                    12120   *
                    12130   *       PARSER ERROR
                    12140   *
    015551          12150   PARSE   ...
    015551          12160           MESS    <PARSER ERROR>
015567  615527      12170           JMP     ERROR           EXIT
                    12180   *
                    12190   *       STACK OVERFLOW
                    12200   *
    015570          12210   $OVF    ...
    015570          12220           MESS    <EXPRESSION TOO COMPLICATED>
015624  615527      12230           JMP     ERROR           EXIT
                    12240   *
                    12250   *       USER FAILED
                    12260   *
    015625          12270   FAIL    ...
    015625          12280           MESS    <SYNTAX ERROR>
015643  615527      12290           JMP     ERROR           EXIT
                    12300   *
                    12310   *       END IS NOT LAST
                    12320   *
    015644          12330   NEND    ...
    015644          12340           MESS    <END IS NOT LAST>
015665  615527      12350           JMP     ERROR           EXIT
                    12360   *
                    12370   *       UNDEFINED LINE NUMBER
                    12380   *
    015666          12390   UND     ...
    015666          12400           MESS    <UNDEFINED LINE NUMBER>
015715  615527      12410           JMP     ERROR
                    12420           .HEAD   0
```

RUN TIME INITIALIZATION

```
                    12430           .STITL  RUN TIME INITIALIZATION
                    12440   *
                    12450   *       THIS ROUTINE INITIALIZED THE INTERPRETER
                    12460   *
        015716      12470   UP      ...
015716  700416      12480           TLS     10              MAKE THE TTY FLAG COME UP
        015717      12482           MESS    <>,CRLF         GET A FRESH LINE
        015724      12484           MESS    <>,CRLF         AND ANOTHER
        015731      12490           MESS    <BASIC HERE>,CRLF
015747  612173      12500           JMP     TSMON           AND START UP THE PROGRAM
                    12510           ;PMC    RESTORE
015750  000030      12520           .LIT
015751  000137
015752  000005
015753  000015
015754  011716
015755  000012
015756  000040
015757  015000
015760  012240
015761  000122
015762  000114
015763  015001
015764  015003
015765  000002
015766  777777
015767  000060
015770  014636
015771  014661
015772  014570
015773  000017
015774  000000
015775  000000
015776  000000
        015777      12530   THIS    .END    SUP

TRANSFER ADDRESS  015716
```

CROSS REFERENCE TABLE

```
13315  DATUM     4830   4510   4680   4710   4750   4770   4070   4360   4650
15666  E  UND   12390   7670
15625  E FAIL   12270   6420
15604  E NEND   12330   5230
15570  E SOVF   12210   4500
15527  EERROR   12070  12170  12230  12290  12350  12410
15551  EPARSE   12150   2110   4730
   12  I    X     300   7610   7630   7650   7680   7720  10340  10350
   13  I    Y     310
   14  I    Z     320
14507  I   EQ    7950   9560
14470  I   IF    7800   9640
14647  I   OP    9400   9340   9360   9380
14571  I  AD1    8810   8760
14574  I  AD2    8830   8800
14554  I  ADD    8690  10040
14531  I  END    8310   9900
14640  I  EXP    9250  10540
14517  I  GTR    8110   9580
14522  I  IFA    8150   8070
14525  I  NEQ    8220   9590
14375  I  PRT    6940   3820   6860   7430  12073
14653  I  REQ    9480   9690
14660  I  ROP    9540   9450   9460   9470   9490   9520
14443  I  TEM    7490   6850   6850   7560   7560   7810   7810   7880   7830   7850   7850   8550   8550
                        8770   8770   8820   8820   8850   8850   9070   9070   9090   9090
14723  I DSTK    9770  11240  11280  11320  11360  11400  11440  11480  11520  11560  11600
14774  I EXIT   10410
14641  I EXP,    9270
14451  I GOTA    7600   7690
14450  I GOTE    7582   7980   8170   8260
14444  I GOTO    7550   9490
14462  I GOTX    7700   7640
14636  I INST    9190   9030
14755  I ITER   10230   8700   8980   9260  10050  10250  10330
14513  I LESS    8030   9570
14606  I MUL,    8990
14605  I MULT    8970  10290
14633  I MULY    9160   9060   9130
14442  I NTEM    7480   6950   7040   7160   7580   7620   7820   8430   8450   8540   8580   8590   8730
                        8840   8860   8880   8890   9010   9100   9140   9900   9990  10880  10130  10150
                       10160
14437  I PRTO    7440   6970   7060
14411  I PRTA    7130   6990
14414  I PRTB    7170   7260
14425  I PRTC    7310   7190
14431  I PRTD    7360   7410
14436  I PRTX    7420   7470
14651  I RGTR    9460   9820
14652  I RLES    9470   9780
14650  I RNEQ    9450   9740
14661  I RTAB    9550   9510
```

CROSS REFERENCE TABLE

14665	I VSTK	9650	10990										
14671	I VTAB	9700	9670										
14601	IARTHX	8870	9180										
14532	IASIGN	8390	9980										
14734	IEVAL1	9960	11100										
14740	IEVAL2	10040	11150										
14726	IEVAL3	9860	11050										
14642	IINSTK	9330	10230	10480									
14466	ITEMO	7742	7860	7960	8060	8120							
14467	ITEM1	7744	7840	7970	8040	8140	8230						
14643	IOPSTK	9350	10190	10440	10690								
14364	IPRINT	6840	9390										
14541	IUNMIN	8520	10890										
13314	MTWO	4820	4720										
13740	O POP	4440	4470	7870	8740	9020							
13745	O SDB	4530	4370	4450	5380	5550	6230						
13744	O IPTR	4520	5370										
13733	O PUSH	4350	4390	9370	9500								
100	O SIZE	4510	4560	4570									
13750	OSTACK	4570	4520	4540	4550								
12	P X	300	6170	6180	6200	6210	6220	6240	6560	6570			
13	P Y	310											
14	P Z	320											
14351	P OK	6660	6890	7740	7990	8180	8250	8460	8610	9390	9530	9690	9800	9920
			10000	10260	10440	9040	9080	9120	9180	9200	9280	10100	10140	10350
			10390	10600	10640	10760	10880	10840	10940	11190	11650	11690	11730	11770
			11810	11850	11890	11930	11970	12010	12050	12090	12130	12170	12210	12250
			12290	12330	12370	12410	12450	12490	12530	12570	12610	12618		
14263	P ALT	5680	5320	5510	5600	6080	6090	6260	6380	6550	6720			
14301	P FAIL	6070	6430											
14324	P FPOP	6330	6110											
14272	P MLIT	5900	5850											
14262	P PART	5670	5350	5530	5590	5620	5830	6280	6490	6520	6530	6580	6700	10270
			10270											
14243	P PUSH	5480	5860											
14264	P TEMP	5690	5940	5960	6540	6590	7150	7250	7320	7350	7400	7880	7890	8410
			8440	9040	9050	9080	9110	9880	9910	10300	10320			
14266	P TEST	5820	5430	5630	6290	6600	8910	10170						
14340	P TRUA	6510	6770											
14336	P TRUE	6480	5920	5970										
14265	PLPART	5700	6500	10240										
13275	POP	4640	4780	4170	4460	4750								
14220	PPARSE	5210	3670	6760										
13257	PUSH	4420	4540	4090	4380	4670								
13323	R POP	4150	4180	6340	6350	6360	6370	6390	6670	6680	6690	6710	6730	
13330	R SDB	4240	4080	4160	5280	6160	6400	6740	10290					
13327	R IPTR	4230	5270	6410	6750									
13316	R PUSH	4060	4100	5300	5330	5360	5390	5420	5500	5520	5540	5560	5580	
400	R SIZE	4220	4270	4280										
13333	RSTACK	4280	4230	4250	4260									
15057	S IF	9500	9150											
20	S TN	5060	5070	5100										

CROSS REFERENCE TABLE

```
 15116   S  END      9830    9190
400015   S  EOL     10630    9380    9480    9630    9890    9970
 15357   S  LET     11610   10980
 15266   S  NUM     11000    9470    9620   10830
 15232   S  SAE     10700   10520   10590   10880   10930
 15134   S  SUM      9990    9370    9550    9570    9960   10740
 15262   S  VAR     10950    9940   10790
 15277   S DTAG     11110   11040
 15210   S FACT     10490   10270   10340
 15215   S FTAG     10550   10530
 15046   S GOTO      9400    9110
 15025   S LETS      9210    9030
 15200   S MLOP     10400   10330
 15162   S PROD     10240   10020   10090
 15167   S PTAG     10300   10280
 15141   S STAG     10050   10030
 14176   S TCNT      5080    4970    5000
 14165   S TEMP      4950    5020    8530    8720    9000    9870
 14177   S TPRT      5090    5010
     1   $ UNIT       210    4960    5070    5070    5100
 15152  $ADDOP     10150   10080
 15125  $ASIGN      9910    9270
 15273  $DHEAD     11060   11030
 15307  $DIGIT     11200   11090   11140
 15305  $DTAG,     11180   10060
777777  $EMPTY     10640   10130   10380   10630   11180
 15226  $EXPOP     10650   10580
 15224  $FTAG,     10630    9270
 15034  $PRINT      9290    9070
 15176  $PTAG,     10380    8990
 15075  $RELOP      9650    9560
 15150  $STAG,     10130    8710
 15001  $STATE      9000    5310    5340
 14175  $TSIZE      5070    4990
     2   T    X       300    2410    2430    3500    3510
     3   T    Y       310
     4   T    Z       320
 11671   T  EOL      1690    1350
   100   T  IBN      1890    1900    1940
 12143   T  MON      3580    3200    3710    3800    8320   12110   12500
 11562   T  MSA       910     960
 11567   T  MSX       970     930
 11577   T  RDA      1190    1370    1650    1830
 12053   T  TEM      2440    2300    2320    2350    2370    2420    2640
 11701   T  WRU      1810    1320
 11660   T BACK      1560    1300
 12134   T BSBF      3110    2750    2760    2990    3020
 12022   T CHAR      2080    2130    2150    2290    2800    2860    3860    5950
    11   T CHRX       270    1150    1330    1580    1610    1720    1740    1760    2090    3270    3350    3400    3460
                              3480    5290    5490    6190   10310
 11713   T FILL      1880     980    1260    3340    3390    5930
 12033   T GNUM      2280    2340    2390    2400    2530    2570
```

CROSS REFERENCE TABLE

11714	T IBLN	1900	1120										
11720	T IBUF	1940	1590	1910									
12062	T LINA	2560	2660										
12074	T LINB	2700	2580										
12104	T LINC	2790	2910										
12105	T LIND	2810	2780										
12055	T LINE	2510	3030	3070	3100	3620							
12117	T LINF	2960	2850	2880									
12126	T LINX	3040	2540										
13247	T LISA	3850	3900										
13241	T LIST	3780	3690	3940									
13254	T LISX	3910	3890										
12240	T LNUM	3720	3220	3810	12072								
11556	T MESS	870	880	900	1420	1510	1820	3590	6870	10420	12071	12080	12160	12220
			12280	12340	12400	12482	12484	12490						
11717	T MONE	1930	920	1570	1620	1730	2180							
12150	T NLNA	3310	3430										
12164	T NLNX	3440	3370	3420									
11572	T READ	1110	1520	1770	2520								
12241	T SBFR	3730	3060	3600	3630	7590							
17	T SBUF	280	2720	2730	2740	2900	2980	3010	3050	3080	3090	3210	3230	3320
			3610	3640	7730									
12054	T TEM1	2450	2550	2600	2650	2710	2830	2890	2970					
10	T TTYX	260	890	950	1000									
11643	TCTRLX	1500	1280	1600										
15777	THIS	12530												
11716	TIBCNT	1920	1130	1360	1630	1640								
11715	TIBFPT	1910	1140	1750	3260	3470								
12032	TLCHAR	2220	2140	2770	3650	9660	9780							
12135	TNLINE	3170	3250	3490	3790	5220								
11551	TPRINT	630	670	940	990	1710	3840	3870	3980	7080	7390	7460		
15716	UP	12470	460	12530										
14055	V POP	4730	4760	6850	7370	7560	7810	7830	7850	8400	8420	8550	8770	8820
			8850	9070	9090	9970	10110							
14062	V SDB	4820	4660	4740	5410	5570	6250							
14061	V IPTR	4810	5400											
14050	V PUSH	4640	4680	7230	8600	8900	9680	9790	9890					
100	V SIZE	4800	4850	4860										
14065	VSTACK	4860	4810	4830	4840									
12	X	300	4430	4450	4480	4490	4650	4670	4690					
13	Y	310	4440	4530	4660	4760								
14	Z	320	4460	4520										

MACRO CROSS REFERENCE TABLE

CHAR	2020	2290	2800	2860	3860	5950							
LCHAR	2190	9660	9780										
MESS	770	1510	1820	3590	12071	12160	12220	12280	12340	12400	12482	12484	12490
POP	4180	6340	6350	6360	6370	6390	6670	6680	6690	6710	6730	6850	7370
		7560	7810	7830	7850	7870	8400	8420	8550	8740	8770	8820	8850
		9020	9070	9090	9970	10110							
POPV	4240	6850	7560	7810	7830	7850	8550	8770	8820	8850	9070	9090	
PRINT	570	940	990	1710	3840	3870	3930	7080	7390	7460			
PUSH	4080	5300	5330	5360	5390	5420	5500	5520	5540	5560	5580	7230	8600
		8900	9370	9500	9680	9790	9890						
SOS	360	10270											