```
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      UUUUUUUUUUUUUUUUU     GGGGGGGGG
DDDDDDDDDDDD    EEEEEEEEEEEEEEE  BBBBBBBBBBBB      UUUUUUUUUUUUUUUUU     GGGGGGGGG
DDDDDDDDDDDD    EEEEEEEEEEEEEEE  BBBBBBBBBBBB      UUUUUUUUUUUUUUUUU     GGGGGGGGG
```

```
DDDDDDDD    BBBBBBBB      GGGGGGGG  NN      NN  HH      HH  EEEEEEEEEE  LL          PPPPPPPP
DDDDDDDD    BBBBBBBB      GGGGGGGG  NN      NN  HH      HH  EEEEEEEEEE  LL          PPPPPPPP
DD      DD  BB      BB  GG          NN      NN  HH      HH  EE          LL          PP      PP
DD      DD  BB      BB  GG          NN      NN  HH      HH  EE          LL          PP      PP
DD      DD  BB      BB  GG          NNNN    NN  HH      HH  EE          LL          PP      PP
DD      DD  BB      BB  GG          NNNN    NN  HH      HH  EE          LL          PP      PP
DD      DD  BBBBBBBB    GG          NN  NN  NN  HHHHHHHHHH  EEEEEEEE    LL          PPPPPPPP
DD      DD  BBBBBBBB    GG          NN  NN  NN  HHHHHHHHHH  EEEEEEEE    LL          PPPPPPPP
DD      DD  BB      BB  GG  GGGGGG  NN    NNNN  HH      HH  EE          LL          PP
DD      DD  BB      BB  GG  GGGGGG  NN    NNNN  HH      HH  EE          LL          PP
DD      DD  BB      BB  GG      GG  NN      NN  HH      HH  EE          LL          PP              ....
DD      DD  BB      BB  GG      GG  NN      NN  HH      HH  EE          LL          PP              ....
DDDDDDDD    BBBBBBBB      GGGGGG    NN      NN  HH      HH  EEEEEEEEEE  LLLLLLLLLL  PP              ....
DDDDDDDD    BBBBBBBB      GGGGGG    NN      NN  HH      HH  EEEEEEEEEE  LLLLLLLLLL  PP              ....

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
```

```
;     1          0001  0 MODULE DBGNHELP (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 ! MODULE FUNCTION
;    31          0031  1 !       This module contains the ATN parse and command execution networks to
;    32          0032  1 !       support the HELP command. The routine DBG$NEXECUTE_HELP consists of
;    33          0033  1 !       the version 2 debugger routine DBG$GET_HELP with a few modifications
;    34          0034  1 !       to allow it to perform correctly in version 3. Specifically, error
;    35          0035  1 !       signals (which unwind the stack) have been replaced with calls to
;    36          0036  1 !       version 3 error output routines.
;    37          0037  1 !
;    38          0038  1 !
;    39          0039  1 ! AUTHOR:       David Plummer, CREATION DATE:   4/9/80
;    40          0040  1 !
;    41          0041  1 ! MODIFIED BY:
;    42          0042  1 !
;    43          0043  1 !               Richard Title   15 Dec 1981     Converted to the new help
;    44          0044  1 !                                               librarian LBR$OUTPUT_HELP,
;    45          0045  1 !                                               which prompts for output
;    46          0046  1 !                                               ("Topic? ", "Subtopic? ")
;    47          0047  1 !               Richard Title   13-Jun 1982     Added support for DBG$HELP
;    48          0048  1 !                                               (logical name telling where
;    49          0049  1 !                                               the help library is)
;    50          0050  1 !
;    51          0051  1
;    52          0052  1 REQUIRE 'SRC$:DBGPROLOG.REQ';
;    53          0186  1
;    54          0187  1 LIBRARY 'LIB$:DBGGEN.L32';
;    55          0188  1
;    56          0189  1 FORWARD ROUTINE
;    57          0190  1     DBG$NPARSE_HELP,                    ! Creates the command execution tree for help
```

```
;   58          0191  1     DBG$NEXECUTE_HELP,          ! Executes the parsed HELP command
;   59          0192  1     PRINT_HELP_LINE,            ! Print a line of HELP text
;   60          0193  1     INPUT_HELP_LINE;            ! Input HELP topic or subtopic request
```

```
;    62          0194  1 EXTERNAL ROUTINE
;    63          0195  1     DBG$GET_TEMPMEM,                    ! Allocates temporary dynamic storage
;    64          0196  1     DBG$NEWLINE: NOVALUE,               ! Go to a new print line
;    65          0197  1     DBG$NMAKE_ARG_VECT,                 ! Constructs a message argument vector
;    66          0198  1     DBG$PRINT: NOVALUE,                 ! Print some debug output
;    67          0199  1     DBG$SCR_OUTPUT_SCREEN: NOVALUE,     ! Output the current screen contents
;    68          0200  1     DBG$SCR_SCREEN_MODE: NOVALUE,       ! Turn screen mode on or off
;    69          0201  1     LBR$INI_CONTROL,                    ! Librarian init control table
;    70          0202  1     LBR$OPEN,                           ! Librarian open library file
;    71          0203  1     LBR$CLOSE,                          ! Librarian close library file
;    72          0204  1     LBR$GET_HELP,                       ! Librarian get help
;    73          0205  1     LBR$OUTPUT_HELP,                    ! Librarian output help
;    74          0206  1     LIB$PUT_OUTPUT,                     ! Library output routine
;    75          0207  1     LIB$GET_INPUT,                      ! Library input routine
;    76          0208  1     SMG$READ_COMPOSED_LINE,             ! Read a line of input in keypad mode
;    77          0209  1     SYS$TRNLOG;                         ! Translate logical name
;    78          0210  1
;    79          0211  1 EXTERNAL
;    80          0212  1     DBG$CHAR_TABLE: VECTOR[,BYTE],      ! Character type table
;    81          0213  1     DBG$GB_KEYPAD_INPUT: BYTE,          ! Flag set if keypad input is active
;    82          0214  1     DBG$GL_CISHEAD: REF CIS$LINK,       ! Head of Command Input Stream
;    83          0215  1     DBG$GL_INPRAB: BLOCK[,BYTE],        ! RAB for DEBUG input (DBG$INPUT)
;    84          0216  1     DBG$GL_KEYBOARD_ID,                 ! The keyboard Id used for keypad input
;    85          0217  1     DBG$GL_KEY_TABLE_ID,                ! The key-table Id used for keypad input
;    86          0218  1     DBG$GL_OUTPRAB: BLOCK[,BYTE],       ! RAB for DEBUG output (DBG$OUTPUT)
;    87          0219  1     DBG$GL_SCREEN_MODE;                 ! Flag set if screen mode is active
;    88          0220  1
;    89          0221  1 EXTERNAL LITERAL
;    90          0222  1     SMG$_EOF;                           ! Keypad input End-of-File code
```

```
;    92          0223  1 GLOBAL ROUTINE DBG$NPARSE_HELP (INPUT_DESC, VERB_NODE, MESSAGE_VECT) =
;    93          0224  1 !
;    94          0225  1 ! FUNCTION
;    95          0226  1 !       DBG$NPARSE_HELP constructs the command execution tree for the HELP
;    96          0227  1 !       command.  Specifically, a Noun Node is allocated and linked to the
;    97          0228  1 !       Verb Node.  A copy of the present input descriptor (describing the
;    98          0229  1 !       input line minus the keyword HELP) is made, and the value of the Noun
;    99          0230  1 !       Node is a pointer to this descriptor.  The actual parsing of the HELP
;   100          0231  1 !       string is done by DBG$NEXECUTE_HELP.
;   101          0232  1 !
;   102          0233  1 ! INPUTS
;   103          0234  1 !       INPUT_DESC - Descriptor of the present input line.
;   104          0235  1 !
;   105          0236  1 !       VERB_NODE - Head of the command execution tree.
;   106          0237  1 !
;   107          0238  1 !       MESSAGE_VECT - The address of a longword to contain the address
;   108          0239  1 !                  of a message argument vector.
;   109          0240  1 !
;   110          0241  1 ! OUTPUTS
;   111          0242  1 !       INPUT_DESC - The input line string descriptor is updated to reflect
;   112          0243  1 !                  the current parse location.  This normally means that the
;   113          0244  1 !                  input descriptor winds up pointing past the end of the line.
;   114          0245  1 !
;   115          0246  1 !       VERB_NODE - The Verb Node is filled in to reflect the parameters
;   116          0247  1 !                  parsed on the HELP command.
;   117          0248  1 !
;   118          0249  1 !       An unsigned integer longword completion code is returned as the
;   119          0250  1 !                  routine's value.  These are the possible values:
;   120          0251  1 !
;   121          0252  1 !                 STS$K_SEVERE (4)  - The input could not be parsed.
;   122          0253  1 !                 STS$K_SUCCESS (1) - The input was parsed and an execution
;   123          0254  1 !                                     tree was created.
;   124          0255  1 !
;   125          0256  1
;   126          0257  2     BEGIN
;   127          0258  2
;   128          0259  2     MAP
;   129          0260  2         INPUT_DESC: REF DBG$STG_DESC,   ! Pointer to input scring descriptor
;   130          0261  2         VERB_NODE: REF DBG$VERB_NODE;   ! Pointer to the Verb Node
;   131          0262  2
;   132          0263  2     LOCAL
;   133          0264  2         NOUN_NODE: REF DBG$NOUN_NODE,   ! Pointer to Noun node for execution tree
;   134          0265  2         COMMAND_DESC: REF DBG$STG_DESC; ! Descriptor of HELP line
;   135          0266  2
;   136          0267  2
;   137          0268  2
;   138          0269  2     ! Get storage for the Noun Node and link it to the Verb Node.
;   139          0270  2     !
;   140          0271  2     NOUN_NODE = DBG$GET_TEMPMEM (DBG$K_NOUN_NODE_SIZE);
;   141          0272  2     VERB_NODE [DBG$L_VERB_OBJECT_PTR] = .NOUN_NODE;
;   142          0273  2
;   143          0274  2
;   144          0275  2     ! Get storage for the command descriptor.
;   145          0276  2     !
;   146          0277  2     COMMAND_DESC = DBG$GET_TEMPMEM (2);
;   147          0278  2
;   148          0279  2
```

```
;  149          0280  2     ! Copy the fields of the input descriptor to the command descriptor.
;  150          0281  2     !
;  151          0282  2     COMMAND_DESC [DSC$W_LENGTH] = .INPUT_DESC [DSC$W_LENGTH];
;  152          0283  2     COMMAND_DESC [DSC$A_POINTER] = .INPUT_DESC [DSC$A_POINTER];
;  153          0284  2
;  154          0285  2
;  155          0286  2     ! Set the value of the Noun Node to the address of the command descriptor.
;  156          0287  2     !
;  157          0288  2     NOUN_NODE [DBG$L_NOUN_VALUE] = .COMMAND_DESC;
;  158          0289  2
;  159          0290  2
;  160          0291  2     ! Eat the rest of the input command and return.
;  161          0292  2     !
;  162          0293  2     INPUT_DESC [DSC$W_LENGTH] = 0;
;  163          0294  2     RETURN STS$K_SUCCESS;
;  164          0295  2
;  165          0296  1     END;
```

```
                                                .TITLE  DBGNHELP
                                                .IDENT  \V04-000\

                                                .EXTRN  DBG$GET_TEMPMEM
                                                .EXTRN  DBG$NEWLINE, DBG$NMAKE_ARG_VECT
                                                .EXTRN  DBG$PRINT, DBG$SCR_OUTPUT_SCREEN
                                                .EXTRN  DBG$SCR_SCREEN_MODE
                                                .EXTRN  LBR$INI_CONTROL
                                                .EXTRN  LBR$OPEN, LBR$CLOSE
                                                .EXTRN  LBR$GET_HELP, LBR$OUTPUT_HELP
                                                .EXTRN  LIB$PUT_OUTPUT, LIB$GET_INPUT
                                                .EXTRN  SMG$READ_COMPOSED_LINE
                                                .EXTRN  SYS$TRNLOG, DBG$CHAR_TABLE
                                                .EXTRN  DBG$GB_KEYPAD_INPUT
                                                .EXTRN  DBG$GL_CISHEAD, DBG$GL_INPRAB
                                                .EXTRN  DBG$GL_KEYBOARD_ID
                                                .EXTRN  DBG$GL_KEY_TABLE_ID
                                                .EXTRN  DBG$GL_OUTPRAB, DBG$GL_SCREEN_MODE
                                                .EXTRN  SMG$_EOF

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

                                    000C 00000  .ENTRY  DBG$NPARSE_HELP, Save R2,R3                     ; 0223
     53 00000000G   00    9E 00002  MOVAB    DBG$GET_TEMPMEM, R3
                    04    DD 00009  PUSHL    #4                                                       ; 0271
                    01    FB 0000B  CALLS    #1, DBG$GET_TEMPMEM
     52             50    D0 0000E  MOVL     R0, NOUN_NODE
     50        08   AC    D0 00011  MOVL     VERB_NODE, R0                                            ; 0272
08   A0             52    D0 00015  MOVL     NOUN_NODE, 8(R0)
                    02    DD 00019  PUSHL    #2                                                       ; 0277
                    01    FB 0001B  CALLS    #1, DBG$GET_TEMPMEM
     51        04   AC    D0 0001E  MOVL     INPUT_DESC, R1                                           ; 0282
     60             61    B0 00022  MOVW     (R1), (COMMAND_DESC)
04   A0        04   A1    D0 00025  MOVL     4(R1), 4(COMMAND_DESC)                                   ; 0283
     62             50    D0 0002A  MOVL     COMMAND_DESC, (NOUN_NODE)                                ; 0288
                    61    B4 0002D  CLRW     (R1)                                                     ; 0293
     50             01    D0 0002F  MOVL     #1, R0                                                   ; 0294
                          04 00032  RET                                                               ; 0296
```

```
; Routine Size:  51 bytes,    Routine Base:  DBG$CODE + 0000
```

```
;  167     0297  1 GLOBAL ROUTINE DBG$NEXECUTE_HELP (VERB_NODE) =
;  168     0298  1 !
;  169     0299  1 ! FUNCTION
;  170     0300  1 !       Invoke the VMS librarian to implement the HELP command.
;  171     0301  1 !
;  172     0302  1 ! INPUTS
;  173     0303  1 !       VERB_NODE - A pointer to the Verb Node that forms the head of the
;  174     0304  1 !                   command execution tree for the HELP command.
;  175     0305  1 !
;  176     0306  1 ! OUTPUTS
;  177     0307  1 !       This routine always returns STS$K_SUCCESS as its value.
;  178     0308  1 !
;  179     0309  1
;  180     0310  2     BEGIN
;  181     0311  2
;  182     0312  2     MAP
;  183     0313  2         VERB_NODE: REF DBG$VERB_NODE;   ! Pointer to the input Verb Node
;  184     0314  2
;  185     0315  2     LOCAL
;  186     0316  2         CHAR,                           ! Temporary placeholder for a char
;  187     0317  2         COUNT,                          ! Counter for leading blanks
;  188     0318  2         DBGHELP_STGDESC: BLOCK[8,BYTE], ! String descriptor for DBG$HELP
;  189     0319  2         DBGHELP_STG: VECTOR[8,BYTE],    ! String with DBG$HELP
;  190     0320  2         DUMMY: VECTOR [2],              ! Output string descriptor for
;  191     0321  2                                         !     SYS$TRNLOG
;  192     0322  2         DUMMY_BUFFER: VECTOR [256, BYTE],! Output buffer for SYS$TRNLOG
;  193     0323  2         INPUT_PTR,                      ! Temporary pointer into input
;  194     0324  2         LIB_NAME: REF DBG$STG_DESC,     ! descriptor for library name
;  195     0325  2         NOUN_NODE: REF DBG$NOUN_NODE,   ! noun node of command execution tree
;  196     0326  2         PARSE_STG_DESC: REF DBG$STG_DESC,! Descriptor of the help command
;  197     0327  2         SAVED_PARSE_STG_DESC,           ! ???
;  198     0328  2         SCREEN_MODE_FLAG,               ! Saved value of screen mode flag
;  199     0329  2         STATUS,                         ! Librarian routines return status
;  200     0330  2         TRNLOG;                         ! Return status from $TRNLOG
;  201     0331  2
;  202     0332  2
;  203     0333  2
;  204     0334  2     ! Recover pointers to the Noun Node and the command descriptor.
;  205     0335  2     !
;  206     0336  2     NOUN_NODE = .VERB_NODE [DBG$L_VERB_OBJECT_PTR];
;  207     0337  2     PARSE_STG_DESC = .NOUN_NODE [DBG$L_NOUN_VALUE];
;  208     0338  2
;  209     0339  2
;  210     0340  2     ! Initialize the library name. If the logical name DBG$HELP is defined,
;  211     0341  2     ! then we use that as the directory in which to find DEBUGHLP.HLB.
;  212     0342  2     ! Otherwise, the help librarian looks in SYS$HELP by default.
;  213     0343  2     !
;  214     0344  2     LIB_NAME = DBG$GET_TEMPMEM (2);
;  215     0345  2     DUMMY[0] = %X'010E0000' + 256;
;  216     0346  2     DUMMY[1] = DUMMY_BUFFER;
;  217     0347  2
;  218     0348  2
;  219     0349  2     ! Set up string descriptor for DBG$HELP.  Use DBG$HELP as the directory
;  220     0350  2     ! name if a logical name translation exists for DBG$HELP.
;  221     0351  2     !
;  222     0352  2     DBGHELP_STGDESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  223     0353  2     DBGHELP_STGDESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
```

```
;  224          0354  2      DBGHELP_STGDESC[DSC$W_LENGTH] = 8;
;  225          0355  2      DBGHELP_STGDESC[DSC$A_POINTER] = DBGHELP_STG;
;  226          0356  2      CH$MOVE (8, UPLIT BYTE (%ASCII 'DBG$HELP'), DBGHELP_STG);
;  227          0357  2      TRNLOG = SYS$TRNLOG (DBGHELP_STGDESC, 0, DUMMY, 0, 0, 0);
;  228          0358  2      IF .TRNLOG EQL SS$_NORMAL
;  229          0359  2      THEN
;  230          0360  3          BEGIN
;  231          0361  3          LIB_NAME [DSC$W_LENGTH] = 17;
;  232          0362  3          LIB_NAME [DSC$A_POINTER] = UPLIT BYTE (%ASCII 'DBG$HELP:DEBUGHLP');
;  233          0363  3          END
;  234          0364  3
;  235          0365  2      ELSE
;  236          0366  3          BEGIN
;  237          0367  3          LIB_NAME [DSC$W_LENGTH] = 8;
;  238          0368  3          LIB_NAME [DSC$A_POINTER] = UPLIT BYTE(%ASCII 'DEBUGHLP');
;  239          0369  2          END;
;  240          0370  2
;  241          0371  2
;  242          0372  2      ! Suppress leading blanks and tabs.
;  243          0373  2      !
;  244          0374  2      INPUT_PTR = CH$PTR(.PARSE_STG_DESC[DSC$A_POINTER]);
;  245          0375  2      CHAR = CH$RCHAR(INPUT_PTR);
;  246          0376  2      COUNT = 0;
;  247          0377  2      WHILE .DBG$CHAR_TABLE[.CHAR] EQL 4 DO
;  248          0378  3          BEGIN
;  249          0379  3          CHAR = CH$RCHAR_A(INPUT_PTR);
;  250          0380  3          COUNT = .COUNT + 1;
;  251          0381  2          END;
;  252          0382  2
;  253          0383  2
;  254          0384  2      ! Update the string descriptor to point to the first non-blank character.
;  255          0385  2      !
;  256          0386  2      PARSE_STG_DESC[DSC$W_LENGTH] = .PARSE_STG_DESC[DSC$W_LENGTH] - .COUNT;
;  257          0387  2      PARSE_STG_DESC[DSC$A_POINTER] = .INPUT_PTR;
;  258          0388  2
;  259          0389  2
;  260          0390  2      ! Save away PARSE_STG_DESC before we clobber it.
;  261          0391  2      !
;  262          0392  2      SAVED_PARSE_STG_DESC = .PARSE_STG_DESC;
;  263          0393  2
;  264          0394  2
;  265          0395  2      ! Remove the trailing carriage return from PARSE_STG_DESC.
;  266          0396  2      !
;  267          0397  2      INPUT_PTR = CH$PTR (.PARSE_STG_DESC[DSC$A_POINTER]);
;  268          0398  2      INPUT_PTR = CH$PLUS (.INPUT_PTR, .PARSE_STG_DESC[DSC$W_LENGTH] - 1);
;  269          0399  2      IF CH$RCHAR(.INPUT_PTR) EQL DBG$K_CAR_RETURN
;  270          0400  2      THEN
;  271          0401  2          PARSE_STG_DESC[DSC$W_LENGTH] = .PARSE_STG_DESC[DSC$W_LENGTH] - 1;
;  272          0402  2
;  273          0403  2
;  274          0404  2      ! Check for all blanks. If so, put zero in PARSE_STG_DESC to tell the
;  275          0405  2      ! HELP librarian that no keys were specified.
;  276          0406  2      !
;  277          0407  2      IF .PARSE_STG_DESC[DSC$W_LENGTH] EQL 0
;  278          0408  2      THEN
;  279          0409  2          PARSE_STG_DESC = 0;
;  280          0410  2
```

```
;  281          0411  2
;  282          0412  2      ! If screen mode is set, turn off screen mode for the duration of the
;  283          0413  2      ! HELP command.  We restore screen mode after the HELP command completes
;  284          0414  2      ! if it was set when the HELP command was entered.
;  285          0415  2      !
;  286          0416  2      SCREEN_MODE_FLAG = .DBG$GL_SCREEN_MODE;
;  287          0417  2      IF .DBG$GL_SCREEN_MODE THEN DBG$SCR_SCREEN_MODE(FALSE);
;  288          0418  2
;  289          0419  2
;  290          0420  2      ! Call the library routine to output help text.  Note that we restore
;  291          0421  2      ! screen mode if appropriate before we signal any error message.
;  292          0422  2      !
;  293          0423  2      STATUS = LBR$OUTPUT_HELP (PRINT_HELP_LINE, 0, .PARSE_STG_DESC,
;  294          0424  2                                .LIB_NAME, OPLIT(HLP$M_PROMPT), INPUT_HELP_LINE);
;  295          0425  2      IF .SCREEN_MODE_FLAG THEN DBG$SCR_SCREEN_MODE(TRUE);
;  296          0426  2      IF NOT .STATUS THEN SIGNAL(DBG$_NOSUCHELP, 0, .STATUS);
;  297          0427  2
;  298          0428  2
;  299          0429  2      ! The HELP has been displayed.  Now cleanup and return.
;  300          0430  2      !
;  301          0431  2      PARSE_STG_DESC = .SAVED_PARSE_STG_DESC;
;  302          0432  2      PARSE_STG_DESC[DSC$A_POINTER] = CH$PLUS(.PARSE_STG_DESC[DSC$A_POINTER],
;  303          0433  2                                              .PARSE_STG_DESC[DSC$W_LENGTH]);
;  304          0434  2      PARSE_STG_DESC[DSC$W_LENGTH] = 0;
;  305          0435  2      RETURN STS$K_SUCCESS;
;  306          0436  2
;  307          0437  1      END;


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

                                50  4C  45  48  24  47  42  44  00000 P.AAA:  .ASCII  \DBG$HELP\                          ;
48  47  55  42  45  44  3A       50  4C  45  48  24  47  42  44  00008 P.AAB:  .ASCII  \DBG$HELP:DEBUGHLP\                ;
                                                                50  4C  00017
                                50  4C  48  47  55  42  45  44  00019 P.AAC:  .ASCII  \DEBUGHLP\                         ;
                                                                00021        .BLKB   3
                                                    00000001    00024 P.AAD:  .LONG   1                                 ;


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

                                                        03FC 00000           .ENTRY  DBG$NEXECUTE_HELP, Save R2,R3,R4,R5,R6,R7,- ; 0297
                                                                             R8,R9                                      ;
                            59 00000000G   00   9E 00002           MOVAB   DBG$SCR_SCREEN_MODE, R9                    ;
                            58 00000000'   EF   9L 00009           MOVAB   P.AAA, R8                                  ;
                            5E      FEE8   CE   9E 00010           MOVAB   -280(SP), SP                               ;
                            50        04   AC   D0 00015           MOVL    VERB_NODE, R0                              ; 0336
                            50        08   A0   D0 00019           MOVL    8(R0), NOUN_NODE                           ;
                            57             60   D0 0001D           MOVL    (NOUN_NODE), PARSE_STG_DESC                ; 0337
                                           02   DD 00020           PUSHL   #2                                         ; 0344
                    00000000G   00         01   FB 00022           CALLS   #1, DBG$GET_TEMPMEM                        ;
                            56             50   D0 00029           MOVL    R0, LIB_NAME                               ;
                        E8  AD 010E0100    8F   D0 0002C           MOVL    #17694976, DUMMY                           ; 0345
                        EC  AD             6E   9E 00034           MOVAB   DUMMY_BUFFER, DUMMY+4                      ; 0346
                        F8  AD 010E0008    8F   D0 00038           MOVL    #17694728, DBGHELP_STGDESC                 ; 0354
```

```
                FC  AD      F0  AD  9E 00040          MOVAB   DBGHELP_STG, DBGHELP_STGDESC+4          ; 0355
        F0  AD          68      08  28 00045          MOVC3   #8, P.AAA, DBGHELP_STG                  ; 0356
                                7E  7C 0004A          CLRQ    -(SP)                                   ; 0357
                                7E  D4 0004C          CLRL    -(SP)                                   ;
                            E8  AD  9F 0004E          PUSHAB  DUMMY                                   ;
                                7E  D4 00051          CLRL    -(SP)                                   ;
                            F8  AD  9F 00053          PUSHAB  DBGHELP_STGDESC                         ;
                00000000G   00      06  FB 00056      CALLS   #6, SYS$TRNLOG                          ;
                            01      50  D1 0005D      CMPL    TRNLOG, #1                              ; 0358
                                    0A  12 00060      BNEQ    1$                                      ;
                            66      11  B0 00062      MOVW    #17, (LIB_NAME)                         ; 0361
                        04  A6      08  A8 9E 00065   MOVAB   P.AAB, 4(LIB_NAME)                     ; 0362
                                    08  11 0006A      BRB     2$                                      ; 0358
                            66      08  B0 0006C  1$:  MOVW    #8, (LIB_NAME)                          ; 0367
                        04  A6      19  A8 9E 0006F   MOVAB   P.AAC, 4(LIB_NAME)                     ; 0368
                            50      04  A7 D0 00074  2$:  MOVL    4(PARSE_STG_DESC), INPUT_PTR            ; 0374
                            51          50 9A 00078      MOVZBL  INPUT_PTR, CHAR                         ; 0375
                                        52 D4 0007B      CLRL    COUNT                                   ; 0376
                            04 00000000G041  91 0007D  3$:  CMPB    DBG$CHAR_TABLE[CHAR], #4                ; 0377
                                        07  12 00085      BNEQ    4$                                      ;
                            51          80 9A 00087      MOVZBL  (INPUT_PTR)+, CHAR                      ; 0379
                                        52 D6 0008A      INCL    COUNT                                   ; 0380
                                        EF 11 0008C      BRB     3$                                      ; 0377
                            67          52 A2 0008E  4$:  SUBW2   COUNT, (PARSE_STG_DESC)                 ; 0386
                        04  A7          50 D0 00091      MOVL    INPUT_PTR, 4(PARSE_STG_DESC)            ; 0387
                            54          57 D0 00095      MOVL    PARSE_STG_DESC, SAVED_PARSE_STG_DESC    ; 0392
                            50      04  A7 D0 00098      MOVL    4(PARSE_STG_DESC), INPUT_PTR            ; 0397
                            51          67 3C 0009C      MOVZWL  (PARSE_STG_DESC), R1                    ; 0398
                            51          50 C0 0009F      ADDL2   INPUT_PTR, R1                           ;
                            50      FF  A1 9E 000A2      MOVAB   -1(R1), INPUT_PTR                       ;
                            0D          60 91 000A6      CMPB    (INPUT_PTR), #13                        ; 0399
                                        02 12 000A9      BNEQ    5$                                      ;
                                        67 B7 000AB      DECW    (PARSE_STG_DESC)                        ; 0401
                                        67 B5 000AD  5$:  TSTW    (PARSE_STG_DESC)                        ; 0407
                                        02 12 000AF      BNEQ    6$                                      ;
                                        57 D4 000B1      CLRL    PARSE_STG_DESC                          ; 0409
                            50  00000000G   00  D0 000B3  6$:  MOVL    DBG$GL_SCREEN_MODE, R0                  ; 0416
                            53          50 D0 000BA      MOVL    R0, SCREEN_MODE_FLAG                    ;
                            05          50 E9 000BD      BLBC    R0, 7$                                  ; 0417
                                        7E D4 000C0      CLRL    -(SP)                                   ;
                            69          01 FB 000C2      CALLS   #1, DBG$SCR_SCREEN_MODE                 ;
                                0000V   CF 9F 000C5  7$:  PUSHAB  INPUT_HELP_LINE                         ; 0423
                                  24    A8 9F 000C9      PUSHAB  P.AAD                                   ; 0424
                                        56 DD 000CC      PUSHL   LIB_NAME                                ;
                                        57 DD 000CE      PUSHL   PARSE_STG_DESC                          ; 0423
                                        7E D4 000D0      CLRL    -(SP)                                   ;
                                0000V   CF 9F 000D2      PUSHAB  PRINT_HELP_LINE                         ;
        00000000G               00      06 FB 000D6      CALLS   #6, LBR$OUTPUT_HELP                     ;
                            52          50 D0 000DD      MOVL    R0, STATUS                              ;
                            05          53 E9 000E0      BLBC    SCREEN_MODE_FLAG, 8$                    ; 0425
                                        01 DD 000E3      PUSHL   #1                                      ;
                            69          01 FB 000E5      CALLS   #1, DBG$SCR_SCREEN_MODE                 ;
                            11          52 E8 000E8  8$:  BLBS    STATUS, 9$                              ; 0426
                                        52 DD 000EB      PUSHL   STATUS                                  ;
                                        7E D4 000ED      CLRL    -(SP)                                   ;
                                00028D28    8F DD 000EF  PUSHL   #167208                                 ;
        00000000G               00      03 FB 000F5      CALLS   #3, LIB$SIGNAL                          ;
```

```
                    57          54  D0 000FC 9$:    MOVL    SAVED_PARSE_STG_DESC, PARSE_STG_DESC    ; 0431
                    50          67  3C 000FF       MOVZWL  (PARSE_STG_DESC), R0                    ; 0433
                04  A7          50  C0 00102       ADDL2   R0, 4(PARSE_STG_DESC)                   ;
                                67  B4 00106       CLRW    (PARSE_STG_DESC)                        ; 0434
                    50          01  D0 00108       MOVL    #1, R0                                  ; 0435
                                    04 0010B       RET                                             ; 0437

; Routine Size:  268 bytes,    Routine Base:  DBG$CODE + 0033
```

```
;  309          0438  1 ROUTINE PRINT_HELP_LINE(LINE_DESC) =
;  310          0439  1 !
;  311          0440  1 ! FUNCTION
;  312          0441  1 !       Print a line of HELP text to the DEBUG output device.  It is necessary
;  313          0442  1 !       to pass this routine to LBR$OUTPUT_HELP, instead of using the default
;  314          0443  1 !       routine LIB$PUT_OUTPUT, because DEBUG may write its output to a log
;  315          0444  1 !       file, to logical name DBG$OUTPUT, or to a screen display, and not
;  316          0445  1 !       necessarily to SYS$OUTPUT.
;  317          0446  1 !
;  318          0447  1 ! INPUTS
;  319          0448  1 !       LINEDESC - A pointer to a string descriptor for the line to be output.
;  320          0449  1 !
;  321          0450  1 ! OUTPUTS
;  322          0451  1 !       This routine always returns SS$_NORMAL as its value.
;  323          0452  1 !
;  324          0453  1
;  325          0454  2     BEGIN
;  326          0455  2
;  327          0456  2     MAP
;  328          0457  2         LINE_DESC: REF DBG$STG_DESC;    ! Pointer to output line string descr.
;  329          0458  2
;  330          0459  2
;  331          0460  2
;  332          0461  2     ! Output the line of HELP text via DBG$PRINT.  Then return.
;  333          0462  2     !
;  334          0463  2     $ABORT_ON_CONTROL_Y;
;  335          0464  2     DBG$PRINT(UPLIT BYTE(%ASCIC '!AD'),
;  336          0465  2                       .LINE_DESC[DSC$W_LENGTH], .LINE_DESC[DSC$A_POINTER]);
;  337          0466  2     DBG$NEWLINE();
;  338          0467  2     RETURN SS$_NORMAL;
;  339          0468  2
;  340          0469  1     END;

                                        .PSECT  DBG$PLIT,NOWRT,  SHR,  PIC,0
                     44  41  21  03  00028 P.AAE:  .ASCII  <3>\!AD\                                  ;
                                        .EXTRN  DBG$GV_CONTROL
                                        .PSECT  DBG$CODE,NOWRT,  SHR,  PIC,0
                             0000 00000 PRINT_HELP_LINE:
                                        .WORD   Save nothing                                        ; 0438
OD 00000000G   00                01  E1 00002          BBC     #1, DBG$GV_CONTROL+1, 1$            ; 0457
                    000280E8     8F  DD 0000A          PUSHL   #164072                             ;
   00000000G   00                01  FB 00010          CALLS   #1, LIB$SIGNAL                      ;
               50         04     AC  D0 00017 1$:      MOVL    LINE_DESC, R0                       ; 0465
                          04     A0  DD 0001B          PUSHL   4(R0)                               ;
               7E                60  3C 0001E          MOVZWL  (R0), -(SP)                         ;
                    00000000'    EF  9F 00021          PUSHAB  P.AAE                               ; 0464
   00000000G   00                03  FB 00027          CALLS   #3, DBG$PRINT                       ;
   00000000G   00                00  FB 0002E          CALLS   #0, DBG$NEWLINE                     ; 0466
               50                01  D0 00035          MOVL    #1, R0                              ; 0467
                                 04  00038          RET                                         ; 0469
```

```
; Routine Size:  57 bytes,    Routine Base:  DBG$CODE + 013F
```

```
;  342          0470  1 ROUTINE INPUT_HELP_LINE (GET_STR, PROMPT_STR) =
;  343          0471  1 !
;  344          0472  1 ! FUNCTION
;  345          0473  1 !       Reads a line of HELP input from the DEBUG input stream.  This routine
;  346          0474  1 !       is used by LBR$OUTPUT_HELP to collect responses to the "Topic? " and
;  347          0475  1 !       "Subtopic? " prompts.  It is necessary to use this instead of the
;  348          0476  1 !       default LIB$GET_INPUT, because DEBUG may read its input from DBG$INPUT,
;  349          0477  1 !       the keypad read routine, or from an indirect command file, and not
;  350          0478  1 !       necessarily from SYS$INPUT.
;  351          0479  1 !
;  352          0480  1 ! INPUTS
;  353          0481  1 !       GET_STR - The address of a string descriptor pointing to the buffer
;  354          0482  1 !                 to receive the input line.
;  355          0483  1 !
;  356          0484  1 !       PROMPT_STR - A string descriptor specifying the prompt string.
;  357          0485  1 !
;  358          0486  1 ! OUTPUTS
;  359          0487  1 !       GET_STR - The actual length of the read input string is returned to the
;  360          0488  1 !                 length field of the GET_STR string descriptor.
;  361          0489  1 !
;  362          0490  1 !       The status returned by the $GET call is returned as this routine's
;  363          0491  1 !                 value.  If $GET was not called, SS$_NORMAL or the keypad
;  364          0492  1 !                 input routine's status is returned.
;  365          0493  1 !
;  366          0494  1
;  367          0495  2     BEGIN
;  368          0496  2
;  369          0497  2     MAP
;  370          0498  2         GET_STR: REF DBG$STG_DESC,      ! String descriptor for input buffer
;  371          0499  2         PROMPT_STR: REF DBG$STG_DESC;   ! Prompt string string descriptor
;  372          0500  2
;  373          0501  2     LOCAL
;  374          0502  2         CIS_PTR: REF CIS$LINK,          ! Pointer to Command Input Stream entry
;  375          0503  2         LENGTH,                         ! The input length on a keypad read
;  376          0504  2         STATUS;                         ! The RMS status code
;  377          0505  2
;  378          0506  2
;  379          0507  2
;  380          0508  2     ! If we are currently reading from a DEBUG command list as on an IF, FOR,
;  381          0509  2     ! or WHILE statement, or a screen display DEBUG command list, we simply
;  382          0510  2     ! return a null line to the HELP librarian.  This means that HELP commands
;  383          0511  2     ! in such command lists do not prompt for topics or subtopics.
;  384          0512  2     !
;  385          0513  2     CIS_PTR = .DBG$GL_CISHEAD;
;  386          0514  2     IF .CIS_PTR[CIS$B_INPUT_TYPE] EQL CIS_INPBUF
;  387          0515  2     THEN
;  388          0516  2         CIS_PTR = .CIS_PTR[CIS$A_NEXT_LINK];
;  389          0517  2
;  390          0518  2     IF (.CIS_PTR[CIS$B_INPUT_TYPE] NEQ CIS_DBG$INPUT) AND
;  391          0519  3        (.CIS_PTR[CIS$B_INPUT_TYPE] NEQ CIS_RAB)
;  392          0520  2     THEN
;  393          0521  3         BEGIN
;  394          0522  3         GET_STR[DSC$W_LENGTH] = 0;
;  395          0523  3         RETURN SS$_NORMAL;
;  396          0524  2         END;
;  397          0525  2
;  398          0526  2
```

```
;  399          0527  2      ! We are reading from the user's input terminal (SYS$INPUT or DBG$INPUT)
;  400          0528  2      ! or we are reading from an indirect command file.  If we are reading from
;  401          0529  2      ! the terminal and we are in screen mode, we update the screen at this
;  402          0530  2      ! point so that the user sees his current HELP output before being prompted
;  403          0531  2      ! for another topic or subtopic.
;  404          0532  2      !
;  405          0533  2      IF .DBG$GL_SCREEN_MODE AND
;  406          0534  3         (.CIS_PTR[CIS$B_INPUT_TYPE] EQL CIS_DBG$INPUT)
;  407          0535  2      THEN
;  408          0536  2          DBG$SCR_OUTPUT_SCREEN();
;  409          0537  2
;  410          0538  2
;  411          0539  2      ! If keypad input mode is enabled and we are reading from the terminal, we
;  412          0540  2      ! read a line of input by calling the keypad input routine.
;  413          0541  2      !
;  414          0542  2      IF .DBG$GB_KEYPAD_INPUT AND
;  415          0543  3         (.CIS_PTR[CIS$B_INPUT_TYPE] EQL CIS_DBG$INPUT)
;  416          0544  2      THEN
;  417          0545  3          BEGIN
;  418          0546  3          STATUS = SMG$READ_COMPOSED_LINE(DBG$GL_KEYBOARD_ID,
;  419          0547  3                                          DBG$GL_KEY_TABLE_ID,
;  420          0548  3                                          .GET_STR,
;  421          0549  3                                          .PROMPT_STR,
;  422          0550  3
;  423          0551  3  ! *** Note - the fifth parameter (DEFAULT_STATE) is being removed from
;  424          0552  3  ! *** this routine, according to Steve Lionel. If Steve's change doesn't
;  425          0553  3  ! *** make it this build, however, the "0" must be restored here.
;  426          0554  3  !
;  427          0555  3  !                               0,
;  428          0556  3
;  429          0557  3                                 LENGTH);
;  430          0558  3          GET_STR[DSC$W_LENGTH] = .LENGTH;
;  431          0559  3          IF .STATUS EQL SMG$_EOF THEN STATUS = RMS$_EOF;
;  432          0560  3          RETURN .STATUS;
;  433          0561  2          END;
;  434          0562  2
;  435          0563  2
;  436          0564  2      ! We are either reading from the user's terminal in the normal way using
;  437          0565  2      ! RMS or we are reading from an indirect command file.  Hence we set up
;  438          0566  2      ! the RAB and call RMS to give the prompt and read a line.
;  439          0567  2      !
;  440          0568  2      DBG$GL_INPRAB[RAB$W_USZ] = .GET_STR[DSC$W_LENGTH];
;  441          0569  2      DBG$GL_INPRAB[RAB$L_UBF] = .GET_STR[DSC$A_POINTER];
;  442          0570  2      DBG$GL_INPRAB[RAB$B_PSZ] = .PROMPT_STR[DSC$W_LENGTH];
;  443          0571  2      DBG$GL_INPRAB[RAB$L_PBF] = .PROMPT_STR[DSC$A_POINTER];
;  444          0572  2      STATUS = $GET(RAB = DBG$GL_INPRAB);
;  445          0573  2
;  446          0574  2
;  447          0575  2      ! Put the number of characters read back into the string descriptor.  Then
;  448          0576  2      ! return with the status we got back from $GET.
;  449          0577  2      !
;  450          0578  2      GET_STR[DSC$W_LENGTH] = .DBG$GL_INPRAB[RAB$W_RSZ];
;  451          0579  2      RETURN .STATUS;
;  452          0580  2
;  453          0581  1      END;
```

```
                                                .EXTRN  SYS$GET

                                        000C 00000 INPUT_HELP_LINE:
                                                   .WORD   Save R2,R3                                   ; 0470
                    53 00000000G    00  9E 00002        MOVAB   DBG$GL_INPRAB+32, R3
                    5E              04  C2 00009        SUBL2   #4, SP
                    50 00000000G    00  D0 0000C        MOVL    DBG$GL_CISHEAD, CIS_PTR                  ; 0513
                    02        02    A0  91 00013        CMPB    2(CIS_PTR), #2                           ; 0514
                                    04  12 00017        BNEQ    1$
                    50        08    A0  D0 00019        MOVL    8(CIS_PTR), CIS_PTR                      ; 0516
                    52        02    A0  9A 0001D 1$:    MOVZBL  2(CIS_PTR), R2                           ; 0518
                                    0C  13 00021        BEQL    2$
                    01              52  91 00023        CMPB    R2, #1                                   ; 0519
                                    07  13 00026        BEQL    2$
                              04    BC  B4 00028        CLRW    @GET_STR                                 ; 0522
                    50              01  D0 0002B        MOVL    #1, R0                                   ; 0523
                                        04 0002E        RET
                    0B 00000000G    00  E9 0002F 2$:    BLBC    DBG$GL_SCREEN_MODE, 3$                   ; 0533
                                    52  D5 00036        TSTL    R2                                       ; 0534
                                    07  12 00038        BNEQ    3$
          00000000G 00              00  FB 0003A        CALLS   #0, DBG$SCR_OUTPUT_SCREEN                ; 0536
                    32 00000000G    00  E9 00041 3$:    BLBC    DBG$GB_KEYPAD_INPUT, 4$                  ; 0542
                                    52  D5 00048        TSTL    R2                                       ; 0543
                                    2E  12 0004A        BNEQ    4$
                                    5E  DD 0004C        PUSHL   SP                                       ; 0546
                    7E        04    AC  7D 0004E        MOVQ    GET_STR, -(SP)                           ; 0548
                        00000000G   00  9F 00052        PUSHAB  DBG$GL_KEY_TABLE_ID                      ; 0546
                        00000000G   00  9F 00058        PUSHAB  DBG$GL_KEYBOARD_ID
          00000000G 00              05  FB 0005E        CALLS   #5, SMG$READ_COMPOSED_LINE
                04  BC              6E  B0 00065        MOVW    LENGTH, @GET_STR                         ; 0558
          00000000G 8F              50  D1 00069        CMPL    STATUS, #SMG$_EOF                        ; 0559
                                    2F  12 00070        BNEQ    5$
                    50 0001827A     8F  D0 00072        MOVL    #98938, STATUS
                                        04 00079        RET                                              ; 0560
                    52        04    AC  D0 0007A 4$:    MOVL    GET_STR, R2                              ; 0568
                    63              62  B0 0007E        MOVW    (R2), DBG$GL_INPRAB+32
              04    A3        04    A2  D0 00081        MOVL    4(R2), DBG$GL_INPRAB+36                  ; 0569
                    51        08    AC  D0 00086        MOVL    PROMPT_STR, R1                           ; 0570
              14    A3              61  90 0008A        MOVB    (R1), DBG$GL_INPRAB+52
              10    A3        04    A1  D0 0008E        MOVL    4(R1), DBG$GL_INPRAB+48                  ; 0571
                              E0    A3  9F 00093        PUSHAB  DBG$GL_INPRAB                            ; 0572
          00000000G 00              01  FB 00096        CALLS   #1, SYS$GET
                    62        02    A3  B0 0009D        MOVW    DBG$GL_INPRAB+34, (R2)                   ; 0578
                                        04 000A1 5$:    RET                                              ; 0581

; Routine Size:  162 bytes,    Routine Base:  DBG$CODE + 0178

;  454          0582  1
;  455          0583  0 END ELUDOM


                                                .EXTRN  LIB$SIGNAL
```

```
;                                PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  DBG$CODE                       538  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(0)
;  DBG$PLIT                        44  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        19         0        1000        00:01.8
;  _$255$DUA28:[DEBUG.OBJ]STRUCDEF.L32;1
;                                        32         0         0           7        00:00.1
;  _$255$DUA28:[DEBUG.OBJ]DBGLIB.L32;1
;                                      1545        48         3          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         2         0          22        00:00.3
;  _$255$DUA28:[DEBUG.OBJ]DBGGEN.L32;1
;                                       150         0         0          12        00:00.3




;                            COMMAND QUALIFIERS

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

; Size:          538 code + 44 data bytes
; Run Time:          00:16.5
; Elapsed Time:      00:56.3
; Lines/CPU Min:      2126
; Lexemes/CPU-Min:  12401
; Memory Used:  137 pages
; Compilation Complete
```

DBGNERMSG
LIS

DBGNHELP
LIS

DBGNPARSE
LIS

DBGNEXCTE
LIS

DBGNPNP
LIS