```
DDDDDDDDDDD     EEEEEEEEEEEEEEE  BBBBBBBBBBB      UUU         UUU     GGGGGGGGGGG
DDDDDDDDDDDD    EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUU         UUU     GGGGGGGGGGGG
DDDDDDDDDDDD    EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUU         UUU     GGGGGGGGGGGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG
DDD         DDD EEEEEEEEEEE      BBBBBBBBBBBB     UUU         UUU GGG
DDD         DDD EEEEEEEEEEE      BBBBBBBBBBBB     UUU         UUU GGG
DDD         DDD EEEEEEEEEEE      BBBBBBBBBBBB     UUU         UUU GGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG    GGGGGGGGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG    GGGGGGGGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG    GGGGGGGGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG          GGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG          GGG
DDD         DDD EEE              BBB         BBB  UUU         UUU GGG          GGG
DDDDDDDDDDDD    EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUUUUUUUUUUUUUUU    GGGGGGGGG
DDDDDDDDDDDD    EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUUUUUUUUUUUUUUU    GGGGGGGGG
DDDDDDDDDDDD    EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUUUUUUUUUUUUUUU    GGGGGGGGG
```

FILEID**DBGNTYPE

```
DDDDDDDD    BBBBBBBB      GGGGGGGG  NN      NN  TTTTTTTTTT  YY      YY  PPPPPPPP    EEEEEEEEEE
DDDDDDDD    BBBBBBBB      GGGGGGGG  NN      NN  TTTTTTTTTT  YY      YY  PPPPPPPP    EEEEEEEEEE
DD      DD  BB      BB  GG          NN      NN      TT      YY      YY  PP      PP  EE
DD      DD  BB      BB  GG          NN      NN      TT      YY      YY  PP      PP  EE
DD      DD  BB      BB  GG          NNNN    NN      TT        YY  YY    PP      PP  EE
DD      DD  BB      BB  GG          NNNN    NN      TT        YY  YY    PP      PP  EE
DD      DD  BBBBBBBB    GG          NN  NN  NN      TT          YY      PPPPPPPP    EEEEEEEE
DD      DD  BBBBBBBB    GG          NN  NN  NN      TT          YY      PPPPPPPP    EEEEEEEE
DD      DD  BB      BB  GG  GGGGGG  NN    NNNN      TT          YY      PP          EE
DD      DD  BB      BB  GG  GGGGGG  NN    NNNN      TT          YY      PP          EE
DD      DD  BB      BB  GG      GG  NN      NN      TT          YY      PP          EE                ....
DD      DD  BB      BB  GG      GG  NN      NN      TT          YY      PP          EE                ....
DDDDDDDD    BBBBBBBB      GGGGGG    NN      NN      TT          YY      PP          EEEEEEEEEE        ....
DDDDDDDD    BBBBBBBB      GGGGGG    NN      NN      TT          YY      PP          EEEEEEEEEE        ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

DBG
V04

: R

: :

```
;     1          0001  0 MODULE DBGNTYPE (IDENT = 'V04-000') =
;     2          0002  0
;     3          0003  1 BEGIN
;     4          0004  1
;     5          0005  1 !
;     6          0006  1 !**********************************************************************
;     7          0007  1 !*                                                                    *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                             *
;    11          0011  1 !*                                                                    *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    17          0017  1 !*  TRANSFERRED.                                                       *
;    18          0018  1 !*                                                                    *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    21          0021  1 !*  CORPORATION.                                                       *
;    22          0022  1 !*                                                                    *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    25          0025  1 !*                                                                    *
;    26          0026  1 !*                                                                    *
;    27          0027  1 !**********************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:
;    32          0032  1 !
;    33          0033  1 !       DEBUG
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       This module contains the parse and execution networks for the TYPE
;    38          0038  1 !       verb. The parsing method employed is that of ATN's.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:
;    41          0041  1 !
;    42          0042  1 !       VAX/VMS
;    43          0043  1 !
;    44          0044  1 ! AUTHOR:
;    45          0045  1 !
;    46          0046  1 !       Richard Title
;    47          0047  1 !
;    48          0048  1 ! CREATION DATE:
;    49          0049  1 !
;    50          0050  1 !       9-14-81
;    51          0051  1 !
;    52          0052  1 ! VERSION:
;    53          0053  1 !
;    54          0054  1 !       V03.0-001
;    55          0055  1 !
;    56          0056  1 ! MODIFIED BY:
;    57          0057  1 !       V. Holt         May, 1982
```

```
;    58          0058  1 !
;    59          0059  1 ! REVISION HISTORY:
;    60          0060  1 ! 3.10  14-May-82      VJH     Added call to DBG$FLUSHBUF, eliminating need to
;    61          0061  1 !                             initialize local output buffer.
;    62          0062  1 ! 3.11   2-Jun-82      VJH     Removed all references to DBG$FAO_PUT and
;    63          0063  1 !                             DBG$OUT_PUT, replacing them with calls to
;    64          0064  1 !                             DBG$PRINT and DBG$NEWLINE, respectively.
;    65          0065  1 !--
```

```
;    67          0066  1 !
;    68          0067  1 ! TABLE OF CONTENTS:
;    69          0068  1 !
;    70          0069  1
;    71          0070  1 FORWARD ROUTINE
;    72          0071  1         DBG$NPARSE_TYPE,                ! Parse routine for TYPE
;    73          0072  1         DBG$NEXECUTE_TYPE,              ! Execution routine for TYPE
;    74          0073  1         DBG$NPARSE_EDIT,                ! Parse routine for EDIT
;    75          0074  1         DBG$NEXECUTE_EDIT;              ! Execution routine for EDIT
;    76          0075  1
;    77          0076  1 !
;    78          0077  1 ! REQUIRE FILES:
;    79          0078  1 !
;    80          0079  1
;    81          0080  1 REQUIRE 'SRC$:DBGPROLOG.REQ';
;    82          0214  1
;    83          0215  1 LIBRARY 'LIB$:DBGGEN.L32';
;    84          0216  1
;    85          0217  1 EXTERNAL ROUTINE
;    86          0218  1     dbg$sta_getsourcemod,               ! Gets module rst pointer
;    87          0219  1     dbg$sta_symname,                    ! Turns module rst pointer back
;    88          0220  1                                         ! into a string
;    89          0221  1     dbg$print:  NOVALUE,                ! Formats ascii output into output buf
;    90          0222  1     dbg$flushbuf:  NOVALUE,             ! Initializes new print line
;    91          0223  1     dbg$nnext_word,                     ! gets next word from input
;    92          0224  1     dbg$newline:  NOVALUE,              ! outputs the output buffer
;    93          0225  1     dbg$nmatch,                         ! Tries to match the next token
;    94          0226  1     dbg$get_tempmem,                    ! Allocates space
;    95          0227  1     dbg$nmake_arg_vect,                 ! Constructs error messages
;    96          0228  1     dbg$nsave_decimal_integer,          ! Reads an integer from the
;    97          0229  1                                         ! input string
;    98          0230  1     dbg$nsyntax_error,                  ! Reports a syntax error
;    99          0231  1
;   100          0232  1     dbg$scr_screen_normal: NOVALUE,     ! Set screen to normal in screen mode
;   101          0233  1     dbg$src_line_to_rec,                ! Translate line number to record num
;   102          0234  1     dbg$src_type_lnum_source,           ! The routine in DBGSOURCE that
;   103          0235  1                                         ! actually outputs the source
;   104          0236  1                                         ! lines to the terminal
;   105          0237  1     lib$do_command,                     ! Execute DCL command
;   106          0238  1     lib$spawn,                          ! Execute DCL command
;   107          0239  1     smg$set_keypad_mode,                ! Set terminal to numeric/application mode
;   108          0240  1     sys$fao;                            ! Formatted ASCII output
;   109          0241  1
;   110          0242  1 EXTERNAL
;   111          0243  1     DBG$GB_KEYPAD_INPUT: BYTE,          ! Set if mode is KEYPAD
;   112          0244  1     DBG$GL_KEYBOARD_ID,                 ! Used by SMG$SET_KEYPAD_MODE
;   113          0245  1     DBG$GV_CONTROL: DBG$CONTROL_FLAGS,  ! Global status flags
;   114          0246  1     DBG$GL_DEVELOPER: BITVECTOR[],      ! Developer bits
;   115          0247  1     DBG$GL_SCREEN_SOURCE,               ! Non-zero pointer if source is being
;   116          0248  1                                         !     directed to a screen display
;   117          0249  1     DBG$SRC_NEXT_MODRSTPTR,             ! Contains module RST pointer
;   118          0250  1                                         !     used by TYPE command
;   119          0251  1     DBG$SRC_NEXT_LNUM;                  ! Contains next line number to be
;   120          0252  1                                         !     output in case of a TYPE command
;   121          0253  1                                         !     with no argument
;   122          0254  1
;   123          0255  1 ! These values can go in the VERB_COMPOSITE field of the EDIT verb
```

```
;  124          0256  1 !
;  125          0257  1 LITERAL
;  126          0258  1     EDIT_EXIT    = 1,    ! EDIT/EXIT
;  127          0259  1     EDIT_NOEXIT = 2;    ! EDIT/NOEXIT
;  128          0260  1
```

```
;  130     0261  1 GLOBAL ROUTINE DBG$NPARSE_TYPE(INPUT_DESC, VERB_NODE, MESSAGE_VECT) =
;  131     0262  1 !
;  132     0263  1 ! Functional Description
;  133     0264  1 !
;  134     0265  1 !       ATN parse network for the TYPE verb.
;  135     0266  1 !       This routine takes a verb node for the TYPE verb, and a string
;  136     0267  1 !       descriptor for the remaining (unparsed) input.
;  137     0268  1 !       It constructs a linked list of noun nodes, each of which
;  138     0269  1 !       contains a module rst pointer and a line number range.
;  139     0270  1 !
;  140     0271  1 ! Formal Parameters
;  141     0272  1 !
;  142     0273  1 !       INPUT_DESC      - A longword containing the address of the
;  143     0274  1 !                         command input descriptor.
;  144     0275  1 !
;  145     0276  1 !       VERB_NODE       - A longword containing the address of the verb node
;  146     0277  1 !
;  147     0278  1 !       MESSAGE_VECT    - The address of a longword to contain the address
;  148     0279  1 !                         of a standard message argument vector.
;  149     0280  1 !
;  150     0281  1 ! Implicit Inputs
;  151     0282  1 !
;  152     0283  1 !       None
;  153     0284  1 !
;  154     0285  1 ! Implicit Outputs
;  155     0286  1 !
;  156     0287  1 !       On success, the command execution tree is constructed.
;  157     0288  1 !       On failure, a message argument vector is constructed or obtained.
;  158     0289  1 !
;  159     0290  1 ! Routine value
;  160     0291  1 !
;  161     0292  1 !       STS$K_SUCCESS (1)       - Success. Command execution tree constructed.
;  162     0293  1 !       STS$K_SEVERE  (4)       - Failure. Error encountered. Message argument
;  163     0294  1 !                                 constructed and returned.
;  164     0295  1 !
;  165     0296  1 ! Side Effects
;  166     0297  1 !
;  167     0298  1 !       None
;  168     0299  1 !
;  169     0300  1
;  170     0301  2 BEGIN
;  171     0302  2
;  172     0303  2 MAP
;  173     0304  2     input_desc  : REF dbg$stg_desc,
;  174     0305  2     verb_node   : REF dbg$verb_node;
;  175     0306  2
;  176     0307  2 BIND
;  177     0308  2     dbg$cs_colon        = UPLIT BYTE (1, dbg$k_colon),
;  178     0309  2     dbg$cs_slash        = UPLIT BYTE (1, dbg$k_slash),
;  179     0310  2     dbg$cs_comma        = UPLIT BYTE (1, dbg$k_comma),
;  180     0311  2     dbg$cs_cr           = UPLIT BYTE (1, dbg$k_car_return),
;  181     0312  2     dbg$cs_backslash    = UPLIT BYTE (1, dbg$k_backslash);
;  182     0313  2
;  183     0314  2 LOCAL
;  184     0315  2     modrstptr,
;  185     0316  2     noun_node       : REF dbg$noun_node,
;  186     0317  2     prev_noun_node  : REF dbg$noun_node,
```

```
;  187     0318  2     low_lnum,
;  188     0319  2     high_lnum,
;  189     0320  2     first_flag;                     ! True if this is the first
;  190     0321  2                                     ! line number range in the list.
;  191     0322  2
;  192     0323  2
;  193     0324  2     ! Create and link a noun node
;  194     0325  2     noun_node = dbg$get_tempmem(dbg$k_noun_node_size);
;  195     0326  2     verb_node[dbg$l_verb_object_ptr] = .noun_node;
;  196     0327  2
;  197     0328  2     first_flag = TRUE;
;  198     0329  2     modrstptr = 0;
;  199     0330  2
;  200     0331  2     ! Check for TYPE <cr>
;  201     0332  2
;  202     0333  2     IF dbg$nmatch(.input_desc, dbg$cs_cr, 1)
;  203     0334  2     THEN
;  204     0335  2
;  205     0336  3         BEGIN ! no argument supplied
;  206     0337  3
;  207     0338  3         ! Try to get default module.
;  208     0339  3
;  209     0340  3         IF .dbg$src_next_modrstptr EQL 0
;  210     0341  3         THEN
;  211     0342  4             BEGIN
;  212     0343  4             ! report an error
;  213     0344  4             .message_vect = dbg$nmake_arg_vect(dbg$_nonxtlin);
;  214     0345  4             RETURN sts$k_severe;
;  215     0346  3             END;
;  216     0347  3
;  217     0348  3         ! The module rst pointer is placed in the adjective field of
;  218     0349  3         ! the noun node.
;  219     0350  3
;  220     0351  3         noun_node [dbg$l_adjective_ptr] = .dbg$src_next_modrstptr;
;  221     0352  3
;  222     0353  3         ! Fill in the line numbers based on global info
;  223     0354  3
;  224     0355  3         noun_node [dbg$l_noun_value] = .dbg$src_next_lnum;
;  225     0356  3         noun_node [dbg$l_noun_value2] = .dbg$src_next_lnum;
;  226     0357  3
;  227     0358  3         ! The link field is zero
;  228     0359  3
;  229     0360  3         noun_node [dbg$l_noun_link] = 0;
;  230     0361  3
;  231     0362  3         END ! no argument supplied
;  232     0363  3
;  233     0364  3
;  234     0365  2     ELSE
;  235     0366  2
;  236     0367  2         WHILE TRUE DO
;  237     0368  2
;  238     0369  3             BEGIN
;  239     0370  3
;  240     0371  3             OWN
;  241     0372  3                 name_buf : VECTOR [81, BYTE]; ! Holds counted string with
;  242     0373  3                                              ! the module name
;  243     0374  3
```

```
;  244     0375  3          LOCAL
;  245     0376  3              string_ptr,
;  246     0377  3              length,
;  247     0378  3              char;
;  248     0379  3
;  249     0380  3          ! Loop through the list of line number ranges.
;  250     0381  3          ! The most general form of the command is:
;  251     0382  3
;  252     0383  3          ! TYPE mod1\l1:h1, mod2\l2:h2, ... , modn\ln:hn
;  253     0384  3
;  254     0385  3          ! The user may omit the module, in which case it
;  255     0386  3          ! defaults to a previously-specified module or
;  256     0387  3          ! to a module determined by the current scope.
;  257     0388  3          ! He may omit the high line num, in which case it
;  258     0389  3          ! is the same as the low line num.
;  259     0390  3
;  260     0391  3          name_buf[0] = 0;
;  261     0392  3          string_ptr = .input_desc[dsc$a_pointer];
;  262     0393  3          length = .input_desc[dsc$w_length];
;  263     0394  3
;  264     0395  3          ! read past leading blanks
;  265     0396  3
;  266     0397  3          WHILE .length GTR 0
;  267     0398  3          DO
;  268     0399  4              BEGIN
;  269     0400  4              char = ch$rchar_a(string_ptr);
;  270     0401  4              length = .length - 1;
;  271     0402  4              IF .char NEQ dbg$k_blank
;  272     0403  4              THEN
;  273     0404  4                  EXITLOOP;
;  274     0405  3              END;
;  275     0406  3
;  276     0407  3          ! If the length reaches zero then it is an error
;  277     0408  3
;  278     0409  3          IF .length EQL 0 AND .char EQL dbg$k_blank
;  279     0410  3          THEN
;  280     0411  4              BEGIN
;  281     0412  4              .message_vect = dbg$nmake_arg_vect(dbg$_needmore);
;  282     0413  4              RETURN sts$k_severe;
;  283     0414  3              END;
;  284     0415  3
;  285     0416  3          ! Read until we reach a separating character.
;  286     0417  3          ! Place the characters into name_buf as we read them.
;  287     0418  3
;  288     0419  3          WHILE .length GTR 0
;  289     0420  3          DO
;  290     0421  4              BEGIN
;  291     0422  4              IF .char EQL '\' OR .char EQL ','
;  292     0423  4              OR .char EQL ':' OR .char EQL '='
;  293     0424  4              THEN
;  294     0425  4                  EXITLOOP;
;  295     0426  4              name_buf[0] = .name_buf[0] + 1;
;  296     0427  4              name_buf[.name_buf[0]] = .char;
;  297     0428  4              char = ch$rchar_a(string_ptr);
;  298     0429  4              length = .length - 1;
;  299     0430  3              END;
;  300     0431  3
```

```
;   301     0432  3          IF .char EQL '\'  ! this signifies that we read a module name
;   302     0433  3          THEN
;   303     0434  4              BEGIN
;   304     0435  4
;   305     0436  4              ! Update the input descriptor
;   306     0437  4              input_desc[dsc$a_pointer] = .string_ptr;
;   307     0438  4              input_desc[dsc$w_length] = .length;
;   308     0439  4
;   309     0440  4              ! convert the name to an rst pointer and put it in the
;   310     0441  4              ! adjective field.
;   311     0442  4
;   312     0443  4              noun_node[dbg$l_adjective_ptr] =
;   313     0444  4                  dbg$sta_getsourcemod(name_buf);
;   314     0445  4
;   315     0446  4              ! If the above routine returned zero then the user entered
;   316     0447  4              ! an invalid module.
;   317     0448  4
;   318     0449  4              IF .noun_node[dbg$l_adjective_ptr] EQL 0
;   319     0450  4              THEN
;   320     0451  5                  BEGIN
;   321     0452  5                  .message_vect = dbg$nmake_arg_vect(
;   322     0453  5                      dbg$_nosuchmodu, 1, name_buf);
;   323     0454  5                  RETURN sts$k_severe;
;   324     0455  4                  END;
;   325     0456  4
;   326     0457  4              ! fill in new value of modrstptr
;   327     0458  4              modrstptr = .noun_node[dbg$l_adjective_ptr];
;   328     0459  4
;   329     0460  4              END ! pick up module name
;   330     0461  4
;   331     0462  3          ELSE
;   332     0463  3
;   333     0464  4              BEGIN ! fill in default module
;   334     0465  4
;   335     0466  4              IF .modrstptr NEQ 0
;   336     0467  4              THEN
;   337     0468  4                  ! use global default
;   338     0469  4                  noun_node[dbg$l_adjective_ptr] = .modrstptr
;   339     0470  4
;   340     0471  4              ELSE
;   341     0472  4
;   342     0473  5                  BEGIN
;   343     0474  5                  ! No default in modrstptr, so try to fill
;   344     0475  5                  ! in a module based on current scope
;   345     0476  5
;   346     0477  5                  modrstptr = dbg$sta_getsourcemod(0);
;   347     0478  5
;   348     0479  5                  IF .modrstptr EQL 0
;   349     0480  5                  THEN
;   350     0481  5
;   351     0482  6                      BEGIN
;   352     0483  6
;   353     0484  6                      ! If this is still zero, then we have no scope with
;   354     0485  6                      ! which to supply a module. Report an error.
;   355     0486  6
;   356     0487  6                      .message_vect = dbg$nmake_arg_vect(
;   357     0488  6                          dbg$_noscope, 1, .dbg$src_next_lnum);
```

```
;  358     0489  6                    RETURN sts$k_severe;
;  359     0490  6
;  360     0491  6                    END
;  361     0492  6
;  362     0493  5                ELSE
;  363     0494  5                    ! we have found a module.
;  364     0495  5                    ! fill in the adjective field.
;  365     0496  5                    noun_node[dbg$l_adjective_ptr] = .modrstptr;
;  366     0497  5
;  367     0498  5                END
;  368     0499  3            END; ! Fill in default module
;  369     0500  3
;  370     0501  3        ! read the low line num
;  371     0502  3
;  372     0503  3        IF NOT dbg$nsave_decimal_integer(.input_desc, low_lnum,
;  373     0504  3            .message_vect)
;  374     0505  3        THEN
;  375     0506  3            RETURN sts$k_severe;
;  376     0507  3
;  377     0508  3        ! Now look for colon which signifies that the user has also
;  378     0509  3        ! specified a high line num.
;  379     0510  3
;  380     0511  3        IF dbg$nmatch(.input_desc,dbg$cs_colon,1)
;  381     0512  3        THEN
;  382     0513  3
;  383     0514  4            BEGIN
;  384     0515  4
;  385     0516  4            ! Get high line num
;  386     0517  4
;  387     0518  4            IF NOT dbg$nsave_decimal_integer(
;  388     0519  4                .input_desc, high_lnum, .message_vect)
;  389     0520  4            THEN
;  390     0521  4                RETURN sts$k_severe;
;  391     0522  4
;  392     0523  4            END
;  393     0524  4
;  394     0525  3        ELSE
;  395     0526  3
;  396     0527  3            ! high line num same as low line num
;  397     0528  3
;  398     0529  3            high_lnum = .low_lnum;
;  399     0530  3
;  400     0531  3        ! Fill in the fields of noun_node
;  401     0532  3
;  402     0533  3        noun_node[dbg$l_noun_value] = .low_lnum;
;  403     0534  3        noun_node[dbg$l_noun_value2] = .high_lnum;
;  404     0535  3
;  405     0536  3        ! Link in the noun node
;  406     0537  3
;  407     0538  3        IF NOT .first_flag
;  408     0539  3        ! The first noun has already been linked to the verb
;  409     0540  3        THEN
;  410     0541  3
;  411     0542  4            BEGIN
;  412     0543  4
;  413     0544  4            prev_noun_node[dbg$l_noun_link] = .noun_node;
;  414     0545  4            noun_node[dbg$l_noun_link] = 0;
```

```
;  415          0546  4
;  416          0547  3                END;
;  417          0548  3
;  418          0549  3            ! set up for next time around loop
;  419          0550  3            first_flag = FALSE;
;  420          0551  3            prev_noun_node = .noun_node;
;  421          0552  3
;  422          0553  3            ! Now look for comma or for end of line
;  423          0554  3
;  424          0555  3            IF dbg$nmatch(.input_desc,dbg$cs_cr,1)
;  425          0556  3            THEN
;  426          0557  3                EXITLOOP;
;  427          0558  3
;  428          0559  3            ! If we did not see end of line, then the next character
;  429          0560  3            ! must be a comma.
;  430          0561  3
;  431          0562  3            IF NOT dbg$nmatch(.input_desc,dbg$cs_comma,1)
;  432          0563  3            THEN
;  433          0564  3
;  434          0565  4                BEGIN
;  435          0566  4
;  436          0567  4                .message_vect =
;  437          0568  4                    dbg$nsyntax_error(dbg$nnext_word(.input_desc));
;  438          0569  4                RETURN sts$k_severe;
;  439          0570  4
;  440          0571  3                END;
;  441          0572  3
;  442          0573  3            ! Now create a new noun node for the next time around the loop.
;  443          0574  3            noun_node = dbg$get_tempmem(dbg$k_noun_node_size);
;  444          0575  2            END; ! While loop
;  445          0576  2
;  446          0577  2        RETURN sts$k_success;
;  447          0578  2
;  448          0579  1 END; ! dbg$nparse_type
; INFO#250                 L1:0544
; Referenced LOCAL symbol PREV_NOUN_NODE is probably not initialized


                                        .TITLE  DBGNTYPE
                                        .IDENT  \V04-000\

                                        .PSECT  DBG$PLIT,NOWRT,  SHR,  PIC,0

3A  01  00000 P.AAA:    .BYTE   1, 58                                           ;
2F  01  00002 P.AAB:    .BYTE   1, 47                                           ;
2C  01  00004 P.AAC:    .BYTE   1, 44                                           ;
0D  01  00006 P.AAD:    .BYTE   1, 13                                           ;
5C  01  00008 P.AAE:    .BYTE   1, 92                                           ;

                                        .PSECT  DBG$OWN,NOEXE,  PIC,2

                00000 NAME_BUF:
                                        .BLKB   81

                      DBG$CS_COLON=     P.AAA
                      DBG$CS_SLASH=     P.AAB
                      DBG$CS_COMMA=     P.AAC
```

```
                        DBG$CS_CR=              P.AAD
                        DBG$CS_BACKSLASH=       P.AAE
                                .EXTRN  DBG$STA_GETSOURCEMOD
                                .EXTRN  DBG$STA_SYMNAME
                                .EXTRN  DBG$PRINT, DBG$FLUSHBUF
                                .EXTRN  DBG$NNEXT_WORD, DBG$NEWLINE
                                .EXTRN  DBG$NMATCH, DBG$GET_TEMPMEM
                                .EXTRN  DBG$NMAKE_ARG_VECT
                                .EXTRN  DBG$NSAVE_DECIMAL_INTEGER
                                .EXTRN  DBG$NSYNTAX_ERROR
                                .EXTRN  DBG$SCR_SCREEN_NORMAL
                                .EXTRN  DBG$SRC_LINE_TO_REC
                                .EXTRN  DBG$SRC_TYPE_LNUM_SOURCE
                                .EXTRN  LIB$DO_COMMAND, LIB$SPAWN
                                .EXTRN  SMG$SET_KEYPAD_MODE
                                .EXTRN  SYS$FAO, DBG$GB_KEYPAD_INPUT
                                .EXTRN  DBG$GL_KEYBOARD_ID
                                .EXTRN  DBG$GV_CONTROL, DBG$GL_DEVELOPER
                                .EXTRN  DBG$GL_SCREEN_SOURCE
                                .EXTRN  DBG$SRC_NEXT_MODRSTPTR
                                .EXTRN  DBG$SRC_NEXT_LNUM

                                .PSECT  DBG$CODE,NOWRT,  SHR,  PIC,0

                    0FFC 00000          .ENTRY  DBG$NPARSE_TYPE, Save R2,R3,R4,R5,R6,R7,R8,-; 0261
                                                R9,R10,R11
                5B 00000000'  EF  9E 00002          MOVAB   DBG$CS_CR, R11                  ;
                5A 00000000'  EF  9E 00009          MOVAB   NAME_BUF, R10                   ;
                5E            08  C2 00010          SUBL2   #8, SP                          ;
                              04  DD 00013          PUSHL   #4                              ; 0325
00000000G       00                01  FB 00015          CALLS   #1, DBG$GET_TEMPMEM         ;
                52                50  D0 0001C          MOVL    R0, NOUN_NODE               ;
                50          08    AC  D0 0001F          MOVL    VERB_NODE, R0               ; 0326
          08    A0                52  D0 00023          MOVL    NOUN_NODE, 8(R0)            ;
                59                01  D0 00027          MOVL    #1, FIRST_FLAG              ; 0328
                                  57  D4 0002A          CLRL    MODRSTPTR                   ; 0329
                                  01  DD 0002C          PUSHL   #1                          ; 0333
                                  5B  DD 0002E          PUSHL   R11                         ;
                54          04    AC  D0 00030          MOVL    INPUT_DESC, R4              ;
                                  54  DD 00034          PUSHL   R4                          ;
00000000G       00                03  FB 00036          CALLS   #3, DBG$NMATCH              ;
                2C                50  E9 0003D          BLBC    R0, 2$                      ;
                      00000000G   00  D5 00040          TSTL    DBG$SRC_NEXT_MODRSTPTR      ; 0340
                                  08  12 00046          BNEQ    1$                          ;
                      00028CE0    8F  DD 00048          PUSHL   #167136                     ; 0344
                                  42  11 0004E          BRB     5$                          ;
          04    A2  00000000G   00  D0 00050  1$:       MOVL    DBG$SRC_NEXT_MODRSTPTR, 4(NOUN_NODE)  ; 0351
                50  00000000G   00  D0 00058          MOVL    DBG$SRC_NEXT_LNUM, R0       ; 0355
                62                50  D0 0005F          MOVL    R0, (NOUN_NODE)             ;
          0C    A2                50  D0 00062          MOVL    R0, 12(NOUN_NODE)           ; 0356
                            08    A2  D4 00066          CLRL    8(NOUN_NODE)                ; 0360
                                0157  31 00069          BRW     19$                         ; 0333
                                  6A  94 0006C  2$:       CLRB    NAME_BUF                ; 0391
                56          04    A4  D0 0006E          MOVL    4(R4), STRING_PTR           ; 0392
                55                64  3C 00072          MOVZWL  (R4), LENGTH                ; 0393
                                  55  D5 00075  3$:       TSTL    LENGTH                  ; 0397
                                  0A  15 00077          BLEQ    4$                      ;
```

```
                53              86  9A 00079          MOVZBL  (STRING_PTR)+, CHAR                     ; 0400
                                55  D7 0007C          DECL    LENGTH                                  ; 0401
                20              53  D1 0007E          CMPL    CHAR, #32                               ; 0402
                                F2  13 00081          BEQL    3$
                                55  D5 00083 4$:      TSTL    LENGTH                                  ; 0409
                                15  12 00085          BNEQ    6$
                20              53  D1 00087          CMPL    CHAR, #32
                                10  12 0008A          BNEQ    6$
                    000280D0    8F  DD 0008C          PUSHL   #164048                                 ; 0412
00000000G       00              01  FB 00092 5$:      CALLS   #1, DBG$NMAKE_ARG_VECT
                              0110  31 00099          BRW     16$
                                55  D5 0009C 6$:      TSTL    LENGTH                                  ; 0419
                                28  15 0009E          BLEQ    7$
0000005C        8F              53  D1 000A0          CMPL    CHAR, #92                               ; 0422
                                1F  13 000A7          BEQL    7$
                2C              53  D1 000A9          CMPL    CHAR, #44
                                1A  13 000AC          BEQL    7$
                3A              53  D1 000AE          CMPL    CHAR, #58                               ; 0423
                                15  13 000B1          BEQL    7$
                20              53  D1 000B3          CMPL    CHAR, #32
                                10  13 000B6          BEQL    7$
                                6A  96 000B8          INCB    NAME_BUF                                ; 0426
                50              6A  9A 000BA          MOVZBL  NAME_BUF, R0                            ; 0427
              6A40              53  90 000BD          MOVB    CHAR, NAME_BUF[R0]
                53              86  9A 000C1          MOVZBL  (STRING_PTR)+, CHAR                     ; 0428
                                55  D7 000C4          DECL    LENGTH                                  ; 0429
                                D4  11 000C6          BRB     6$                                      ; 0419
0000005C        8F              53  D1 000C8 7$:      CMPL    CHAR, #92                               ; 0432
                                28  12 000CF          BNEQ    9$
          04    A4              56  D0 000D1          MOVL    STRING_PTR, 4(R4)                       ; 0437
                64              55  B0 000D5          MOVW    LENGTH, (R4)                            ; 0438
                                5A  DD 000D8          PUSHL   R10                                     ; 0444
00000000G       00              01  FB 000DA          CALLS   #1, DBG$STA_GETSOURCEMOD
          04    A2              50  D0 000E1          MOVL    R0, 4(NOUN_NODE)
                                0C  12 000E5          BNEQ    8$                                      ; 0449
                                5A  DD 000E7          PUSHL   R10                                     ; 0452
                                01  DD 000E9          PUSHL   #1
                    000281E8    8F  DD 000EB          PUSHL   #164328
                                26  11 000F1          BRB     10$
                57        04    A2  D0 000F3 8$:      MOVL    4(NOUN_NODE), MODRSTPTR                 ; 0458
                                2E  11 000F7          BRB     12$                                     ; 0432
                                57  D5 000F9 9$:      TSTL    MODRSTPTR                               ; 0466
                                26  12 000FB          BNEQ    11$
                                7E  D4 000FD          CLRL    -(SP)                                   ; 0477
00000000G       00              01  FB 000FF          CALLS   #1, DBG$STA_GETSOURCEMOD
                57              50  D0 00106          MOVL    R0, MODRSTPTR
                                18  12 00109          BNEQ    11$                                     ; 0479
                    00000000G   00  DD 0010B          PUSHL   DBG$SRC_NEXT_LNUM                       ; 0488
                                01  DD 00111          PUSHL   #1                                      ; 0487
                    00028972    8F  DD 00113          PUSHL   #166258
00000000G       00              03  FB 00119 10$:     CALLS   #3, DBG$NMAKE_ARG_VECT
                              0089  31 00120          BRW     16$
          04    A2              57  D0 00123 11$:     MOVL    MODRSTPTR, 4(NOUN_NODE)                 ; 0496
                        0C      AC  DD 00127 12$:     PUSHL   MESSAGE_VECT                            ; 0504
                        04      AE  9F 0012A          PUSHAB  LOW_LNUM                                ; 0503
                                54  DD 0012D          PUSHL   R4
00000000G       00              03  FB 0012F          CALLS   #3, DBG$NSAVE_DECIMAL_INTEGER           ;
```

```
                77              50  E9 00136          BLBC    R0, 17$                                 ;
                                01  DD 00139          PUSHL   #1                                      ; 0511
                            FA  AB  9F 0013B          PUSHAB  DBG$CS_COLON                            ;
                                54  DD 0013E          PUSHL   R4                                      ;
00000000G       00              03  FB 00140          CALLS   #3, DBG$NMATCH                          ;
                14              50  E9 00147          BLBC    R0, 13$                                 ;
                            0C  AC  DD 0014A          PUSHL   MESSAGE_VECT                            ; 0519
                            08  AE  9F 0014D          PUSHAB  HIGH_LNUM                               ; 0518
                                54  DD 00150          PUSHL   R4                                      ; 0519
00000000G       00              03  FB 00152          CALLS   #3, DBG$NSAVE_DECIMAL_INTEGER           ;
                06              50  E8 00159          BLBS    R0, 14$                                 ;
                                52  11 0015C          BRB     17$                                     ; 0521
                04  AE          6E  D0 0015E 13$:      MOVL    LOW_LNUM, HIGH_LNUM                     ; 0529
                62              6E  D0 00162 14$:      MOVL    LOW_LNUM, (NOUN_NODE)                   ; 0533
                0C  A2      04  AE  D0 00165          MOVL    HIGH_LNUM, 12(NOUN_NODE)                ; 0534
                07              59  E8 0016A          BLBS    FIRST_FLAG, 15$                         ; 0538
                08  A8          52  D0 0016D          MOVL    NOUN_NODE, 8(PREV_NOUN_NODE)            ; 0544
                            08  A2  D4 00171          CLRL    8(NOUN_NODE)                            ; 0545
                                59  D4 00174 15$:      CLRL    FIRST_FLAG                              ; 0550
                58              52  D0 00176          MOVL    NOUN_NODE, PREV_NOUN_NODE               ; 0551
                                01  DD 00179          PUSHL   #1                                      ; 0555
                          0810  8F  BB 0017B          PUSHR   #^M<R4,R11>                             ;
00000000G       00              03  FB 0017F          CALLS   #3, DBG$NMATCH                          ;
                3A              50  E8 00186          BLBS    R0, 19$                                 ;
                                01  DD 00189          PUSHL   #1                                      ; 0562
                            FE  AB  9F 0018B          PUSHAB  DBG$CS_COMMA                            ;
                                54  DD 0018E          PUSHL   R4                                      ;
00000000G       00              03  FB 00190          CALLS   #3, DBG$NMATCH                          ;
                1A              50  E8 00197          BLBS    R0, 18$                                 ;
                                54  DD 0019A          PUSHL   R4                                      ; 0568
00000000G       00              01  FB 0019C          CALLS   #1, DBG$NNEXT_WORD                      ;
                                50  DD 001A3          PUSHL   R0                                      ;
00000000G       00              01  FB 001A5          CALLS   #1, DBG$NSYNTAX_ERROR                   ;
                0C  BC          50  D0 001AC 16$:      MOVL    R0, @MESSAGE_VECT                       ;
                50              04  D0 001B0 17$:      MOVL    #4, R0                                  ; 0569
                                    04 001B3          RET                                             ;
                                04  DD 001B4 18$:      PUSHL   #4                                      ; 0574
00000000G       00              01  FB 001B6          CALLS   #1, DBG$GET_TEMPMEM                     ;
                52              50  D0 001BD          MOVL    R0, NOUN_NODE                           ;
                              FEA9  31 001C0          BRW     2$                                      ; 0367
                50              01  D0 001C3 19$:      MOVL    #1, R0                                  ; 0577
                                    04 001C6          RET                                             ; 0579

; Routine Size:  455 bytes,    Routine Base:  DBG$CODE + 0000

;  449          0580  1
;  450          0581  1
```

```
;  452     0582  1 GLOBAL ROUTINE DBG$NEXECUTE_TYPE(VERB_NODE, MESSAGE_VECT) =
;  453     0583  1 !
;  454     0584  1 ! Functional Description
;  455     0585  1 !
;  456     0586  1 !       This routine performs the action associated with the TYPE
;  457     0587  1 !       command.
;  458     0588  1 !
;  459     0589  1 ! Formal Parameters
;  460     0590  1 !
;  461     0591  1 !       VERB_NODE       - A longword containing the address of the
;  462     0592  1 !                         head (verb) node.
;  463     0593  1 !
;  464     0594  1 !       MESSAGE_VECT    - The address of a longword to contain the
;  465     0595  1 !                         address of an error message vector
;  466     0596  1 !
;  467     0597  1 ! Implicit Inputs
;  468     0598  1 !
;  469     0599  1 !       The command tree contains a verb node and a linked list
;  470     0600  1 !       of one or more noun nodes.
;  471     0601  1 !
;  472     0602  1 ! Implicit Outputs
;  473     0603  1 !
;  474     0604  1 !       This routine calls a routine in DBGSOURCE which displays the
;  475     0605  1 !       source lines to the user.
;  476     0606  1 !
;  477     0607  1 ! Routine Value
;  478     0608  1 !
;  479     0609  1 !       A completion code.
;  480     0610  1 !
;  481     0611  1 ! Completion Codes
;  482     0612  1 !
;  483     0613  1 !       STS$K_SUCCESS (1)       - Success. Command executed
;  484     0614  1 !       STS$K_SEVERE (4)        - Failure. The command could not be
;  485     0615  1 !                                 executed. An error message is constructed.
;  486     0616  1 !
;  487     0617  1 ! Side Effects
;  488     0618  1 !
;  489     0619  1 !       None
;  490     0620  1 !
;  491     0621  1
;  492     0622  2 BEGIN
;  493     0623  2
;  494     0624  2 MAP
;  495     0625  2     verb_node : REF dbg$verb_node;
;  496     0626  2
;  497     0627  2 LOCAL
;  498     0628  2     noun_node : REF dbg$noun_node,
;  499     0629  2     modnameptr;
;  500     0630  2
;  501     0631  2     noun_node = .verb_node[dbg$l_verb_object_ptr];
;  502     0632  2
;  503     0633  2     DO
;  504     0634  3         BEGIN
;  505     0635  3
;  506     0636  3         ! loop through linked list of noun nodes.
;  507     0637  3
;  508     0638  3         ! First output the module name.
```

```
;  509     0639  3
;  510     0640  3          ! Set up for FAO_PUT.
;  511     0641  3          !
;  512     0642  3          DBG$FLUSHBUF();
;  513     0643  3
;  514     0644  3
;  515     0645  3          ! Get the module name and print it out.  Note that the printing of
;  516     0646  3          ! the module name is suppressed if the source line output is being
;  517     0647  3          ! directed to a source screen display.
;  518     0648  3          !
;  519     0649  3          IF .DBG$GL_SCREEN_SOURCE EQL 0
;  520     0650  3          THEN
;  521     0651  4              BEGIN
;  522     0652  4              DBG$STA_SYMNAME(.NOUN_NODE[DBG$L_ADJECTIVE_PTR], MODNAMEPTR);
;  523     0653  4              DBG$PRINT(UPLIT BYTE(%ASCIC 'module !AC'), .MODNAMEPTR);
;  524     0654  4              DBG$NEWLINE();
;  525     0655  3              END;
;  526     0656  3
;  527     0657  3
;  528     0658  3          ! Call the routine which actually displays the source lines.
;  529     0659  3          !
;  530     0660  3          DBG$SRC_TYPE_LNUM_SOURCE(.NOUN_NODE[DBG$L_ADJECTIVE_PTR],
;  531     0661  3                                   .NOUN_NODE[DBG$L_NOUN_VALUE], 0,
;  532     0662  3                                   .NOUN_NODE[DBG$L_NOUN_VALUE2], 0,
;  533     0663  3                                   0, FALSE);
;  534     0664  3          END
;  535     0665  3
;  536     0666  2      UNTIL 0 EQL (NOUN_NODE = .NOUN_NODE[DBG$L_NOUN_LINK]);
;  537     0667  2
;  538     0668  2      RETURN STS$K_SUCCESS
;  539     0669  2
;  540     0670  1      END;


                                                     .PSECT  DBG$PLIT,NOWRT,  SHR,  PIC,0
43 41 21 20 65 6C 75 64 6F 6D 0A  0000A P.AAF:       .ASCII  <10>\module !AC\                                    ;

                                                     .PSECT  DBG$CODE,NOWRT,  SHR,  PIC,0
                                   0004 00000        .ENTRY  DBG$NEXECUTE_TYPE, Save R2                          ; 0582
                              5E     04   C2 00002    SUBL2   #4, SP                                              ;
                              50 04  AC   D0 00005    MOVL    VERB_NODE, R0                                       ; 0631
                              52 08  A0   D0 00009    MOVL    8(R0), NOUN_NODE                                    ;
                00000000G     00     00   FB 0000D 1$: CALLS   #0, DBG$FLUSHBUF                                   ; 0642
                         00000000G     00   D5 00014    TSTL    DBG$GL_SCREEN_SOURCE                                ; 0649
                                       22   12 0001A    BNEQ    2$                                                  ;
                                       5E   DD 0001C    PUSHL   SP                                                  ; 0652
                              04       A2   DD 0001E    PUSHL   4(NOUN_NODE)                                        ;
                00000000G     00       02   FB 00021    CALLS   #2, DBG$STA_SYMNAME                                 ;
                                       6E   DD 00028    PUSHL   MODNAMEPTR                                          ; 0653
                         00000000'     EF   9F 0002A    PUSHAB  P.AAF                                               ;
                00000000G     00       02   FB 00030    CALLS   #2, DBG$PRINT                                       ;
                00000000G     00       00   FB 00037    CALLS   #0, DBG$NEWLINE                                     ; 0654
```

```
                          7E  7C 0003E 2$:     CLRQ    -(SP)                                   ; 0660
                          7E  D4 00040        CLRL    -(SP)                                   ;
                       0C A2  DD 00042        PUSHL   12(NOUN_NODE)                           ; 0662
                          7E  D4 00045        CLRL    -(SP)                                   ; 0660
                          62  DD 00047        PUSHL   (NOUN_NODE)                             ; 0661
                       04 A2  DD 00049        PUSHL   4(NOUN_NODE)                            ; 0660
            00000000G  00     07  FB 0004C    CALLS   #7, DBG$SRC_TYPE_LNUM_SOURCE            ;
                       52  08 A2  D0 00053    MOVL    8(NOUN_NODE), NOUN_NODE                 ; 0666
                           B4  12 00057        BNEQ    1$                                      ;
                       50      01  D0 00059    MOVL    #1, R0                                  ; 0668
                               04 0005C        RET                                             ; 0670

; Routine Size:  93 bytes,    Routine Base:  DBG$CODE + 01C7
```

```
;  542     0671  1 GLOBAL ROUTINE DBG$NPARSE_EDIT (INPUT_DESC, VERB_NODE, MESSAGE_VECT) =
;  543     0672  1 !
;  544     0673  1 ! Functional Description
;  545     0674  1 !
;  546     0675  1 !       Parse routine for the EDIT command. EDIT is the DEBUG interface
;  547     0676  1 !       to the EDITH editor.
;  548     0677  1 !
;  549     0678  1 !       The routine DBG$NPARSE_CMD in DBGNPARSE has already recognized
;  550     0679  1 !       the "EDIT" verb and has called this routine to parse the
;  551     0680  1 !       rest of the command.
;  552     0681  1 !
;  553     0682  1 !       This routine takes a verb node for the EDIT verb, and a string
;  554     0683  1 !       descriptor for the remaining (unparsed) input.
;  555     0684  1 !
;  556     0685  1 !       The rest of the command execution tree is constructed.
;  557     0686  1 !
;  558     0687  1 ! Formal Parameters
;  559     0688  1 !
;  560     0689  1 !       INPUT_DESC      - A longword containing the address of the
;  561     0690  1 !                         command input descriptor.
;  562     0691  1 !
;  563     0692  1 !       VERB_NODE       - A longword containing the address of the verb node
;  564     0693  1 !
;  565     0694  1 !       MESSAGE_VECT    - The address of a longword to contain the address
;  566     0695  1 !                         of a standard message argument vector.
;  567     0696  1 !
;  568     0697  1 ! Implicit Inputs
;  569     0698  1 !
;  570     0699  1 !       None
;  571     0700  1 !
;  572     0701  1 ! Implicit Outputs
;  573     0702  1 !
;  574     0703  1 !       On success, the command execution tree is constructed.
;  575     0704  1 !       On failure, a message argument vector is constructed or obtained.
;  576     0705  1 !
;  577     0706  1 ! Routine value
;  578     0707  1 !
;  579     0708  1 !       STS$K_SUCCESS (1)       - Success. Command execution tree constructed.
;  580     0709  1 !       STS$K_SEVERE  (4)       - Failure. Error encountered. Message argument
;  581     0710  1 !                                 constructed and returned.
;  582     0711  1 !
;  583     0712  1 ! Side Effects
;  584     0713  1 !
;  585     0714  1 !       None
;  586     0715  1 !
;  587     0716  1
;  588     0717  2     BEGIN
;  589     0718  2
;  590     0719  2     MAP
;  591     0720  2         INPUT_DESC      : REF DBG$STG_DESC,
;  592     0721  2         VERB_NODE       : REF DBG$VERB_NODE;
;  593     0722  2
;  594     0723  2     BIND
;  595     0724  2         DBG$CS_EXIT             = UPLIT BYTE (4, 'EXIT'),
;  596     0725  2         DBG$CS_NOEXIT           = UPLIT BYTE (6, 'NOEXIT'),
;  597     0726  2         DBG$CS_BACKSLASH        = UPLIT BYTE (1, DBG$K_BACKSLASH),
;  598     0727  2         DBG$CS_CR               = UPLIT BYTE (1, DBG$K_CAR_RETURN),
```

```
;  599     0728  2          DBG$CS_SLASH            = UPLIT BYTE (1, DBG$K_SLASH);
;  600     0729  2
;  601     0730  2      LOCAL
;  602     0731  2          CHAR,
;  603     0732  2          DLINE_PTR: REF DBG$DLINE_ENTRY,
;  604     0733  2          DROW,
;  605     0734  2          LENGTH,
;  606     0735  2          LINENO,
;  607     0736  2          LINUM,
;  608     0737  2          MODRSTPTR,
;  609     0738  2          NAME_BUF: VECTOR [81, BYTE],
;  610     0739  2          NOUN_NODE: REF DBG$NOUN_NODE,
;  611     0740  2          RBEG,
;  612     0741  2          RLEN,
;  613     0742  2          SRC_DISP: REF DBG$DISP_ENTRY,
;  614     0743  2          STRING_PTR;
;  615     0744  2
;  616     0745  2
;  617     0746  2      ! Create and link a noun node.
;  618     0747  2      !
;  619     0748  2      NOUN_NODE = DBG$GET_TEMPMEM(DBG$K_NOUN_NODE_SIZE);
;  620     0749  2      VERB_NODE[DBG$L_VERB_OBJECT_PTR] = .NOUN_NODE;
;  621     0750  2
;  622     0751  2
;  623     0752  2      ! Pick up optional /EXIT or /NOEXIT. /NOEXIT is the default.
;  624     0753  2      !
;  625     0754  2      VERB_NODE[DBG$B_VERB_COMPOSITE] = EDIT_NOEXIT;
;  626     0755  2      WHILE DBG$NMATCH (.INPUT_DESC, DBG$CS_SLASH, 1) DO
;  627     0756  3          BEGIN
;  628     0757  3
;  629     0758  3          SELECTONE TRUE OF
;  630     0759  3              SET
;  631     0760  3
;  632     0761  3              [DBG$NMATCH (.INPUT_DESC, DBG$CS_EXIT, 1)]:
;  633     0762  3                  VERB_NODE[DBG$B_VERB_COMPOSITE] = EDIT_EXIT;
;  634     0763  3
;  635     0764  3              [DBG$NMATCH (.INPUT_DESC, DBG$CS_NOEXIT, 1)]:
;  636     0765  3                  VERB_NODE[DBG$B_VERB_COMPOSITE] = EDIT_NOEXIT;
;  637     0766  3
;  638     0767  3              [DBG$NMATCH (.INPUT_DESC, DBG$CS_CR, 1)]:
;  639     0768  3                  SIGNAL(DBG$_NEEDMORE);
;  640     0769  3
;  641     0770  3              [OTHERWISE]:
;  642     0771  3                  SIGNAL(DBG$_CMDSYNERR, 1, DBG$NNEXT_WORD(.INPUT_DESC));
;  643     0772  3
;  644     0773  3              TES;
;  645     0774  2          END;
;  646     0775  2
;  647     0776  2
;  648     0777  2      ! Initialize module and line number.
;  649     0778  2      !
;  650     0779  2      MODRSTPTR = 0;
;  651     0780  2      LINENO = 0;
;  652     0781  2
;  653     0782  2
;  654     0783  2      ! Check for EDIT <cr>
;  655     0784  2      !
```

```
;  656     0785  2     IF DBG$NMATCH(.INPUT_DESC, DBG$CS_CR, 1)
;  657     0786  2     THEN
;  658     0787  2         BEGIN
;  659     0788  2
;  660     0789  2
;  661     0790  2         ! Try to get default module and line number. If screen mode is set,
;  662     0791  2         ! then get this info from the source display on the screen. Otherwise,
;  663     0792  2         ! use defaulting rules based on the most recent TYPE command. If we
;  664     0793  2         ! are unable to default module name and line number then leave
;  665     0794  2         ! zeros in the variables to indicate this.
;  666     0795  2         !
;  667     0796  2         SRC_DISP = .DBG$GL_SCREEN_SOURCE;
;  668     0797  2         IF .SRC_DISP NEQ 0
;  669     0798  3         THEN
;  670     0799  4             BEGIN
;  671     0800  4             MODRSTPTR = .SRC_DISP[DBG$L_DISP_MODPTR];
;  672     0801  4
;  673     0802  4
;  674     0803  4             ! Do the computation to figure out what line number is
;  675     0804  4             ! the center of the source display.
;  676     0805  4             !
;  677     0806  4             RBEG = .SRC_DISP[DBG$W_DISP_RBEG];
;  678     0807  4             RLEN = .SRC_DISP[DBG$W_DISP_RLEN];
;  679     0808  4             DROW = .SRC_DISP[DBG$W_DISP_DROW];
;  680     0809  4             DLINE_PTR = .SRC_DISP[DBG$L_DISP_START_LINE_PTR];
;  681     0810  4             LINUM = .DLINE_PTR[DBG$L_DLINE_LINUM];
;  682     0811  4             LINENO = (.DROW-.RBEG) + .RLEN/2 + .LINUM;
;  683     0812  4             END
;  684     0813  4
;  685     0814  3         ELSE IF .DBG$SRC_NEXT_MODRSTPTR NEQ 0
;  686     0815  3         THEN
;  687     0816  4             BEGIN
;  688     0817  4             MODRSTPTR = .DBG$SRC_NEXT_MODRSTPTR;
;  689     0818  4             LINENO = .DBG$SRC_NEXT_LNUM - 1;
;  690     0819  3             END;
;  691     0820  3         END
;  692     0821  3
;  693     0822  3
;  694     0823  2     ELSE
;  695     0824  3         BEGIN
;  696     0825  3
;  697     0826  3
;  698     0827  3         ! Initialize the buffer which will hold the module name.
;  699     0828  3         ! Set up to scan the input.
;  700     0829  3         !
;  701     0830  3         name_buf[0] = 0;
;  702     0831  3         string_ptr = .input_desc[dsc$a_pointer];
;  703     0832  3         length = .input_desc[dsc$w_length];
;  704     0833  3
;  705     0834  3
;  706     0835  3         ! Read past leading blanks
;  707     0836  3         !
;  708     0837  3         WHILE .length GTR 0 DO
;  709     0838  4             BEGIN
;  710     0839  4             char = ch$rchar_a(string_ptr);
;  711     0840  4             length = .length - 1;
;  712     0841  4             IF .char NEQ dbg$k_blank
```

```
;  713          0842  4          THEN
;  714          0843  4              EXITLOOP;
;  715          0844  3          END;
;  716          0845  3
;  717          0846  3
;  718          0847  3      ! If the length reaches zero then it is an error.
;  719          0848  3      !
;  720          0849  3      IF .length EQL 0
;  721          0850  3      THEN
;  722          0851  3          $DBG_ERROR('DBGNTYPE\DBG$NPARSE_EDIT');
;  723          0852  3
;  724          0853  3
;  725          0854  3      ! Read until we reach a separating character.
;  726          0855  3      ! Place the characters into name_buf as we read them.
;  727          0856  3      !
;  728          0857  3      WHILE .length GTR 0  DO
;  729          0858  4          BEGIN
;  730          0859  4          IF .char EQL '\' OR .char EQL ' '
;  731          0860  4          THEN
;  732          0861  4              EXITLOOP;
;  733          0862  4          name_buf[0] = .name_buf[0] + 1;
;  734          0863  4          IF .name_buf[0] GTR 80
;  735          0864  4          THEN
;  736          0865  4              SIGNAL(dbg$_cmdsynerr, 1, dbg$nnext_word(.input_desc));
;  737          0866  4          name_buf[.name_buf[0]] = .char;
;  738          0867  4          char = ch$rchar_a(string_ptr);
;  739          0868  4          length = .length - 1;
;  740          0869  3          END;
;  741          0870  3
;  742          0871  3
;  743          0872  3      ! Backslash signifies that we just read a module name.
;  744          0873  3      !
;  745          0874  3      IF .char EQL '\'
;  746          0875  3      THEN
;  747          0876  4          BEGIN
;  748          0877  4
;  749          0878  4
;  750          0879  4          ! Update the input descriptor
;  751          0880  4          !
;  752          0881  4          input_desc[dsc$a_pointer] = .string_ptr;
;  753          0882  4          input_desc[dsc$w_length] = .length;
;  754          0883  4
;  755          0884  4
;  756          0885  4          ! Convert the name to an rst pointer.
;  757          0886  4          !
;  758          0887  4          modrstptr = dbg$sta_getsourcemod(name_buf);
;  759          0888  4
;  760          0889  4
;  761          0890  4          ! If the above routine returned zero then the user entered
;  762          0891  4          ! an invalid module.
;  763          0892  4          !
;  764          0893  4          IF .modrstptr EQL 0
;  765          0894  4          THEN
;  766          0895  4              SIGNAL(dbg$_nosuchmodu, 1, name_buf);
;  767          0896  4          END
;  768          0897  4
;  769          0898  3      ELSE
```

```
;  4                    BEGIN
;  4
;  4
;  4                    ! No module was specified, so try to fill
;  4                    ! in a module based on (1) module in current
;  4                    ! source display, or (2) module given by previous
;  4                    ! TYPE command, or (3) current scope
;  4                    !
;  4                    SRC_DISP = .DBG$GL_SCREEN_SOURCE;
;  4                    IF .SRC_DISP NEQ 0
;  4                    THEN
;  4                        MODRSTPTR = .SRC_DISP[DBG$L_DISP_MODPTR]
;  4
;  4                    ELSE IF .DBG$SRC_NEXT_MODRSTPTR NEQ 0
;  4                    THEN
;  4                        MODRSTPTR = .DBG$SRC_NEXT_MODRSTPTR
;  4
;  4                    ELSE
;  4                        modrstptr = dbg$sta_getsourcemod(0);
;  3                    END;
;  3
;  3
;  3                ! Read the line number.
;  3                !
;  3                IF NOT dbg$nsave_decimal_integer(.input_desc, lineno, .message_vect)
;  3                THEN
;  3                    RETURN sts$k_severe;
;  2                END;
;  2
;  2
;  2            ! Fill in the fields of the noun node with the module and
;  2            ! line number.
;  2            !
;  2            NOUN_NODE[DBG$L_ADJECTIVE_PTR] = .MODRSTPTR;
;  2            NOUN_NODE[DBG$L_NOUN_VALUE] = .LINENO;
;  2
;  2
;  2            ! We have finished parsing the EDIT command. Return success.
;  2            !
;  2            RETURN STS$K_SUCCESS;
;  1            END;


                                                            .PSECT  DBG$PLIT,NOWRT,  SHR,  PIC,0

                                        04  00015 P.AAG:    .BYTE   4                                           ;
                            54  49  58  45  00016           .ASCII  \EXIT\                                      ;
                                        06  0001A P.AAH:    .BYTE   6                                           ;
                    54  49  58  45  4F  4E  0001B           .ASCII  \NOEXIT\                                    ;
                                    5C  01  00021 P.AAI:    .BYTE   1, 92                                       ;
                                    0D  01  00023 P.AAJ:    .BYTE   1, 13                                       ;
                                    2F  01  00025 P.AAK:    .BYTE   1, 47                                       ;
4E  24  47  42  44  5C  45  50  59  54  4E  47  42  44  18  00027 P.AAL:    .ASCII  <24>\DBGNTYPE\<92>\DBG$NPARSE_EDIT\  ;
                54  49  44  45  5F  45  53  52  41  50  00036                                                   ;

                                                    DBG$CS_EXIT=            P.AAG
```

```
                        DBG$CS_NOEXIT=      P.AAH
                        DBG$CS_BACKSLASH=   P.AAI
                        DBG$CS_CR=          P.AAJ
                        DBG$CS_SLASH=       P.AAK

                                            .PSECT  DBG$CODE,NOWRT,  SHR,  PIC,0

                                    0FFC 00000      .ENTRY  DBG$NPARSE_EDIT, Save R2,R3,R4,R5,R6,R7,R8,-; 0671
                                                            R9,R10,R11
                    5B 00000000G    00   9E 00002      MOVAB   DBG$NMATCH, R11                          ;
                    5A 00000000'    EF   9E 00009      MOVAB   DBG$CS_CR, R10                           ;
                    5E        A8    AE   9E 00010      MOVAB   -88(SP), SP                              ;
                                    04   DD 00014      PUSHL   #4                                       ; 0748
          00000000G 00              01   FB 00016      CALLS   #1, DBG$GET_TEMPMEM                      ;
                    59              50   D0 0001D      MOVL    R0, NOUN_NODE                            ;
                    52        08    AC   D0 00020      MOVL    VERB_NODE, R2                            ; 0749
                 08 A2              59   D0 00024      MOVL    NOUN_NODE, 8(R2)                         ;
                 01 A2              02   90 00028      MOVB    #2, T(R2)                                ; 0754
                    54        04    AC   D0 0002C      MOVL    INPUT_DESC, R4                           ; 0755
                                    01   DD 00030 1$:   PUSHL   #1                                       ;
                              02    AA   9F 00032      PUSHAB  DBG$CS_SLASH                             ;
                                    54   DD 00035      PUSHL   R4                                       ;
                    6B              03   FB 00037      CALLS   #3, DBG$NMATCH                           ;
                    63              50   E9 0003A      BLBC    R0, 5$                                   ;
                                    01   DD 0003D      PUSHL   #1                                       ; 0761
                              F2    AA   9F 0003F      PUSHAB  DBG$CS_EXIT                              ;
                                    54   DD 00042      PUSHL   R4                                       ;
                    6B              03   FB 00044      CALLS   #3, DBG$NMATCH                           ;
                    01              50   D1 00047      CMPL    R0, #1                                   ;
                                    06   12 0004A      BNEQ    2$                                       ;
                 01 A2              01   90 0004C      MOVB    #1, 1(R2)                                ; 0762
                                    DE   11 00050      BRB     1$                                       ;
                                    01   DD 00052 2$:   PUSHL   #1                                       ; 0764
                           F7       AA   9F 00054      PUSHAB  DBG$CS_NOEXIT                            ;
                                    54   DD 00057      PUSHL   R4                                       ;
                    6B              03   FB 00059      CALLS   #3, DBG$NMATCH                           ;
                    01              50   D1 0005C      CMPL    R0, #1                                   ;
                                    06   12 0005F      BNEQ    3$                                       ;
                 01 A2              02   90 00061      MOVB    #2, 1(R2)                                ; 0765
                                    C9   11 00065      BRB     1$                                       ;
                                    01   DD 00067 3$:   PUSHL   #1                                       ; 0767
                         0410       8F   BB 00069      PUSHR   #^M<R4,R10>                              ;
                    6B              03   FB 0006D      CALLS   #3, DBG$NMATCH                           ;
                    01              50   D1 00070      CMPL    R0, #1                                   ;
                                    0F   12 00073      BNEQ    4$                                       ;
                       000280D0     8F   DD 00075      PUSHL   #164048                                  ; 0768
          00000000G 00              01   FB 0007B      CALLS   #1, LIB$SIGNAL                           ;
                                    AC   11 00082      BRB     1$                                       ;
                                    54   DD 00084 4$:   PUSHL   R4                                       ; 0771
          00000000G 00              01   FB 00086      CALLS   #1, DBG$NNEXT_WORD                       ;
                                    50   DD 0008D      PUSHL   R0                                       ;
                                    01   DD 0008F      PUSHL   #1                                       ;
                       00028EB8     8F   DD 00091      PUSHL   #167608                                  ;
          00000000G 00              03   FB 00097      CALLS   #3, LIB$SIGNAL                           ;
                                    90   11 0009E      BRB     1$                                       ; 0755
                                    56   D4 000A0 5$:   CLRL    MODRSTPTR                                ; 0779
```

```
                                    6E  D4 000A2          CLRL    LINENO                                  ; 0780
                                    01  DD 000A4          PUSHL   #1                                      ; 0785
                              0410  8F  BB 000A6          PUSHR   #^M<R4,R10>                             ;
                         6B         03  FB 000AA          CALLS   #3, DBG$NMATCH                          ;
                         47         50  E9 000AD          BLBC    R0, 8$                                  ;
                         52 00000000G 00  D0 000B0          MOVL    DBG$GL_SCREEN_SOURCE, SRC_DISP          ; 0796
                                    27  13 000B7          BEQL    6$                                      ; 0797
                         56      34 A2  D0 000B9          MOVL    52(SRC_DISP), MODRSTPTR                 ; 0800
                         55      10 A2  3C 000BD          MOVZWL  16(SRC_DISP), RBEG                      ; 0806
                         50      12 A2  3C 000C1          MOVZWL  18(SRC_DISP), RLEN                      ; 0807
                         51      18 A2  3C 000C5          MOVZWL  24(SRC_DISP), DROW                      ; 0808
                         53      20 A2  D0 000C9          MOVL    32(SRC_DISP), DLINE_PTR                 ; 0809
                         53      10 A3  D0 000CD          MOVL    16(DLINE_PTR), LINUM                    ; 0810
                         51         55  C2 000D1          SUBL2   RBEG, R1                                ; 0811
                         50         02  C6 000D4          DIVL2   #2, R0                                  ;
                         50         51  C0 000D7          ADDL2   R1, R0                                  ;
                   6E    50         53  C1 000DA          ADDL3   LINUM, R0, LINENO                       ;
                                    14  11 000DE          BRB     7$                                      ; 0797
                         50 00000000G 00  D0 000E0 6$:      MOVL    DBG$SRC_NEXT_MODRSTPTR, R0              ; 0814
                                    0B  13 000E7          BEQL    7$                                      ;
                         56         50  D0 000E9          MOVL    R0, MODRSTPTR                           ; 0817
                   6E 00000000G 00      01  C3 000EC          SUBL3   #1, DBG$SRC_NEXT_LNUM, LINENO           ; 0818
                                  00E0  31 000F4 7$:       BRW     18$                                     ; 0785
                                  04 AE 94 000F7 8$:       CLRB    NAME_BUF                                ; 0830
                         58      04 A4  D0 000FA          MOVL    4(R4), STRING_PTR                       ; 0831
                         57         64  3C 000FE          MOVZWL  (R4), LENGTH                            ; 0832
                                    57  D5 00101 9$:       TSTL    LENGTH                                  ; 0837
                                    0A  15 00103          BLEQ    10$                                     ;
                         55         88  9A 00105          MOVZBL  (STRING_PTR)+, CHAR                     ; 0839
                                    57  D7 00108          DECL    LENGTH                                  ; 0840
                         20         55  D1 0010A          CMPL    CHAR, #32                               ; 0841
                                    F2  13 0010D          BEQL    9$                                      ;
                                    57  D5 0010F 10$:      TSTL    LENGTH                                  ; 0849
                                    12  12 00111          BNEQ    11$                                     ;
                               04   AA  9F 00113          PUSHAB  P.AAL                                   ; 0851
                                    01  DD 00116          PUSHL   #1                                      ;
                         00028362   8F  DD 00118          PUSHL   #164706                                 ;
                   00000000G 00     03  FB 0011E          CALLS   #3, LIB$SIGNAL                          ;
                                    57  D5 00125 11$:      TSTL    LENGTH                                  ; 0857
                                    41  15 00127          BLEQ    13$                                     ;
                   0000005C 8F      55  D1 00129          CMPL    CHAR, #92                               ; 0859
                                    38  13 00130          BEQL    13$                                     ;
                         20         55  D1 00132          CMPL    CHAR, #32                               ;
                                    33  13 00135          BEQL    13$                                     ;
                               04   AE  96 00137          INCB    NAME_BUF                                ; 0862
                         53      04 AE  9A 0013A          MOVZBL  NAME_BUF, R3                            ; 0863
                      50 8F         53  91 0013E          CMPB    R3, #80                                 ;
                                    1A  1B 00142          BLEQU   12$                                     ;
                                    54  DD 00144          PUSHL   R4                                      ; 0865
                   00000000G 00     01  FB 00146          CALLS   #1, DBG$NNEXT_WORD                      ;
                                    50  DD 0014D          PUSHL   R0                                      ;
                                    01  DD 0014F          PUSHL   #1                                      ;
                         00028EB8   8F  DD 00151          PUSHL   #167608                                 ;
                   00000000G 00     03  FB 00157          CALLS   #3, LIB$SIGNAL                          ;
                         04 AE43    55  90 0015E 12$:      MOVB    CHAR, NAME_BUF[R3]                      ; 0866
                         55         88  9A 00163          MOVZBL  (STRING_PTR)+, CHAR                     ; 0867
                                    57  D7 00166          DECL    LENGTH                                  ; 0868
```

```
                                 BB  11 00168          BRB     11$                                             ; 0857
0000005C   8F                    55  D1 0016A 13$:     CMPL    CHAR, #92                                       ; 0874
                                 2A  12 00171          BNEQ    14$                                             ;
      04   A4                    58  D0 00173          MOVL    STRING_PTR, 4(R4)                               ; 0881
           64                    57  B0 00177          MOVW    LENGTH, (R4)                                    ; 0882
                      04         AE  9F 0017A          PUSHAB  NAME_BUF                                        ; 0887
00000000G  00                    01  FB 0017D          CALLS   #1, DBG$STA_GETSOURCEMOD                        ;
           56                    50  D0 00184          MOVL    R0, MODRSTPTR                                   ;
                                 38  12 00187          BNEQ    17$                                             ; 0893
                      04         AE  9F 00189          PUSHAB  NAME_BUF                                        ; 0895
                      01         DD  0018C             PUSHL   #1                                              ;
                 000281E8        8F  DD 0018E          PUSHL   #164328                                         ;
00000000G  00                    03  FB 00194          CALLS   #3, LIB$SIGNAL                                  ;
                                 24  11 0019B          BRB     17$                                             ; 0874
           52 00000000G          00  D0 0019D 14$:     MOVL    DBG$GL_SCREEN_SOURCE, SRC_DISP                  ; 0907
                                 06  13 001A4          BEQL    15$                                             ; 0908
           56         34         A2  D0 001A6          MOVL    52(SRC_DISP), MODRSTPTR                         ; 0910
                                 15  11 001AA          BRB     17$                                             ;
           50 00000000G          00  D0 001AC 15$:     MOVL    DBG$SRC_NEXT_MODRSTPTR, R0                      ; 0912
                                 09  12 001B3          BNEQ    16$                                             ;
                                 7E  D4 001B5          CLRL    -(SP)                                           ; 0917
00000000G  00                    01  FB 001B7          CALLS   #1, DBG$STA_GETSOURCEMOD                        ;
           56                    50  D0 001BE 16$:     MOVL    R0, MODRSTPTR                                   ;
                      0C         AC  DD 001C1 17$:     PUSHL   MESSAGE_VECT                                    ; 0923
                      04         AE  9F 001C4          PUSHAB  LINENO                                          ;
                                 54  DD 001C7          PUSHL   R4                                              ;
00000000G  00                    03  FB 001C9          CALLS   #3, DBG$NSAVE_DECIMAL_INTEGER                   ;
           04                    50  E8 001D0          BLBS    R0, 18$                                         ;
           50                    04  D0 001D3          MOVL    #4, R0                                          ; 0925
                                 04  001D6             RET                                                     ;
      04   A9                    56  D0 001D7 18$:     MOVL    MODRSTPTR, 4(NOUN_NODE)                         ; 0932
           69                    6E  D0 001DB          MOVL    LINENO, (NOUN_NODE)                             ; 0933
           50                    01  D0 001DE          MOVL    #1, R0                                          ; 0938
                                 04  001E1             RET                                                     ; 0939

; Routine Size:  482 bytes,    Routine Base:  DBG$CODE + 0224
```

```
;  812     0940  1 GLOBAL ROUTINE DBG$NEXECUTE_EDIT (VERB_NODE, MESSAGE_VECT) =
;  813     0941  1 !
;  814     0942  1 ! Functional Description
;  815     0943  1 !
;  816     0944  1 !       This routine performs the action associated with the EDIT
;  817     0945  1 !       command. EDIT is the interface to the EDITH editor.
;  818     0946  1 !
;  819     0947  1 !       This routine constructs a DCL command line which invokes EDITH.
;  820     0948  1 !       Either LIB$SPAWN or LIB$DO_COMMAND is called to execute the
;  821     0949  1 !       command line.
;  822     0950  1 !
;  823     0951  1 ! Formal Parameters
;  824     0952  1 !
;  825     0953  1 !       VERB_NODE       - A longword containing the address of the
;  826     0954  1 !                         verb node of the command execution tree.
;  827     0955  1 !                         This tree has already been built by the
;  828     0956  1 !                         DBG$NPARSE_EDIT routine.
;  829     0957  1 !
;  830     0958  1 !       MESSAGE_VECT    - The address of a longword to contain the
;  831     0959  1 !                         address of an error message vector
;  832     0960  1 !
;  833     0961  1 ! Implicit Inputs
;  834     0962  1 !
;  835     0963  1 !       The command tree contains a verb node and a noun node.
;  836     0964  1 !
;  837     0965  1 ! Implicit Outputs
;  838     0966  1 !
;  839     0967  1 !       This routine constructs a command line and calls either
;  840     0968  1 !       LIB$SPAWN or LIB$DO_COMMAND to execute the command line.
;  841     0969  1 !
;  842     0970  1 ! Routine Value
;  843     0971  1 !
;  844     0972  1 !       A completion code.
;  845     0973  1 !
;  846     0974  1 ! Completion Codes
;  847     0975  1 !
;  848     0976  1 !       STS$K_SUCCESS (1)       - Success. Command executed
;  849     0977  1 !       STS$K_SEVERE (4)        - Failure. The command could not be
;  850     0978  1 !                                 executed. An error message is constructed.
;  851     0979  1 !
;  852     0980  1 ! Side Effects
;  853     0981  1 !
;  854     0982  1 !       If the EDIT/EXIT form of the command was given, the debugging
;  855     0983  1 !       session is terminated.
;  856     0984  1 !
;  857     0985  1
;  858     0986  2     BEGIN
;  859     0987  2
;  860     0988  2     MAP
;  861     0989  2         VERB_NODE : REF DBG$VERB_NODE;
;  862     0990  2
;  863     0991  2     LOCAL
;  864     0992  2         COMMAND_LINE: VECTOR[132,BYTE], ! Command line to invoke EDITH
;  865     0993  2         CTRLBUF,
;  866     0994  2         CTRLDESC: DBG$STG_DESC,
;  867     0995  2         FABPTR: REF $FAB_DECL,          ! FAB for searching for newer file ver.
;  868     0996  2         FAOBUF: VECTOR[80, BYTE],
```

```
;  869     0997  2          FAODESC: DBG$STG_DESC,
;  870     0998  2          FAOLEN,
;  871     0999  2          LENGTH,                         ! Length of command line
;  872     1000  2          LINENO,                         ! Line number given in EDIT command
;  873     1001  2          MODRSTPTR,                      ! Module given in EDIT command
;  874     1002  2          NAMESA,                         ! Ptr to filename buff for $PARSE/$SEARCH
;  875     1003  2          NAMPTR: REF $NAM_DECL,
;  876     1004  2          NAMRSA,                         ! Ptr to filename buff for $PARSE/$SEARCH
;  877     1005  2          NOUN_NODE : REF DBG$NOUN_NODE,  ! Noun node in execution tree
;  878     1006  2          OLDNAMPTR : REF $NAM_DECL,      ! NAM block of current source file
;  879     1007  2          PTR,
;  880     1008  2          RECNO,                          ! Record number in file
;  881     1009  2          STATUS,                         ! Return status temp.
;  882     1010  2          STG_DESC: DBG$STG_DESC;         ! String descriptor
;  883     1011  2
;  884     1012  2
;  885     1013  2      ! Recover the noun node. From this, obtain the module RST pointer and
;  886     1014  2      ! line number.
;  887     1015  2      !
;  888     1016  2      NOUN_NODE = .VERB_NODE[DBG$L_VERB_OBJECT_PTR];
;  889     1017  2      MODRSTPTR = .NOUN_NODE[DBG$L_ADJECTIVE_PTR];
;  890     1018  2      LINENO = .NOUN_NODE[DBG$L_NOUN_VALUE];
;  891     1019  2
;  892     1020  2      ! Build the command line and the string descriptor
;  893     1021  2      ! with which we will invoke EDITH. We start by initializing
;  894     1022  2      ! the current length of the command line and a pointer into
;  895     1023  2      ! the command line.
;  896     1024  2      !
;  897     1025  2      LENGTH = 0;
;  898     1026  2      PTR = COMMAND_LINE;
;  899     1027  2
;  900     1028  2      ! Append the LSEDIT command to the line.
;  901     1029  2      ! Note - the command name may change. We leave a few extra
;  902     1030  2      ! spaces so we can patch in whatever command name they choose,
;  903     1031  2      ! up to 15 characters.
;  904     1032  2      ! Additional note - it now appears that the command name will
;  905     1033  2      ! be EDIT/LSE. However, this change will not happen until at least
;  906     1034  2      ! version 4.1 of VMS.
;  907     1035  2      !
;  908     1036  2      LENGTH = .LENGTH + 15;
;  909     1037  2      PTR = CH$MOVE(15, UPLIT BYTE(%ASCII 'LSEDIT         '), .PTR);
;  910     1038  2
;  911     1039  2      ! The following block is only executed when a source file is present.
;  912     1040  2      ! Translate the module and line number into a NAM block pointer and record
;  913     1041  2      ! number within the file. This is done by calling a routine in DBGSOURCE.
;  914     1042  2      ! Also, do the checking for a more recent version of the source file,
;  915     1043  2      ! and issue any appropriate messages.  After this is done, add the
;  916     1044  2      ! file name and the appropriate /START_POSITION qualifier.
;  917     1045  2      !
;  918     1046  2      IF .MODRSTPTR NEQ 0
;  919     1047  2      THEN
;  920     1048  3          BEGIN
;  921     1049  3          DBG$SRC_LINE_TO_REC(.MODRSTPTR, .LINENO, OLDNAMPTR, RECNO);
```

```
;  926          1054  3
;  927          1055  3
;  928          1056  3          ! Get memory for the FAB and NAM blocks.
;  929          1057  3          !
;  930          1058  3          FABPTR = DBG$GET_TEMPMEM((FAB$C_BLN+3)/4);
;  931          1059  3          NAMPTR = DBG$GET_TEMPMEM((NAM$C_BLN+3)/4);
;  932          1060  3          NAMESA = DBG$GET_TEMPMEM((NAM$C_MAXRSS+3)/4);
;  933          1061  3          NAMRSA = DBG$GET_TEMPMEM((NAM$C_MAXRSS+3)/4);
;  934          1062  3
;  935          1063  3
;  936          1064  3          ! Initialize the FAB and NAM blocks for $PARSE and $SEARCH.
;  937          1065  3          !
;  938        P 1066  3          $FAB_INIT(FAB = .FABPTR,
;  939        P 1067  3                    NAM = .NAMPTR,
;  940        P 1068  3                    FNA = .OLDNAMPTR[NAM$L_ESA],
;  941          1069  3                    FNS = (.OLDNAMPTR[NAM$B_ESL] - .OLDNAMPTR[NAM$B_VER]) );
;  942          1070  3
;  943        P 1071  3          $NAM_INIT(NAM = .NAMPTR,
;  944        P 1072  3                    ESS = NAM$C_MAXRSS,
;  945        P 1073  3                    ESA = .NAMESA,
;  946        P 1074  3                    RSS = NAM$C_MAXRSS,
;  947          1075  3                    RSA = .NAMRSA );
;  948          1076  3
;  949          1077  3          ! Parse the spec, and then search for the most recent version.
;  950          1078  3          ! The PARSE should never fail, since we are using a string from a
;  951          1079  3          ! previous file OPEN.  However, under very unusual circumstances,
;  952          1080  3          ! the SEARCH mail fail (if a user spawned out from DEBUG and renamed
;  953          1081  3          ! or deleted the file, for example).
;  954          1082  3          !
;  955          1083  3          STATUS = $PARSE(FAB = .FABPTR);
;  956          1084  3          IF NOT .STATUS
;  957          1085  3          THEN
;  958          1086  3              $DBG_ERROR('DBGNTYPE\DBG$NEXECUTE_EDIT - RMS Parse error',.STATUS);
;  959          1087  3
;  960          1088  3          STATUS = $SEARCH(FAB = .FABPTR);
;  961          1089  3          IF NOT .STATUS
;  962          1090  3          THEN
;  963          1091  3              SIGNAL(DBG$_EDITNOFILE,2,
;  964          1092  3                          .OLDNAMPTR[NAM$B_ESL],.OLDNAMPTR[NAM$L_ESA],.STATUS)
;  965          1093  3          ELSE
;  966          1094  3
;  967          1095  4              BEGIN
;  968          1096  4              ! Issue a message about the file we are editing.
;  969          1097  4              !
;  970          1098  4              SIGNAL(DBG$_EDITFILE,2,.NAMPTR[NAM$B_RSL],.NAMPTR[NAM$L_RSA]);
;  971          1099  4
;  972          1100  4
;  973          1101  4              ! Now compare the result of the SEARCH with the old filename, and
;  974          1102  4              ! issue a pair of messages if the version differs.
;  975          1103  4              !
;  976          1104  4              IF CH$NEQ(.OLDNAMPTR[NAM$B_ESL], .OLDNAMPTR[NAM$L_ESA],
;  977          1105  4                        .NAMPTR[NAM$B_RS[],    .NAMPTR[NAM$L_RSA] )
;  978          1106  4              THEN
;  979          1107  5                  BEGIN
;  980          1108  5                  SIGNAL(DBG$_EDITREVVER);
;  981          1109  5                  SIGNAL(DBG$_EDITDISVER,2,.OLDNAMPTR[NAM$B_ESL],.OLDNAMPTR[NAM$L_ESA]);
;  982          1110  4                  END;
```

```
;  983          1111  4
;  984          1112  4
;  985          1113  4              ! Append the start position qualifier. Again, leave some
;  986          1114  4              ! extra space for flexibiity.
;  987          1115  4              !
;  988          1116  4              LENGTH = .LENGTH + 21;
;  989          1117  4              PTR = CH$MOVE(21, UPLIT BYTE(%ASCII '/START_POSITION   =  '), .PTR);
;  990          1118  4
;  991          1119  4
;  992          1120  4              ! We need to call $FAO
;  993          1121  4              ! to translate the record number that we have into a
;  994          1122  4              ! string.
;  995          1123  4              !
;  996          1124  4              CTRLBUF = UPLIT BYTE(%ASCII '(!UL,1) ');
;  997          1125  4              CTRLDESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  998          1126  4              CTRLDESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  999          1127  4              CTRLDESC[DSC$W_LENGTH] = 8;
; 1000          1128  4              CTRLDESC[DSC$A_POINTER] = .CTRLBUF;
; 1001          1129  4              FAOLEN = 0;
; 1002          1130  4              FAODESC[DSC$B_CLASS] = DSC$K_CLASS_S;
; 1003          1131  4              FAODESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 1004          1132  4              FAODESC[DSC$W_LENGTH] = 80;
; 1005          1133  4              FAODESC[DSC$A_POINTER] = FAOBUF;
; 1006          1134  4              IF NOT SYS$FAO (CTRLDESC, FAOLEN, FAODESC, .RECNO)
; 1007          1135  4              THEN
; 1008          1136  4                  $DBG_ERROR('DBGNTYPE\DBG$NEXECUTE_EDIT $FAO failed');
; 1009          1137  4              LENGTH = .LENGTH + .FAOLEN;
; 1010          1138  4              IF .LENGTH GTR 132
; 1011          1139  4              THEN
; 1012          1140  4                  $DBG_ERROR('EDIT command: Command line buffer has overflowed');
; 1013          1141  4              PTR = CH$MOVE(.FAOLEN, FAOBUF, .PTR);
; 1014          1142  4
; 1015          1143  4
; 1016          1144  4                      ! Append the file name to the command line.
; 1017          1145  4                      !
; 1018          1146  4                      LENGTH = .LENGTH + .NAMPTR[NAM$B_RSL];
; 1019          1147  4                      IF .LENGTH GTR 132
; 1020          1148  4                      THEN
; 1021          1149  4                          $DBG_ERROR('EDIT command: Command line buffer has overflowed');
; 1022          1150  4                      PTR = CH$MOVE(.NAMPTR[NAM$B_RSL], .NAMPTR[NAM$L_RSA], .PTR);
; 1023          1151  4                      END
; 1024          1152  4
; 1025          1153  3                  END
; 1026          1154  3
; 1027          1155  2          ELSE
; 1028          1156  2              SIGNAL(DBG$_EDITNOFILE);
; 1029          1157  2
; 1030          1158  2
; 1031          1159  2          ! Now build a string descriptor with the command line.
; 1032          1160  2          !
; 1033          1161  2          STG_DESC[DSC$B_CLASS] = DSC$K_CLASS_S;
; 1034          1162  2          STG_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 1035          1163  2          STG_DESC[DSC$W_LENGTH] = .LENGTH;
; 1036          1164  2          STG_DESC[DSC$A_POINTER] = COMMAND_LINE;
; 1037          1165  2
; 1038          1166  2
; 1039          1167  2          ! Developer bit 1 causes us to print the result but not
```

```
;  1040          1168  2     ! do the action - (for testing purposes).
;  1041          1169  2     !
;  1042          1170  2     IF .DBG$GL_DEVELOPER[1]
;  1043          1171  2     THEN
;  1044          1172  3         BEGIN
;  1045          1173  3         DBG$PRINT(UPLIT BYTE(%ASCIC '!AS'), STG_DESC);
;  1046          1174  3         DBG$NEWLINE();
;  1047          1175  3         END
;  1048          1176  3
;  1049          1177  2     ELSE
;  1050          1178  3         BEGIN
;  1051          1179  3
;  1052          1180  3         ! Set the screen back to normal
;  1053          1181  3         !
;  1054          1182  3         DBG$SCR_SCREEN_NORMAL();
;  1055          1183  3
;  1056          1184  3         ! Invoke EDITH with either LIB$DO_COMMAND or LIB$SPAWN (depending
;  1057          1185  3         ! on whether the /EXIT form of the command was used).
;  1058          1186  3         !
;  1059          1187  3         SELECTONE .VERB_NODE[DBG$B_VERB_COMPOSITE] OF
;  1060          1188  4             SET
;  1061          1189  4
;  1062          1190  4             [EDIT_EXIT]:
;  1063          1191  4                 BEGIN
;  1064          1192  4                 DBG$GV_CONTROL[DBG$V_CONTROL_EXIT] = TRUE;
;  1065          1193  4                 LIB$DO_COMMAND(STG_DESC);
;  1066          1194  4                 END;
;  1067          1195  4
;  1068          1196  3             [EDIT_NOEXIT]:
;  1069          1197  4                 BEGIN
;  1070          1198  4                 LOCAL
;  1071          1199  4                     X;
;  1072          1200  4                 LIB$SPAWN(STG_DESC);
;  1073          1201  4                 IF .DBG$GB_KEYPAD_INPUT
;  1074          1202  4                     THEN X = 1
;  1075          1203  4                     ELSE X = 0;
;  1076          1204  4                 SMG$SET_KEYPAD_MODE(DBG$GL_KEYBOARD_ID, X);
;  1077          1205  4                 END;
;  1078          1206  4
;  1079          1207  4             [OTHERWISE]:
;  1080          1208  4                 $DBG_ERROR('DBGNTYPE\DBG$NEXECUTE_EDIT');
;  1081          1209  4             TES;
;  1082          1210  3         END;
;  1083          1211  3
;  1084          1212  3
;  1085          1213  3     ! We are all done. Return success.
;  1086          1214  3     !
;  1087          1215  3     RETURN STS$K_SUCCESS
;  1088          1216  2     END;


                                                                .PSECT  DBG$PLIT,NOWRT,  SHR,  PIC,0

20  20  20  20  20  20  20  20  20  54  49  44  45  53  4C  00040 P.AAM:  .ASCII  \LSEDIT          \
4E  24  47  42  44  5C  45  50  59  54  4E  47  42  44  2C  0004F P.AAN:  .ASCII  \,DBGNTYPE\<92>\DBG$NEXECUTE_EDIT - RMS \
20  2D  20  54  49  44  45  5F  45  54  55  43  45  58  45  0005E
```

```
                                    20  53  4D  52  0006D                                                        ;
                72  6F  72  72  65  20  65  73  72  61  50  00071            .ASCII  \Parse error\               ;
4E  4F  49  54  49  53  4F  50  5F  54  52  41  54  53  2F  0007C P.AAO:     .ASCII  \/START_POSITION   =  \     ;
                                20  20  3D  20  20  20  0008B                                                    ;
                        20  29  31  2C  4C  55  21  28  00091 P.AAP:     .ASCII  \(!UL,1) \                      ;
4E  24  47  42  44  5C  45  50  59  54  4E  47  42  44  26  00099 P.AAQ:     .ASCII  \&DBGNTYPE\<92>\DBG$NEXECUTE_EDIT $FAO f\ ;
46  24  20  54  49  44  45  5F  45  54  55  43  45  58  45  000A8                                                ;
                                        66  20  4F  41  000B7                                                    ;
                                64  65  6C  69  61  000BB            .ASCII  \ailed\                             ;
20  3A  64  6E  61  6D  6D  6F  63  20  54  49  44  45  30  000C0 P.AAR:     .ASCII  \0EDIT command: Command line buffer has o\ ;
75  62  20  65  6E  69  6C  20  64  6E  61  6D  6D  6F  43  000CF                                                ;
                    6F  20  73  61  68  20  72  65  66  66  000DE                                                ;
                        64  65  77  6F  6C  66  72  65  76  000E8            .ASCII  \verflowed\                 ;
20  3A  64  6E  61  6D  6D  6F  63  20  54  49  44  45  30  000F1 P.AAS:     .ASCII  \0EDIT command: Command line buffer has o\ ;
75  62  20  65  6E  69  6C  20  64  6E  61  6D  6D  6F  43  00100                                                ;
                    6F  20  73  61  68  20  72  65  66  66  0010F                                                ;
                        64  65  77  6F  6C  66  72  65  76  00119            .ASCII  \verflowed\                 ;
                                        53  41  21  03  00122 P.AAT:     .ASCII  <3>\!AS\                        ;
4E  24  47  42  44  5C  45  50  59  54  4E  47  42  44  1A  00126 P.AAU:     .ASCII  <26>\DBGNTYPE\<92>\DBG$NEXECUTE_EDIT\ ;
            54  49  44  45  5F  45  54  55  43  45  58  45  00135                                                ;

                                                                 .EXTRN  SYS$PARSE, SYS$SEARCH

                                                                 .PSECT  DBG$CODE,NOWRT,  SHR,  PIC,0

                                            0FFC 00000           .ENTRY  DBG$NEXECUTE_EDIT, Save R2,R3,R4,R5,R6,R7,- ; 0940
                                                                         R8,R9,R10,R11                      ;
                          5E      FEF0  CE  9E 00002             MOVAB   -272(SP), SP                       ;
                          5B        04  AC  D0 00007             MOVL    VERB_NODE, R11                     ; 1017
                          50        08  AB  D0 0000B             MOVL    8(R11), NOUN_NODE                  ;
                          56        04  A0  D0 0000F             MOVL    4(NOUN_NODE), MODRSTPTR            ; 1018
                          57            60  D0 00013             MOVL    (NOUN_NODE), LINENO                ; 1019
                                        59  D4 00016             CLRL    LENGTH                             ; 1027
                          5A      FF7C  CD  9E 00018             MOVAB   COMMAND_LINE, PTR                  ; 1028
                          59            0F  C0 0001D             ADDL2   #15, LENGTH                        ; 1039
            6A 00000000'  EF            0F  28 00020             MOVC3   #15, P.AAM, (PTR)                  ; 1040
                          5A            53  D0 00028             MOVL    R3, PTR                            ;
                                        56  D5 0002B             TSTL    MODRSTPTR                          ; 1050
                                        03  12 0002D             BNEQ    1$                                 ;
                                      01E4  31 0002F             BRW     8$                                 ;
                          08            AE  9F 00032 1$:         PUSHAB  RECNO                              ; 1053
                          10            AE  9F 00035             PUSHAB  OLDNAMPTR                          ;
                          7E            56  7D 00038             MOVQ    MODRSTPTR, -(SP)                   ;
               00000000G  00            04  FB 0003B             CALLS   #4, DBG$SRC_LINE_TO_REC            ;
                                        14  DD 00042             PUSHL   #20                                ; 1058
               00000000G  00            01  FB 00044             CALLS   #1, DBG$GET_TEMPMEM                ;
                          58            50  D0 0004B             MOVL    R0, FABPTR                         ;
                                        18  DD 0004E             PUSHL   #24                                ; 1059
               00000000G  00            01  FB 00050             CALLS   #1, DBG$GET_TEMPMEM                ;
                          57            50  D0 00057             MOVL    R0, NAMPTR                         ;
                          7E        40  8F  9A 0005A             MOVZBL  #64, -(SP)                         ; 1060
               00000000G  00            01  FB 0005E             CALLS   #1, DBG$GET_TEMPMEM                ;
                        04  AE          50  D0 00065             MOVL    R0, NAMESA                         ;
                          7E        40  8F  9A 00069             MOVZBL  #64, -(SP)                         ; 1061
               00000000G  00            01  FB 0006D             CALLS   #1, DBG$GET_TEMPMEM                ;
                          6E            50  D0 00074             MOVL    R0, NAMRSA                         ;
    0050   8F           00  6E          00  2C 00077             MOVC5   #0, (SP), #0, #80, (FABPTR)        ; 1069
```

```
                                           68       0007E
                          68      5003     8F   B0  0007F         MOVW    #20483, (FABPTR)                          ;
                     16   A8               02   90  00084         MOVB    #2, 22(FABPTR)                            ;
                     1F   A8               02   90  00088         MOVB    #2, 31(FABPTR)                            ;
                     28   A8               57   D0  0008C         MOVL    NAMPTR, 40(FABPTR)                        ;
                     56              0C    AE   D0  00090         MOVL    OLDNAMPTR, R6                             ;
                     2C   A8         0C    A6   D0  00094         MOVL    12(R6), 44(FABPTR)                        ;
  0060    8F     34   A8   0B         3D    A6   83  00099         SUBB3   61(R6), 11(R6), 52(FABPTR)                ;
                      00              6E   00   2C  000A0         MOVC5   #0, (SP), #0, #96, (NAMPTR)               ; 1075
                                           67       000A7
                          67      6002     8F   B0  000A8         MOVW    #24578, (NAMPTR)                          ;
                     02   A7               01   8E  000AD         MNEGB   #1, 2(NAMPTR)                             ;
                     04   A7               6E   D0  000B1         MOVL    NAMRSA, 4(NAMPTR)                         ;
                     0A   A7               01   8E  000B5         MNEGB   #1, 10(NAMPTR)                            ;
                     0C   A7         04    AE   D0  000B9         MOVL    NAMESA, 12(NAMPTR)                        ;
                                           58   DD  000BE         PUSHL   FABPTR                                    ; 1083
              00000000G   00               01   FB  000C0         CALLS   #1, SYS$PARSE                             ;
                          52               50   D0  000C7         MOVL    R0, STATUS                                ;
                          17               52   E8  000CA         BLBS    STATUS, 2$                                ; 1084
                                           52   DD  000CD         PUSHL   STATUS                                    ; 1086
                               00000000'   EF   9F  000CF         PUSHAB  P.AAN                                     ;
                                           01   DD  000D5         PUSHL   #1                                        ;
                               00028362    8F   DD  000D7         PUSHL   #164706                                   ;
              00000000G   00               04   FB  000DD         CALLS   #4, LIB$SIGNAL                            ;
                                           58   DD  000E4  2$:    PUSHL   FABPTR                                    ; 1088
              00000000G   00               01   FB  000E6         CALLS   #1, SYS$SEARCH                            ;
                          52               50   D0  000ED         MOVL    R0, STATUS                                ;
                          1B               52   E8  000F0         BLBS    STATUS, 3$                                ; 1089
                                           52   DD  000F3         PUSHL   STATUS                                    ; 1092
                                     0C    A6   DD  000F5         PUSHL   12(R6)                                    ;
                          7E         0B    A6   9A  000F8         MOVZBL  11(R6), -(SP)                             ;
                                           02   DD  000FC         PUSHL   #2                                        ; 1091
                               0002900B    8F   DD  000FE         PUSHL   #167947                                   ;
              00000000G   00               05   FB  00104         CALLS   #5, LIB$SIGNAL                            ;
                                         0115   31  0010B         BRW     9$                                        ;
                                     04    A7   DD  0010E  3$:    PUSHL   4(NAMPTR)                                 ; 1098
                          7E         03    A7   9A  00111         MOVZBL  3(NAMPTR), -(SP)                          ;
                                           02   DD  00115         PUSHL   #2                                        ;
                               00028FEB    8F   DD  00117         PUSHL   #167915                                   ;
              00000000G   00               04   FB  0011D         CALLS   #4, LIB$SIGNAL                            ;
                          51         0B    A6   9A  00124         MOVZBL  11(R6), R1                                ; 1104
                          50         03    A7   9A  00128         MOVZBL  3(NAMPTR), R0                             ; 1105
        50            00         0C   B6   51   2D  0012C         CMPC5   R1, @12(R6), #0, R0, @4(NAMPTR)           ; 1104
                                     04    B7       00132
                                           23   13  00134         BEQL    4$                                        ;
                               00028FF3    8F   DD  00136         PUSHL   #167923                                   ; 1108
              00000000G   00               01   FB  0013C         CALLS   #1, LIB$SIGNAL                            ;
                                     0C    A6   DD  00143         PUSHL   12(R6)                                    ; 1109
                          7E         0B    A6   9A  00146         MOVZBL  11(R6), -(SP)                             ;
                                           02   DD  0014A         PUSHL   #2                                        ;
                               00028FFB    8F   DD  0014C         PUSHL   #167931                                   ;
              00000000G   00               04   FB  00152         CALLS   #4, LIB$SIGNAL                            ;
                          59               15   C0  00159  4$:    ADDL2   #21, LENGTH                               ; 1116
              6A 00000000'  EF             15   28  0015C         MOVC3   #21, P.AAO, (PTR)                         ; 1117
                          5A               53   D0  00164         MOVL    R3, PTR                                   ;
                          50  00000000'    EF   9E  00167         MOVAB   P.AAP, CTRLBUF                            ; 1124
                  0080    CE  010E0008     8F   D0  0016E         MOVL    #17694728, CTRLDESC                       ; 1127
```

```
                0084   CE                50  D0 00177          MOVL    CTRLBUF, CTRLDESC+4                     ; 1128
                     10   AE              D4  00017C          CLRL    FAOLEN                                  ; 1129
                     24   AE 010E0050     8F  D0 0017F        MOVL    #17694800, FAODESC                      ; 1132
                     28   AE          30  AE  9E 00187        MOVAB   FAOBUF, FAODESC+4                       ; 1133
                                     08  AE  DD 0018C        PUSHL   RECNO                                   ; 1134
                                     28  AE  9F 0018F        PUSHAB  FAODESC
                                     18  AE  9F 00192        PUSHAB  FAOLEN
                                   008C  CE  9F 00195        PUSHAB  CTRLDESC
          00000000G   00                  04  FB 00199        CALLS   #4, SYS$FAO
                      15                  50  E8 001A0        BLBS    R0, 5$
                              00000000'   EF  9F 001A3        PUSHAB  P.AAQ                                   ; 1136
                                          01  DD 001A9        PUSHL   #1
                              00028362    8F  DD 001AB        PUSHL   #164706
          00000000G   00                  03  FB 001B1        CALLS   #3, LIB$SIGNAL
                      59          10  AE  C0  001B8 5$:       ADDL2   FAOLEN, LENGTH                          ; 1137
          00000084    8F                  59  D1 001BC        CMPL    LENGTH, #132                            ; 1138
                                          15  15 001C3        BLEQ    6$
                              00000000'   EF  9F 001C5        PUSHAB  P.AAR                                   ; 1140
                                          01  DD 001CB        PUSHL   #1
                              00028362    8F  DD 001CD        PUSHL   #164706
          00000000G   00                  03  FB 001D3        CALLS   #3, LIB$SIGNAL
    6A          30  AE          10  AE  28  001DA 6$:       MOVC3   FAOLEN, FAOBUF, (PTR)                   ; 1141
                    5A                  53  D0 001E0        MOVL    R3, PTR
                    50          03  A7  9A  001E3        MOVZBL  3(NAMPTR), R0                           ; 1146
                    59                  50  C0 001E7        ADDL2   R0, LENGTH
          00000084    8F                  59  D1 001EA        CMPL    LENGTH, #132                            ; 1147
                                          15  15 001F1        BLEQ    7$
                              00000000'   EF  9F 001F3        PUSHAB  P.AAS                                   ; 1149
                                          01  DD 001F9        PUSHL   #1
                              00028362    8F  DD 001FB        PUSHL   #164706
          00000000G   00                  03  FB 00201        CALLS   #3, LIB$SIGNAL
                      50          03  A7  9A  00208 7$:       MOVZBL  3(NAMPTR), R0                           ; 1150
    6A          04  B7                  50  28  0020C        MOVC3   R0, @4(NAMPTR), (PTR)
                    5A                  53  D0 00211        MOVL    R3, PTR
                                        0D  11 00214        BRB     9$                                      ; 1089
                              0002900B    8F  DD 00216 8$:       PUSHL   #167947                         ; 1156
          00000000G   00                  01  FB 0021C        CALLS   #1, LIB$SIGNAL
                  1A  AE          010E    8F  B0 00223 9$:       MOVW    #270, STG_DESC+2                        ; 1162
                  18  AE              59  B0 00229        MOVW    LENGTH, STG_DESC                        ; 1163
                  1C  AE          FF7C    CD  9E 0022D        MOVAB   COMMAND_LINE, STG_DESC+4                ; 1164
    19 00000000G      00                  01  E1 00233        BBC     #1, DBG$GL_DEVELOPER, 10$               ; 1170
                                     18  AE  9F 0023B        PUSHAB  STG_DESC                                ; 1173
                              00000000'   EF  9F 0023E        PUSHAB  P.AAT
          00000000G   00                  02  FB 00244        CALLS   #2, DBG$PRINT
          00000000G   00                  00  FB 0024B        CALLS   #0, DBG$NEWLINE                         ; 1174
                                          69  11 00252        BRB     15$                                     ; 1170
          00000000G   00                  00  FB 00254 10$:       CALLS   #0, DBG$SCR_SCREEN_NORMAL       ; 1182
                      50          01  AB  9A  0025B        MOVZBL  1(R11), R0                              ; 1187
                      01                  50  91 0025F        CMPB    R0, #1                                  ; 1190
                                          13  12 00262        BNEQ    11$
          00000000G   00                  10  88 00264        BISB2   #16, DBG$GV_CONTROL                     ; 1192
                                     18  AE  9F 0026B        PUSHAB  STG_DESC                                ; 1193
          00000000G   00                  01  FB 0026E        CALLS   #1, LIB$DO_COMMAND
                                          46  11 00275        BRB     15$                                     ; 1187
                      02                  50  91 00277 11$:       CMPB    R0, #2                          ; 1196
                                          2C  12 0027A        BNEQ    14$
                                     18  AE  9F 0027C        PUSHAB  STG_DESC                                ; 1200
```

```
                00000000G  00                    01  FB  0027F              CALLS   #1, LIB$SPAWN                                   ;
                           06 00000000G           00  E9  00286              BLBC    DBG$GB_KEYPAD_INPUT, 12$                        ; 1201
                      14   AE                     01  D0  0028D              MOVL    #1, X                                           ; 1202
                                                  03  11  00291              BRB     13$                                             ;
                                14   AE           D4  00293  12$:           CLRL    X                                               ; 1203
                                14   AE           9F  00296  13$:           PUSHAB  X                                               ; 1204
                           00000000G              00  9F  00299              PUSHAB  DBG$GL_KEYBOARD_ID                              ;
                00000000G  00                     02  FB  0029F              CALLS   #2, SMG$SET_KEYPAD_MODE                         ;
                                                  15  11  002A6              BRB     15$                                             ; 1187
                           00000000'              EF  9F  002A8  14$:        PUSHAB  P.AAU                                           ; 1208
                                                  01  DD  002AE              PUSHL   #1                                              ;
                           00028362               8F  DD  002B0              PUSHL   #164706                                         ;
                00000000G  00                     03  FB  002B6              CALLS   #3, LIB$SIGNAL                                  ;
                           50                     01  D0  002BD  15$:        MOVL    #1, R0                                          ; 1215
                                                  04  002C0                  RET                                                     ; 1216

; Routine Size:  705 bytes,    Routine Base:  DBG$CODE + 0406

; 1089          1217  0 END ELUDOM


                                                                  .EXTRN  LIB$SIGNAL

;                                  PSECT SUMMARY
;
;       Name                            Bytes                           Attributes
;
;  DBG$PLIT                               321  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(0)
;  DBG$OWN                                 81  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  DBG$CODE                              1735  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(0)


;                               Library Statistics
;
;                                        --------  Symbols  --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped       Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1          18619        68         0        1000      00:01.9
;  _$255$DUA28:[DEBUG.OBJ]STRUCDEF.L32;1     32         0         0           7      00:00.1
;  _$255$DUA28:[DEBUG.OBJ]DBGLIB.L32;1     1545        84         5          97      00:02.1
;  _$255$DUA28:[DEBUG.OBJ]DSTRECRDS.L32;1
;                                           418         0         0          31      00:00.3
;  _$255$DUA28:[DEBUG.OBJ]DBGMSG.L32;1      386        10         2          22      00:00.3
;  _$255$DUA28:[DEBUG.OBJ]DBGGEN.L32;1      150         0         0          12      00:00.3


; Information:  1
; Warnings:     0
; Errors:       0
```

```
;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:DBGNTYPE/OBJ=OBJ$:DBGNTYPE MSRC$:DBGNTYPE/UPDATE=(ENH$:DBGNTYPE)

; Size:         1735 code + 402 data bytes
; Run Time:       00:34.6
; Elapsed Time:   01:35.8
; Lines/CPU Min:   2109
; Lexemes/CPU-Min: 15119
; Memory Used:  252 pages
; Compilation Complete
```

DBGNTYPE
LIS

DBGNSHOW
LIS

DBGPARSER
LIS