```
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUU         UUU     GGGGGGGGGGGG
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  EEE              BBB         BBB  UUU         UUU  GGG
DDD         DDD  EEEEEEEEEEEE     BBBBBBBBBBBB     UUU         UUU  GGG
DDD         DDD  EEEEEEEEEEEE     BBBBBBBBBBBB     UUU         UUU  GGG
DDD         DDD  EEEEEEEEEEEE     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     UUUUUUUUUUUUUUU     GGGGGGGGG
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUUUUUUUUUUUUUU     GGGGGGGGG
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUUUUUUUUUUUUUU     GGGGGGGGG
```

```
DDDDDDDD    BBBBBBBB      GGGGGGGG  NN      NN    SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT
DDDDDDDD    BBBBBBBB      GGGGGGGG  NN      NN    SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT
DD      DD  BB      BB  GG          NN      NN  SS          EE              TT
DD      DD  BB      BB  GG          NN      NN  SS          EE              TT
DD      DD  BB      BB  GG          NNNN    NN  SS          EE              TT
DD      DD  BB      BB  GG          NNNN    NN  SS          EE              TT
DD      DD  BBBBBBBB    GG          NN  NN  NN    SSSSSS    EEEEEEEE        TT
DD      DD  BBBBBBBB    GG          NN  NN  NN    SSSSSS    EEEEEEEE        TT
DD      DD  BB      BB  GG  GGGGGG  NN    NNNN          SS  EE              TT
DD      DD  BB      BB  GG  GGGGGG  NN    NNNN          SS  EE              TT
DD      DD  BB      BB  GG      GG  NN      NN          SS  EE              TT          ....
DD      DD  BB      BB  GG      GG  NN      NN          SS  EE              TT          ....
DDDDDDDD    BBBBBBBB      GGGGGG    NN      NN  SSSSSSSS    EEEEEEEEEE      TT          ....
DDDDDDDD    BBBBBBBB      GGGGGG    NN      NN  SSSSSSSS    EEEEEEEEEE      TT          ....

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 DBGNSET (IDENT = 'V04-000') =
;     2          0002  1 BEGIN
;     3          0003  1
;     4          0004  1 !
;     5          0005  1 !**************************************************************************
;     6          0006  1 !*                                                                        *
;     7          0007  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
;     8          0008  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
;     9          0009  1 !*  ALL RIGHTS RESERVED.                                                   *
;    10          0010  1 !*                                                                        *
;    11          0011  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    12          0012  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    13          0013  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    14          0014  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    15          0015  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    16          0016  1 !*  TRANSFERRED.                                                           *
;    17          0017  1 !*                                                                        *
;    18          0018  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    19          0019  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    20          0020  1 !*  CORPORATION.                                                           *
;    21          0021  1 !*                                                                        *
;    22          0022  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    23          0023  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;    24          0024  1 !*                                                                        *
;    25          0025  1 !*                                                                        *
;    26          0026  1 !**************************************************************************
;    27          0027  1 !
;    28          0028  1 !
;    29          0029  1 ! FACILITY:     DEBUG
;    30          0030  1 !
;    31          0031  1 ! ABSTRACT:
;    32          0032  1 !
;    33          0033  1 !       This module contains the ATN parse network and the command execution network
;    34          0034  1 !       for the SET verb. These are currently the largest and most complicated
;    35          0035  1 !       networks. Briefly, the parse network constructs a command execution tree
;    36          0036  1 !       which consists of at least a verb node, and possibly one or more noun nodes
;    37          0037  1 !       The execution network accepts this tree as input and decodes the arguments
;    38          0038  1 !       of the various nodes to perform the specified command.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:  VAX/VMS
;    41          0041  1 !
;    42          0042  1 ! AUTHOR:       David Plummer   , CREATION DATE:        3/28/80
;    43          0043  1 !
;    44          0044  1 ! VERSION:      V02.2-001
;    45          0045  1 !
;    46          0046  1 ! MODIFIED BY:
;    47          0047  1 !
;    48          0048  1 !               Richard Title   31-Jul-81
;    49          0049  1 !               Sid Maxwell     07-May-82
;    50          0050  1 !               Brad Becker     13-Sep-83
;    51          0051  1 !
;    52          0052  1 ! REVISION HISTORY:
;    53          0053  1 ! 3.0   31-JUL-81       RT      Added new nouns FLOAT, D_FLOAT, etc.
;    54          0054  1 !                               in the SET TYPE command.
;    55          0055  1 ! 3.1   13-SEP-81       RT      Implemented new SET commands:
;    56          0056  1 !                               SET STEP SOURCE
;    57          0057  1 !                               SET SOURCE dir-list
```

```
;    58          0058  1 !                                        SET MAX_SOURCE_FILES n
;    59          0059  1 !                                        as part of source line display
;    60          0060  1 ! 3.2    9-Oct-81        RT              Implemented SET MARGINS
;    61          0061  1 ! 3.3   13-Oct-81        RT              Implemented SET SEARCH
;    62          0062  1 ! 3B.0  18-Feb-82        PS              Implemented SET DEVELOPER
;    63          0063  1 ! ???   06-May-82        RT              Implemented SET DEFINE
;    64          0064  1 !       07-May-82        SRM             Added hooks for new:
;    65          0065  1 !                                            SET BREAK
;    66          0066  1 !                                            SET TRACE
;    67          0067  1 !                                            SET WATCH
;    68          0068  1 !       09-Aug-82        RT              Added implementation level 3 code for
;    69          0069  1 !                                        SET BREAK, SET TRACE, SET WATCH
;    70          0070  1 !       15-Apr-83        RT              Added SET SCOPE/MODULE
;    71          0071  1 !       22-Apr-83        RT              Added SET MODULE/ALLOCATE
;    72          0072  1 !       15-Jul-83        WC3             Added SET TYPE PACKED:n and SET TYPE/OVERRIDE PACKED:n
;    73          0073  1 !       18-Jul-83        WC3             Added SET TYPE DATE_TIME and SET TYPE/OVERRIDE DATE_TIME
;    74          0074  1 !       17-Aug-83        PS              Added SET MODE G/NOG_FLOAT.
;    75          0075  1 ! 4.0   13-Sep-83        BAB             Added SET KEY.
;    76          0076  1 !
;    77          0077  1
;    78          0078  1 REQUIRE 'SRC$:DBGPROLOG.REQ';
;    79          0212  1
;    80          0213  1 LIBRARY 'LIB$:DBGGEN.L32';
;    81          0214  1
;    82          0215  1 FORWARD ROUTINE
;    83          0216  1     DBG$NPARSE_SET,                     ! SET verb parse network
;    84          0217  1     DBG$NEXECUTE_SET,                   ! SET verb execution network
;    85          0218  1     DBG$NSET_LOG,                       ! Sets up an output log
;    86          0219  1     DBG$NSETUP_LOG,                     ! Opens and connects an output log
;    87          0220  1     DUMMY: NOVALUE;                     ! Dummy routine to foil BLISS
;    88          0221  1                                         !      compiler optimization
```

```
;    90          0222  1 EXTERNAL ROUTINE
;    91          0223  1     DBG$EVENT_SYNTAX,                  ! Event syntax (parser)
;    92          0224  1     DBG$EVENT_SEMANTICS,               ! Event semantics
;    93          0225  1     DBG$NGET_LENGTH,                   ! Returns bit length for a primary
;    94          0226  1     DBG$NGET_DIR_LIST,                 ! Parses a directory list for the
;    95          0227  1                                        !      SET SOURCE dir-list command
;    96          0228  1     DBG$NGET_TRANS_RADIX,              ! Translate radix
;    97          0229  1     DBG$SCR_EXECUTE_DISPLAY_CMD:NOVALUE,! Execute the SET DISPLAY command
;    98          0230  1     DBG$SCR_EXECUTE_SETTERM_CMD:NOVALUE,! Execute the SET TERMINAL command
;    99          0231  1     DBG$SCR_EXECUTE_SETWIND_CMD:NOVALUE,! Execute the SET WINDOW command
;   100          0232  1     DBG$SCR_PARSE_DISPLAY_CMD: NOVALUE,! Parse the SET DISPLAY command
;   101          0233  1     DBG$SCR_PARSE_SETTERM_CMD: NOVALUE,! Parse the SET TERMINAL command
;   102          0234  1     DBG$SCR_PARSE_SETWIND_CMD: NOVALUE,! Parse the SET WINDOW command
;   103          0235  1     DBG$SCR_SCREEN_MODE: NOVALUE,      ! Set or clear screen mode
;   104          0236  1     DBG$STA_GETSOURCEMOD,              ! Finds a Module RST pointer
;   105          0237  1     DBG$SRC_SET_SOURCE: NOVALUE,       ! Implements the SET SOURCE command
;   106          0238  1     DBG$NSAVE_DECIMAL_INTEGER,         ! Saves an inline numeric literal
;   107          0239  1     DBG$NSAVE_INTEGER,                 ! Saves a numeric literal in the current radix
;   108          0240  1     DBG$NSAVE_STRING,                  ! Saves a counted string from input
;   109          0241  1     DBG$NGET_TYPE,                     ! Obtains type of primary
;   110          0242  1     DBG$PRINT: NOVALUE,                ! Format print output
;   111          0243  1     DBG$NEWLINE: NOVALUE,              ! Output a line of print output
;   112          0244  1     DBG$SET_SEARCH_LVL: NOVALUE,       ! Sets search data structure pointer
;   113          0245  1     DBG$SET_STP_LVL: NOVALUE,          ! Sets step data structure pointer
;   114          0246  1     DBG$SET_DEFINE_LVL: NOVALUE,       ! Sets define data structure pointer
;   115          0247  1     DBG$SET_MOD_LVL,                   ! Sets mode structure level
;   116          0248  1     DBG$NPARSE_SET_TASK: NOVALUE,      ! Parses SET TASK command
;   117          0249  1     DBG$NEXECUTE_SET_TASK: NOVALUE,    ! Handles SET TASK command
;   118          0250  1     DBG$RST_SETSCOPE: NOVALUE,         ! Sets user specified scopes
;   119          0251  1     DBG$NPARSE_SCOPE_LIST,             ! Parses a scope list
;   120          0252  1     DBG$NSHOW_MARGINS: NOVALUE,        ! Shows the margins
;   121          0253  1     DBG$SRC_SET_MAX_FILES: NOVALUE,    ! Performs SET MAX command
;   122          0254  1     DBG$NPATHDESC_TO_CS: NOVALUE,      ! Translates a pathname desc to a c.s.
;   123          0255  1     DBG$RST_SETMOD,                    ! Sets a module
;   124          0256  1     DBG$GET_MEMORY,                    ! Allocates a permanent memory block
;   125          0257  1     DBG$REL_MEMORY: NOVALUE,           ! Releases a memory block
;   126          0258  1     DBG$GET_TEMPMEM,                   ! Allocates and lists dynamic storage
;   127          0259  1     DBG$NSAVE_FILESP,                  ! Eats and stores a filespec string
;   128          0260  1     DBG$NMATCH,                        ! Routine to match keywords
;   129          0261  1     DBG$NPARSE_ADDRESS,                ! Interface routine to Address Expression Interpreter
;   130          0262  1     DBG$NNEXT_WORD,                    ! Isolates next word of input for syntax errors
;   131          0263  1     DBG$NOUT_INFO,                     ! Outputs an informational message
;   132          0264  1     DBG$NMAKE_ARG_VECT,                ! Constructs a message argument vector
;   133          0265  1     DBG$NSYNTAX_ERROR,                 ! Constructs a syntax error message
;   134          0266  1     DBG$SET_LANG,                      ! Version 2 routine to change language
;   135          0267  1     DBG$READ_KEY_INFO,                 ! Reads in the state-name string for SET KEY
;   136          0268  1     SMG$SET_DEFAULT_STATE,             ! Sets the default keypad state
;   137          0269  1     SMG$SET_KEYPAD_MODE;               ! Sets terminal to numeric/applicat. mode
;   138          0270  1
;   139          0271  1 EXTERNAL
;   140          0272  1     DBG$GB_KEYPAD_INPUT: BYTE,         ! TRUE if keypad input is enabled
;   141          0273  1     DBG$GL_KEY_TABLE_ID,               ! Used in SET KEY
;   142          0274  1     DBG$GL_KEYBOARD_ID,                ! Used in SET MODE KEY
;   143          0275  1     DBG$GB_RADIX: VECTOR[3, BYTE],     ! Override and default radix
;   144          0276  1     DBG$GL_GBLTYP,                     ! Override type
;   145          0277  1     DBG$GW_GBLLNGTH: WORD,             ! Override length
;   146          0278  1     DBG$GL_DFLTTYP,                    ! Default type
```

```
;  147          0279  1     DBG$GW_DFLTLENG: WORD,                  ! Default length
;  148          0280  1     DBG$GB_LANGUAGE: BYTE,                  ! Current language
;  149          0281  1     DBG$GB_SEARCH_PTR: REF VECTOR[,BYTE],   ! Search structure
;  150          0282  1     DBG$GB_MOD_PTR: REF VECTOR [,BYTE],     ! Mode structure
;  151          0283  1     DBG$GB_DEFINE_PTR: REF VECTOR [,BYTE],  ! Define structure
;  152          0284  1     DBG$RUNFRAME: BLOCK [,BYTE],            ! Current runframe
;  153          0285  1     DBG$GB_RESIGNAL: BYTE,                  ! If TRUE, then resignal exeptions
;  154          0286  1     DBG$GL_LOG_BUF,                         ! Old debugger pointer to counted string filespec
;  155          0287  1     DBG$GL_CONTEXT: BITVECTOR,              ! Old debugger context switches
;  156          0288  1     DBG$GB_DEF_OUT: VECTOR [,BYTE],         ! Old debugger output control vector
;  157          0289  1     DBG$GL_LOGFAB: BLOCK [,BYTE],           ! FAB for log file
;  158          0290  1     DBG$GL_LOGRAB: BLOCK [,BYTE],           ! RAB for log file
;  159          0291  1     DBG$GL_LOGNAM: REF $NAM_DECL,           ! NAM block for file spec
;  160          0292  1     DBG$GB_LOGFSR: REF VECTOR [,BYTE],      ! Resultant LOG filespec buffer
;  161          0293  1     DBG$GB_LOGFSE: REF VECTOR [,BYTE],      ! Expanded LOG filespec buffer
;  162          0294  1     DBG$GL_SCREEN_LOG,                      ! Flag set to activate screen logging
;  163          0295  1     DBG$GB_STP_PTR:                         ! Pointer to current stepping descriptor
;  164          0296  1         REF EVENT$STEPPING_DESCRIPTOR,      !
;  165          0297  1     DBG$SRC_LEFT_MARGIN,                    ! Left margin for source display
;  166          0298  1     DBG$SRC_RIGHT_MARGIN,                   ! Right margin for source display
;  167          0299  1     DBG$GL_ORIG_COMMAND_PTR,                ! Pointer to original command string
;  168          0300  1     DBG$GL_UPCASE_COMMAND_PTR: VECTOR[2];   !
;  169          0301  1                                             ! Pointers to start and end
;  170          0302  1                                             !    of current command string
;  171          0303  1
;  172          0304  1
;  173          0305  1
;  174          0306  1
;  175          0307  1 ! These are composite verb literals used in parsing and execution.
;  176          0308  1 !
;  177          0309  1 ! Note - you may cause yourself problems if you try to renumber these,
;  178          0310  1 ! because the EVENT$K_SET_XXX definitions in DBGLIB.REQ must correspond
;  179          0311  1 ! to the numbers here.
;  180          0312  1 !
;  181          0313  1 LITERAL
;  182          0314  1     SET_MINIMUM          = 1,
;  183          0315  1     SET_BREAK            = 1,          ! Also defined as EVENT$K_SET_BREAK
;  184          0316  1     SET_BREAK_DO         = 2,
;  185          0317  1     SET_EXCEPTION_BREAK  = 3,          ! Also defined as EVENT$K_SET_BREAK_EXC
;  186          0318  1     SET_LANGUAGE         = 4,
;  187          0319  1     SET_LOG              = 5,
;  188          0320  1     SET_MODE             = 6,
;  189          0321  1     SET_MODULE           = 7,
;  190          0322  1     SET_MODULE_ALL       = 8,
;  191          0323  1     SET_OUTPUT           = 9,
;  192          0324  1     SET_RADIX            = 27,
;  193          0325  1     SET_RADIX_OVERRIDE   = 28,
;  194          0326  1     SET_SCOPE            = 10,
;  195          0327  1     SET_STEP             = 11,
;  196          0328  1     SET_TYPE             = 12,
;  197          0329  1     SET_TYPE_OVERRIDE    = 13,
;  198          0330  1     SET_TRACE            = 14,         ! Also defined as EVENT$K_SET_TRACE
;  199          0331  1     SET_TRACE_CALLS      = 15,
;  200          0332  1     SET_TRACE_BRANCH     = 16,
;  201          0333  1     SET_WATCH            = 17,         ! Also defined as EVENT$K_SET_WATCH
;  202          0334  1     SET_SOURCE           = 18,
;  203          0335  1     SET_MAX_SOURCE_FILES= 19,
```

```
;  204          0336  1      SET_MARGINS          = 20,
;  205          0337  1      SET_SEARCH           = 21,
;  206          0338  1      SET_DEVELOPER        = 22,
;  207          0339  1      SET_DEFINE           = 23,
;  208          0340  1      SET_DISPLAY          = 24,
;  209          0341  1      SET_TERMINAL         = 25,
;  210          0342  1      SET_WINDOW           = 26,
;  211          0343  1      SET_KEY              = 29,
;  212          0344  1      SET_TASK             = 30,
;  213          0345  1      SET_MAXIMUM          = 30;
;  214          0346  1
;  215          0347  1  ! These are the legal mode literals for the SET MODE command.
;  216          0348  1  !
;  217          0349  1  LITERAL
;  218          0350  1      MODE_LOWEST          = 1,
;  219          0351  1      MODE_BINARY          = 1,
;  220          0352  1      MODE_OCTAL           = 2,
;  221          0353  1      MODE_DECIMAL         = 3,
;  222          0354  1      MODE_HEX             = 4,
;  223          0355  1      MODE_SYMBLS          = 5,
;  224          0356  1      MODE_NOSYMBLS        = 6,
;  225          0357  1      MODE_DEFAULT         = 7,
;  226          0358  1      MODE_SCREEN          = 8,
;  227          0359  1      MODE_NOSCREEN        = 9,
;  228          0360  1      MODE_KEYPAD          = 10,
;  229          0361  1      MODE_NOKEYPAD        = 11,
;  230          0362  1      MODE_G_FLOAT         = 12,
;  231          0363  1      MODE_NOG_FLOAT       = 13,
;  232          0364  1      MODE_HIGHEST         = 13;
;  233          0365  1
;  234          0366  1  GLOBAL
;  235          0367  1      DBG$GL_DEVELOPER:                   ! Special mode bits for developers
;  236          0368  1          BITVECTOR[32] INITIAL(0);       !
;  237          0369  1
;  238          0370  1  MACRO
;  239          0371  1
;  240          0372  1          ! OUTPUT_RMS_ERROR reports RMS errors caused by $CREATE, $OPEN, and $CONNECT
;  241          0373  1          !
;  242        M 0374  1          OUTPUT_RMS_ERROR (BLOCK_TYPE) =
;  243        M 0375  1              BEGIN
;  244        M 0376  1                  LOCAL
;  245        M 0377  1                      MSG_DESC : REF dbg$stg_desc,
;  246        M 0378  1                      STS,
;  247        M 0379  1                      STV;
;  248        M 0380  1
;  249        M 0381  1                  msg_desc = dbg$get_tempmem (2);
;  250        M 0382  1
;  251        M 0383  1                  build_error_desc (msg_desc);
;  252        M 0384  1                  IF block_type EQL 1
;  253        M 0385  1                  THEN
;  254        M 0386  1                      BEGIN
;  255        M 0387  1                          sts = .dbg$gl_logfab [fab$l_sts];
;  256        M 0388  1                          stv = .dbg$gl_logfab [fab$l_stv];
;  257        M 0389  1                      END
;  258        M 0390  1                  ELSE
;  259        M 0391  1                      BEGIN
;  260        M 0392  1                          sts = .dbg$gl_lograb [rab$l_sts];
```

```
;  261        M 0393  1                    stv = .dbg$gl_lograb [rab$l_stv];
;  262        M 0394  1                END;
;  263        M 0395  1
;  264        M 0396  1            .message_vect = dbg$nmake_arg_vect (shr$_openout+dbg_fac_code,
;  265        M 0397  1                                                1, .msg_desc, .sts, .stv);
;  266        M 0398  1
;  267        M 0399  1            RETURN sts$k_severe;
;  268        M 0400  1
;  269          0401  1        END %,
;  270          0402  1
;  271          0403  1    ! Build_error_desc creates a descriptor for RMS error reporting
;  272          0404  1
;  273        M 0405  1    BUILD_ERROR_DESC (MSG_DESC) =
;  274        M 0406  1        IF .dbg$gl_log_buf NEQ 0
;  275        M 0407  1        THEN
;  276        M 0408  1            BEGIN
;  277        M 0409  1                msg_desc [dsc$w_length] = .dbg$gl_logfab [fab$b_fns];
;  278        M 0410  1                msg_desc [dsc$a_pointer] = .dbg$gl_logfab [fab$l_fna];
;  279        M 0411  1            END
;  280        M 0412  1        ELSE
;  281        M 0413  1            BEGIN
;  282        M 0414  1                msg_desc [dsc$w_length] = .dbg$gl_logfab [fab$b_dns];
;  283        M 0415  1                msg_desc [dsc$a_pointer] = .dbg$gl_logfab [fab$l_dna];
;  284          0416  1            END %,
;  285          0417  1
;  286          0418  1    ! Restore_nam restores an  RMS NAM block
;  287          0419  1
;  288        M 0420  1    RESTORE_NAM =
;  289        M 0421  1
;  290        M 0422  1        BEGIN
;  291        M 0423  1
;  292        M 0424  1        LOCAL
;  293        M 0425  1            ERR,
;  294        M 0426  1            TYPE_B;
;  295        M 0427  1
;  296        M 0428  1        ! Release storage for the aborted log file
;  297        M 0429  1
;  298        M 0430  1        dbg$rel_memory (.temp_nam);
;  299        M 0431  1        dbg$rel_memory (.temp_fsr);
;  300        M 0432  1        dbg$rel_memory (.temp_fse);
;  301        M 0433  1
;  302        M 0434  1        dbg$gl_lognam = .old_nam_ptr;
;  303        M 0435  1        old_nam_ptr = 0;
;  304        M 0436  1        dbg$gl_logfab [fab$l_nam] = .dbg$gl_lognam;
;  305        M 0437  1        dbg$gl_logfab [fab$l_fna] = .fna;
;  306        M 0438  1        dbg$gl_logfab [fab$b_fns] = .fns;
;  307        M 04 ?  1
;  308        M 0440  1        dbg$gl_logfab [fab$v_nam] = 1;
;  309        M 0441  1        dbg$gl_logfab [fab$v_cif] = 1;
;  310        M 0442  1        dbg$gl_logfab [fab$v_mxv] = 0;
;  311        M 0443  1
;  312        M 0444  1        err = dbg$nsetup_log (type_b);
;  313        M 0445  1
;  314        M 0446  1        IF NOT .err
;  315        M 0447  1        THEN output_rms_error (.type_b);
;  316        M 0448  1
;  317        M 0449  1        dbg$gb_def_out [out_log] = .log_temp;
```

```
;  318        M 0450  1
;  319          0451  1              END %;
```

```
;  321          0452  1 GLOBAL ROUTINE DBG$NPARSE_SET (INPUT_DESC, VERB_NODE, MESSAGE_VECT) =
;  322          0453  1
;  323          0454  1 !++
;  324          0455  1 ! FUNCTIONAL DESCRIPTION:
;  325          0456  1 !
;  326          0457  1 !       This routine comprises the ATN parse network for the SET verb. The network
;  327          0458  1 !       recognizes various keywords and contructs appropriate noun nodes which it
;  328          0459  1 !       links to the verb node. Many composite verbs are recognized in this network
;  329          0460  1 !       as well. The value of these composite verbs are placed in the verb node (verb composite)
;  330          0461  1 !       field.
;  331          0462  1 !
;  332          0463  1 ! FORMAL PARAMETERS:
;  333          0464  1 !
;  334          0465  1 !       INPUT_DESC -            The present command line input descriptor
;  335          0466  1 !
;  336          0467  1 !       VERB_NODE _             The first node in the command execution tree.
;  337          0468  1 !                               Value field has already been filled in by
;  338          0469  1 !                               DBG$NPARSE_CMD.
;  339          0470  1 !
;  340          0471  1 !       MESSAGE_VECT -          The address of a longword to contain the address
;  341          0472  1 !                               of a message argument vector
;  342          0473  1 !
;  343          0474  1 ! IMPLICIT INPUTS:
;  344          0475  1 !
;  345          0476  1 !       NONE
;  346          0477  1 !
;  347          0478  1 ! IMPLICIT OUTPUTS:
;  348          0479  1 !
;  349          0480  1 !       The command execution tree corresponding to the input command is constructed.
;  350          0481  1 !
;  351          0482  1 ! ROUTINE VALUE:
;  352          0483  1 !
;  353          0484  1 !       An unsigned integer longword completion code
;  354          0485  1 !
;  355          0486  1 ! COMPLETION CODES:
;  356          0487  1 !
;  357          0488  1 !       STS$K_SEVERE (4)        - Unsuccessful parse
;  358          0489  1 !
;  359          0490  1 !       STS$K_SUCCESS (1)       - Successful parse
;  360          0491  1 !
;  361          0492  1 ! SIDE EFFECTS:
;  362          0493  1 !
;  363          0494  1 !       NONE
;  364          0495  1 !
;  365          0496  1 !--
;  366          0497  1
;  367          0498  2     BEGIN
;  368          0499  2
;  369          0500  2     ! Define strings used at this level.
;  370          0501  2     !
;  371          0502  2     BIND
;  372          0503  2         DBG$CS_BREAK            = UPLIT BYTE (%ASCIC 'BREAK'),
;  373          0504  2         DBG$CS_DEFINE           = UPLIT BYTE (%ASCIC 'DEFINE'),
;  374          0505  2         DBG$CS_DEVELOPER        = UPLIT BYTE (%ASCIC 'DEVELOPER'),
;  375          0506  2         DBG$CS_DISPLAY          = UPLIT BYTE (%ASCIC 'DISPLAY'),
;  376          0507  2         DBG$CS_EXCEPTION        = UPLIT BYTE (%ASCIC 'EXCEPTION'),
;  377          0508  2         DBG$CS_LANGUAGE         = UPLIT BYTE (%ASCIC 'LANGUAGE'),
```

```
;  378          0509  2          DBG$CS_LOG              = UPLIT BYTE (%ASCIC 'LOG'),
;  379          0510  2          DBG$CS_KEY              = UPLIT BYTE (%ASCIC 'KEY'),
;  380          0511  2          DBG$CS_MARGINS          = UPLIT BYTE (%ASCIC 'MARGINS'),
;  381          0512  2          DBG$CS_MAX_SOURCE_FILES = UPLIT BYTE (%ASCIC 'MAX_SOURCE_FILES'),
;  382          0513  2          DBG$CS_MODE             = UPLIT BYTE (%ASCIC 'MODE'),
;  383          0514  2          DBG$CS_MODULE           = UPLIT BYTE (%ASCIC 'MODULE'),
;  384          0515  2          DBG$CS_OUTPUT           = UPLIT BYTE (%ASCIC 'OUTPUT'),
;  385          0516  2          DBG$CS_RADIX            = UPLIT BYTE (%ASCIC 'RADIX'),
;  386          0517  2          DBG$CS_SCOPE            = UPLIT BYTE (%ASCIC 'SCOPE'),
;  387          0518  2          DBG$CS_SEARCH           = UPLIT BYTE (%ASCIC 'SEARCH'),
;  388          0519  2          DBG$CS_SOURCE           = UPLIT BYTE (%ASCIC 'SOURCE'),
;  389          0520  2          DBG$CS_STEP             = UPLIT BYTE (%ASCIC 'STEP'),
;  390          0521  2          DBG$CS_TASK             = UPLIT BYTE (%ASCIC 'TASK'),
;  391          0522  2          DBG$CS_TERMINAL         = UPLIT BYTE (%ASCIC 'TERMINAL'),
;  392          0523  2          DBG$CS_TRACE            = UPLIT BYTE (%ASCIC 'TRACE'),
;  393          0524  2          DBG$CS_TYPE             = UPLIT BYTE (%ASCIC 'TYPE'),
;  394          0525  2          DBG$CS_WATCH            = UPLIT BYTE (%ASCIC 'WATCH'),
;  395          0526  2          DBG$CS_WINDOW           = UPLIT BYTE (%ASCIC 'WINDOW'),
;  396          0527  2          DBG$CS_COLON            = UPLIT BYTE (%ASCIC ':'),
;  397          0528  2          DBG$CS_COMMA            = UPLIT BYTE (%ASCIC ','),
;  398          0529  2          DBG$CS_EQUAL            = UPLIT BYTE (%ASCIC '='),
;  399          0530  2          DBG$CS_SLASH            = UPLIT BYTE (%ASCIC '/'),
;  400          0531  2          DBG$CS_CR               = UPLIT BYTE (1, DBG$K_CAR_RETURN);
;  401          0532  2
;  402          0533  2      MAP
;  403          0534  2          VERB_NODE: REF DBG$VERB_NODE;   ! Pointer to command Verb Node
;  404          0535  2
;  405          0536  2      LOCAL
;  406          0537  2          NOUN_NODE: REF DBG$NOUN_NODE;   ! Pointer to current Noun Node
;  407          0538  2
;  408          0539  2
;  409          0540  2
;  410          0541  2      ! Construct the noun node.
;  411          0542  2      !
;  412          0543  2      NOUN_NODE = DBG$GET_TEMPMEM (DBG$K_NOUN_NODE_SIZE);
;  413          0544  2
;  414          0545  2
;  415          0546  2      ! Link the noun node to the verb node.
;  416          0547  2      !
;  417          0548  2      VERB_NODE [DBG$L_VERB_OBJECT_PTR] = .NOUN_NODE;
;  418          0549  2
;  419          0550  2
;  420          0551  2      ! Recognize keyword and transfer control to subnetwork if appropriate.
;  421          0552  2      !
;  422          0553  2      SELECTONE TRUE OF
;  423          0554  2          SET
;  424          0555  2
;  425          0556  2
;  426          0557  2          ! Parse the SET BREAK command.
;  427          0558  2          !
;  428          0559  2          [DBG$NMATCH (.INPUT_DESC, DBG$CS_BREAK, 1)]:
;  429          0560  3              BEGIN
;  430          0561  3              VERB_NODE [DBG$B_VERB_COMPOSITE] = EVENT$K_SET_BREAK;
;  431          0562  3              RETURN DBG$EVENT_SYNTAX (.INPUT_DESC,
;  432          0563  3                                       .VERB_NODE,
;  433          0564  3                                       .MESSAGE_VECT
;  434          0565  3                                      );
```

```
;  435          0566  2          END;
;  436          0567  2
;  437          0568  2
;  438          0569  2      [dbg$nmatch (.input_desc, dbg$cs_define, 1)] :          ! Set Define
;  439          0570  2          BEGIN
;  440          0571  2
;  441          0572  2          ! Define the keywords that are legal after SET DEFINE
;  442          0573  2          !
;  443          0574  2          BIND
;  444          0575  2              dbg$cs_address  = UPLIT BYTE (7, 'ADDRESS'),
;  445          0576  2              dbg$cs_command  = UPLIT BYTE (7, 'COMMAND'),
;  446          0577  2              dbg$cs_procedure= UPLIT BYTE (9, 'PROCEDURE'),
;  447          0578  2              dbg$cs_string   = UPLIT BYTE (6, 'STRING'),
;  448          0579  2              dbg$cs_value    = UPLIT BYTE (5, 'VALUE');
;  449          0580  2
;  450          0581  2          verb_node [dbg$b_verb_composite] = set_define;
;  451          0582  2
;  452          0583  2          ! Pick up one keyword.
;  453          0584  2          ! *** Two of the five are commented out because they are
;  454          0585  2          ! *** not yet supported.
;  455          0586  2          !
;  456          0587  2          SELECTONE TRUE OF
;  457          0588  2              SET
;  458          0589  2
;  459          0590  2              [dbg$nmatch (.input_desc, dbg$cs_address, 1)]:
;  460          0591  2                  noun_node [dbg$l_noun_value] = define_address;
;  461          0592  2
;  462          0593  2              [dbg$nmatch (.input_desc, dbg$cs_command, 1)]:
;  463          0594  2                  noun_node [dbg$l_noun_value] = define_command;
;  464          0595  2
;  465          0596  3              ! [dbg$nmatch (.input_desc, dbg$cs_procedure, 1)]:
;  466          0597  3              !     noun_node [dbg$l_noun_value] = define_procedure;
;  467          0598  3              !
;  468          0599  3              ! [dbg$nmatch (.input_desc, dbg$cs_string, 1)]:
;  469          0600  3              !     noun_node [dbg$l_noun_value] = define_string;
;  470          0601  3              !
;  471          0602  3              [dbg$nmatch (.input_desc, dbg$cs_value, 1)]:
;  472          0603  3                  noun_node [dbg$l_noun_value] = define_value;
;  473          0604  3
;  474          0605  3              [OTHERWISE]:
;  475          0606  4                  BEGIN
;  476          0607  4                  IF dbg$nmatch (.input_desc, dbg$cs_cr, 1)
;  477          0608  4                  THEN
3 ;  478          0609  4                      .message_vect = dbg$nmake_arg_vect (dbg$_needmore)
;  479          0610  4                  ELSE
;  480          0611  4                      .message_vect =
5 ;  481          0612  4                        dbg$nsyntax_error (
;  482          0613  4                          dbg$nnext_word (.input_desc));
;  483          0614  4                  RETURN sts$k_severe;
3 ;  484          0615  3                  END;
;  485          0616  3
4 ;  486          0617  3              TES;
;  487          0618  2          END;
;  488          0619  2
;  489          0620  2      [dbg$nmatch (.input_desc, dbg$cs_developer, 9)] :       ! Set Developer
;  490          0621  3          BEGIN
5 ;  491          0622  3          LOCAL
```

```
;  492          0623  3              link;
;  493          0624  3
;  494          0625  3          verb_node [dbg$b_verb_composite] = set_developer;
;  495          0626  3          link = verb_node[dbg$l_verb_object_ptr];
;  496          0627  3          IF NOT dbg$nmatch(.input_desc, dbg$cs_cr, 1)
;  497          0628  3          THEN
;  498          0629  4              BEGIN
;  499          0630  4              WHILE true DO
;  500          0631  5                  BEGIN
;  501          0632  5                  IF NOT dbg$nsave_integer(.input_desc, noun_node[dbg$l_noun_value])
;  502          0633  5                  THEN
;  503          0634  5                      RETURN sts$k_severe;
;  504          0635  5
;  505          0636  5                  IF (.noun_node[dbg$l_noun_value] LSS 0) OR
;  506          0637  6                     (.noun_node[dbg$l_noun_value] GTR 31)
;  507          0638  5                  THEN
;  508          0639  6                      BEGIN
;  509          0640  6                      .message_vect = dbg$nmake_arg_vect(dbg$_bitrange);
;  510          0641  6                      RETURN sts$k_severe;
;  511          0642  5                      END;
;  512          0643  5
;  513          0644  5                  link = noun_node[dbg$l_noun_link];
;  514          0645  5                  IF NOT dbg$nmatch(.input_desc, dbg$cs_comma, 1)
;  515          0646  5                  THEN
;  516          0647  6                      BEGIN
;  517          0648  6                      IF NOT dbg$nmatch(.input_desc, dbg$cs_cr, 1)
;  518          0649  6                      THEN
;  519          0650  7                          BEGIN
;  520          0651  7                          .message_vect = dbg$nsyntax_error(dbg$nnext_word(.input_desc));
;  521          0652  7                          RETURN sts$k_severe;
;  522          0653  7                          END
;  523          0654  7
;  524          0655  6                      ELSE
;  525          0656  6                          EXITLOOP;
;  526          0657  6
;  527          0658  5                      END;
;  528          0659  5
;  529          0660  5                  noun_node = dbg$get_tempmem (dbg$k_noun_node_size);
;  530          0661  5                  .link = .noun_node;
;  531          0662  4                  END;                            ! End of WHILE loop.
;  532          0663  4
;  533          0664  4              END                                 ! End of SET DEVELOPER 0,1...
;  534          0665  4
;  535          0666  3              ELSE
;  536          0667  4                  BEGIN
;  537          0668  4                  .message_vect = dbg$nmake_arg_vect(dbg$_needmore);
;  538          0669  4                  RETURN sts$k_severe;
;  539          0670  3                  END;
;  540          0671  3
;  541          0672  3              .link = 0;
;  542          0673  3
;  543          0674  2              END;
;  544          0675  2
;  545          0676  2
;  546          0677  2          ! Parse the SET DISPLAY command.
;  547          0678  2          !
;  548          0679  2          [DBG$NMATCH(.INPUT_DESC, DBG$CS_DISPLAY, 3)]:
```

```
;  549          0680  3              BEGIN
;  550          0681  3              VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_DISPLAY;
;  551          0682  3              DBG$SCR_PARSE_DISPLAY_CMD(.INPUT_DESC, TRUE, .VERB_NODE);
;  552          0683  2              END;
;  553          0684  2
;  554          0685  2
;  555          0686  2          ! Parse the SET EXCEPTION BREAK command.
;  556          0687  2          !
;  557          0688  2          [DBG$NMATCH (.INPUT_DESC, DBG$CS_EXCEPTION, 1)]:
;  558          0689  3              BEGIN
;  559          0690  3
;  560          0691  3              ! We are looking for SET EXCEPTION BREAK
;  561          0692  3
;  562          0693  3              IF NOT dbg$nmatch (.input_desc, dbg$cs_break, 1)
;  563          0694  3              THEN
;  564          0695  4                  BEGIN
;  565          0696  4                  .message_vect =
;  566          0697  5                  ( IF dbg$nmatch (.input_desc, dbg$cs_cr, 1)
;  567          0698  5                    THEN
;  568          0699  5                        dbg$nmake_arg_vect (dbg$_needmore)
;  569          0700  5                    ELSE
;  570          0701  4                        dbg$nsyntax_error (dbg$nnext_word (.input_desc)));
;  571          0702  4                  RETURN sts$k_severe;
;  572          0703  3                  END;
;  573          0704  3
;  574          0705  3              ! Set the verb composite
;  575          0706  3
;  576          0707  3              verb_node [dbg$b_verb_composite] = set_exception_break;
;  577          0708  3
;  578          0709  3
;  579          0710  3              ! Reset the noun and adverb pointers.
;  580          0711  3              !
;  581          0712  3              verb_node [dbg$l_verb_object_ptr] = 0;
;  582          0713  3              verb_node [dbg$l_verb_adverb_ptr] = 0;
;  583          0714  2              END;
;  584          0715  2
;  585          0716  2          ! Parse SET KEY
;  586          0717  2          !
;  587          0718  2
;  588          0719  2          [dbg$nmatch (.input_desc, dbg$cs_key, 1)]:
;  589          0720  3              BEGIN
;  590          0721  3
;  591          0722  3              MAP
;  592          0723  3                  input_desc              : REF BLOCK [,BYTE];
;  593          0724  3
;  594          0725  3              BIND
;  595          0726  3                  dbg$cs_state            = UPLIT BYTE (5, 'STATE'),
;  596          0727  3                  dbg$cs_nostate          = UPLIT BYTE (7, 'NOSTATE'),
;  597          0728  3                  dbg$cs_nolog            = UPLIT BYTE (5, 'NOLOG');
;  598          0729  3
;  599          0730  3              LOCAL
;  600          0731  3                  temp_key_desc           : REF dbg$stg_desc,
;  601          0732  3                  status;
;  602          0733  3
;  603          0734  3
;  604          0735  3              IF NOT .dbg$gb_keypad_input
;  605          0736  3              THEN
```

```
;  606     0737  3          SIGNAL(dbg$_nokeydef);
;  607     0738  3
;  608     0739  3      verb_node [dbg$b_verb_composite] = set_key;
;  609     0740  3
;  610     0741  3      ! Initialize the noun node to point to a descriptor of the
;  611     0742  3      ! current state.
;  612     0743  3      !
;  613     0744  3      temp_key_desc = dbg$get_tempmem(2);
;  614     0745  3      temp_key_desc [dsc$w_length] = 0;
;  615     0746  3      temp_key_desc [dsc$b_dtype]  = dsc$k_dtype_t;
;  616     0747  3      temp_key_desc [dsc$b_class]  = dsc$k_class_d;
;  617     0748  3      temp_key_desc [dsc$a_pointer]= 0;
;  618     0749  3
;  619     0750  3      smg$set_default_state (dbg$gl_key_table_id, 0, .temp_key_desc);
;  620     0751  3
;  621     0752  3      ! Let this field indicate whether to output a log of the SET KEY command
;  622     0753  3      !
;  623     0754  3      verb_node [dbg$l_verb_adverb_ptr] = 1;
;  624     0755  3
;  625     0756  3      noun_node [dbg$l_noun_value] = .temp_key_desc;
;  626     0757  3
;  627     0758  3      WHILE (NOT dbg$nmatch (.input_desc, dbg$cs_cr, 1)) AND
;  628     0759  3            (.input_desc [dsc$w_length] GTR 0) DO
;  629     0760  3
;  630     0761  3          IF dbg$nmatch (.input_desc, dbg$cs_slash,1)
;  631     0762  3          THEN
;  632     0763  3              SELECTONE TRUE OF
;  633     0764  3                  SET
;  634     0765  3
;  635     0766  3                  [dbg$nmatch(.input_desc, dbg$cs_nolog, 3)] :
;  636     0767  4                      BEGIN
;  637     0768  4                      verb_node [dbg$l_verb_adverb_ptr] = 0;
;  638     0769  3                      END;
;  639     0770  3
;  640     0771  3                  [dbg$nmatch(.input_desc, dbg$cs_nostate, 3)] :
;  641     0772  4                      BEGIN
;  642     0773  4                      ! In this case the noun node field that points to
;  643     0774  4                      ! the state name is reset anyway, as above.
;  644     0775  4                      !
;  645     0776  4                      smg$set_default_state(dbg$gl_key_table_id, 0, .temp_key_desc);
;  646     0777  4                      noun_node [dbg$l_noun_value] = .temp_key_desc;
;  647     0778  3                      END;
;  648     0779  3
;  649     0780  3                  [dbg$nmatch(.input_desc, dbg$cs_log, 1)] :
;  650     0781  4                      BEGIN
;  651     0782  4                      verb_node [dbg$l_verb_adverb_ptr] = 1;
;  652     0783  3                      END;
;  653     0784  3
;  654     0785  3                  [dbg$nmatch(.input_desc, dbg$cs_state, 1)] :
;  655     0786  4                      BEGIN
;  656     0787  4                      ! Check for =
;  657     0788  4                      !
;  658     0789  4                      IF NOT dbg$nmatch(.input_desc, dbg$cs_equal, 1)
;  659     0790  4                      THEN
;  660     0791  5                          BEGIN
;  661     0792  5                          IF dbg$nmatch(.input_desc, dbg$cs_cr, 1)
;  662     0793  5                          THEN
```

```
;  663          0794  5                        .message_vect = dbg$nmake_arg_vect (dbg$_needmore)
;  664          0795  5                    ELSE
;  665          0796  5                        .message_vect = dbg$nsyntax_error (dbg$nnext_word(.input_desc));
;  666          0797  5                    RETURN sts$k_severe;
;  667          0798  4                    END;
;  668          0799  4
;  669          0800  4                temp_key_desc [dsc$a_pointer] = 0;
;  670          0801  4                temp_key_desc [dsc$w_length]  = 0;
;  671          0802  4                status = dbg$read_key_info (.input_desc,
;  672          0803  4                                           .temp_key_desc,
;  673          0804  4                                           .message_vect);
;  674          0805  4                IF NOT .status
;  675          0806  4                THEN
;  676          0807  4                    RETURN .status;
;  677          0808  3                END;
;  678          0809  3
;  679          0810  3            [OTHERWISE] :
;  680          0811  4                BEGIN
;  681          0812  4                IF dbg$nmatch(.input_desc, dbg$cs_cr, 1)
;  682          0813  4                THEN
;  683          0814  4                    .message_vect = dbg$nmake_arg_vect (dbg$_needmore)
;  684          0815  4                ELSE
;  685          0816  4                    .message_vect = dbg$nsyntax_error (dbg$nnext_word(.input_desc));
;  686          0817  4                RETURN sts$k_severe;
;  687          0818  3                END;
;  688          0819  3            TES
;  689          0820  3        ELSE
;  690          0821  4            BEGIN
;  691          0822  4            .message_vect = dbg$nsyntax_error (dbg$nnext_word(.input_desc));
;  692          0823  4            RETURN sts$k_severe;
;  693          0824  3            END;
;  694          0825  2        END;                ! End of SET KEY parse
;  695          0826  2
;  696          0827  2
;  697          0828  2        [dbg$nmatch (.input_desc, dbg$cs_language, 2)]:
;  698          0829  3            BEGIN
;  699          0830  3
;  700          0831  3            BIND
;  701          0832  3                DBG$CS_MACRO    = UPLIT BYTE(%ASCIC 'MACRO'),
;  702          0833  3                DBG$CS_FORTRAN  = UPLIT BYTE(%ASCIC 'FORTRAN'),
;  703          0834  3                DBG$CS_BLISS    = UPLIT BYTE(%ASCIC 'BLISS'),
;  704          0835  3                DBG$CS_COBOL    = UPLIT BYTE(%ASCIC 'COBOL'),
;  705          0836  3                DBG$CS_BASIC    = UPLIT BYTE(%ASCIC 'BASIC'),
;  706          0837  3                DBG$CS_PLI      = UPLIT BYTE(%ASCIC 'PLI'),
;  707          0838  3                DBG$CS_PASCAL   = UPLIT BYTE(%ASCIC 'PASCAL'),
;  708          0839  3                DBG$CS_C        = UPLIT BYTE(%ASCIC 'C'),
;  709          0840  3                DBG$CS_RPG      = UPLIT BYTE(%ASCIC 'RPG'),
;  710          0841  3                DBG$CS_ADA      = UPLIT BYTE(%ASCIC 'ADA'),
;  711          0842  3                DBG$CS_UNKNOWN  = UPLIT BYTE(%ASCIC 'UNKNOWN');
;  712          0843  3
;  713          0844  3
;  714          0845  3            VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_LANGUAGE;
;  715          0846  3
;  716          0847  3            SELECTONE TRUE OF
;  717          0848  3                SET
;  718          0849  3
;  719          0850  3                [DBG$NMATCH (.INPUT_DESC, DBG$CS_MACRO, 2)] :
```

```
;  720     0851  1                NOUN_NODE [DBG$L_NOUN_VALUE] = DBG$K_MACRO;
;  721     0852  1
;  722     0853  1            [DBG$NMATCH (.INPUT_DESC, DBG$CS_FORTRAN, 2)] :
;  723     0854  1                NOUN_NODE [DBG$[_NOUN_VALUE] = DBG$K_FORTRAN;
;  724     0855  1
;  725     0856  1            [DBG$NMATCH (.INPUT_DESC, DBG$CS_BLISS, 2)] :
;  726     0857  1                NOUN_NODE [DBG$[_NOUN_VALUE] = DBG$K_BLISS;
;  727     0858  1
;  728     0859  1            [DBG$NMATCH (.INPUT_DESC, DBG$CS_COBOL, 2)] :
;  729     0860  1                NOUN_NODE [DBG$[_NOUN_VALUE] = DBG$K_COBOL;
;  730     0861  1
;  731     0862  1            [DBG$NMATCH (.INPUT_DESC, DBG$CS_BASIC, 2)] :
;  732     0863  1                NOUN_NODE [DBG$[_NOUN_VALUE] = DBG$K_BASIC;
;  733     0864  1
;  734     0865  1            [DBG$NMATCH (.INPUT_DESC, DBG$CS_PLI, 2)] :
;  735     0866  1                NOUN_NODE [DBG$[_NOUN_VALUE] = DBG$K_PLI;
;  736     0867  1
;  737     0868  1            [DBG$NMATCH (.INPUT_DESC, DBG$CS_PASCAL, 2)] :
;  738     0869  1                NOUN_NODE [DBG$[_NOUN_VALUE] = DBG$K_PASCAL;
;  739     0870  1
;  740     0871  1            [DBG$NMATCH(.INPUT_DESC, DBG$CS_C, 1)]:
;  741     0872  1                NOUN_NODE[DBG$[_NOUN_VALUE] = DBG$K_C;
;  742     0873  1
;  743     0874  1            [DBG$NMATCH(.INPUT_DESC, DBG$CS_RPG, 2)]:
;  744     0875  1                NOUN_NODE[DBG$[_NOUN_VALUE] = DBG$K_RPG;
;  745     0876  1
;  746     0877  1            [IF NOT .DBG$GL_DEVELOPER[0] THEN FALSE ELSE
;  747     0878  1             DBG$NMATCH(.INPUT_DESC, DBG$CS_ADA, 2)]:
;  748     0879  1                NOUN_NODE[DBG$[_NOUN_VALUE] = DBG$K_ADA;
;  749     0880  1
;  750     0881  1            [DBG$NMATCH(.INPUT_DESC, DBG$CS_UNKNOWN, 3)]:
;  751     0882  1                NOUN_NODE[DBG$[_NOUN_VALUE] = DBG$K_UNKNOWN;
;  752     0883  1
;  753     0884  1                [OTHERWISE] :
;  754     0885  2                    BEGIN
;  755     0886  2                    IF DBG$NMATCH (.INPUT_DESC, DBG$CS_CR, 1)
;  756     0887  2                    THEN
;  757     0888  2                        .MESSAGE_VECT = DBG$NMAKE_ARG_VECT (DBG$_NEEDMORE)
;  758     0889  2
;  759     0890  2                    ELSE
;  760     0891  2                        .MESSAGE_VECT =
;  761     0892  2                                DBG$NSYNTAX_ERROR(DBG$NNEXT_WORD(.INPUT_DESC));
;  762     0893  2
;  763     0894  2                    RETURN STS$K_SEVERE;
;  764     0895  2                    END;
;  765     0896  2
;  766     0897  2                TES;
;  767     0898  1
;  768     0899  1            END;
;  769     0900  1
;  770     0901  1        [DBG$NMATCH (.INPUT_DESC, DBG$CS_LOG, 2)] :
;  771     0902  1            BEGIN
;  772     0903  1            VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_LOG;
;  773     0904  1            IF NOT DBG$NSAVE_FILESP (.INPUT_DESC,
;  774     0905  1                                     NOUN_NODE [DBG$L_NOUN_VALUE],
;  775     0906  1                                     .MESSAGE_VECT)
;  776     0907  1            THEN
```

```
;  777     0908  3            RETURN STS$K_SEVERE;
;  778     0909  2        END;
;  779     0910  2
;  780     0911  2    [DBG$NMATCH (.INPUT_DESC, DBG$CS_MARGINS, 3)] :
;  781     0912  2        BEGIN
;  782     0913  2        LOCAL
;  783     0914  2            LEFT_MARGIN,
;  784     0915  2            RIGHT_MARGIN;
;  785     0916  2
;  786     0917  2        VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_MARGINS;
;  787     0918  2
;  788     0919  2
;  789     0920  2        ! Legal forms of the SET MARGINS command are:
;  790     0921  2        ! 1. SET MARGINS l:r
;  791     0922  2        ! 2. SET MARGINS r
;  792     0923  2        ! 3. SET MARGINS l:
;  793     0924  2        ! 4. SET MARGINS :r
;  794     0925  2        !
;  795     0926  2        IF DBG$NMATCH (.INPUT_DESC, DBG$CS_COLON, 1)
;  796     0927  2        THEN
;  797     0928  3            BEGIN
;  798     0929  3
;  799     0930  3
;  800     0931  3            ! This must be case four, above
;  801     0932  3            !
;  802     0933  3            IF NOT DBG$NSAVE_DECIMAL_INTEGER(.INPUT_DESC,
;  803     0934  3                    RIGHT_MARGIN, .MESSAGE_VECT)
;  804     0935  3            THEN
;  805     0936  3                RETURN STS$K_SEVERE;
;  806     0937  3            LEFT_MARGIN = .DBG$SRC_LEFT_MARGIN;
;  807     0938  3            END
;  808     0939  3
;  809     0940  2        ELSE
;  810     0941  3            BEGIN
;  811     0942  3            IF NOT DBG$NSAVE_DECIMAL_INTEGER (.INPUT_DESC,
;  812     0943  3                RIGHT_MARGIN, .MESSAGE_VECT)
;  813     0944  3            THEN
;  814     0945  3                RETURN STS$K_SEVERE;
;  815     0946  3
;  816     0947  3
;  817     0948  3            ! Look for colon
;  818     0949  3            !
;  819     0950  3            IF DBG$NMATCH (.INPUT_DESC, DBG$CS_COLON, 1)
;  820     0951  3            THEN
;  821     0952  4                BEGIN
;  822     0953  4
;  823     0954  4
;  824     0955  4                ! This must be case 1 or case 3 above
;  825     0956  4                !
;  826     0957  4                LEFT_MARGIN = .RIGHT_MARGIN;
;  827     0958  4                IF DBG$NMATCH (.INPUT_DESC, DBG$CS_CR, 1)
;  828     0959  4                THEN ! case 3
;  829     0960  4                    RIGHT_MARGIN = .DBG$SRC_RIGHT_MARGIN
;  830     0961  4
;  831     0962  4                ELSE ! case 1
;  832     0963  5                    BEGIN
;  833     0964  5                    IF NOT DBG$NSAVE_DECIMAL_INTEGER (.INPUT_DESC,
```

```
;  834     0965  6                    RIGHT_MARGIN, .MESSAGE_VECT)
;  835     0966  6                THEN
;  836     0967  6                    RETURN STS$K_SEVERE;
;  837     0968  6
;  838     0969  5                END;
;  839     0970  5
;  840     0971  5            END
;  841     0972  5
;  842     0973  4        ELSE
;  843     0974  4            ! case 2
;  844     0975  5            BEGIN
;  845     0976  5            LEFT_MARGIN = 1;
;  846     0977  4            END;
;  847     0978  4
;  848     0979  3        END;
;  849     0980  3
;  850     0981  3
;  851     0982  3    ! Check that left margin is less than right margin
;  852     0983  3    !
;  853     0984  3    IF .LEFT_MARGIN GEQ .RIGHT_MARGIN
;  854     0985  3    THEN
;  855     0986  4        BEGIN
;  856     0987  4        DBG$NSHOW_MARGINS();
;  857     0988  4        SIGNAL(DBG$_INVMAR);
;  858     0989  3        END;
;  859     0990  3
;  860     0991  3
;  861     0992  3    ! Fill in the noun node
;  862     0993  3    !
;  863     0994  3        NOUN_NODE[DBG$L_NOUN_VALUE] = .LEFT_MARGIN;
;  864     0995  3        NOUN_NODE[DBG$L_NOUN_VALUE2] = .RIGHT_MARGIN;
;  865     0996  3
;  866     0997  2        END;
;  867     0998  2
;  868     0999  2    [DBG$NMATCH (.INPUT_DESC, DBG$CS_MAX_SOURCE_FILES, 3)] :
;  869     1000  2        BEGIN
;  870     1001  2        LOCAL
;  871     1002  2            MAX_FILES;
;  872     1003  2        VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_MAX_SOURCE_FILES;
;  873     1004  2        IF NOT DBG$NSAVE_DECIMAL_INTEGER(
;  874     1005  2            .INPUT_DESC, MAX_FILES, .MESSAGE_VECT)
;  875     1006  2        THEN
;  876     1007  2            RETURN STS$K_SEVERE;
;  877     1008  2        NOUN_NODE[DBG$L_NOUN_VALUE] = .MAX_FILES;
;  878     1009  2        END;
;  879     1010  2
;  880     1011  2    [DBG$NMATCH (.INPUT_DESC, DBG$CS_MODE, 1)] :
;  881     1012  2        BEGIN
;  882     1013  2
;  883     1014  2        LOCAL
;  884     1015  2            LINK;
;  885     1016  2
;  886     1017  2        BIND
;  887     1018  2            DBG$CS_BINARY           = UPLIT BYTE(%ASCIC 'BINARY'),
;  888     1019  2            DBG$CS_DECIMAL          = UPLIT BYTE(%ASCIC 'DECIMAL'),
;  889     1020  2            DBG$CS_DEFAULT          = UPLIT BYTE(%ASCIC 'DEFAULT'),
;  890     1021  3            DBG$CS_G_FLOAT          = UPLIT BYTE(%ASCIC 'G_FLOAT'),
```

```
;  891          1022  3          DBG$CS_HEXADECIMAL      = UPLIT BYTE(%ASCIC 'HEXADECIMAL'),
;  892          1023  3          DBG$CS_KEYPAD           = UPLIT BYTE(%ASCIC 'KEYPAD'),
;  893          1024  3          DBG$CS_NOG_FLOAT        = UPLIT BYTE(%ASCIC 'NOG_FLOAT'),
;  894          1025  3          DBG$CS_NOKEYPAD         = UPLIT BYTE(%ASCIC 'NOKEYPAD'),
;  895          1026  3          DBG$CS_NOSCREEN         = UPLIT BYTE(%ASCIC 'NOSCREEN'),
;  896          1027  3          DBG$CS_NOSYMBOLS        = UPLIT BYTE(%ASCIC 'NOSYMBOLIC'),
;  897          1028  3          DBG$CS_OCTAL            = UPLIT BYTE(%ASCIC 'OCTAL'),
;  898          1029  3          DBG$CS_SCREEN           = UPLIT BYTE(%ASCIC 'SCREEN'),
;  899          1030  3          DBG$CS_SYMBOLS          = UPLIT BYTE(%ASCIC 'SYMBOLIC');
;  900          1031  3
;  901          1032  3      VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_MODE;
;  902          1033  3
;  903          1034  3      ! Accept the mode switches
;  904          1035  3      !
;  905          1036  3      WHILE TRUE DO
;  906          1037  4          BEGIN
;  907          1038  4
;  908          1039  4          SELECTONE TRUE OF
;  909          1040  4              SET
;  910          1041  4
;  911          1042  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_BINARY, 1)] :
;  912          1043  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_BINARY;
;  913          1044  4
;  914          1045  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_DEFAULT, 3)]:
;  915          1046  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_DEFAULT;
;  916          1047  4
;  917          1048  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_DECIMAL, 1)] :
;  918          1049  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_DECIMAL;
;  919          1050  4
;  920          1051  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_HEXADECIMAL, 1)] :
;  921          1052  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_HEX;
;  922          1053  4
;  923          1054  4              [DBG$NMATCH(.INPUT_DESC, DBG$CS_KEYPAD, 1)]:
;  924          1055  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_KEYPAD;
;  925          1056  4
;  926          1057  4              [DBG$NMATCH(.INPUT_DESC, DBG$CS_NOKEYPAD, 3)]:
;  927          1058  4                  NOUN_NODE[DBG$L_NOUN_VALUE] = MODE_NOKEYPAD;
;  928          1059  4
;  929          1060  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_NOSCREEN, 5)] :
;  930          1061  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_NOSCREEN;
;  931          1062  4
;  932          1063  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_NOSYMBOLS, 1)] :
;  933          1064  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_NOSYMBLS;
;  934          1065  4
;  935          1066  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_OCTAL, 1)] :
;  936          1067  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_OCTAL;
;  937          1068  4
;  938          1069  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_SCREEN, 3)] :
;  939          1070  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_SCREEN;
;  940          1071  4
;  941          1072  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_SYMBOLS, 1)] :
;  942          1073  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_SYMBLS;
;  943          1074  4
;  944          1075  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_G_FLOAT, 1)] :
;  945          1076  4                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_G_FLOAT;
;  946          1077  4
;  947          1078  4              [DBG$NMATCH (.INPUT_DESC, DBG$CS_NOG_FLOAT, 3)] :
```

```
;  948          1079  4                NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_NOG_FLOAT;
;  949          1080  4
;  950          1081  4            [OTHERWISE] :
;  951          1082  5                BEGIN
;  952          1083  5                IF DBG$NMATCH(.INPUT_DESC, DBG$CS_CR, 1)
;  953          1084  5                THEN
;  954          1085  5                    .MESSAGE_VECT = DBG$NMAKE_ARG_VECT(DBG$_NEEDMORE)
;  955          1086  5
;  956          1087  5                ELSE
;  957          1088  5                    .MESSAGE_VECT = DBG$NSYNTAX_ERROR(
;  958          1089  5                                        DBG$NNEXT_WORD(.INPUT_DESC));
;  959          1090  5
;  960          1091  5                RETURN STS$K_SEVERE;
;  961          1092  4                END;
;  962          1093  4
;  963          1094  4            TES;
;  964          1095  4
;  965          1096  4
;  966          1097  4        ! Check for a comma to see if more input should follow
;  967          1098  4        !
;  968          1099  4        IF NOT DBG$NMATCH (.INPUT_DESC, DBG$CS_COMMA, 1)
;  969          1100  4        THEN
;  970          1101  4            EXITLOOP;
;  971          1102  4
;  972          1103  4
;  973          1104  4        ! More input should follow. Create another noun node and link
;  974          1105  4        !
;  975          1106  4        LINK = NOUN_NODE [DBG$L_NOUN_LINK];
;  976          1107  4        NOUN_NODE = DBG$GET_TEMPMEM (DBG$K_NOUN_NODE_SIZE);
;  977          1108  4        .LINK = .NOUN_NODE;
;  978          1109  4
;  979          1110  3        END;                        ! End of loop
;  980          1111  3
;  981          1112  3
;  982          1113  3            ! Place a 0 in the last noun link field
;  983          1114  3            !
;  984          1115  3            NOUN_NODE [DBG$L_NOUN_LINK] = 0;
;  985          1116  3
;  986          1117  2            END;
;  987          1118  2
;  988          1119  2        [DBG$NMATCH (.INPUT_DESC, DBG$CS_MODULE, 4)] :
;  989          1120  3            BEGIN
;  990          1121  3
;  991          1122  3            LOCAL
;  992          1123  3                ALL_FLAG,
;  993          1124  3                ALLOCATE_FLAG,
;  994          1125  3                NOWAIT_FLAG;
;  995          1126  3
;  996          1127  3
;  997          1128  3            ! Initialize the flags for SET MODULE/ALL and SET MODULE/NOWAIT
;  998          1129  3            !
;  999          1130  3            ALL_FLAG = FALSE;
; 1000          1131  3            ALLOCATE_FLAG = FALSE;
; 1001          1132  3            NOWAIT_FLAG = FALSE;
; 1002          1133  3
; 1003          1134  3
; 1004          1135  3            ! Check for SET MODULE/ALL or SET MODULE/NOWAIT.
```

```
;  1005          1136  3          !
;  1006          1137  3          WHILE DBG$NMATCH (.INPUT_DESC, DBG$CS_SLASH, 1) DO
;  1007          1138  4              BEGIN
;  1008          1139  4              BIND
;  1009          1140  4                  DBG$CS_AL      =     UPLIT BYTE (3, 'ALL'),
;  1010          1141  4                  DBG$CS_ALLOCATE =    UPLIT BYTE (8, 'ALLOCATE'),
;  1011          1142  4                  DBG$CS_NOWAIT =      UPLIT BYTE (6, 'NOWAIT');
;  1012          1143  4
;  1013          1144  4              IF DBG$NMATCH (.INPUT_DESC, DBG$CS_ALLOCATE, 4)
;  1014          1145  4              THEN
;  1015          1146  4                  ALLOCATE_FLAG = TRUE
;  1016          1147  4
;  1017          1148  4              ELSE IF DBG$NMATCH (.INPUT_DESC, DBG$CS_ALL, 1)
;  1018          1149  4              THEN
;  1019          1150  4                  ALL_FLAG = TRUE
;  1020          1151  4
;  1021          1152  5              ELSE IF (IF NOT .DBG$GL_DEVELOPER[0] THEN FALSE ELSE
;  1022          1153  5                       DBG$NMATCH (.INPUT_DESC, DBG$CS_NOWAIT, 1))
;  1023          1154  4              THEN
;  1024          1155  4                  NOWAIT_FLAG = TRUE
;  1025          1156  4
;  1026          1157  4              ELSE
;  1027          1158  5                  BEGIN
;  1028          1159  5                  IF DBG$NMATCH(.INPUT_DESC, DBG$CS_CR, 1)
;  1029          1160  5                  THEN
;  1030          1161  5                      .MESSAGE_VECT = DBG$NMAKE_ARG_VECT(DBG$_NEEDMORE)
;  1031          1162  5
;  1032          1163  5                  ELSE
;  1033          1164  5                      .MESSAGE_VECT = DBG$NSYNTAX_ERROR(
;  1034          1165  5                                          DBG$NNEXT_WORD(.INPUT_DESC));
;  1035          1166  5
;  1036          1167  5                  RETURN STS$K_SEVERE;
;  1037          1168  4                  END;
;  1038          1169  4
;  1039          1170  3              END;
;  1040          1171  3
;  1041          1172  3
;  1042          1173  3              ! Use the ADVERB_POINTER field to indicate whether /NOWAIT
;  1043          1174  3              ! was specified and/or whether /ALLOCATE was specified.
;  1044          1175  3              ! The codes are
;  1045          1176  3              !   0 - neither /NOWAIT or /ALLOCATE
;  1046          1177  3              !   1 - /NOWAIT, not /ALLOCATE
;  1047          1178  3              !   2 - not /NOWAIT, /ALLOCATE
;  1048          1179  3              !   3 - both were specified
;  1049          1180  3              !
;  1050          1181  3              VERB_NODE [DBG$L_VERB_ADVERB_PTR] = .NOWAIT_FLAG +
;  1051          1182  3                                                  2*.ALLOCATE_FLAG;
;  1052          1183  3
;  1053          1184  3
;  1054          1185  3              ! If /ALL was specified, indicate this in the verb_composite node.
;  1055          1186  3              !
;  1056          1187  3              IF .ALL_FLAG
;  1057          1188  3              THEN
;  1058          1189  3                  VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_MODULE_ALL
;  1059          1190  3
;  1060          1191  3
;  1061          1192  3              ! Only pick up a module if /ALL was not specified.
```

```
;  1062          1193  3          ! We have a module name list to parse.
;  1063          1194  3          !
;  1064          1195  3          ELSE
;  1065          1196  4              BEGIN
;  1066          1197  4
;  1067          1198  4              BIND
;  1068          1199  4                  DBG$CS_COMMA = UPLIT BYTE (1, dbg$k_comma);
;  1069          1200  4
;  1070          1201  4              LOCAL
;  1071          1202  4                  LINK;               ! Temporary pointer
;  1072          1203  4
;  1073          1204  4
;  1074          1205  4              ! Accept strings and commas
;  1075          1206  4              !
;  1076          1207  4              WHILE TRUE DO
;  1077          1208  5                  BEGIN
;  1078          1209  5
;  1079          1210  5                  ! For language C, we do some fancy footwork to
;  1080          1211  5                  ! make sure we preserve the original casing of
;  1081          1212  5                  ! the identifier (since casing is significant
;  1082          1213  5                  ! in C).
;  1083          1214  5                  !
;  1084          1215  5                  IF .dbg$gb_language EQL dbg$k_c
;  1085          1216  5                  THEN
;  1086          1217  6                      BEGIN
;  1087          1218  6                      MAP
;  1088          1219  6                          input_desc: REF dbg$stg_desc;
;  1089          1220  6                      LOCAL
;  1090          1221  6                          length,
;  1091          1222  6                          new_pointer: REF VECTOR [,BYTE],
;  1092          1223  6                                       ! Pointer to orig. command input
;  1093          1224  6                          pointer,     ! Pointer into input string
;  1094          1225  6                          stg_desc: dbg$stg_desc, ! String descriptor
;  1095          1226  6                          temp_ptr;
;  1096          1227  6
;  1097          1228  6                      pointer = .input_desc[dsc$a_pointer];
;  1098          1229  6                      length = .input_desc[dsc$w_length];
;  1099          1230  6                      IF (.pointer LSS .dbg$gl_upcase_command_ptr[0]) OR
;  1100          1231  7                         (.pointer GTR .dbg$gl_upcase_command_ptr[1])
;  1101          1232  6                      THEN
;  1102          1233  6                          $DBG_ERROR('DBGNSHOW\DBG$NPARSE_SET 10');
;  1103          1234  6
;  1104          1235  6                      ! We unfortunately have to allocate memory
;  1105          1236  6                      ! and copy strings in order to stuff a
;  1106          1237  6                      ! trailing carriage return at the end.
;  1107          1238  6                      !
;  1108          1239  6                      new_pointer = dbg$get_tempmem((.length+3)/4);
;  1109          1240  6                      temp_ptr = (.pointer - .dbg$gl_upcase_command_ptr[0]) +
;  1110          1241  6                                 .dbg$gl_orig_command_ptr;
;  1111          1242  6                      CH$MOVE (.length, .temp_ptr, .new_pointer);
;  1112          1243  6                      new_pointer[.length-1] = dbg$k_car_return;
;  1113          1244  6
;  1114          1245  6                      ! Fill in the string descriptor.
;  1115          1246  6                      !
;  1116          1247  6                      stg_desc[dsc$b_class] = dsc$k_class_s;
;  1117          1248  6                      stg_desc[dsc$b_dtype] = dsc$k_dtype_t;
;  1118          1249  6                      stg_desc[dsc$w_length] = .length;
```

```
;  1119          1250  6                    stg_desc[dsc$a_pointer] = .new_pointer;
;  1120          1251  6                    stg_desc[dsc$l_pos] = 0;
;  1121          1252  6
;  1122          1253  6                    ! Pick up the symbol name.
;  1123          1254  6                    !
;  1124          1255  6                    IF NOT dbg$nsave_string( stg_desc,
;  1125          1256  6                                             NOUN_NODE [DBG$L_NOUN_VALUE],
;  1126          1257  6                                            .message_vect)
;  1127          1258  6                    THEN
;  1128          1259  6                        RETURN sts$k_severe;
;  1129          1260  6
;  1130          1261  6                    ! Update the input descriptor.
;  1131          1262  6                    !
;  1132          1263  6                    input_desc[dsc$w_length] = .input_desc[dsc$w_length] -
;  1133          1264  6                        (.length - .stg_desc[dsc$w_length]);
;  1134          1265  6                    input_desc[dsc$a_pointer] = .input_desc[dsc$a_pointer] +
;  1135          1266  6                        (.length - .stg_desc[dsc$w_length]);
;  1136          1267  6                    END
;  1137          1268  6
;  1138          1269  6                ! All other languages besides C ...
;  1139          1270  6                !
;  1140          1271  5                ELSE
;  1141          1272  5
;  1142          1273  5
;  1143          1274  5                    IF NOT DBG$NSAVE_STRING(.INPUT_DESC,
;  1144          1275  5                          NOUN_NODE [DBG$L_NOUN_VALUE], .MESSAGE_VECT)
;  1145          1276  5                    THEN
;  1146          1277  5                        RETURN STS$K_SEVERE;
;  1147          1278  5
;  1148          1279  5
;  1149          1280  5                    ! Check for a comma
;  1150          1281  5                    !
;  1151          1282  5                    IF NOT DBG$NMATCH (.INPUT_DESC, DBG$CS_COMMA, 1)
;  1152          1283  5                    THEN
;  1153          1284  5                        EXITLOOP;
;  1154          1285  5
;  1155          1286  5
;  1156          1287  5                    ! Create a new noun node to hold the next string
;  1157          1288  5                    !
;  1158          1289  5                    LINK = NOUN_NODE [DBG$L_NOUN_LINK];
;  1159          1290  5                    NOUN_NODE = DBG$GET_TEMPMEM (DBG$K_NOUN_NODE_SIZE);
;  1160          1291  5                    .LINK = .NOUN_NODE;
;  1161          1292  5
;  1162          1293  4                    END;                ! End of loop
;  1163          1294  4
;  1164          1295  4
;  1165          1296  4                ! Place a zero in the last link field
;  1166          1297  4                !
;  1167          1298  4                NOUN_NODE [DBG$L_NOUN_LINK] = 0;
;  1168          1299  4                VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_MODULE;
;  1169          1300  3                END;
;  1170          1301  3
;  1171          1302  2            END;
;  1172          1303  2
;  1173          1304  2        [DBG$NMATCH (.INPUT_DESC, DBG$CS_OUTPUT, 1)] :
;  1174          1305  3            BEGIN
;  1175          1306  3
```

```
;  1176          1307  3          LITERAL
;  1177          1308  3              NOUN_LITERAL_LOG        = 1,
;  1178          1309  3              NOUN_LITERAL_NOLOG      = 2,
;  1179          1310  3              NOUN_LITERAL_SCREEN     = 3,
;  1180          1311  3              NOUN_LITERAL_NOSCREEN   = 4,
;  1181          1312  3              NOUN_LITERAL_TERMINAL   = 5,
;  1182          1313  3              NOUN_LITERAL_NOTERMINAL = 6,
;  1183          1314  3              NOUN_LITERAL_VERIFY     = 7,
;  1184          1315  3              NOUN_LITERAL_NOVERIFY   = 8;
;  1185          1316  3
;  1186          1317  3          BIND
;  1187          1318  3              DBG$CS_LOG              = UPLIT BYTE (%ASCIC 'LOG'),
;  1188          1319  3              DBG$CS_NOLOG            = UPLIT BYTE (%ASCIC 'NOLOG'),
;  1189          1320  3              DBG$CS_SCREEN_LOG       = UPLIT BYTE (%ASCIC 'SCREEN_LOG'),
;  1190          1321  3              DBG$CS_NOSCREEN_LOG     = UPLIT BYTE (%ASCIC 'NOSCREEN_LOG'),
;  1191          1322  3              DBG$CS_TERMINAL         = UPLIT BYTE (%ASCIC 'TERMINAL'),
;  1192          1323  3              DBG$CS_NOTERMINAL       = UPLIT BYTE (%ASCIC 'NOTERMINAL'),
;  1193          1324  3              DBG$CS_VERIFY           = UPLIT BYTE (%ASCIC 'VERIFY'),
;  1194          1325  3              DBG$CS_NOVERIFY         = UPLIT BYTE (%ASCIC 'NOVERIFY');
;  1195          1326  3
;  1196          1327  3          LOCAL
;  1197          1328  3              LINK;                   ! Used as a link pointer
;  1198          1329  3
;  1199          1330  3
;  1200          1331  3          ! Set the verb composite.
;  1201          1332  3          !
;  1202          1333  3          VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_OUTPUT;
;  1203          1334  3
;  1204          1335  3
;  1205          1336  3          ! Accept the SET OUTPUT xxx objects.
;  1206          1337  3          !
;  1207          1338  3          WHILE TRUE DO
;  1208          1339  4              BEGIN
;  1209          1340  4              SELECTONE TRUE OF
;  1210          1341  4                  SET
;  1211          1342  4
;  1212          1343  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_LOG, 1)] :
;  1213          1344  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_LOG;
;  1214          1345  4
;  1215          1346  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_NOLOG, 3)] :
;  1216          1347  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_NOLOG;
;  1217          1348  4
;  1218          1349  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_SCREEN_LOG, 1)] :
;  1219          1350  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_SCREEN;
;  1220          1351  4
;  1221          1352  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_NOSCREEN_LOG, 3)] :
;  1222          1353  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_NOSCREEN;
;  1223          1354  4
;  1224          1355  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_TERMINAL, 1)] :
;  1225          1356  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_TERMINAL;
;  1226          1357  4
;  1227          1358  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_NOTERMINAL, 3)] :
;  1228          1359  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_NOTERMINAL;
;  1229          1360  4
;  1230          1361  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_VERIFY, 1)] :
;  1231          1362  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_VERIFY;
;  1232          1363  4
```

```
;  1233          1364  4                [DBG$NMATCH (.INPUT_DESC, DBG$CS_NOVERIFY, 3)] :
;  1234          1365  4                    NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_NOVERIFY;
;  1235          1366  4
;  1236          1367  4                [OTHERWISE] :
;  1237          1368  5                    BEGIN
;  1238          1369  5                    IF DBG$NMATCH (.INPUT_DESC, DBG$CS_CR, 1)
;  1239          1370  5                    THEN
;  1240          1371  5                        .MESSAGE_VECT = DBG$NMAKE_ARG_VECT (DBG$_NEEDMORE)
;  1241          1372  5
;  1242          1373  5                    ELSE
;  1243          1374  5                        .MESSAGE_VECT = DBG$NSYNTAX_ERROR (DBG$NNEXT_WORD
;  1244          1375  5                                                           (.INPUT_DESC));
;  1245          1376  5                    RETURN STS$K_SEVERE;
;  1246          1377  4                    END;
;  1247          1378  4
;  1248          1379  4                TES;
;  1249          1380  4
;  1250          1381  4
;  1251          1382  4            ! Look for a comma and exit the loop if none found
;  1252          1383  4            !
;  1253          1384  4            IF NOT DBG$NMATCH (.INPUT_DESC, DBG$CS_COMMA, 1)
;  1254          1385  4            THEN
;  1255          1386  4                EXITLOOP;
;  1256          1387  4
;  1257          1388  4
;  1258          1389  4            ! Since we expect more input, create another noun node
;  1259          1390  4            !
;  1260          1391  4            LINK = NOUN_NODE [DBG$L_NOUN_LINK];
;  1261          1392  4            NOUN_NODE = DBG$GET_TEMPMEM (DBG$K_NOUN_NODE_SIZE);
;  1262          1393  4            .LINK = .NOUN_NODE;
;  1263          1394  4
;  1264          1395  3            END;                        ! End of loop
;  1265          1396  3
;  1266          1397  3
;  1267          1398  3            ! Put a zero in the last link field of the noun node chain
;  1268          1399  3            !
;  1269          1400  3            NOUN_NODE [DBG$L_NOUN_LINK] = 0;
;  1270          1401  3
;  1271          1402  2            END;
;  1272          1403  2
;  1273          1404  2        [DBG$NMATCH (.INPUT_DESC, DBG$CS_RADIX, 1)] :
;  1274          1405  3            BEGIN
;  1275          1406  3            BIND
;  1276          1407  3                DBG$CS_BINARY           = UPLIT BYTE(%ASCIC 'BINARY'),
;  1277          1408  3                DBG$CS_DECIMAL          = UPLIT BYTE(%ASCIC 'DECIMAL'),
;  1278          1409  3                DBG$CS_DEFAULT          = UPLIT BYTE(%ASCIC 'DEFAULT'),
;  1279          1410  3                DBG$CS_HEXADECIMAL      = UPLIT BYTE(%ASCIC 'HEXADECIMAL'),
;  1280          1411  3                DBG$CS_INPUT            = UPLIT BYTE(%ASCIC 'INPUT'),
;  1281          1412  3                DBG$CS_OCTAL            = UPLIT BYTE(%ASCIC 'OCTAL'),
;  1282          1413  3                DBG$CS_OUTPUT           = UPLIT BYTE(%ASCIC 'OUTPUT'),
;  1283          1414  3                DBG$CS_OVERRIDE         = UPLIT BYTE(%ASCIC 'OVERRIDE');
;  1284          1415  3
;  1285          1416  3            ! Check for the /OVERRIDE switch. Then fill in the verb
;  1286          1417  3            ! composite field accordingly. Also check for /INPUT or
;  1287          1418  3            ! /OUTPUT.
;  1288          1419  3            !
;  1289          1420  3            VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_RADIX;
```

```
;  1290     1421  3          WHILE DBG$NMATCH (.INPUT_DESC, DBG$CS_SLASH, 1) DO
;  1291     1422  3              SELECTONE TRUE OF
;  1292     1423  3                  SET
;  1293     1424  3
;  1294     1425  3                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_INPUT, 1)]:
;  1295     1426  3                      NOUN_NODE[DBG$L_NOUN_VALUE2] = 1;
;  1296     1427  3
;  1297     1428  3                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_OUTPUT, 2)]:
;  1298     1429  3                      NOUN_NODE[DBG$L_NOUN_VALUE2] = 2;
;  1299     1430  3
;  1300     1431  3                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_OVERRIDE, 2)]:
;  1301     1432  3                      VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_RADIX_OVERRIDE;
;  1302     1433  3
;  1303     1434  3                  [OTHERWISE]:
;  1304     1435  4                      BEGIN
;  1305     1436  4                      LOCAL
;  1306     1437  4                          CS: REF VECTOR[,BYTE],
;  1307     1438  4                          STG_DESC: DBG$STG_DESC;
;  1308     1439  4                      IF DBG$NMATCH (.INPUT_DESC, DBG$CS_CR, 1)
;  1309     1440  4                      THEN
;  1310     1441  4                          SIGNAL(DBG$_NEEDMORE)
;  1311     1442  4                      ELSE
;  1312     1443  5                          BEGIN
;  1313     1444  5                          CS = DBG$NNEXT_WORD(.INPUT_DESC);
;  1314     1445  5                          STG_DESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  1315     1446  5                          STG_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  1316     1447  5                          STG_DESC[DSC$W_LENGTH] = .CS[0];
;  1317     1448  5                          STG_DESC[DSC$A_POINTER] = CS[1];
;  1318     1449  5                          SIGNAL(DBG$_SYNTAX, 1, STG_DESC);
;  1319     1450  4                          END;
;  1320     1451  3                      END;
;  1321     1452  3                  TES;
;  1322     1453  3
;  1323     1454  3          ! Accept the radix argument.
;  1324     1455  3          !
;  1325     1456  3          SELECTONE TRUE OF
;  1326     1457  3              SET
;  1327     1458  3
;  1328     1459  3              [DBG$NMATCH (.INPUT_DESC, DBG$CS_BINARY, 1)] :
;  1329     1460  3                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_BINARY;
;  1330     1461  3
;  1331     1462  3              [DBG$NMATCH (.INPUT_DESC, DBG$CS_DEFAULT, 3)]:
;  1332     1463  3                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_DEFAULT;
;  1333     1464  3
;  1334     1465  3              [DBG$NMATCH (.INPUT_DESC, DBG$CS_DECIMAL, 1)] :
;  1335     1466  3                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_DECIMAL;
;  1336     1467  3
;  1337     1468  3              [DBG$NMATCH (.INPUT_DESC, DBG$CS_HEXADECIMAL, 1)] :
;  1338     1469  3                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_HEX;
;  1339     1470  3
;  1340     1471  3              [DBG$NMATCH (.INPUT_DESC, DBG$CS_OCTAL, 1)] :
;  1341     1472  3                  NOUN_NODE [DBG$L_NOUN_VALUE] = MODE_OCTAL;
;  1342     1473  3
;  1343     1474  3              [OTHERWISE] :
;  1344     1475  4                  BEGIN
;  1345     1476  4                  IF DBG$NMATCH(.INPUT_DESC, DBG$CS_CR, 1)
;  1346     1477  4                  THEN
```

```
;  1347     1478  4                    .MESSAGE_VECT = DBG$NMAKE_ARG_VECT(DBG$_NEEDMORE)
;  1348     1479  4
;  1349     1480  4                ELSE
;  1350     1481  4                    .MESSAGE_VECT = DBG$NSYNTAX_ERROR(
;  1351     1482  4                                        DBG$NNEXT_WORD(.INPUT_DESC));
;  1352     1483  4                RETURN STS$K_SEVERE;
;  1353     1484  3                END;
;  1354     1485  3
;  1355     1486  3            TES;
;  1356     1487  2        END;
;  1357     1488  2
;  1358     1489  2
;  1359     1490  2        [DBG$NMATCH (.INPUT_DESC, DBG$CS_SCOPE, 2)] :
;  1360     1491  3            BEGIN
;  1361     1492  3            VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_SCOPE;
;  1362     1493  3
;  1363     1494  3            ! Fill in 1 to the ADVERB field if SET SCOPE/MODULE was
;  1364     1495  3            ! specified; 0 otherwise.
;  1365     1496  3            !
;  1366     1497  3            IF DBG$NMATCH (.INPUT_DESC, DBG$CS_SLASH, 1)
;  1367     1498  3            THEN
;  1368     1499  3                IF DBG$NMATCH (.INPUT_DESC, DBG$CS_MODULE, 1)
;  1369     1500  3                THEN
;  1370     1501  3                    VERB_NODE[DBG$' _VERB_ADVERB_PTR] = 1
;  1371     1502  3                ELSE
;  1372     1503  3                    SIGNAL(DBG$_SETSCOMOD, 1, DBG$NNEXT_WORD(.INPUT_DESC))
;  1373     1504  3            ELSE
;  1374     1505  3                VERB_NODE[DBG$L_VERB_ADVERB_PTR] = 0;
;  1375     1506  3
;  1376     1507  3            IF NOT DBG$NPARSE_SCOPE_LIST(.INPUT_DESC,
;  1377     1508  3                                NOUN_NODE[DBG$L_NOUN_VALUE], .MESSAGE_VECT)
;  1378     1509  3            THEN
;  1379     1510  3                RETURN STS$K_SEVERE;
;  1380     1511  3
;  1381     1512  2            END;
;  1382     1513  2
;  1383     1514  2        [DBG$NMATCH (.INPUT_DESC, DBG$CS_SEARCH, 2)] :
;  1384     1515  3            BEGIN
;  1385     1516  3
;  1386     1517  3            LITERAL
;  1387     1518  3                NOUN_LITERAL_ALL = 1,
;  1388     1519  3                NOUN_LITERAL_NEXT = 2,
;  1389     1520  3                NOUN_LITERAL_STRING = 3,
;  1390     1521  3                NOUN_LITERAL_IDENT = 4;
;  1391     1522  3
;  1392     1523  3            BIND
;  1393     1524  3                DBG$CS_ALL      = UPLIT BYTE (3, 'ALL'),
;  1394     1525  3                DBG$CS_NEXT     = UPLIT BYTE (4, 'NEXT'),
;  1395     1526  3                DBG$CS_STRING   = UPLIT BYTE (6, 'STRING'),
;  1396     1527  3                DBG$CS_IDENT    = UPLIT BYTE(10, 'IDENTIFIER');
;  1397     1528  3
;  1398     1529  3            LOCAL
;  1399     1530  3                LINK;                   ! Used as a link pointer
;  1400     1531  3
;  1401     1532  3
;  1402     1533  3            ! Set up the verb composite
;  1403     1534  3            !
```

```
;  1404          1535  3          VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_SEARCH;
;  1405          1536  3
;  1406          1537  3          ! Accept the SET SEARCH xxx objects
;  1407          1538  3          !
;  1408          1539  3          WHILE TRUE DO
;  1409          1540  4              BEGIN
;  1410          1541  4              SELECTONE TRUE OF
;  1411          1542  4                  SET
;  1412          1543  4
;  1413          1544  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_ALL, 1)] :
;  1414          1545  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_ALL;
;  1415          1546  4
;  1416          1547  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_IDENT, 1)] :
;  1417          1548  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_IDENT;
;  1418          1549  4
;  1419          1550  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_NEXT, 1)] :
;  1420          1551  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_NEXT;
;  1421          1552  4
;  1422          1553  4                  [DBG$NMATCH (.INPUT_DESC, DBG$CS_STRING, 1)] :
;  1423          1554  4                      NOUN_NODE [DBG$L_NOUN_VALUE] = NOUN_LITERAL_STRING;
;  1424          1555  4
;  1425          1556  4                  [OTHERWISE] :
;  1426          1557  5                      BEGIN
;  1427          1558  5                      IF DBG$NMATCH (.INPUT_DESC, DBG$CS_CR, 1)
;  1428          1559  5                      THEN
;  1429          1560  5                          .MESSAGE_VECT = DBG$NMAKE_ARG_VECT (DBG$_NEEDMORE)
;  1430          1561  5
;  1431          1562  5                      ELSE
;  1432          1563  5                          .MESSAGE_VECT = DBG$NSYNTAX_ERROR(
;  1433          1564  5                                              DBG$NNEXT_WORD(.INPUT_DESC));
;  1434          1565  5
;  1435          1566  5                      RETURN STS$K_SEVERE;
;  1436          1567  4                      END;
;  1437          1568  4
;  1438          1569  4                  TES;
;  1439          1570  4
;  1440          1571  4
;  1441          1572  4              ! Look for comma and exit the loop if none found.
;  1442          1573  4              !
;  1443          1574  4              IF NOT DBG$NMATCH (.INPUT_DESC, DBG$CS_COMMA, 1)
;  1444          1575  4              THEN
;  1445          1576  4                  EXITLOOP;
;  1446          1577  4
;  1447          1578  4
;  1448          1579  4              ! Since we expect more input, create another noun node.
;  1449          1580  4              !
;  1450          1581  4              LINK = NOUN_NODE[DBG$L_NOUN_LINK];
;  1451          1582  4              NOUN_NODE = DBG$GET_TEMPMEM (DBG$K_NOUN_NODE_SIZE);
;  1452          1583  4              .LINK = .NOUN_NODE;
;  1453          1584  4
;  1454          1585  3              END;                        ! End of loop
;  1455          1586  3
;  1456          1587  3
;  1457          1588  3          ! Put a zero in the last link field of the noun node chain.
;  1458          1589  3          !
;  1459          1590  3          NOUN_NODE [DBG$L_NOUN_LINK] = 0;
;  1460          1591  3
```

```
;  1461          1592  2          END;
;  1462          1593  2
;  1463          1594  2
;  1464          1595  2      ! Parse the SET SOURCE command.
;  1465          1596  2      !
;  1466          1597  2      [DBG$NMATCH (.INPUT_DESC, DBG$CS_SOURCE, 2)] :
;  1467          1598  3          BEGIN
;  1468          1599  3          VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_SOURCE;
;  1469          1600  3
;  1470          1601  3
;  1471          1602  3          ! Check for SET SOURCE/MODULE=dir-list
;  1472          1603  3          !
;  1473          1604  3          IF DBG$NMATCH (.INPUT_DESC, DBG$CS_SLASH, 1)
;  1474          1605  3          THEN
;  1475          1606  4              BEGIN
;  1476          1607  4
;  1477          1608  4              LOCAL
;  1478          1609  4                  MODNAMEPTR;
;  1479          1610  4
;  1480          1611  4              BIND
;  1481          1612  4                  DBG$CS_MODULE = UPLIT BYTE (6, 'MODULE');
;  1482          1613  4
;  1483          1614  4
;  1484          1615  4              ! Read the string MODULE.
;  1485          1616  4              !
;  1486          1617  4              IF NOT DBG$NMATCH (.INPUT_DESC, DBG$CS_MODULE, 4)
;  1487          1618  4              THEN
;  1488          1619  5                  BEGIN
;  1489          1620  5                  .MESSAGE_VECT = DBG$NSYNTAX_ERROR(
;  1490          1621  5                                  DBG$NNEXT_WORD(.INPUT_DESC));
;  1491          1622  5                  RETURN STS$K_SEVERE;
;  1492          1623  4                  END;
;  1493          1624  4
;  1494          1625  4
;  1495          1626  4              ! Read the = sign.
;  1496          1627  4              !
;  1497          1628  4              IF NOT DBG$NMATCH (.INPUT_DESC, DBG$CS_EQUAL, 1)
;  1498          1629  4              THEN
;  1499          1630  5                  BEGIN
;  1500          1631  5                  .MESSAGE_VECT = DBG$NSYNTAX_ERROR(
;  1501          1632  5                                  DBG$NNEXT_WORD(.INPUT_DESC));
;  1502          1633  5                  RETURN STS$K_SEVERE;
;  1503          1634  4                  END;
;  1504          1635  4
;  1505          1636  4
;  1506          1637  4              ! Read the module name.
;  1507          1638  4              !
;  1508          1639  4              IF NOT DBG$NSAVE_STRING(.INPUT_DESC, MODNAMEPTR, .MESSAGE_VECT)
;  1509          1640  4              THEN
;  1510          1641  4                  RETURN STS$K_SEVERE;
;  1511          1642  4
;  1512          1643  4
;  1513          1644  4              ! Convert the module name into an rst pointer and
;  1514          1645  4              ! save into the adjective field of the noun node.
;  1515          1646  4              !
;  1516          1647  4              NOUN_NODE[DBG$L_ADJECTIVE_PTR] =
;  1517          1648  4                  DBG$STA_GETSOURCEMOD(.MODNAMEPTR);
```

```
;  1518          1649  4
;  1519          1650  4
;  1520          1651  4                ! If the above routine returns zero then the user has
;  1521          1652  4                ! entered an invalid module.
;  1522          1653  4                !
;  1523          1654  4                IF .NOUN_NODE[DBG$L_ADJECTIVE_PTR] EQL 0
;  1524          1655  4                THEN
;  1525          1656  5                    BEGIN
;  1526          1657  5                    .MESSAGE_VECT = DBG$NMAKE_ARG_VECT(
;  1527          1658  5                                    DBG$_NOSUCHMODU, 1, .MODNAMEPTR);
;  1528          1659  5                    RETURN STS$K_SEVERE;
;  1529          1660  4                    END;
;  1530          1661  4
;  1531          1662  3                END;                        ! SET SOURCE/MODULE=
;  1532          1663  3
;  1533          1664  3
;  1534          1665  3            ! Now read the directory list.
;  1535          1666  3            !
;  1536          1667  3            IF NOT DBG$NGET_DIR_LIST (.INPUT_DESC,
;  1537          1668  3                        NOUN_NODE[DBG$L_NOUN_VALUE], .MESSAGE_VECT)
;  1538          1669  3            THEN
;  1539          1670  3                RETURN STS$K_SEVERE;
;  1540          1671  3
;  1541          1672  3            END;                            ! SET SOURCE
;  1542          1673  2
;  1543          1674  2
;  1544          1675  2        ! Parse the SET STEP command.
;  1545          1676  2        !
;  1546          1677  2        [DBG$NMATCH (.INPUT_DESC, DBG$CS_STEP, 1)] :
;  1547          1678  3            BEGIN
;  1548          1679  3            VERB_NODE [DBG$B_VERB_COMPOSITE] = EVENT$K_SET_STEP;
;  1549          1680  3            RETURN DBG$EVENT_SYNTAX (.INPUT_DESC, .VERB_NODE, .MESSAGE_VECT);
;  1550          1681  2            END;
;  1551          1682  2
;  1552          1683  2
;  1553          1684  2        ! Parse the SET TASK command.
;  1554          1685  2        !
;  1555          1686  2        [IF NOT .DBG$GL_DEVELOPER[0] THEN FALSE ELSE
;  1556          1687  2         DBG$NMATCH (.INPUT_DESC, DBG$CS_TASK, 2)]:
;  1557          1688  3            BEGIN
;  1558          1689  3            VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_TASK;
;  1559          1690  3            DBG$NPARSE_SET_TASK(.INPUT_DESC, .VERB_NODE);
;  1560          1691  2            END;
;  1561          1692  2
;  1562          1693  2
;  1563          1694  2        ! Parse the SET TERMINAL command.
;  1564          1695  2        !
;  1565          1696  2        [DBG$NMATCH(.INPUT_DESC, DBG$CS_TERMINAL, 4)]:
;  1566          1697  3            BEGIN
;  1567          1698  3            VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_TERMINAL;
;  1568          1699  3            DBG$SCR_PARSE_SETTERM_CMD(.INPUT_DESC, .VERB_NODE);
;  1569          1700  2            END;
;  1570          1701  2
;  1571          1702  2
;  1572          1703  2        ! Parse the SET TRACE command.
;  1573          1704  2        !
;  1574          1705  2        [DBG$NMATCH(.INPUT_DESC, DBG$CS_TRACE, 1)]:
```

```
;  1575     1706  3              BEGIN
;  1576     1707  3              VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_TRACE;
;  1577     1708  3              RETURN DBG$EVENT_SYNTAX (.INPUT_DESC,
;  1578     1709  3                                       .VERB_NODE,
;  1579     1710  3                                       .MESSAGE_VECT
;  1580     1711  3                                      );
;  1581     1712  2              END;
;  1582     1713  2
;  1583     1714  2
;  1584     1715  2          [dbg$nmatch (.input_desc, dbg$cs_type, 2)] :    ! SET TYPE or SET TYPE/OVERRIDE
;  1585     1716  3              BEGIN
;  1586     1717  3              LOCAL
;  1587     1718  3                  LENGTH_NODE : REF dbg$noun_node;
;  1588     1719  3
;  1589     1720  3              BIND
;  1590     1721  3                  DBG$CS_AC       = UPLIT BYTE (2, 'AC'),
;  1591     1722  3                  DBG$CS_AD       = UPLIT BYTE (2, 'AD'),
;  1592     1723  3                  DBG$CS_ASCIC    = UPLIT BYTE (5, 'ASCIC'),
;  1593     1724  3                  DBG$CS_ASCID    = UPLIT BYTE (5, 'ASCID'),
;  1594     1725  3                  DBG$CS_ASCII    = UPLIT BYTE (5, 'ASCII'),
;  1595     1726  3                  DBG$CS_ASCIZ    = UPLIT BYTE (5, 'ASCIZ'),
;  1596     1727  3                  DBG$CS_AZ       = UPLIT BYTE (2, 'AZ'),
;  1597     1728  3                  DBG$CS_BYTE     = UPLIT BYTE (4, 'BYTE'),
;  1598     1729  3                  DBG$CS_COLON    = UPLIT BYTE (1, dbg$k_colon),
;  1599     1730  3                  DBG$CS_D_FLOAT  = UPLIT BYTE (7, 'D_FLOAT'),
;  1600     1731  3                  DBG$CS_FLOAT    = UPLIT BYTE (5, 'FLOAT'),
;  1601     1732  3                  DBG$CS_F_FLOAT  = UPLIT BYTE (7, 'F_FLOAT'),
;  1602     1733  3                  DBG$CS_G_FLOAT  = UPLIT BYTE (7, 'G_FLOAT'),
;  1603     1734  3                  DBG$CS_H_FLOAT  = UPLIT BYTE (7, 'H_FLOAT'),
;  1604     1735  3                  DBG$CS_PACKED   = UPLIT BYTE (6, 'PACKED'),
;  1605     1736  3                  DBG$CS_INSTRUCTION = UPLIT BYTE (11, 'INSTRUCTION'),
;  1606     1737  3                  DBG$CS_DATE_TIME = UPLIT BYTE (9, 'DATE_TIME'),
;  1607     1738  3                  DBG$CS_LONGWORD = UPLIT BYTE (8, 'LONGWORD'),
;  1608     1739  3                  DBG$CS_OCTAWORD = UPLIT BYTE (8, 'OCTAWORD'),
;  1609     1740  3                  DBG$CS_OVERRIDE = UPLIT BYTE (8, 'OVERRIDE'),
;  1610     1741  3                  DBG$CS_QUADWORD = UPLIT BYTE (8, 'QUADWORD'),
;  1611     1742  3                  DBG$CS_WORD     = UPLIT BYTE (4, 'WORD');
;  1612     1743  3
;  1613     1744  3              ! Create and link the length node (the second noun node).
;  1614     1745  3              !
;  1615     1746  3              length_node = dbg$get_tempmem (dbg$k_noun_node_size);
;  1616     1747  3              noun_node [dbg$l_noun_link] = .length_node;
;  1617     1748  3
;  1618     1749  3              ! Look for slash
;  1619     1750  3              !
;  1620     1751  3              IF dbg$nmatch (.input_desc, dbg$cs_slash, 1)
;  1621     1752  3              THEN
;  1622     1753  4                  BEGIN
;  1623     1754  4
;  1624     1755  4                  ! Must find 'override'
;  1625     1756  4                  !
;  1626     1757  4                  IF NOT dbg$nmatch (.input_desc, dbg$cs_override, 1)
;  1627     1758  4                  THEN
;  1628     1759  5                      BEGIN
;  1629     1760  5                      .message_vect =
;  1630     1761  6                      (IF dbg$nmatch (.input_desc, dbg$cs_cr, 1)
;  1631     1762  6                      THEN
```

```
;  1632          1763  6                       dbg$nmake_arg_vect (dbg$_needmore)
;  1633          1764  6                   ELSE
;  1634          1765  5                       dbg$nsyntax_error (dbg$nnext_word (.input_desc)));
;  1635          1766  5                   RETURN sts$k_severe;
;  1636          1767  4                   END;
;  1637          1768  4
;  1638          1769  4                   ! Set the verb composite
;  1639          1770  4                   !
;  1640          1771  4                   verb_node [dbg$b_verb_composite] = set_type_override;
;  1641          1772  4                   END
;  1642          1773  4
;  1643          1774  3               ELSE
;  1644          1775  3                   verb_node [dbg$b_verb_composite] = set_type;
;  1645          1776  3
;  1646          1777  3               ! The verb composite has been set. We must recognize the type specified.
;  1647          1778  3               !
;  1648          1779  3               SELECTONE true
;  1649          1780  3                   OF
;  1650          1781  3                   SET
;  1651          1782  3
;  1652          1783  3                   [dbg$nmatch (.input_desc, dbg$cs_ascic, 5),
;  1653          1784  3                    dbg$nmatch (.input_desc, dbg$cs_ac, 2)] :
;  1654          1785  4                       BEGIN
;  1655          1786  4                       noun_node [dbg$l_noun_value] = dsc$k_dtype_ac;
;  1656          1787  4                       length_node [dbg$l_noun_value] = 0;
;  1657          1788  3                       END;
;  1658          1789  3
;  1659          1790  3 !                 [dbg$nmatch (.input_desc, dbg$cs_ascid, 5) OR
;  1660          1791  3 !                  dbg$nmatch (.input_desc, dbg$cs_ad, 2)] :
;  1661          1792  3 !                     BEGIN
;  1662          1793  3 !                     noun_node [dbg$l_noun_value] = dsc$k_dtype_ad;
;  1663          1794  3 !                     length_node [dbg$l_noun_value] = 8;
;  1664          1795  3 !                     END;
;  1665          1796  3
;  1666          1797  3                   [dbg$nmatch (.input_desc, dbg$cs_packed, 1)] :
;  1667          1798  4                       BEGIN
;  1668          1799  4                       noun_node [dbg$l_noun_value] = dsc$k_dtype_p;
;  1669          1800  4
;  1670          1801  4                       ! Accept the ':' and integer. If there is no colon, use
;  1671          1802  4                       ! a length or default of 10.
;  1672          1803  4                       !
;  1673          1804  4                       IF NOT dbg$nmatch (.input_desc, dbg$cs_colon, 1)
;  1674          1805  4                       THEN
;  1675          1806  4                           SIGNAL(dbg$_pacsizreq)
;  1676          1807  4                       ELSE
;  1677          1808  5                           BEGIN
;  1678          1809  5
;  1679          1810  5                           ! We found the colon. Accept the integer
;  1680          1811  5                           !
;  1681          1812  5                           IF NOT dbg$nsave_integer (.input_desc,
;  1682          1813  5                                                     length_node [dbg$l_noun_value])
;  1683          1814  5                           THEN
;  1684          1815  5                               RETURN sts$k_severe;
;  1685          1816  5
;  1686          1817  5                           ! Check to see that the length is within 0 and 31
;  1687          1818  5                           !
;  1688          1819  6                           IF (.length_node [dbg$l_noun_value] LSS 0)  OR  (.length_node [dbg$l_noun_value] GTR
```

```
;  1689          1820  5                    THEN
;  1690          1821  5                        SIGNAL(dbg$_illpacsiz, 1, .length_node [dbg$l_noun_value]);
;  1691          1822  4                    END;
;  1692          1823  3                END;
;  1693          1824  3
;  1694          1825  3            [dbg$nmatch (.input_desc  dbg$cs_ascii, 1)] :
;  1695          1826  4                BEGIN
;  1696          1827  4                noun_node [dbg$l_noun_value] = dsc$k_dtype_t;
;  1697          1828  4
;  1698          1829  4                ! There may be a colon followed by integer. If not,
;  1699          1830  4                ! we use a default length of 4.
;  1700          1831  4                !
;  1701          1832  4                length_node [dbg$l_noun_value] = 4;
;  1702          1833  4
;  1703          1834  4                ! See if a color follows
;  1704          1835  4                !
;  1705          1836  4                IF dbg$nmatch (.input_desc, dbg$cs_colon, 1)
;  1706          1837  4                THEN
;  1707          1838  5                    BEGIN
;  1708          1839  5
;  1709          1840  5                    ! Obtain the integer.
;  1710          1841  5                    !
;  1711          1842  5                    IF NOT dbg$nsave_integer (.input_desc,
;  1712          1843  5                                              length_node [dbg$l_noun_value])
;  1713          1844  5                    THEN
;  1714          1845  5                        RETURN sts$k_severe;
;  1715          1846  5
;  1716          1847  5                    ! Check for a zero length
;  1717          1848  5                    !
;  1718          1849  5                    IF .length_node [dbg$l_noun_value] EQL 0
;  1719          1850  5                    THEN
;  1720          1851  6                        BEGIN
;  1721          1852  6
;  1722          1853  6                        ! Set up for an error message
;  1723          1854  6                        !
;  1724          1855  6                        OWN
;  1725          1856  6                            ZERO_DESC : dbg$stg_desc;
;  1726          1857  6                        zero_desc [dsc$w_length] = 1;
;  1727          1858  6                        zero_desc [dsc$a_pointer] = UPLIT BYTE ('0');
;  1728          1859  6                        .message_vect = dbg$nmake_arg_vect (dbg$_invnumber,
;  1729          1860  6                                                            1,
;  1730          1861  6                                                            zero_desc);
;  1731          1862  6                        RETURN sts$k_severe;
;  1732          1863  5                        END;
;  1733          1864  4                    END;
;  1734          1865  3                END;
;  1735          1866  3
;  1736          1867  3            [dbg$nmatch (.input_desc, dbg$cs_asciz, 5),
;  1737          1868  3             dbg$nmatch (.input_desc, dbg$cs_az, 2)] :
;  1738          1869  4                BEGIN
;  1739          1870  4                noun_node [dbg$l_noun_value] = dsc$k_dtype_az;
;  1740          1871  4                length_node [dbg$l_noun_value] = 0;
;  1741          1872  3                END;
;  1742          1873  3
;  1743          1874  3            [dbg$nmatch (.input_desc, dbg$cs_byte, 1)] :
;  1744          1875  4                BEGIN
;  1745          1876  4                noun_node [dbg$l_noun_value] = dsc$k_dtype_b;
```

```
;  1746          1877  4                 length_node [dbg$l_noun_value] = 1;
;  1747          1878  3                 END;
;  1748          1879  3
;  1749          1880  3             [dbg$nmatch( '.input_desc, dbg$cs_date_time, 2)] :
;  1750          1881  4                 BEGIN
;  1751          1882  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_adt;
;  1752          1883  4                 length_node [dbg$l_noun_value] = 8;
;  1753          1884  3                 END;
;  1754          1885  3
;  1755          1886  3             [dbg$nmatch (.input_desc, dbg$cs_d_float, 1) ] :
;  1756          1887  4                 BEGIN
;  1757          1888  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_d;
;  1758          1889  4                 length_node [dbg$l_noun_value] = 8;
;  1759          1890  3                 END;
;  1760          1891  3
;  1761          1892  3             [dbg$nmatch (.input_desc, dbg$cs_float, 1),
;  1762          1893  3              dbg$nmatch (.input_desc, dbg$cs_f_float, 1)] :
;  1763          1894  4                 BEGIN
;  1764          1895  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_f;
;  1765          1896  4                 length_node [dbg$l_noun_value] = 4;
;  1766          1897  3                 END;
;  1767          1898  3
;  1768          1899  3             [dbg$nmatch (.input_desc, dbg$cs_g_float, 1) ] :
;  1769          1900  4                 BEGIN
;  1770          1901  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_g;
;  1771          1902  4                 length_node [dbg$l_noun_value] = 8;
;  1772          1903  3                 END;
;  1773          1904  3
;  1774          1905  3             [dbg$nmatch (.input_desc, dbg$cs_h_float, 1) ] :
;  1775          1906  4                 BEGIN
;  1776          1907  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_h;
;  1777          1908  4                 length_node [dbg$l_noun_value] = 16;
;  1778          1909  3                 END;
;  1779          1910  3
;  1780          1911  3             [dbg$nmatch (.input_desc, dbg$cs_instruction, 1)] :
;  1781          1912  4                 BEGIN
;  1782          1913  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_zi;
;  1783          1914  4                 length_node [dbg$l_noun_value] = 0;
;  1784          1915  3                 END;
;  1785          1916  3
;  1786          1917  3             [dbg$nmatch (.input_desc, dbg$cs_longword, 1)] :
;  1787          1918  4                 BEGIN
;  1788          1919  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_l;
;  1789          1920  4                 length_node [dbg$l_noun_value] = 4;
;  1790          1921  3                 END;
;  1791          1922  3
;  1792          1923  3             [dbg$nmatch (.input_desc, dbg$cs_octaword, 1)] :
;  1793          1924  4                 BEGIN
;  1794          1925  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_o;
;  1795          1926  4                 length_node [dbg$l_noun_value] = 16;
;  1796          1927  3                 END;
;  1797          1928  3
;  1798          1929  3             [dbg$nmatch (.input_desc, dbg$cs_quadword, 1)] :
;  1799          1930  4                 BEGIN
;  1800          1931  4                 noun_node [dbg$l_noun_value] = dsc$k_dtype_q;
;  1801          1932  4                 length_node [dbg$l_noun_value] = 8;
;  1802          1933  3                 END;
```

```
;  1803          1934  3                [dbg$nmatch (.input_desc, dbg$cs_word, 1)] :
;  1804          1935  3                    BEGIN
;  1805          1936  4                    noun_node [dbg$l_noun_value] = dsc$k_dtype_w;
;  1806          1937  4                    length_node [dbg$l_noun_value] = 2;
;  1807          1938  4                    END;
;  1808          1939  3
;  1809          1940  3
;  1810          1941  3                [OTHERWISE] :
;  1811          1942  4                    BEGIN
;  1812          1943  4                    .message_vect =
;  1813          1944  5                    (IF dbg$nmatch (.input_desc, dbg$cs_cr, 1)
;  1814          1945  5                    THEN
;  1815          1946  5                        dbg$nmake_arg_vect (dbg$_needmore)
;  1816          1947  5                    ELSE
;  1817          1948  4                        dbg$nsyntax_error (dbg$nnext_word (.input_desc)));
;  1818          1949  4                    RETURN sts$k_severe;
;  1819          1950  3                    END;
;  1820          1951  3
;  1821          1952  3                TES;
;  1822          1953  3
;  1823          1954  2            END;
;  1824          1955  2
;  1825          1956  2
;  1826          1957  2            ! Parse the SET WATCH command.
;  1827          1958  2            !
;  1828          1959  2            [DBG$NMATCH (.INPUT_DESC, DBG$CS_WATCH, 1)]:
;  1829          1960  3                BEGIN
;  1830          1961  3                VERB_NODE [DBG$B_VERB_COMPOSITE] = SET_WATCH;
;  1831          1962  3                RETURN DBG$EVENT_SYNTAX(.INPUT_DESC, .VERB_NODE, .MESSAGE_VECT);
;  1832          1963  2                END;
;  1833          1964  2
;  1834          1965  2
;  1835          1966  2            ! Parse the SET WINDOW command.
;  1836          1967  2            !
;  1837          1968  2            [DBG$NMATCH(.INPUT_DESC, DBG$CS_WINDOW, 3)]:
;  1838          1969  3                BEGIN
;  1839          1970  3                VERB_NODE[DBG$B_VERB_COMPOSITE] = SET_WINDOW;
;  1840          1971  3                DBG$SCR_PARSE_SETWIND_CMD(.INPUT_DESC, .VERB_NODE);
;  1841          1972  2                END;
;  1842          1973  2
;  1843          1974  2
;  1844          1975  2            ! Any other SET command keyword constitutes a syntax error.
;  1845          1976  2            !
;  1846          1977  2            [OTHERWISE]:
;  1847          1978  3                BEGIN
;  1848          1979  3                IF DBG$NMATCH (.INPUT_DESC, DBG$CS_CR, 1)
;  1849          1980  3                THEN
;  1850          1981  3                    .MESSAGE_VECT = DBG$NMAKE_ARG_VECT(DBG$_NEEDMORE)
;  1851          1982  3
;  1852          1983  3                ELSE
;  1853          1984  3                    .MESSAGE_VECT = DBG$NSYNTAX_ERROR(DBG$NNEXT_WORD (.INPUT_DESC));
;  1854          1985  3
;  1855          1986  3                RETURN STS$K_SEVERE;
;  1856          1987  2                END;
;  1857          1988  2
;  1858          1989  2            TES;
;  1859          1990  2
```

```
;  1860          1991  2
;  1861          1992  2      ! The SET command has been successfully parsed.  Return to the caller.
;  1862          1993  2      !
;  1863          1994  2      RETURN STS$K_SUCCESS;
;  1864          1995  2
;  1865          1996  1      END;


                                                                                .TITLE  DBGNSET
                                                                                .IDENT  \V04-000\

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

                                                    4B 41 45 52 42 05 00000 P.AAA:  .ASCII  <5>\BREAK\
                                                 45 4E 49 46 45 44 06 00006 P.AAB:  .ASCII  <6>\DEFINE\
                                              52 45 50 4F 4C 45 56 45 44 09 0000D P.AAC:  .ASCII  <9>\DEVELOPER\
                                                 59 41 4C 50 53 49 44 07 00017 P.AAD:  .ASCII  <7>\DISPLAY\
                                              4E 4F 49 54 50 45 43 58 45 09 0001F P.AAE:  .ASCII  <9>\EXCEPTION\
                                                 45 47 41 55 47 4E 41 4C 08 00029 P.AAF:  .ASCII  <8>\LANGUAGE\
                                                                47 4F 4C 03 00032 P.AAG:  .ASCII  <3>\LOG\
                                                                59 45 4B 03 00036 P.AAH:  .ASCII  <3>\KEY\
                                                 53 4E 49 47 52 41 4D 07 0003A P.AAI:  .ASCII  <7>\MARGINS\
4C 49 46 5F 45 43 52 55 4F 53 5F 58 41 4D 10 00042 P.AAJ:  .ASCII  <16>\MAX_SOURCE_FILES\
                                                                53 45 00051
                                                       45 44 4F 4D 04 00053 P.AAK:  .ASCII  <4>\MODE\
                                                    45 4C 55 44 4F 4D 06 00058 P.AAL:  .ASCII  <6>\MODULE\
                                                    54 55 50 54 55 4F 06 0005F P.AAM:  .ASCII  <6>\OUTPUT\
                                                       58 49 44 41 52 05 00066 P.AAN:  .ASCII  <5>\RADIX\
                                                       45 50 4F 43 53 05 0006C P.AAO:  .ASCII  <5>\SCOPE\
                                                 48 43 52 41 45 53 06 00072 P.AAP:  .ASCII  <6>\SEARCH\
                                                 45 43 52 55 4F 53 06 00079 P.AAQ:  .ASCII  <6>\SOURCE\
                                                       50 45 54 53 04 00080 P.AAR:  .ASCII  <4>\STEP\
                                                       4B 53 41 54 04 00085 P.AAS:  .ASCII  <4>\TASK\
4C 41 4E 49 4D 52 45 54 08 0008A P.AAT:  .ASCII  <8>\TERMINAL\
                                                    45 43 41 52 54 05 00093 P.AAU:  .ASCII  <5>\TRACE\
                                                       45 50 59 54 04 00099 P.AAV:  .ASCII  <4>\TYPE\
                                                 48 43 54 41 57 05 0009E P.AAW:  .ASCII  <5>\WATCH\
                                                 57 4F 44 4E 49 57 06 000A4 P.AAX:  .ASCII  <6>\WINDOW\
                                                                3A 01 000AB P.AAY:  .ASCII  <1>\:\
                                                                2C 01 000AD P.AAZ:  .ASCII  <1>\,\
                                                                3D 01 000AF P.ABA:  .ASCII  <1>\=\
                                                                2F 01 000B1 P.ABB:  .ASCII  <1>\/\
                                                                0D 01 000B3 P.ABC:  .BYTE   1, 13
                                                                   07 000B5 P.ABD:  .BYTE   7
                                                 53 53 45 52 44 44 41 000B6       .ASCII  \ADDRESS\
                                                                   07 000BD P.ABE:  .BYTE   7
                                                 44 4E 41 4D 4D 4F 43 000BE       .ASCII  \COMMAND\
                                                                   09 000C5 P.ABF:  .BYTE   9
45 52 55 44 45 43 4F 52 50 000C6       .ASCII  \PROCEDURE\
                                                                   06 000CF P.ABG:  .BYTE   6
                                                    47 4E 49 52 54 53 000D0       .ASCII  \STRING\
                                                                   05 000D6 P.ABH:  .BYTE   5
                                                       45 55 4C 41 56 000D7       .ASCII  \VALUE\
                                                                   05 000DC P.ABI:  .BYTE   5
                                                       45 54 41 54 53 000DD       .ASCII  \STATE\
                                                                   07 000E2 P.ABJ:  .BYTE   7
                                                 45 54 41 54 53 4F 4E 000E3       .ASCII  \NOSTATE\
```

```
                                                 05  000EA P.ABK:  .BYTE   5
                                 47 4F 4C 4F  4E  000EB            .ASCII  \NOLOG\
                              4F 52 43 41 4D  05  000F0 P.ABL:     .ASCII  <5>\MACRO\
                        4E 41 52 54 52 4F 46  07  000F6 P.ABM:     .ASCII  <7>\FORTRAN\
                              53 53 49 4C 42  05  000FE P.ABN:     .ASCII  <5>\BLISS\
                              4C 4F 42 4F 43  05  00104 P.ABO:     .ASCII  <5>\COBOL\
                              43 49 53 41 42  05  0010A P.ABP:     .ASCII  <5>\BASIC\
                                    49 4C 50  03  00110 P.ABQ:     .ASCII  <3>\PLI\
                           4C 41 43 53 41 50  06  00114 P.ABR:     .ASCII  <6>\PASCAL\
                                          43  01  0011B P.ABS:     .ASCII  <1>\C\
                                    47 50 52  03  0011D P.ABT:     .ASCII  <3>\RPG\
                                    41 44 41  03  00121 P.ABU:     .ASCII  <3>\ADA\
                        4E 57 4F 4E 4B 4E 55  07  00125 P.ABV:     .ASCII  <7>\UNKNOWN\
                           59 52 41 4E 49 42  06  0012D P.ABW:     .ASCII  <6>\BINARY\
                        4C 41 4D 49 43 45 44  07  00134 P.ABX:     .ASCII  <7>\DECIMAL\
                        54 4C 55 41 46 45 44  07  0013C P.ABY:     .ASCII  <7>\DEFAULT\
                        54 41 4F 4C 46 5F 47  07  00144 P.ABZ:     .ASCII  <7>\G_FLOAT\
            4C 41 4D 49 43 45 44 41 58 45 48  0B  0014C P.ACA:     .ASCII  <11>\HEXADECIMAL\
                           44 41 50 59 45 4B  06  00158 P.ACB:     .ASCII  <6>\KEYPAD\
                  54 41 4F 4C 46 5F 47 4F 4E  09  0015F P.ACC:     .ASCII  <9>\NOG_FLOAT\
                     44 41 50 59 45 4B 4F 4E  08  00169 P.ACD:     .ASCII  <8>\NOKEYPAD\
                     4E 45 45 52 43 53 4F 4E  08  00172 P.ACE:     .ASCII  <8>\NOSCREEN\
               43 49 4C 4F 42 4D 59 53 4F 4E  0A  0017B P.ACF:     .ASCII  <10>\NOSYMBOLIC\
                              4C 41 54 43 4F  05  00186 P.ACG:     .ASCII  <5>\OCTAL\
                           4E 45 45 52 43 53  06  0018C P.ACH:     .ASCII  <6>\SCREEN\
                     43 49 4C 4F 42 4D 59 53  08  00193 P.ACI:     .ASCII  <8>\SYMBOLIC\
                                              03  0019C P.ACJ:     .BYTE   3
                                       4C 4C  41  0019D            .ASCII  \ALL\
                                              08  001A0 P.ACK:     .BYTE   8
                        45 54 41 43 4F 4C 4C  41  001A1            .ASCII  \ALLOCATE\
                                              06  001A9 P.ACL:     .BYTE   6
                              54 49 41 57 4F  4E  001AA            .ASCII  \NOWAIT\
                                          2C  01  001B0 P.ACM:     .BYTE   1, 44
4E 24 47 42 44 5C 57 4F 48 53 4E 47 42 44  1A  001B2 P.ACN:     .ASCII  <26>\DBGNSHOW\<92>\DBG$NPARSE_SET 10\
         30 31 20 54 45 53 5F 45 53 52 41  50  001C1
                                    47 4F 4C  03  001CD P.ACO:     .ASCII  <3>\LOG\
                              47 4F 4C 4F 4E  05  001D1 P.ACP:     .ASCII  <5>\NOLOG\
               47 4F 4C 5F 4E 45 45 52 43 53  0A  001D7 P.ACQ:     .ASCII  <10>\SCREEN_LOG\
         47 4F 4C 5F 4E 45 45 52 43 53 4F 4E  0C  001E2 P.ACR:     .ASCII  <12>\NOSCREEN_LOG\
                     4C 41 4E 49 4D 52 45 54  08  001EF P.ACS:     .ASCII  <8>\TERMINAL\
               4C 41 4E 49 4D 52 45 54 4F 4E  0A  001F8 P.ACT:     .ASCII  <10>\NOTERMINAL\
                           59 46 49 52 45 56  06  00203 P.ACU:     .ASCII  <6>\VERIFY\
                     59 46 49 52 45 56 4F 4E  08  0020A P.ACV:     .ASCII  <8>\NOVERIFY\
                           59 52 41 4E 49 42  06  00213 P.ACW:     .ASCII  <6>\BINARY\
                        4C 41 4D 49 43 45 44  07  0021A P.ACX:     .ASCII  <7>\DECIMAL\
                        54 4C 55 41 46 45 44  07  00222 P.ACY:     .ASCII  <7>\DEFAULT\
            4C 41 4D 49 43 45 44 41 58 45 48  0B  0022A P.ACZ:     .ASCII  <11>\HEXADECIMAL\
                              54 55 50 4E 49  05  00236 P.ADA:     .ASCII  <5>\INPUT\
                              4C 41 54 43 4F  05  0023C P.ADB:     .ASCII  <5>\OCTAL\
                           54 55 50 54 55 4F  06  00242 P.ADC:     .ASCII  <6>\OUTPUT\
                     45 44 49 52 52 45 56 4F  08  00249 P.ADD:     .ASCII  <8>\OVERRIDE\
                                              03  00252 P.ADE:     .BYTE   3
                                       4C 4C  41  00253            .ASCII  \ALL\
                                              04  00256 P.ADF:     .BYTE   4
                                    54 58 45  4E  00257            .ASCII  \NEXT\
                                              06  0025B P.ADG:     .BYTE   6
                           47 4E 49 52 54 53  0025C               .ASCII  \STRING\
```

```
                                                    0A 00262 P.ADH:  .BYTE   10
                52  45  49  46  49  54  4E  45  44   49 00263          .ASCII  \IDENTIFIER\
                                                    06 0026D P.ADI:  .BYTE   6
                                    45  4C  55  44  4F  4D 0026E          .ASCII  \MODULE\
                                                    02 00274 P.ADJ:  .BYTE   2
                                                43  41  00275          .ASCII  \AC\
                                                    02 00277 P.ADK:  .BYTE   2
                                                44  41  00278          .ASCII  \AD\
                                                    05 0027A P.ADL:  .BYTE   5
                                        43  49  43  53  41 0027B          .ASCII  \ASCIC\
                                                    05 00280 P.ADM:  .BYTE   5
                                        44  49  43  53  41 00281          .ASCII  \ASCID\
                                                    05 00286 P.ADN:  .BYTE   5
                                        49  49  43  53  41 00287          .ASCII  \ASCII\
                                                    05 0028C P.ADO:  .BYTE   5
                                        5A  49  43  53  41 0028D          .ASCII  \ASCIZ\
                                                    02 00292 P.ADP:  .BYTE   2
                                                5A  41  00293          .ASCII  \AZ\
                                                    04 00295 P.ADQ:  .BYTE   4
                                            45  54  59  42 00296          .ASCII  \BYTE\
                                                3A  01 0029A P.ADR:  .BYTE   1, 58
                                                    07 0029C P.ADS:  .BYTE   7
                        54  41  4F  4C  46  5F  44  0029D          .ASCII  \D_FLOAT\
                                                    05 002A4 P.ADT:  .BYTE   5
                                    54  41  4F  4C  46 002A5          .ASCII  \FLOAT\
                                                    07 002AA P.ADU:  .BYTE   7
                        54  41  4F  4C  46  5F  46 002AB          .ASCII  \F_FLOAT\
                                                    07 002B2 P.ADV:  .BYTE   7
                        54  41  4F  4C  46  5F  47 002B3          .ASCII  \G_FLOAT\
                                                    07 002BA P.AD_   .BYTE   7
                        54  41  4F  4C  46  5F  48 002BB          .ASCII  \H_FLOAT\
                                                    06 002C2 P.ADX:  .BYTE   6
                                44  45  4B  43  41  50 002C3          .ASCII  \PACKED\
                                                    0B 002C9 P.ADY:  .BYTE   11
    4E  4F  49  54  43  55  52  54  53  4E  49  4F  4E 002CA          .ASCII  \INSTRUCTION\
                                                    09 002D5 P.ADZ:  .BYTE   9
                45  4D  49  54  5F  45  54  41  44  54 002D6          .ASCII  \DATE_TIME\
                                                    08 002DF P.AEA:  .BYTE   8
                    44  52  4F  57  47  4E  4F  4C  4F 002E0          .ASCII  \LONGWORD\
                                                    08 002E8 P.AEB:  .BYTE   8
                    44  52  4F  57  41  54  43  4F  4F 002E9          .ASCII  \OCTAWORD\
                                                    08 002F1 P.AEC:  .BYTE   8
                    45  44  49  52  52  45  56  4F  52 002F2          .ASCII  \OVERRIDE\
                                                    08 002FA P.AED:  .BYTE   8
                    44  52  4F  57  44  41  55  51  44 002FB          .ASCII  \QUADWORD\
                                                    04 00303 P.AEE:  .BYTE   4
                                        44  52  4F  57  57 00304          .ASCII  \WORD\
                                                    30 00308 P.AEF:  .ASCII  \0\

                                                                     .PSECT  DBG$OWN,NOEXE,  PIC,2

                                                    00000 ZERO_DESC:
                                                                     .BLKB   12

                                                                     .PSECT  DBG$GLOBAL,NOEXE,  PIC,2

                                                00000000  00000 DBG$GL_DEVELOPER::
```

```
                .LONG   0                                       ;

DBG$CS_BREAK=           P.AAA
DBG$CS_DEFINE=          P.AAB
DBG$CS_DEVELOPER=       P.AAC
DBG$CS_DISPLAY=         P.AAD
DBG$CS_EXCEPTION=       P.AAE
DBG$CS_LANGUAGE=        P.AAF
DBG$CS_LOG=             P.AAG
DBG$CS_KEY=             P.AAH
DBG$CS_MARGINS-         P.AAI
DBG$CS_MAX_SOURCE_FILES=
                        P.AAJ
DBG$CS_MODE=            P.AAK
DBG$CS_MODULE=          P.AAL
DBG$CS_OUTPUT=          P.AAM
DBG$CS_RADIX=           P.AAN
DBG$CS_SCOPE=           P.AAO
DBG$CS_SEARCH=          P.AAP
DBG$CS_SOURCE=          P.AAQ
DBG$CS_STEP=            P.AAR
DBG$CS_TASK=            P.AAS
DBG$CS_TERMINAL=        P.AAT
DBG$CS_TRACE=           P.AAU
DBG$CS_TYPE=            P.AAV
DBG$CS_WATCH=           P.AAW
DBG$CS_WINDOW=          P.AAX
DBG$CS_COLON=           P.AAY
DBG$CS_COMMA=           P.AAZ
DBG$CS_EQUAL=           P.ABA
DBG$CS_SLASH=           P.ABB
DBG$CS_CR=              P.ABC
DBG$CS_ADDRESS=         P.ABD
DBG$CS_COMMAND=         P.ABE
DBG$CS_PROCEDURE=       P.ABF
DBG$CS_STRING=          P.ABG
DBG$CS_VALUE=           P.ABH
DBG$CS_STATE=           P.ABI
DBG$CS_NOSTATE=         P.ABJ
DBG$CS_NOLOG=           P.ABK
DBG$CS_MACRO=           P.ABL
DBG$CS_FORTRAN=         P.ABM
DBG$CS_BLISS=           P.ABN
DBG$CS_COBOL=           P.ABO
DBG$CS_BASIC=           P.ABP
DBG$CS_PLI=             P.ABQ
DBG$CS_PASCAL=          P.ABR
DBG$CS_C=               P.ABS
DBG$CS_RPG=             P.ABT
DBG$CS_ADA=             P.ABU
DBG$CS_UNKNOWN=         P.ABV
DBG$CS_BINARY=          P.ABW
DBG$CS_DECIMAL=         P.ABX
DBG$CS_DEFAULT=         P.ABY
DBG$CS_G_FLOAT=         P.ABZ
DBG$CS_HEXADECIMAL=     P.ACA
DBG$CS_KEYPAD=          P.ACB
```

```
DBG$CS_NOG_FLOAT=    P.ACC
DBG$CS_NOKEYPAD=     P.ACD
DBG$CS_NOSCREEN=     P.ACE
DBG$CS_NOSYMBOLS=    P.ACF
DBG$CS_OCTAL=        P.ACG
DBG$CS_SCREEN=       P.ACH
DBG$CS_SYMBOLS=      P.ACI
DBG$CS_ALL=          P.ACJ
DBG$CS_ALLOCATE=     P.ACK
DBG$CS_NOWAIT=       P.ACL
DBG$CS_COMMA=        P.ACM
DBG$CS_LOG=          P.ACO
DBG$CS_NOLOG=        P.ACP
DBG$CS_SCREEN_LOG=   P.ACQ
DBG$CS_NOSCREEN_LOG=P.ACR
DBG$CS_TERMINAL=     P.ACS
DBG$CS_NOTERMINAL=   P.ACT
DBG$CS_VERIFY=       P.ACU
DBG$CS_NOVERIFY=     P.ACV
DBG$CS_BINARY=       P.ACW
DBG$CS_DECIMAL=      P.ACX
DBG$CS_DEFAULT=      P.ACY
DBG$CS_HEXADECIMAL=  P.ACZ
DBG$CS_INPUT=        P.ADA
DBG$CS_OCTAL=        P.ADB
DBG$CS_OUTPUT=       P.ADC
DBG$CS_OVERRIDE=     P.ADD
DBG$CS_ALL=          P.ADE
DBG$CS_NEXT=         P.ADF
DBG$CS_STRING=       P.ADG
DBG$CS_IDENT=        P.ADH
DBG$CS_MODULE=       P.ADI
DBG$CS_AC=           P.ADJ
DBG$CS_AD=           P.ADK
DBG$CS_ASCIC=        P.ADL
DBG$CS_ASCID=        P.ADM
DBG$CS_ASCII=        P.ADN
DBG$CS_ASCIZ=        P.ADO
DBG$CS_AZ=           P.ADP
DBG$CS_BYTE=         P.ADQ
DBG$CS_COLON=        P.ADR
DBG$CS_D_FLOAT=      P.ADS
DBG$CS_FLOAT=        P.ADT
DBG$CS_F_FLOAT=      P.ADU
DBG$CS_G_FLOAT=      P.ADV
DBG$CS_H_FLOAT=      P.ADW
DBG$CS_PACKED=       P.ADX
DBG$CS_INSTRUCTION=  P.ADY
DBG$CS_DATE_TIME=    P.ADZ
DBG$CS_LONGWORD=     P.AEA
DBG$CS_OCTAWORD=     P.AEB
DBG$CS_OVERRIDE=     P.AEC
DBG$CS_QUADWORD=     P.AED
DBG$CS_WORD=         P.AEE
        .EXTRN  DBG$EVENT_SYNTAX
        .EXTRN  DBG$EVENT_SEMANTICS
        .EXTRN  DBG$NGET_LENGTH
```

```
.EXTRN  DBG$NGET_DIR_LIST
.EXTRN  DBG$NGET_TRANS_RADIX
.EXTRN  DBG$SCR_EXECUTE_DISPLAY_CMD
.EXTRN  DBG$SCR_EXECUTE_SETTERM_CMD
.EXTRN  DBG$SCR_EXECUTE_SETWIND_CMD
.EXTRN  DBG$SCR_PARSE_DISPLAY_CMD
.EXTRN  DBG$SCR_PARSE_SETTERM_CMD
.EXTRN  DBG$SCR_PARSE_SETWIND_CMD
.EXTRN  DBG$SCR_SCREEN_MODE
.EXTRN  DBG$STA_GETSOURCEMOD
.EXTRN  DBG$SRC_SET_SOURCE
.EXTRN  DBG$NSAVE_DECIMAL_INTEGER
.EXTRN  DBG$NSAVE_INTEGER
.EXTRN  DBG$NSAVE_STRING
.EXTRN  DBG$NGET_TYPE, DBG$PRINT
.EXTRN  DBG$NEWLINE, DBG$SET_SEARCH_LVL
.EXTRN  DBG$SET_STP_LVL
.EXTRN  DBG$SET_DEFINE_LVL
.EXTRN  DBG$SET_MOD_LVL
.EXTRN  DBG$NPARSE_SET_TASK
.EXTRN  DBG$NEXECUTE_SET_TASK
.EXTRN  DBG$RST_SETSCOPE
.EXTRN  DBG$NPARSE_SCOPE_LIST
.EXTRN  DBG$NSHOW_MARGINS
.EXTRN  DBG$SRC_SET_MAX_FILES
.EXTRN  DBG$NPATHDESC_TO_CS
.EXTRN  DBG$RST_SETMOD, DBG$GET_MEMORY
.EXTRN  DBG$REL_MEMORY, DBG$GET_TEMPMEM
.EXTRN  DBG$NSAVE_FILESP
.EXTRN  DBG$NMATCH, DBG$NPARSE_ADDRESS
.EXTRN  DBG$NNEXT_WORD, DBG$NOUT_INFO
.EXTRN  DBG$NMAKE_ARG_VECT
.EXTRN  DBG$NSYNTAX_ERROR
.EXTRN  DBG$SET_LANG, DBG$READ_KEY_INFO
.EXTRN  SMG$SET_DEFAULT_STATE
.EXTRN  SMG$SET_KEYPAD_MODE
.EXTRN  DBG$GB_KEYPAD_INPUT
.EXTRN  DBG$GL_KEY_TABLE_ID
.EXTRN  DBG$GL_KEYBOARD_ID
.EXTRN  DBG$GB_RADIX, DBG$GL_GBLTYP
.EXTRN  DBG$GW_GBLLNGTH
.EXTRN  DBG$GL_DFLTTYP, DBG$GW_DFLTLENG
.EXTRN  DBG$GB_LANGUAGE
.EXTRN  DBG$GB_SEARCH_PTR
.EXTRN  DBG$GB_MOD_PTR, DBG$GB_DEFINE_PTR
.EXTRN  DBG$RUNFRAME, DBG$GB_RESIGNAL
.EXTRN  DBG$GL_LOG_BUF, DBG$GL_CONTEXT
.EXTRN  DBG$GB_DEF_OUT, DBG$GL_LOGFAB
.EXTRN  DBG$GL_LOGRAB, DBG$GL_LOGNAM
.EXTRN  DBG$GB_LOGFSR, DBG$GB_LOGFSE
.EXTRN  DBG$GL_SCREEN_LOG
.EXTRN  DBG$GB_STP_PTR, DBG$SRC_LEFT_MARGIN
.EXTRN  DBG$SRC_RIGHT_MARGIN
.EXTRN  DBG$GL_ORIG_COMMAND_PTR
.EXTRN  DBG$GL_UPCASE_COMMAND_PTR

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

```
                          0FFC 00000          .ENTRY  DBG$NPARSE_SET, Save R2,R3,R4,R5,R6,R7,R8,- ; 0452
                                                      R9,R10,R11
                5E              18   C2 00002          SUBL2   #24, SP
                                04   DD 00005          PUSHL   #4                                ; 0543
00000000G       00              01   FB 00007          CALLS   #1, DBG$GET_TEMPMEM
                58              50   D0 0000E          MOVL    R0, NOUN_NODE
                59        08    AC   D0 00011          MOVL    VERB_NODE, R9                     ; 0548
          08    A9              58   D0 00015          MOVL    NOUN_NODE, 8(R9)
                                01   DD 00019          PUSHL   #1                                ; 0559
                    00000000'   EF   9F 0001B          PUSHAB  DBG$CS_BREAK
                57        04    AC   D0 00021          MOVL    INPUT_DESC, R7
                                57   DD 00025          PUSHL   R7
00000000G       00              03   FB 00027          CALLS   #3, DBG$NMATCH
                01              50   D1 0002E          CMPL    R0, #1
                                07   12 00031          BNEQ    1$
          01    A9              01   90 00033          MOVB    #1, 1(R9)                         ; 0561
                              1054   31 00037          BRW     205$                              ; 0564
                                01   DD 0003A 1$:      PUSHL   #1                                ; 0569
                    00000000'   EF   9F 0003C          PUSHAB  DBG$CS_DEFINE
                                57   DD 00042          PUSHL   R7
00000000G       00              03   FB 00044          CALLS   #3, DBG$NMATCH
                01              50   D1 0004B          CMPL    R0, #1
                                52   12 0004E          BNEQ    5$
          01    A9              17   90 00050          MOVB    #23, 1(R9)                        ; 0581
                                01   DD 00054          PUSHL   #1                                ; 0590
                    00000000'   EF   9F 00056          PUSHAB  DBG$CS_ADDRESS
                                57   DD 0005C          PUSHL   R7
00000000G       00              03   FB 0005E          CALLS   #3, DBG$NMATCH
                01              50   D1 00065          CMPL    R0, #1
                                03   12 00068          BNEQ    2$
                              0A17   31 0006A          BRW     130$
                                01   DD 0006D 2$:      PUSHL   #1                                ; 0593
                    00000000'   EF   9F 0006F          PUSHAB  DBG$CS_COMMAND
                                57   DD 00075          PUSHL   R7
00000000G       00              03   FB 00077          CALLS   #3, DBG$NMATCH
                01              50   D1 0007E          CMPL    R0, #1
                                03   12 00081          BNEQ    3$
                              0A6D   31 00083          BRW     138$
                                01   DD 00086 3$:      PUSHL   #1                                ; 0602
                    00000000'   EF   9F 00088          PUSHAB  DBG$CS_VALUE
                                57   DD 0008E          PUSHL   R7
00000000G       00              C3   FB 00090          CALLS   #3, DBG$NMATCH
                01              50   D1 00097          CMPL    R0, #1
                                03   13 0009A          BEQL    4$
                              0D0E   31 0009C          BRW     169$
                              02E9   31 0009F 4$:       BRW     37$
                                09   DD 000A2 5$:      PUSHL   #9                                ; 0620
                    00000000'   EF   9F 000A4          PUSHAB  DBG$CS_DEVELOPER
                                57   DD 000AA          PUSHL   R7
00000000G       00              03   FB 000AC          CALLS   #3, DBG$NMATCH
                01              50   D1 000B3          CMPL    R0, #1
                                03   13 000B6          BEQL    6$
                              0084   31 000B8          BRW     13$
          01    A9              16   90 000BB 6$:      MOVB    #22, 1(R9)                        ; 0625
                52        08    A9   9E 000BF          MOVAB   8(R9), LINE                       ; 0626
                                01   DD 000C3          PUSHL   #1                                ; 0627
```

```
                    00000000'   EF  9F 000C5          PUSHAB  DBG$CS_CR                                ;
                                57  DD 000CB          PUSHL   R7                                       ;
00000000G   00                  03  FB 000CD          CALLS   #3, DBG$NMATCH                           ;
            03                  50  E9 000D4          BLBC    R0, 7$                                   ;
                          0CEA  31  000D7          BRW     170$                                     ;
            7E                  57  7D 000DA 7$:      MOVQ    R7, -(SP)                                ; 0632
00000000G   00                  02  FB 000DD          CALLS   #2, DBG$NSAVE_INTEGER                    ;
            03                  50  E8 000E4          BLBS    R0, 8$                                   ;
                          0FF3  31  000E7          BRW     212$                                     ;
                                68  D5 000EA 8$:      TSTL    (NOUN_NODE)                              ; 0636
                                05  19 000EC          BLSS    9$                                       ;
            1F                  68  D1 000EE          CMPL    (NOUN_NODE), #31                         ; 0637
                                09  15 000F1          BLEQ    10$                                      ;
                    00028248    8F  DD 000F3 9$:      PUSHL   #164424                                  ; 0640
                          0CCE  31  000F9          BRW     171$                                     ;
            52          08      A8  9E 000FC 10$:     MOVAB   8(R8), LINK                              ; 0644
                                01  DD 00100          PUSHL   #1                                       ; 0645
                    00000000'   EF  9F 00102          PUSHAB  DBG$CS_COMMA                             ;
                                57  DD 00108          PUSHL   R7                                       ;
00000000G   00                  03  FB 0010A          CALLS   #3, DBG$NMATCH                           ;
            1A                  50  E8 00111          BLBS    R0, 12$                                  ;
                                01  DD 00114          PUSHL   #1                                       ; 0648
                    00000000'   EF  9F 00116          PUSHAB  DBG$CS_CR                                ;
                                57  DD 0011C          PUSHL   R7                                       ;
00000000G   00                  03  FB 0011E          CALLS   #3, DBG$NMATCH                           ;
            03                  50  E8 00125          BLBS    R0, 11$                                  ;
                          0F9C  31  00128          BRW     210$                                     ;
                          0EC9  31  0012B 11$:      BRW     195$                                     ;
                                04  DD 0012E 12$:      PUSHL   #4                                       ; 0660
00000000G   00                  01  FB 00130          CALLS   #1, DBG$GET_TEMPMEM                      ;
            58                  50  D0 00137          MOVL    R0, NOUN_NODE                            ;
            62                  58  D0 0013A          MOVL    NOUN_NODE, (LINK)                        ; 0661
                                9B  11 0013D          BRB     7$                                       ; 0630
                                03  DD 0013F 13$:      PUSHL   #3                                       ; 0679
                    00000000'   EF  9F 00141          PUSHAB  DBG$CS_DISPLAY                           ;
                                57  DD 00147          PUSHL   R7                                       ;
00000000G   00                  03  FB 00149          CALLS   #3, DBG$NMATCH                           ;
            01                  50  D1 00150          CMPL    R0, #1                                   ;
                                13  12 00153          BNEQ    14$                                      ;
      01    A9                  18  90 00155          MOVB    #24, 1(R9)                               ; 0681
                                59  DD 00159          PUSHL   R9                                       ; 0682
                                01  DD 0015B          PUSHL   #1                                       ;
                                57  DD 0015D          PUSHL   R7                                       ;
00000000G   00                  03  FB 0015F          CALLS   #3, DBG$SCR_PARSE_DISPLAY_CMD            ;
                                34  11 00166          BRB     16$                                      ; 0553
                                01  DD 00168 14$:      PUSHL   #1                                       ; 0688
                    00000000'   EF  9F 0016A          PUSHAB  DBG$CS_EXCEPTION                         ;
                                57  DD 00170          PUSHL   R7                                       ;
00000000G   00                  03  FB 00172          CALLS   #3, DBG$NMATCH                           ;
            01                  50  D1 00179          CMPL    R0, #1                                   ;
                                21  12 0017C          BNEQ    17$                                      ;
                                01  DD 0017E          PUSHL   #1                                       ; 0693
                    00000000'   EF  9F 00180          PUSHAB  DBG$CS_BREAK                             ;
                                57  DD 00186          PUSHL   R7                                       ;
00000000G   00                  03  FB 00188          CALLS   #3, DBG$NMATCH                           ;
            03                  50  E8 0018F          BLBS    R0, 15$                                  ;
                          0C18  31  00192          BRW     169$                                     ;
```

```
                01    A9              03  90 00195 15$:    MOVB    #3, 1(R9)                       ; 0707
                            04          A9  7C 00199      CLRQ    4(R9)                           ; 0713
                                      0F42  31 0019C 16$:   BRW     213$                            ; 0553
                                        01  DD 0019F 17$:   PUSHL   #1                             ; 0719
                            00000000'   EF  9F 001A1      PUSHAB  DBG$CS_KEY
                                        57  DD 001A7      PUSHL   R7
00000000G       00                      03  FB 001A9      CALLS   #3, DBG$NMATCH
                01                      50  D1 001B0      CMPL    R0, #1
                                        03  13 001B3      BEQL    18$
                                      0122  31 001B5      BRW     30$
                0D  00000000G           00  E8 001B8 18$: BLBS    DBG$GB_KEYPAD_INPUT, 19$        ; 0735
                    00028D18            8F  DD 001BF      PUSHL   #167192                         ; 0737
00000000G       00                      01  FB 001C5      CALLS   #1, LIB$SIGNAL
                01    A9                1D  90 001CC 19$: MOVB    #29, 1(R9)                      ; 0739
                                        02  DD 001D0      PUSHL   #2                              ; 0744
00000000G       00                      01  FB 001D2      CALLS   #1, DBG$GET_TEMPMEM
                52                      50  D0 001D9      MOVL    R0, TEMP_KEY_DESC
                62  020E0000            8F  D0 001DC      MOVL    #34471936, (TEMP_KEY_DESC)      ; 0745
                            04          A2  D4 001E3      CLRL    4(TEMP_KEY_DESC)                ; 0748
                                        52  DD 001E6      PUSHL   TEMP_KEY_DESC                   ; 0750
                                        7E  D4 001E8      CLRL    -(SP)
                            00000000G   00  9F 001EA      PUSHAB  DBG$GL_KEY_TABLE_ID
00000000G       00                      03  FB 001F0      CALLS   #3, SMG$SET_DEFAULT_STATE
                04    A9                01  D0 001F7      MOVL    #1, 4(R9)                       ; 0754
                68                      52  D0 001FB 20$: MOVL    TEMP_KEY_DESC, (NOUN_NODE)      ; 0756
                                        01  DD 001FE 21$: PUSHL   #1                              ; 0758
                            00000000'   EF  9F 00200      PUSHAB  DBG$CS_CR
                                        57  DD 00206      PUSHL   R7
00000000G       00                      03  FB 00208      CALLS   #3, DBG$NMATCH
                8A                      50  E8 0020F      BLBS    R0, 16$
                                        67  B5 00212      TSTW    (R7)                            ; 0759
                                        86  13 00214      BEQL    16$
                                        01  DD 00216      PUSHL   #1                              ; 0761
                            00000000'   EF  9F 00218      PUSHAB  DBG$CS_SLASH
                                        57  DD 0021E      PUSHL   R7
00000000G       00                      03  FB 00220      CALLS   #3, DBG$NMATCH
                03                      50  E8 00227      BLBS    R0, 22$
                                      0E9A  31 0022A      BRW     210$
                                        03  DD 0022D 22$: PUSHL   #3                              ; 0766
                            00000000'   EF  9F 0022F      PUSHAB  DBG$CS_NOLOG
                                        57  DD 00235      PUSHL   R7
00000000G       00                      03  FB 00237      CALLS   #3, DBG$NMATCH
                01                      50  D1 0023E      CMPL    R0, #1
                                        05  12 00241      BNEQ    24$
                            04          A9  D4 00243      CLRL    4(R9)                           ; 0768
                                        B6  11 00246 23$: BRB     21$                             ; 0763
                                        03  DD 00248 24$: PUSHL   #3                              ; 0771
                            00000000'   EF  9F 0024A      PUSHAB  DBG$CS_NOSTATE
                                        57  DD 00250      PUSHL   R7
00000000G       00                      03  FB 00252      CALLS   #3, DBG$NMATCH
                01                      50  D1 00259      CMPL    R0, #1
                                        13  12 0025C      BNEQ    25$
                                        52  DD 0025E      PUSHL   TEMP_KEY_DESC                   ; 0776
                                        7E  D4 00260      CLRL    -(SP)
                            00000000G   00  9F 00262      PUSHAB  DBG$GL_KEY_TABLE_ID
00000000G       00                      03  FB 00268      CALLS   #3, SMG$SET_DEFAULT_STATE
                                        8A  11 0026F      BRB     20$                             ; 0777
```

```
                                01 DD 00271 25$:    PUSHL   #1                                      ; 0780
                    00000000'   EF 9F 00273         PUSHAB  DBG$CS_LOG
                                57 DD 00279         PUSHL   R7
00000000G   00                  03 FB 0027B         CALLS   #3, DBG$NMATCH
            01                  50 D1 00282         CMPL    R0, #1
                                06 12 00285         BNEQ    26$
      04    A9                  01 D0 00287         MOVL    #1, 4(R9)                               ; 0782
                                B9 11 0028B         BRB     23$                                     ; 0763
                                01 DD 0028D 26$:    PUSHL   #1                                      ; 0785
                    00000000'   EF 9F 0028F         PUSHAB  DBG$CS_STATE
                                57 DD 00295         PUSHL   R7
00000000G   00                  03 FB 00297         CALLS   #3, DBG$NMATCH
            01                  50 D1 0029E         CMPL    R0, #1
                                03 13 002A1         BEQL    28$
                            0B07 31 002A3 27$:      BRW     169$
                                01 DD 002A6 28$:    PUSHL   #1                                      ; 0789
                    00000000'   EF 9F 002A8         PUSHAB  DBG$CS_EQUAL
                                57 DD 002AE         PUSHL   R7
00000000G   00                  03 FB 002B0         CALLS   #3, DBG$NMATCH
            E9                  50 E9 002B7         BLBC    R0, 27$
                          04    A2 D4 002BA         CLRL    4(TEMP_KEY_DESC)                        ; 0800
                                62 B4 002BD         CLRW    (TEMP_KEY_DESC)                         ; 0801
                          0C    AC DD 002BF         PUSHL   MESSAGE_VECT                            ; 0804
                                52 DD 002C2         PUSHL   TEMP_KEY_DESC                           ; 0803
                                57 DD 002C4         PUSHL   R7                                      ; 0802
00000000G   00                  03 FB 002C6         CALLS   #3, DBG$READ_KEY_INFO
            53                  50 D0 002CD         MOVL    R0, STATUS
            03                  53 E9 002D0         BLBC    STATUS, 29$                             ; 0805
                            FF28 31 002D3         BRW     21$
            50                  53 D0 002D6 29$:    MOVL    STATUS, R0                              ; 0807
                                04 0C2D9            RET
                                02 DD 002DA 30$:    PUSHL   #2                                      ; 0828
                    00000000'   EF 9F 002DC         PUSHAB  DBG$CS_LANGUAGE
                                57 DD 002E2         PUSHL   R7
00000000G   00                  03 FB 002E4         CALLS   #3, DBG$NMATCH
            01                  50 D1 002EB         CMPL    R0, #1
                                03 13 002EE         BEQL    31$
                            0131 31 002F0         BRW     49$
      01    A9                  04 90 002F3 31$:    MOVB    #4, 1(R9)                               ; 0845
                                02 DD 002F7         PUSHL   #2                                      ; 0850
                    00000000'   EF 9F 002F9         PUSHAB  DBG$CS_MACRO
                                57 DD 002FF         PUSHL   R7
00000000G   00                  03 FB 00301         CALLS   #3, DBG$NMATCH
            01                  50 D1 00308         CMPL    R0, #1
                                04 12 0030B         BNEQ    32$
                                68 D4 0030D         CLRL    (NOUN_NODE)                             ; 0851
                                7D 11 0030F         BRB     38$
                                02 DD 00311 32$:    PUSHL   #2                                      ; 0853
                    00000000'   EF 9F 00313         PUSHAB  DBG$CS_FORTRAN
                                57 DD 00319         PUSHL   R7
00000000G   00                  03 FB 0031B         CALLS   #3, DBG$NMATCH
            01                  50 D1 00322         CMPL    R0, #1
                                03 12 00325         BNEQ    33$
                            075A 31 00327         BRW     130$
                                02 DD 0032A 33$:    PUSHL   #2                                      ; 0856
                    00000000'   EF 9F 0032C         PUSHAB  DBG$CS_BLISS
                                57 DD 00332         PUSHL   R7
```

```
00000000G  00              03  FB 00334          CALLS   #3, DBG$NMATCH                    ;
           01              50  D1 0033B          CMPL    R0, #1                           ;
                           03  12 0033E          BNEQ    34$                              ;
                       07B0  31 00340          BRW     138$                             ;
                           02  DD 00343 34$:     PUSHL   #2                               ; 0859
                00000000'  EF  9F 00345          PUSHAB  DBG$CS_COBOL                     ;
                           57  DD 0034B          PUSHL   R7                               ;
00000000G  00              03  FB 0034D          CALLS   #3, DBG$NMATCH                   ;
           01              50  D1 00354          CMPL    R0, #1                           ;
                           03  12 00357          BNEQ    35$                              ;
                       075E  31 00359          BRW     134$                             ;
                           02  DD 0035C 35$:     PUSHL   #2                               ; 0862
                00000000'  EF  9F 0035E          PUSHAB  DBG$CS_BASIC                     ;
                           57  DD 00364          PUSHL   R7                               ;
00000000G  00              03  FB 00366          CALLS   #3, DBG$NMATCH                   ;
           01              50  D1 0036D          CMPL    R0, #1                           ;
                           03  12 00370          BNEQ    36$                              ;
                       0760  31 00372          BRW     136$                             ;
                           02  DD 00375 36$:     PUSHL   #2                               ; 0865
                00000000'  EF  9F 00377          PUSHAB  DBG$CS_PLI                       ;
                           57  DD 0037D          PUSHL   R7                               ;
00000000G  00              03  FB 0037F          CALLS   #3, DBG$NMATCH                   ;
           01              50  D1 00386          CMPL    R0, #1                           ;
                           05  12 00389          BNEQ    39$                              ;
           68              05  D0 0038B 37$:     MOVL    #5, (NOUN_NODE)                  ; 0866
                           73  11 0038E 38$:     BRB     45$                              ;
                           02  DD 00390 39$:     PUSHL   #2                               ; 0868
                00000000'  EF  9F 00392          PUSHAB  DBG$CS_PASCAL                    ;
                           57  DD 00398          PUSHL   R7                               ;
00000000G  00              03  FB 0039A          CALLS   #3, DBG$NMATCH                   ;
           01              50  D1 003A1          CMPL    R0, #1                           ;
                           05  12 003A4          BNEQ    40$                              ;
           68              06  D0 003A6          MOVL    #6, (NOUN_NODE)                  ; 0869
                           76  11 003A9          BRB     48$                              ;
                           01  DD 003AB 40$:     PUSHL   #1                               ; 0871
                00000000'  EF  9F 003AD          PUSHAB  DBG$CS_C                         ;
                           57  DD 003B3          PUSHL   R7                               ;
00000000G  00              03  FB 003B5          CALLS   #3, DBG$NMATCH                   ;
           01              50  D1 003BC          CMPL    R0, #1                           ;
                           03  12 003BF          BNEQ    41$                              ;
                       06DB  31 003C1          BRW     132$                             ;
                           02  DD 003C4 41$:     PUSHL   #2                               ; 0874
                00000000'  EF  9F 003C6          PUSHAB  DBG$CS_RPG                       ;
                           57  DD 003CC          PUSHL   R7                               ;
00000000G  00              03  FB 003CE          CALLS   #3, DBG$NMATCH                   ;
           01              50  D1 003D5          CMPL    R0, #1                           ;
                           05  12 003D8          BNEQ    42$                              ;
           68              08  D0 003DA          MOVL    #8, (NOUN_NODE)                  ; 0875
                           42  11 003DD          BRB     48$                              ;
           04 00000000'    EF  E8 003DF 42$:     BLBS    DBG$GL_DEVELOPER, 43$            ; 0877
                           50  D4 003E6          CLRL    R0                               ;
                           11  11 003E8          BRB     44$                              ;
                           02  DD 003EA 43$:     PUSHL   #2                               ; 0878
                00000000'  EF  9F 003EC          PUSHAB  DBG$CS_ADA                       ;
                           57  DD 003F2          PUSHL   R7                               ;
00000000G  00              03  FB 003F4          CALLS   #3, DBG$NMATCH                   ;
           01              50  D1 003FB 44$:     CMPL    R0, #1                           ; 0877
```

```
                        05  12 003FE           BNEQ    46$                                 ;
                    68  09  D0 00400           MOVL    #9, (NOUN_NODE)                     ; 0879
                        1C  11 00403 45$:       BRB     48$                                 ;
                        03  DD 00405 46$:       PUSHL   #3                                  ; 0881
                00000000'   EF  9F 00407        PUSHAB  DBG$CS_UNKNOWN                      ;
                        57  DD 0040D           PUSHL   R7                                  ;
00000000G               00  FB 0040F           CALLS   #3, DBG$NMATCH                      ;
                        01  50 D1 00416        CMPL    R0, #1                              ;
                        03  13 00419           BEQL    47$                                 ;
                      098F  31 0041B           BRW     169$                                ;
                    68  0A  D0 0041E 47$:       MOVL    #10, (NOUN_NODE)                    ; 0882
                      0CBD  31 00421 48$:       BRW     213$                                ;
                        02  DD 00424 49$:       PUSHL   #2                                  ; 0901
                00000000'   EF  9F 00426        PUSHAB  DBG$CS_LOG                          ;
                        57  DD 0042C           PUSHL   R7                                  ;
00000000G               00  FB 0042E           CALLS   #3, DBG$NMATCH                      ;
                        01  50 D1 00435        CMPL    R0, #1                              ;
                        17  12 00438           BNEQ    52$                                 ;
                01      A9  05 90 0043A        MOVB    #5, 1(R9)                           ; 0903
                        0C  AC DD 0043E        PUSHL   MESSAGE_VECT                        ; 0906
                    7E      57 7D 00441        MOVQ    R7, -(SP)                           ; 0905
00000000G               00  FB 00444           CALLS   #3, DBG$NSAVE_FILESP                ;
                        D3  50 E8 0044B 50$:    BLBS    R0, 48$                             ;
                      0C8C  31 0044E 51$:       BRW     212$                                ; 0908
                        03  DD 00451 52$:       PUSHL   #3                                  ; 0911
                00000000'   EF  9F 00453        PUSHAB  DBG$CS_MARGINS                      ;
                        57  DD 00459           PUSHL   R7                                  ;
00000000G               00  FB 0045B           CALLS   #3, DBG$NMATCH                      ;
                        01  50 D1 00462        CMPL    R0, #1                              ;
                        03  13 00465           BEQL    53$                                 ;
                      00B3  31 00467           BRW     60$                                 ;
                01      A9  14 90 0046A 53$:    MOVB    #20, 1(R9)                          ; 0917
                        01  DD 0046E           PUSHL   #1                                  ; 0926
                00000000'   EF  9F 00470        PUSHAB  DBG$CS_COLON                        ;
                        57  DD 00476           PUSHL   R7                                  ;
00000000G               00  FB 00478           CALLS   #3, DBG$NMATCH                      ;
                        1B  50 E9 0047F        BLBC    R0, 54$                             ;
                        0C  AC DD 00482        PUSHL   MESSAGE_VECT                        ; 0934
                        04  AE 9F 00485        PUSHAB  RIGHT_MARGIN                        ; 0933
                        57  DD 00488           PUSHL   R7                                  ;
00000000G               00  FB 0048A           CALLS   #3, DBG$NSAVE_DECIMAL_INTEGER       ;
                        BA  50 E9 00491        BLBC    R0, 51$                             ;
                    52  00000000G  00  D0 00494 MOVL    DBG$SRC_LEFT_MARGIN, LEFT_MARGIN    ; 0937
                        5E  11 0049B           BRB     58$                                 ; 0926
                        0C  AC DD 0049D 54$:    PUSHL   MESSAGE_VECT                        ; 0943
                        04  AE 9F 004A0        PUSHAB  RIGHT_MARGIN                        ; 0942
                        57  DD 004A3           PUSHL   R7                                  ;
00000000G               00  FB 004A5           CALLS   #3, DBG$NSAVE_DECIMAL_INTEGER       ;
                        9F  50 E9 004AC        BLBC    R0, 51$                             ;
                        01  DD 004AF           PUSHL   #1                                  ; 0950
                00000000'   EF  9F 004B1        PUSHAB  DBG$CS_COLON                        ;
                        57  DD 004B7           PUSHL   R7                                  ;
00000000G               00  FB 004B9           CALLS   #3, DBG$NMATCH                      ;
                        35  50 E9 004C0        BLBC    R0, 57$                             ;
                        52  6E D0 004C3        MOVL    RIGHT_MARGIN, LEFT_MARGIN           ; 0957
                        01  DD 004C6           PUSHL   #1                                  ; 0958
                00000000'   EF  9F 004C8        PUSHAB  DBG$CS_CR                           ;
```

```
                          57  DD 004CE          PUSHL   R7                                                      ;
00000000G   00            03  FB 004D0          CALLS   #3, DBG$NMATCH                                          ;
            09            50  E9 004D7          BLBC    R0, 55$                                                 ;
            6E  00000000G 00  D0 004DA          MOVL    DBG$SRC_RIGHT_MARGIN, RIGHT_MARGIN                      ; 0960
                          18  11 004E1          BRB     58$                                                     ;
                    0C    AC  DD 004E3 55$:     PUSHL   MESSAGE_VECT                                            ; 0965
                    04    AE  9F 004E6          PUSHAB  RIGHT_MARGIN                                            ; 0964
                          57  DD 004E9          PUSHL   R7                                                      ;
00000000G   00            03  FB 004EB          CALLS   #3, DBG$NSAVE_DECIMAL_INTEGER                           ;
            06            50  E8 004F2          BLBS    R0, 58$                                                 ;
                    0BE5  31  004F5 56$:        BRW     212$                                                    ; 0967
            52            01  D0 004F8 57$:     MOVL    #1, LEFT_MARGIN                                         ; 0976
            6E            52  D1 004FB 58$:     CMPL    LEFT_MARGIN, RIGHT_MARGIN                               ; 0984
                          14  19 004FE          BLSS    59$                                                     ;
00000000G   00            00  FB 00500          CALLS   #0, DBG$NSHOW_MARGINS                                   ; 0987
                00028E10  8F  DD 00507          PUSHL   #167440                                                 ; 0988
00000000G   00            01  FB 0050D          CALLS   #1, LIB$SIGNAL                                          ;
            68            52  D0 00514 59$:     MOVL    LEFT_MARGIN, (NOUN_NODE)                                ; 0994
      0C    A8            6E  D0 00517          MOVL    RIGHT_MARGIN, 12(NOUN_NODE)                             ; 0995
                          30  11 0051B          BRB     61$                                                     ; 0553
                          03  DD 0051D 60$:     PUSHL   #3                                                      ; 0999
                00000000' EF  9F 0051F          PUSHAB  DBG$CS_MAX_SOURCE_FILES                                 ;
                          57  DD 00525          PUSHL   R7                                                      ;
00000000G   00            03  FB 00527          CALLS   #3, DBG$NMATCH                                          ;
            01            50  D1 0052E          CMPL    R0, #1                                                  ;
                          1D  12 00531          BNEQ    62$                                                     ;
      01    A9            13  90 00533          MOVB    #19, 1(R9)                                              ; 1003
                    0C    AC  DD 00537          PUSHL   MESSAGE_VECT                                            ; 1005
                    08    AE  9F 0053A          PUSHAB  MAX_FILES                                               ; 1004
                          57  DD 0053D          PUSHL   R7                                                      ; 1005
00000000G   00            03  FB 0053F          CALLS   #3, DBG$NSAVE_DECIMAL_INTEGER                           ;
            AC            50  E9 00546          BLBC    R0, 56$                                                 ;
            68      04    AE  D0 00549          MOVL    MAX_FILES, (NOUN_NODE)                                  ; 1008
                    0B91  31  0054D 61$:        BRW     213$                                                    ; 0553
                          01  DD 00550 62$:     PUSHL   #1                                                      ; 1011
                00000000' EF  9F 00552          PUSHAB  DBG$CS_MODE                                             ;
                          57  DD 00558          PUSHL   R7                                                      ;
00000000G   00            03  FB 0055A          CALLS   #3, DBG$NMATCH                                          ;
            01            50  D1 00561          CMPL    R0, #1                                                  ;
                          03  13 00564          BEQL    63$                                                     ;
                    0197  31  00566          BRW     87$                                                     ;
      01    A9            06  90 00569 63$:     MOVB    #6, 1(R9)                                               ; 1032
                          01  DD 0056D 64$:     PUSHL   #1                                                      ; 1042
                00000000' EF  9F 0056F          PUSHAB  DBG$CS_BINARY                                           ;
                          57  DD 00575          PUSHL   R7                                                      ;
00000000G   00            03  FB 00577          CALLS   #3, DBG$NMATCH                                          ;
            01            50  D1 0057E          CMPL    R0, #1                                                  ;
                          06  12 00581          BNEQ    65$                                                     ;
            68            01  D0 00583          MOVL    #1, (NOUN_NODE)                                         ; 1043
                    0089  31  00586          BRW     70$                                                     ;
                          03  DD 00589 65$:     PUSHL   #3                                                      ; 1045
                00000000' EF  9F 0058B          PUSHAB  DBG$CS_DEFAULT                                          ;
                          57  DD 00591          PUSHL   R7                                                      ;
00000000G   00            03  FB 00593          CALLS   #3, DBG$NMATCH                                          ;
            01            50  D1 0059A          CMPL    R0, #1                                                  ;
                          06  12 0059D          BNEQ    66$                                                     ;
            68            07  D0 0059F          MOVL    #7, (NOUN_NODE)                                         ; 1046
```

```
                 0089  31 005A2          BRW     72$
                   01  DD 005A5 66$:     PUSHL   #1                                      1048
        00000000'  EF  9F 005A7          PUSHAB  DBG$CS_DECIMAL
                   57  DD 005AD          PUSHL   R7
00000000G  00      03  FB 005AF          CALLS   #3, DBG$NMATCH
           01      50  D1 005B6          CMPL    R0, #1
                   06  12 005B9          BNEQ    67$
           68      03  D0 005BB          MOVL    #3, (NOUN_NODE)                         1049
                 0088  31 005BE          BRW     74$
                   01  DD 005C1 67$:     PUSHL   #1                                      1051
        00000000'  EF  9F 005C3          PUSHAB  DBG$CS_HEXADECIMAL
                   57  DD 005C9          PUSHL   R7
00000000G  00      03  FB 005CB          CALLS   #3, DBG$NMATCH
           01      50  D1 005D2          CMPL    R0, #1
                   06  12 005D5          BNEQ    68$
           68      04  D0 005D7          MOVL    #4, (NOUN_NODE)                         1052
                 0087  31 005DA          BRW     76$
                   01  DD 005DD 68$:     PUSHL   #1                                      1054
        00000000'  EF  9F 005DF          PUSHAB  DBG$CS_KEYPAD
                   57  DD 005E5          PUSHL   R7
00000000G  00      03  FB 005E7          CALLS   #3, DBG$NMATCH
           01      50  D1 005EE          CMPL    R0, #1
                   06  12 005F1          BNEQ    69$
           68      0A  D0 005F3          MOVL    #10, (NOUN_NODE)                        1055
                 0086  31 005F6          BRW     78$
                   03  DD 005F9 69$:     PUSHL   #3                                      1057
        00000000'  EF  9F 005FB          PUSHAB  DBG$CS_NOKEYPAD
                   57  DD 00601          PUSHL   R7
00000000G  00      03  FB 00603          CALLS   #3, DBG$NMATCH
           01      50  D1 0060A          CMPL    R0, #1
                   06  12 0060D          BNEQ    71$
           68      0B  D0 0060F          MOVL    #11, (NOUN_NODE)                        1058
                 0085  31 00612 70$:     BRW     80$
                   05  DD 00615 71$:     PUSHL   #5                                      1060
        00000000'  EF  9F 00617          PUSHAB  DBG$CS_NOSCREEN
                   57  DD 0061D          PUSHL   R7
00000000G  00      03  FB 0061F          CALLS   #3, DBG$NMATCH
           01      50  D1 00626          CMPL    R0, #1
                   05  12 00629          BNEQ    73$
           68      09  D0 0062B          MOVL    #9, (NOUN_NODE)                         1061
                   19  11 0062E 72$:     BRB     74$
                   01  DD 00630 73$:     PUSHL   #1                                      1063
        00000000'  EF  9F 00632          PUSHAB  DBG$CS_NOSYMBOLS
                   57  DD 00638          PUSHL   R7
00000000G  00      03  FB 0063A          CALLS   #3, DBG$NMATCH
           01      50  D1 00641          CMPL    R0, #1
                   05  12 00644          BNEQ    75$
           68      06  D0 00646          MOVL    #6, (NOUN_NODE)                         1064
                   6A  11 00649 74$:     BRB     82$
                   01  DD 0064B 75$:     PUSHL   #1                                      1066
        00000000'  EF  9F 0064D          PUSHAB  DBG$CS_OCTAL
                   57  DD 00653          PUSHL   R7
00000000G  00      03  FB 00655          CALLS   #3, DBG$NMATCH
           01      50  D1 0065C          CMPL    R0, #1
                   05  12 0065F          BNEQ    77$
           68      02  D0 00661          MOVL    #2, (NOUN_NODE)                         1067
                   6D  11 00664 76$:     BRB     85$
```

```
                    03  DD 00666 77$:    PUSHL   #3                                      ;1069
         00000000'  EF  9F 00668          PUSHAB  DBG$CS_SCREEN                          ;
                    57  DD 0066E          PUSHL   R7                                     ;
00000000G   00      03  FB 00670          CALLS   #3, DBG$NMATCH                         ;
            01      50  D1 00677          CMPL    R0, #1                                 ;
                    05  12 0067A          BNEQ    79$                                    ;
            68      08  D0 0067C          MOVL    #8, (NOUN_NODE)                        ;1070
                    52  11 0067F 78$:     BRB     85$                                    ;
                    01  DD 00681 79$:     PUSHL   #1                                     ;1072
         00000000'  EF  9F 00683          PUSHAB  DBG$CS_SYMBOLS                         ;
                    57  DD 00689          PUSHL   R7                                     ;
00000000G   00      03  FB 0068B          CALLS   #3, DBG$NMATCH                         ;
            01      50  D1 00692          CMPL    R0, #1                                 ;
                    05  12 00695          BNEQ    81$                                    ;
            68      05  D0 00697          MOVL    #5, (NOUN_NODE)                        ;1073
                    37  11 0069A 80$:     BRB     85$                                    ;
                    01  DD 0069C 81$:     PUSHL   #1                                     ;1075
         00000000'  EF  9F 0069E          PUSHAB  DBG$CS_G_FLOAT                         ;
                    57  DD 006A4          PUSHL   R7                                     ;
00000000G   00      03  FB 006A6          CALLS   #3, DBG$NMATCH                         ;
            01      50  D1 006AD          CMPL    R0, #1                                 ;
                    05  12 006B0          BNEQ    83$                                    ;
            68      0C  D0 006B2          MOVL    #12, (NOUN_NODE)                       ;1076
                    1C  11 006B5 82$:     BRB     85$                                    ;
                    03  DD 006B7 83$:     PUSHL   #3                                     ;1078
         00000000'  EF  9F 006B9          PUSHAB  DBG$CS_NOG_FLOAT                       ;
                    57  DD 006BF          PUSHL   R7                                     ;
00000000G   00      03  FB 006C1          CALLS   #3, DBG$NMATCH                         ;
            01      50  D1 006C8          CMPL    R0, #1                                 ;
                    03  13 006CB          BEQL    84$                                    ;
                06DD  31 006CD          BRW     169$                                   ;
            68      0D  D0 006D0 84$:     MOVL    #13, (NOUN_NODE)                       ;1079
                    01  DD 006D3 85$:     PUSHL   #1                                     ;1099
         00000000'  EF  9F 006D5          PUSHAB  DBG$CS_COMMA                           ;
                    57  DD 006DB          PUSHL   R7                                     ;
00000000G   00      03  FB 006DD          CALLS   #3, DBG$NMATCH                         ;
            03      50  E8 006E4          BLBS    R0, 86$                                ;
                053A  31 006E7          BRW     152$                                   ;
            5B  08  A8  9E 006EA 86$:     MOVAB   8(R8), LINK                            ;1106
                    04  DD 006EE          PUSHL   #4                                     ;1107
00000000G   00      01  FB 006F0          CALLS   #1, DBG$GET_TEMPMEM                    ;
            58      50  D0 006F7          MOVL    R0, NOUN_NODE                          ;
            6B      58  D0 006FA          MOVL    NOUN_NODE, (LINK)                      ;1108
                FE6D  31 006FD          BRW     64$                                    ;1036
                    04  DD 00700 87$:     PUSHL   #4                                     ;1119
         00000000'  EF  9F 00702          PUSHAB  DBG$CS_MODULE                          ;
                    57  DD 00708          PUSHL   R7                                     ;
00000000G   00      03  FB 0070A          CALLS   #3, DBG$NMATCH                         ;
            01      50  D1 00711          CMPL    R0, #1                                 ;
                    03  13 00714          BEQL    88$                                    ;
                015A  31 00716          BRW     103$                                   ;
                    52  D4 00719 88$:     CLRL    ALLOCATE_FLAG                          ;1131
                    53  7C 0071B          CLRQ    NOWAIT_FLAG                            ;1132
                    01  DD 0071D 89$:     PUSHL   #1                                     ;1137
         00000000'  EF  9F 0071F          PUSHAB  DBG$CS_SLASH                           ;
                    57  DD 00725          PUSHL   R7                                     ;
00000000G   00      03  FB 00727          CALLS   #3, DBG$NMATCH                         ;
```

```
                55              50  E9 0072E          BLBC    R0, 94$                                                 ;
                                04  DD 00731          PUSHL   #4                                                      ; 1144
                    00000000'   EF  9F 00733          PUSHAB  DBG$CS_ALLOCATE                                         ;
                                57  DD 00739          PUSHL   R7                                                      ;
00000000G       00              03  FB 0073B          CALLS   #3, DBG$NMATCH                                          ;
                05              50  E9 00742          BLBC    R0, 90$                                                 ;
                52              01  D0 00745          MOVL    #1, ALLOCATE_FLAG                                       ; 1146
                                D3  11 00748          BRB     89$                                                     ;
                                01  DD 0074A 90$:     PUSHL   #1                                                      ; 1148
                    00000000'   EF  9F 0074C          PUSHAB  DBG$CS_ALL                                              ;
                                57  DD 00752          PUSHL   R7                                                      ;
00000000G       00              03  FB 00754          CALLS   #3, DBG$NMATCH                                          ;
                05              50  E9 0075B          BLBC    R0, 91$                                                 ;
                54              01  D0 0075E          MOVL    #1, ALL_FLAG                                            ; 1150
                                BA  11 00761          BRB     89$                                                     ;
                03 00000000'    EF  E8 00763 91$:     BLBS    DBG$GL_DEVELOPER, 93$                                   ; 1152
                          0640  31  00766 92$:     BRW     169$                                                    ;
                                01  DD 00769 93$:     PUSHL   #1                                                      ; 1153
                    00000000'   EF  9F 0076B          PUSHAB  DBG$CS_NOWAIT                                           ;
                                57  DD 00771          PUSHL   R7                                                      ;
00000000G       00              03  FB 00773          CALLS   #3, DBG$NMATCH                                          ;
                E9              50  E9 0077A          BLBC    R0, 92$                                                 ;
                53              01  D0 0077D          MOVL    #1, NOWAIT_FLAG                                         ; 1155
                                97  11 00780          BRB     89$                                                     ;
            04  A9          6342   3E 00782 94$:     MOVAW   (NOWAIT_FLAG)[ALLOCATE_FLAG], 4(R9)                     ; 1181
                07              54  E9 00787          BLBC    ALL_FLAG, 95$                                           ; 1187
            01  A9              08  90 0078A          MOVB    #8, 1(R9)                                               ; 1189
                          094C  31  0078E          BRW     213$                                                    ;
                07 00000000G    00  91 00791 95$:     CMPB    DBG$GB_LANGUAGE, #7                                     ; 1215
                                03  13 00798          BEQL    96$                                                     ;
                          008B  31  0079A          BRW     99$                                                     ;
                    52      04  A7  D0 0079D 96$:     MOVL    4(R7), POINTER                                          ; 1228
                    56          67  3C 007A1          MOVZWL  (R7), LENGTH                                            ; 1229
00000000G           00          52  D1 007A4          CMPL    POINTER, DBG$GL_UPCASE_COMMAND_PTR                      ; 1230
                                09  19 007AB          BLSS    97$                                                     ;
00000000G           00          52  D1 007AD          CMPL    POINTER, DBG$GL_UPCASE_COMMAND_PTR+4                    ; 1231
                                15  15 007B4          BLEQ    98$                                                     ;
                    00000000'   EF  9F 007B6 97$:     PUSHAB  P.ACN                                                   ; 1233
                                01  DD 007BC          PUSHL   #1                                                      ;
                    00028362    8F  DD 007BE          PUSHL   #164706                                                 ;
00000000G           00              03  FB 007C4          CALLS   #3, LIB$SIGNAL                                          ;
                    50      03  A6  9E 007CB 98$:     MOVAB   3(R6), R0                                               ; 1239
7E                  50              04  C7 007CF          DIVL3   #4, R0, -(SP)                                           ;
00000000G           00              01  FB 007D3          CALLS   #1, DBG$GET_TEMPMEM                                     ;
                    5A              50  D0 007DA          MOVL    R0, NEW_POINTER                                         ;
                    52  00000000G   00  C2 007DD          SUBL2   DBG$GL_UPCASE_COMMAND_PTR, R2                           ; 1240
50                  52  00000000G   00  C1 007E4          ADDL3   DBG$GL_ORIG_COMMAND_PTR, R2, TEMP_PTR                   ; 1241
6A                  60              56  28 007EC          MOVC3   LENGTH, (TEMP_PTR), (NEW_POINTER)                       ; 1242
            FF  A64A                0D  90 007F0          MOVB    #13, -1(LENGTH)[NEW_POINTER]                            ; 1243
            0E  AE          010E    8F  B0 007F5          MOVW    #270, STG_DESC+2                                        ; 1248
            0C  AE                  56  B0 007FB          MOVW    LENGTH, STG_DESC                                        ; 1249
            10  AE                  5A  D0 00803          MOVL    NEW_POINTER, STG_DESC+4                                 ; 1250
                              14    AE  D4 00807          CLRL    STG_DESC+8                                              ; 1251
                              0C    AC  DD 0080A          PUSHL   MESSAGE_VECT                                            ; 1257
                                    58  DD 0080D          PUSHL   NOUN_NODE                                               ; 1256
                              14    AE  9F 0080F          PUSHAB  STG_DESC                                                ; 1255
00000000G           00              03  FB 00812          CALLS   #3, DBG$NSAVE_STRING                                    ; 1256
```

```
                1D            50  E9 00819          BLBC    R0, 100$                                ;
                50        0C  AE  3C 0081C          MOVZWL  STG_DESC, R0                            ; 1264
                50            56  C2 00820          SUBL2   LENGTH, R0                              ;
                67            50  A0 00823          ADDW2   R0, (R7)                                ;
              04 A7           50  C2 00826          SUBL2   R0, 4(R7)                               ; 1266
                              13  11 0082A          BRB     101$                                    ; 1215
                        0C    AC  DD 0082C 99$:     PUSHL   MESSAGE_VECT                            ; 1275
                7E            57  7D 0082F          MOVQ    R7, -(SP)                               ;
00000000G       00            03  FB 00832          CALLS   #3, DBG$NSAVE_STRING                    ;
                03            50  E8 00839 100$:     BLBS    R0, 101$                                ;
                          089E    31 0083C          BRW     212$                                    ;
                            01    DD 0083F 101$:     PUSHL   #1                                      ; 1282
                  00000000'   EF  9F 00841          PUSHAB  DBG$CS_COMMA                            ;
                            57  DD 00847          PUSHL   R7                                      ;
00000000G       00            03  FB 00849          CALLS   #3, DBG$NMATCH                          ;
                16            50  E9 00850          BLBC    R0, 102$                                ;
                5B        08  A8  9E 00853          MOVAB   8(R8), LINK                             ; 1289
                            04  DD 00857          PUSHL   #4                                      ; 1290
00000000G       00            01  FB 00859          CALLS   #1, DBG$GET_TEMPMEM                     ;
                58            50  D0 00860          MOVL    R0, NOUN_NODE                           ;
                6B            58  D0 00863          MOVL    NOUN_NODE, (LINK)                       ; 1291
                          FF2C    31 00866          BRW     95$                                     ; 1207
                        08  A8  D4 00869 102$:     CLRL    8(NOUN_NODE)                            ; 1298
              01 A9             07  90 0086C          MOVB    #7, 1(R9)                               ; 1299
                          086E    31 00870          BRW     213$                                    ; 0553
                            01    DD 00873 103$:     PUSHL   #1                                      ; 1304
                  00000000'   EF  9F 00875          PUSHAB  DBG$CS_OUTPUT                           ;
                            57  DD 00878          PUSHL   R7                                      ;
00000000G       00            03  FB 0087D          CALLS   #3, DBG$NMATCH                          ;
                01            50  D1 00884          CMPL    R0, #1                                  ;
                              03  13 00887          BEQL    104$                                    ;
                          010B    31 00889          BRW     119$                                    ;
              01 A9             09  90 0088C 104$:     MOVB    #9, 1(R9)                               ; 1333
                            01    DD 00890 105$:     PUSHL   #1                                      ; 1343
                  00000000'   EF  9F 00892          PUSHAB  DBG$CS_LOG                              ;
                            57  DD 00898          PUSHL   R7                                      ;
00000000G       00            03  FB 0089A          CALLS   #3, DBG$NMATCH                          ;
                01            50  D1 008A1          CMPL    R0, #1                                  ;
                              06  12 008A4          BNEQ    106$                                    ;
                68            01  D0 008A6          MOVL    #1, (NOUN_NODE)                         ; 1344
                          0085    31 008A9          BRW     112$                                    ;
                            03    DD 008AC 106$:     PUSHL   #3                                      ; 1346
                  00000000'   EF  9F 008AE          PUSHAB  DBG$CS_NOLOG                            ;
                            57  DD 008B4          PUSHL   R7                                      ;
00000000G       00            03  FB 008B6          CALLS   #3, DBG$NMATCH                          ;
                01            50  D1 008BD          CMPL    R0, #1                                  ;
                              05  12 008C0          BNEQ    107$                                    ;
                68            02  D0 008C2          MOVL    #2, (NOUN_NODE)                         ; 1347
                              19  11 008C5          BRB     108$                                    ;
                            01    DD 008C7 107$:     PUSHL   #1                                      ; 1349
                  00000000'   EF  9F 008C9          PUSHAB  DBG$CS_SCREEN_LOG                       ;
                            57  DD 008CF          PUSHL   R7                                      ;
00000000G       00            03  FB 008D1          CALLS   #3, DBG$NMATCH                          ;
                01            50  D1 008D8          CMPL    R0, #1                                  ;
                              05  12 008DB          BNEQ    109$                                    ;
                68            03  D0 008DD          MOVL    #3, (NOUN_NODE)                         ; 1350
                              6A  11 008E0 108$:     BRB     114$                                    ;
```

```
                        03   DD 008E2 109$:   PUSHL   #3                                    ; 1352
                00000000'   EF   9F 008E4     PUSHAB  DBG$CS_NOSCREEN_LOG
                        57   DD 008EA         PUSHL   R7
00000000G       00          03   FB 008EC     CALLS   #3, DBG$NMATCH
                01          50   D1 008F3     CMPL    R0, #1
                            05   12 008F6     BNEQ    110$
                68          04   D0 008F8     MOVL    #4, (NOUN_NODE)                    ; 1353
                            6D   11 008FB     BRB     117$
                            01   DD 008FD 110$:   PUSHL   #1                             ; 1355
                00000000'   EF   9F 008FF     PUSHAB  DBG$CS_TERMINAL
                        57   DD 00905         PUSHL   R7
00000000G       00          03   FB 00907     CALLS   #3, DBG$NMATCH
                01          50   D1 0090E     CMPL    R0, #1
                            05   12 00911     BNEQ    111$
                68          05   D0 00913     MOVL    #5, (NOUN_NODE)                    ; 1356
                            52   11 00916     BRB     117$
                            03   DD 00918 111$:   PUSHL   #3                             ; 1358
                00000000'   EF   9F 0091A     PUSHAB  DBG$CS_NOTERMINAL
                        57   DD 00920         PUSHL   R7
00000000G       00          03   FB 00922     CALLS   #3, DBG$NMATCH
                01          50   D1 00929     CMPL    R0, #1
                            05   12 0092C     BNEQ    113$
                68          06   D0 0092E     MOVL    #6, (NOUN_NODE)                    ; 1359
                            37   11 00931 112$:   BRB     117$
                            01   DD 00933 113$:   PUSHL   #1                             ; 1361
                00000000'   EF   9F 00935     PUSHAB  DBG$CS_VERIFY
                        57   DD 0093B         PUSHL   R7
00000000G       00          03   FB 0093D     CALLS   #3, DBG$NMATCH
                01          50   D1 00944     CMPL    R0, #1
                            05   12 00947     BNEQ    115$
                68          07   D0 00949     MOVL    #7, (NOUN_NODE)                    ; 1362
                            1C   11 0094C 114$:   BRB     117$
                            03   DD 0094E 115$:   PUSHL   #3                             ; 1364
                00000000'   EF   9F 00950     PUSHAB  DBG$CS_NOVERIFY
                        57   DD 00956         PUSHL   R7
00000000G       00          03   FB 00958     CALLS   #3, DBG$NMATCH
                01          50   D1 0095F     CMPL    R0, #1
                            03   13 00962     BEQL    116$
                        0446   31 00964     BRW     169$
                68          08   D0 00967 116$:   MOVL    #8, (NOUN_NODE)                ; 1365
                            01   DD 0096A 117$:   PUSHL   #1                             ; 1384
                00000000'   EF   9F 0096C     PUSHAB  DBG$CS_COMMA
                        57   DD 00972         PUSHL   R7
00000000G       00          03   FB 00974     CALLS   #3, DBG$NMATCH
                03          50   E8 0097B     BLBS    R0, 118$
                        02A3   31 0097E     BRW     152$
                52      08  A8   9E 00981 118$:   MOVAB   8(R8), LINK                    ; 1391
                            04   DD 00985     PUSHL   #4                                 ; 1392
00000000G       00          01   FB 00987     CALLS   #1, DBG$GET_TEMPMEM
                58          50   D0 0098E     MOVL    R0, NOUN_NODE
                62          58   D0 00991     MOVL    NOUN_NODE, (LINK)                  ; 1393
                        FEF9   31 00994     BRW     105$                               ; 1338
                            01   DD 00997 119$:   PUSHL   #1                             ; 1404
                00000000'   EF   9F 00999     PUSHAB  DBG$CS_RADIX
                        57   DD 0099F         PUSHL   R7
00000000G       00          03   FB 009A1     CALLS   #3, DBG$NMATCH
                01          50   D1 009A8     CMPL    R0, #1
```

```
                    03  13 009AB           BEQL    120$
                  0149  31 009AD           BRW     140$
      01    A9          1B  90 009B0 120$:  MOVB    #27, 1(R9)                      1420
                        01  DD 009B4 121$:  PUSHL   #1                              1421
                00000000' EF  9F 009B6       PUSHAB  DBG$CS_SLASH
                        57  DD 009BC         PUSHL   R7
00000000G       00        03  FB 009BE       CALLS   #3, DBG$NMATCH
                03        50  E8 009C5       BLBS    R0, 122$
                        00A3  31 009C8       BRW     129$
                        01  DD 009CB 122$:  PUSHL   #1                              1425
                00000000' EF  9F 009CD       PUSHAB  DBG$CS_INPUT
                        57  DD 009D3         PUSHL   R7
00000000G       00        03  FB 009D5       CALLS   #3, DBG$NMATCH
                01        50  D1 009DC       CMPL    R0, #1
                        06  12 009DF         BNEQ    124$
      0C    A8          01  D0 009E1         MOVL    #1, 12(NOUN_NODE)               1426
                        CD  11 009E5 123$:  BRB     121$                             1428
                        02  DD 009E7 124$:  PUSHL   #2
                00000000' EF  9F 009E9       PUSHAB  DBG$CS_OUTPUT
                        57  DD 009EF         PUSHL   R7
00000000G       00        03  FB 009F1       CALLS   #3, DBG$NMATCH
                01        50  D1 009F8       CMPL    R0, #1
                        06  12 009FB         BNEQ    126$
      0C    A8          02  D0 009FD         MOVL    #2, 12(NOUN_NODE)               1429
                        B1  11 00A01 125$:  BRB     121$                             1431
                        02  DD 00A03 126$:  PUSHL   #2
                00000000' EF  9F 00A05       PUSHAB  DBG$CS_OVERRIDE
                        57  DD 00A0B         PUSHL   R7
00000000G       00        03  FB 00A0D       CALLS   #3, DBG$NMATCH
                01        50  D1 00A14       CMPL    R0, #1
                        06  12 00A17         BNEQ    127$
      01    A9          1C  90 00A19         MOVB    #28, 1(R9)                      1432
                        95  11 00A1D         BRB     121$
                        01  DD 00A1F 127$:  PUSHL   #1                              1439
                00000000' EF  9F 00A21       PUSHAB  DBG$CS_CR
                        57  DD 00A27         PUSHL   R7
00000000G       00        03  FB 00A29       CALLS   #3, DBG$NMATCH
                0F        50  E9 00A30       BLBC    R0, 128$
                000280D0  8F  DD 00A33       PUSHL   #164048                         1441
00000000G       00        01  FB 00A39       CALLS   #1, LIB$SIGNAL
                        A3  11 00A40         BRB     123$
                        57  DD 00A42 128$:  PUSHL   R7                              1444
00000000G       00        01  FB 00A44       CALLS   #1, DBG$NNEXT_WORD
      0E    AE      010E    8F  B0 00A4B       MOVW    #270, STG_DESC+2             1446
      0C    AE              60  9B 00A51       MOVZBW  (CS), STG_DESC               1447
      10    AE        01    A0  9E 00A55       MOVAB   1(R0), STG_DESC+4            1448
                      0C    AE  9F 00A5A       PUSHAB  STG_DESC                     1449
                        01  DD 00A5D         PUSHL   #1
                00028238  8F  DD 00A5F       PUSHL   #164408
00000000G       00        03  FB 00A65       CALLS   #3, LIB$SIGNAL
                        93  11 00A6C         BRB     125$                         1422
                        01  DD 00A6E 129$:  PUSHL   #1                              1459
                00000000' EF  9F 00A70       PUSHAB  DBG$CS_BINARY
                        57  DD 00A76         PUSHL   R7
00000000G       00        03  FB 00A78       CALLS   #3, DBG$NMATCH
                01        50  D1 00A7F       CMPL    R0, #1
                        05  12 00A82         BNEQ    131$
```

```
                68              01  D0 00A84 130$:  MOVL    #1, (NOUN_NODE)                 ; 1460
                                6D  11 00A87        BRB     139$
                                03  DD 00A89 131$:  PUSHL   #3                              ; 1462
                    00000000'   EF  9F 00A8B        PUSHAB  DBG$CS_DEFAULT
                                57  DD 00A91        PUSHL   R7
    00000000G   00              03  FB 00A93        CALLS   #3, DBG$NMATCH
                01              50  D1 00A9A        CMPL    R0, #1
                                05  12 00A9D        BNEQ    133$
                68              07  D0 00A9F 132$:  MOVL    #7, (NOUN_NODE)                 ; 1463
                                52  11 00AA2        BRB     139$
                                01  DD 00AA4 133$:  PUSHL   #1                              ; 1465
                    00000000'   EF  9F 00AA6        PUSHAB  DBG$CS_DECIMAL
                                57  DD 00AAC        PUSHL   R7
    00000000G   00              03  FB 00AAE        CALLS   #3, DBG$NMATCH
                01              50  D1 00AB5        CMPL    R0, #1
                                05  12 00AB8        BNEQ    135$
                68              03  D0 00ABA 134$:  MOVL    #3, (NOUN_NODE)                 ; 1466
                                37  11 00ABD        BRB     139$
                                01  DD 00ABF 135$:  PUSHL   #1                              ; 1468
                    00000000'   EF  9F 00AC1        PUSHAB  DBG$CS_HEXADECIMAL
                                57  DD 00AC7        PUSHL   R7
    00000000G   00              03  FB 00AC9        CALLS   #3, DBG$NMATCH
                01              50  D1 00AD0        CMPL    R0, #1
                                05  12 00AD3        BNEQ    137$
                68              04  D0 00AD5 136$:  MOVL    #4, (NOUN_NODE)                 ; 1469
                                1C  11 00AD8        BRB     139$
                                01  DD 00ADA 137$:  PUSHL   #1                              ; 1471
                    00000000'   EF  9F 00ADC        PUSHAB  DBG$CS_OCTAL
                                57  DD 00AE2        PUSHL   R7
    00000000G   00              03  FB 00AE4        CALLS   #3, DBG$NMATCH
                01              50  D1 00AEB        CMPL    R0, #1
                                03  13 00AEE        BEQL    138$
                            02BA  31 00AF0        BRW     169$
                68              02  D0 00AF3 138$:  MOVL    #2, (NOUN_NODE)                 ; 1472
                            05E8  31 00AF6 139$:  BRW     213$
                                02  DD 00AF9 140$:  PUSHL   #2                              ; 1490
                    00000000'   EF  9F 00AFB        PUSHAB  DBG$CS_SCOPE
                                57  DD 00B01        PUSHL   R7
    00000000G   00              03  FB 00B03        CALLS   #3, DBG$NMATCH
                01              50  D1 00B0A        CMPL    R0, #1
                                61  12 00B0D        BNEQ    144$
          01    A9              0A  90 00B0F        MOVB    #10, 1(R9)                      ; 1492
                                01  DD 00B13        PUSHL   #1                              ; 1497
                    00000000'   EF  9F 00B15        PUSHAB  DBG$CS_SLASH
                                57  DD 00B1B        PUSHL   R7
    00000000G   00              03  FB 00B1D        CALLS   #3, DBG$NMATCH
                36              50  E9 00B24        BLBC    R0, 142$
                                01  DD 00B27        PUSHL   #1                              ; 1499
                    00000000'   EF  9F 00B29        PUSHAB  DBG$CS_MODULE
                                57  DD 00B2F        PUSHL   R7
    00000000G   00              03  FB 00B31        CALLS   #3, DBG$NMATCH
                06              50  E9 00B38        BLBC    R0, 141$
          04    A9              01  D0 00B3B        MOVL    #1, 4(R9)                       ; 1501
                                1F  11 00B3F        BRB     143$
                                57  DD 00B41 141$:  PUSHL   R7                              ; 1503
    00000000G   00              01  FB 00B43        CALLS   #1, DBG$NNEXT_WORD
                                50  DD 00B4A        PUSHL   R0
```

```
                                01  DD 00B4C          PUSHL   #1                                      ;
                    00028A8A    8F  DD 00B4E          PUSHL   #166538                                 ;
00000000G   00                  03  FB 00B54          CALLS   #3, LIB$SIGNAL                          ;
                                03  11 00B5B          BRB     143$                                    ;  1499
                          04    A9  D4 00B5D 142$:     CLRL    4(R9)                                   ;  1505
                          0C    AC  DD 00B60 143$:     PUSHL   MESSAGE_VECT                            ;  1508
                       7E       57  7D 00B63          MOVQ    R7, -(SP)                               ;
00000000G   00                  03  FB 00B66          CALLS   #3, DBG$NPARSE_SCOPE_LIST               ;
                          F8DB  31  00B6D          BRW     50$                                     ;
                                02  DD 00B70 144$:     PUSHL   #2                                      ;  1514
                    00000000'   EF  9F 00B72          PUSHAB  DBG$CS_SEARCH                           ;
                                57  DD 00B78          PUSHL   R7                                      ;
00000000G   00                  03  FB 00B7A          CALLS   #3, DBG$NMATCH                          ;
            01                  50  D1 00B81          CMPL    R0, #1                                  ;
                                03  13 00B84          BEQL    145$                                    ;
                          00A1  31  00B86          BRW     153$                                    ;
         01 A9                  15  90 00B89 145$:     MOVB    #21, 1(R9)                              ;  1535
                                01  DD 00B8D 146$:     PUSHL   #1                                      ;  1544
                    00000000'   EF  9F 00B8F          PUSHAB  DBG$CS_ALL                              ;
                                57  DD 00B95          PUSHL   R7                                      ;
00000000G   00                  03  FB 00B97          CALLS   #3, DBG$NMATCH                          ;
            01                  50  D1 00B9E          CMPL    R0, #1                                  ;
                                05  12 00BA1          BNEQ    147$                                    ;
            68                  01  D0 00BA3          MOVL    #1, (NOUN_NODE)                         ;  1545
                                52  11 00BA6          BRB     151$                                    ;
                                01  DD 00BA8 147$:     PUSHL   #1                                      ;  1547
                    00000000'   EF  9F 00BAA          PUSHAB  DBG$CS_IDENT                            ;
                                57  DD 00BB0          PUSHL   R7                                      ;
00000000G   00                  03  FB 00BB2          CALLS   #3, DBG$NMATCH                          ;
            01                  50  D1 00BB9          CMPL    R0, #1                                  ;
                                05  12 00BBC          BNEQ    148$                                    ;
            68                  04  D0 00BBE          MOVL    #4, (NOUN_NODE)                         ;  1548
                                37  11 00BC1          BRB     151$                                    ;
                                01  DD 00BC3 148$:     PUSHL   #1                                      ;  1550
                    00000000'   EF  9F 00BC5          PUSHAB  DBG$CS_NEXT                             ;
                                57  DD 00BCB          PUSHL   R7                                      ;
00000000G   00                  03  FB 00BCD          CALLS   #3, DBG$NMATCH                          ;
            01                  50  D1 00BD4          CMPL    R0, #1                                  ;
                                05  12 00BD7          BNEQ    149$                                    ;
            68                  02  D0 00BD9          MOVL    #2, (NOUN_NODE)                         ;  1551
                                1C  11 00BDC          BRB     151$                                    ;
                                01  DD 00BDE 149$:     PUSHL   #1                                      ;  1553
                    00000000'   EF  9F 00BE0          PUSHAB  DBG$CS_STRING                           ;
                                57  DD 00BE6          PUSHL   R7                                      ;
00000000G   00                  03  FB 00BE8          CALLS   #3, DBG$NMATCH                          ;
            01                  50  D1 00BEF          CMPL    R0, #1                                  ;
                                03  13 00BF2          BEQL    150$                                    ;
                          01B6  31  00BF4          BRW     169$                                    ;
            68                  03  D0 00BF7 150$:     MOVL    #3, (NOUN_NODE)                         ;  1554
                                01  DD 00BFA 151$:     PUSHL   #1                                      ;  1574
                    00000000'   EF  9F 00BFC          PUSHAB  DBG$CS_COMMA                            ;
                                57  DD 00C02          PUSHL   R7                                      ;
00000000G   00                  03  FB 00C04          CALLS   #3, DBG$NMATCH                          ;
            16                  50  E9 00C0B          BLBC    R0, 152$                                ;
            52          08      A8  9E 00C0E          MOVAB   8(R8), LINK                             ;  1581
                                04  DD 00C12          PUSHL   #4                                      ;  1582
00000000G   00                  01  FB 00C14          CALLS   #1, DBG$GET_TEMPMEM                     ;
```

```
                58              50  D0 00C1B          MOVL    R0, NOUN_NODE
                62              58  D0 00C1E          MOVL    NOUN_NODE, (LINK)                   ;  1583
                            FF69    31 00C21          BRW     146$                                ;  1539
                        08      A8  D4 00C24 152$:    CLRL    8(NOUN_NODE)                        ;  1590
                            04B7    31 00C27          BRW     213$                                ;  0553
                                02  DD 00C2A 153$:    PUSHL   #2                                  ;  1597
                    00000000'   EF  9F 00C2C          PUSHAB  DBG$CS_SOURCE                       ;
                                57  DD 00C32          PUSHL   R7                                  ;
00000000G       00              03  FB 00C34          CALLS   #3, DBG$NMATCH                      ;
                01              50  D1 00C3B          CMPL    R0, #1                              ;
                                03  13 00C3E          BEQL    154$                                ;
                            0086    31 00C40          BRW     159$                                ;
            01  A9              12  90 00C43 154$:    MOVB    #18, 1(R9)                          ;  1599
                                01  DD 00C47          PUSHL   #1                                  ;  1604
                    00000000'   EF  9F 00C49          PUSHAB  DBG$CS_SLASH                        ;
                                57  DD 00C4F          PUSHL   R7                                  ;
00000000G       00              03  FB 00C51          CALLS   #3, DBG$NMATCH                      ;
                5E              50  E9 00C58          BLBC    R0, 158$                            ;
                                04  DD 00C5B          PUSHL   #4                                  ;  1617
                    00000000'   EF  9F 00C5D          PUSHAB  DBG$CS_MODULE                       ;
                                57  DD 00C63          PUSHL   R7                                  ;
00000000G       00              03  FB 00C65          CALLS   #3, DBG$NMATCH                      ;
                11              50  E9 00C6C          BLBC    R0, 155$                            ;
                                01  DD 00C6F          PUSHL   #1                                  ;  1628
                    00000000'   EF  9F 00C71          PUSHAB  DBG$CS_EQUAL                        ;
                                57  DD 00C77          PUSHL   R7                                  ;
00000000G       00              03  FB 00C79          CALLS   #3, DBG$NMATCH                      ;
                03              50  E8 00C80 155$:    BLBS    R0, 156$                            ;
                            0441    31 00C83          BRW     210$                                ;
                    0C      AC  DD 00C86 156$:    PUSHL   MESSAGE_VECT                        ;  1639
                    0C      AE  9F 00C89          PUSHAB  MODNAMEPTR                          ;
                                57  DD 00C8C          PUSHL   R7                                  ;
00000000G       00              03  FB 00C8E          CALLS   #3, DBG$NSAVE_STRING                ;
                03              50  E8 00C95          BLBS    R0, 157$                            ;
                            0442    31 00C98          BRW     212$                                ;
                    08      AE  DD 00C9B 157$:    PUSHL   MODNAMEPTR                          ;  1648
00000000G       00              01  FB 00C9E          CALLS   #1, DBG$STA_GETSOURCEMOD            ;
            04  A8              50  D0 00CA5          MOVL    R0, 4(NOUN_NODE)                    ;
                                0E  12 00CA9          BNEQ    158$                                ;  1654
                    08      AE  DD 00CAB          PUSHL   MODNAMEPTR                          ;  1656
                                01  DD 00CAE          PUSHL   #1                                  ;  1657
                000281E8        8F  DD 00CB0          PUSHL   #164328                             ;
                            0226    31 00CB6          BRW     183$                                ;
                    0C      AC  DD 00CB9 158$:    PUSHL   MESSAGE_VECT                        ;  1668
                7E              57  7D 00CBC          MOVQ    R7, -(SP)                           ;
00000000G       00              03  FB 00CBF          CALLS   #3, DBG$NGET_DIR_LIST               ;
                            F782    31 00CC6          BRW     50$                                 ;
                                01  DD 00CC9 159$:    PUSHL   #1                                  ;  1677
                    00000000'   EF  9F 00CCB          PUSHAB  DBG$CS_STEP                         ;
                                57  DD 00CD1          PUSHL   R7                                  ;
00000000G       00              03  FB 00CD3          CALLS   #3, DBG$NMATCH                      ;
                01              50  D1 00CDA          CMPL    R0, #1                              ;
                                06  12 00CDD          BNEQ    160$                                ;
            01  A9              0B  90 00CDF          MOVB    #11, 1(R9)                          ;  1679
                                74  11 00CE3          BRB     166$                                ;  1680
                04 00000000'    EF  E8 00CE5 160$:    BLBS    DBG$GL_DEVELOPER, 161$              ;  1686
                                50  D4 00CEC          CLRL    R0                                  ;
```

```
                        11  11 00CEE              BRB     162$                                    ;
                        02  DD 00CF0 161$:        PUSHL   #2                                      ; 1687
                00000000' EF  9F 00CF2          PUSHAB  DBG$CS_TASK                             ;
                        57  DD 00CF8             PUSHL   R7                                      ;
00000000G       00      03  FB 00CFA             CALLS   #3, DBG$NMATCH                          ;
                01      50  D1 00D01 162$:      CMPL    R0, #1                                  ; 1686
                        11  12 00D04             BNEQ    163$                                    ;
      01        A9      1E  90 00D06             MOVB    #30, 1(R9)                              ; 1689
                0280    8F  BB 00D0A             PUSHR   #^M<R7,R9>                              ; 1690
00000000G       00      02  FB 00D0E             CALLS   #2, DBG$NPARSE_SET_TASK                 ;
                        25  11 00D15             BRB     164$                                    ; 0553
                        04  DD 00D17 163$:        PUSHL   #4                                      ; 1696
                00000000' EF  9F 00D19          PUSHAB  DBG$CS_TERMINAL                         ;
                        57  DD 00D1F             PUSHL   R7                                      ;
00000000G       00      03  FB 00D21             CALLS   #3, DBG$NMATCH                          ;
                01      50  D1 00D28             CMPL    R0, #1                                  ;
                        12  12 00D2B             BNEQ    165$                                    ;
      01        A9      19  90 00D2D             MOVB    #25, 1(R9)                              ; 1698
                0280    8F  BB 00D31             PUSHR   #^M<R7,R9>                              ; 1699
00000000G       00      02  FB 00D35             CALLS   #2, DBG$SCR_PARSE_SETTERM_CMD           ;
                    03A2 31  00D3C 164$:        BRW     213$                                    ; 0553
                        01  DD 00D3F 165$:        PUSHL   #1                                      ; 1705
                00000000' EF  9F 00D41          PUSHAB  DBG$CS_TRACE                            ;
                        57  DD 00D47             PUSHL   R7                                      ;
00000000G       00      03  FB 00D49             CALLS   #3, DBG$NMATCH                          ;
                01      50  D1 00D50             CMPL    R0, #1                                  ;
                        07  12 00D53             BNEQ    167$                                    ;
      01        A9      0E  90 00D55             MOVB    #14, 1(R9)                              ; 1707
                    0332 31  00D59 166$:        BRW     205$                                    ; 1710
                        02  DD 00D5C 167$:        PUSHL   #2                                      ; 1715
                00000000' EF  9F 00D5E          PUSHAB  DBG$CS_TYPE                             ;
                        57  DD 00D64             PUSHL   R7                                      ;
00000000G       00      03  FB 00D66             CALLS   #3, DBG$NMATCH                          ;
                01      50  D1 00D6D             CMPL    R0, #1                                  ;
                        03  13 00D70             BEQL    168$                                    ;
                    02FF 31  00D72             BRW     204$                                    ;
                        04  DD 00D75 168$:        PUSHL   #4                                      ; 1746
00000000G       00      01  FB 00D77             CALLS   #1, DBG$GET_TEMPMEM                     ;
                52      50  D0 00D7E             MOVL    R0, LENGTH_NODE                         ;
      08        A8      52  D0 00D81             MOVL    LENGTH_NODE, 8(NOUN_NODE)               ; 1747
                        01  DD 00D85             PUSHL   #1                                      ; 1751
                00000000' EF  9F 00D87          PUSHAB  DBG$CS_SLASH                            ;
                        57  DD 00D8D             PUSHL   R7                                      ;
00000000G       00      03  FB 00D8F             CALLS   #3, DBG$NMATCH                          ;
                41      50  E9 00D96             BLBC    R0, 173$                                ;
                        01  DD 00D99             PUSHL   #1                                      ; 1757
                00000000' EF  9F 00D9B          PUSHAB  DBG$CS_OVERRIDE                         ;
                        57  DD 00DA1             PUSHL   R7                                      ;
00000000G       00      03  FB 00DA3             CALLS   #3, DBG$NMATCH                          ;
                27      50  E8 00DAA             BLBS    R0, 172$                                ;
                        01  DD 00DAD 169$:        PUSHL   #1                                      ; 1761
                00000000' EF  9F 00DAF          PUSHAB  DBG$CS_CR                               ;
                        57  DD 00DB5             PUSHL   R7                                      ;
00000000G       00      03  FB 00DB7             CALLS   #3, DBG$NMATCH                          ;
                03      50  E8 00DBE             BLBS    R0, 170$                                ;
                    0303 31  00DC1             BRW     210$                                    ;
                00028000  8F  DD 00DC4 170$:        PUSHL   #164048                                 ; 1763
```

```
                                01 FB 00DCA 171$:   CALLS   #1, DBG$NMAKE_ARG_VECT
                              0305 31 00DD1         BRW     211$
      01 A9                     0D 90 00DD4 172$:   MOVB    #13, 1(R9)                  ;1771
                                04 11 00DD8         BRB     174$                        ;1751
      01 A9                     0C 90 00DDA 173$:   MOVB    #12, 1(R9)                  ;1775
                                05 DD 00DDE 174$:   PUSHL   #5                          ;1783
                 00000000'      EF 9F 00DE0         PUSHAB  DBG$CS_ASCIC
                                57 DD 00DE6         PUSHL   R7
00000000G 00                    03 FB 00DE8         CALLS   #3, DBG$NMATCH
          53                    50 D0 00DEF         MOVL    R0, R3
          01                    53 D1 00DF2         CMPL    R3, #1
                                16 13 00DF5         BEQL    175$
                                02 DD 00DF7         PUSHL   #2                          ;1784
                 00000000'      EF 9F 00DF9         PUSHAB  DBG$CS_AC
                                57 DD 00DFF         PUSHL   R7
00000000G 00                    03 FB 00E01         CALLS   #3, DBG$NMATCH
          01                    50 D1 00E08         CMPL    R0, #1
                                06 12 00E0B         BNEQ    176$
          68                    26 D0 00E0D 175$:   MOVL    #38, (NOUN_NODE)            ;1786
                              01E4 31 00E10         BRW     195$                        ;1787
                                01 DD 00E13 176$:   PUSHL   #1                          ;1797
                 00000000'      EF 9F 00E15         PUSHAB  DBG$CS_PACKED
                                57 DD 00E1B         PUSHL   R7
00000000G 00                    03 FB 00E1D         CALLS   #3, DBG$NMATCH
          01                    50 D1 00E24         CMPL    R0, #1
                                51 12 00E27         BNEQ    180$
          68                    15 D0 00E29         MOVL    #21, (NOUN_NODE)            ;1799
                                01 DD 00E2C         PUSHL   #1                          ;1804
                 00000000'      EF 9F 00E2E         PUSHAB  DBG$CS_COLON
                                57 DD 00E34         PUSHL   R7
00000000G 00                    03 FB 00E36         CALLS   #3, DBG$NMATCH
          0F                    50 E8 00E3D         BLBS    R0, 177$
                 00028830       8F DD 00E40         PUSHL   #165936                     ;1806
00000000G 00                    01 FB 00E46         CALLS   #1, LIB$SIGNAL
                                28 11 00E4D         BRB     179$
                                52 DD 00E4F 177$:   PUSHL   LENGTH_NODE                 ;1813
                                57 DD 00E51         PUSHL   R7
00000000G 00                    02 FB 00E53         CALLS   #2, DBG$NSAVE_INTEGER
          58                    50 E9 00E5A         BLBC    R0, 181$
                                62 D5 00E5D         TSTL    (LENGTH_NODE)               ;1819
                                05 19 00E5F         BLSS    178$
          1F                    62 D1 00E61         CMPL    (LENGTH_NODE), #31
                                11 15 00E64         BLEQ    179$
                                62 DD 00E66 178$:   PUSHL   (LENGTH_NODE)               ;1821
                                01 DD 00E68         PUSHL   #1
                 00028828       8F DD 00E6A         PUSHL   #165928
00000000G 00                    03 FB 00E70         CALLS   #3, LIB$SIGNAL
                              0267 31 00E77 179$:   BRW     213$                        ;1779
                                01 DD 00E7A 180$:   PUSHL   #1                          ;1825
                 00000000'      EF 9F 00E7C         PUSHAB  DBG$CS_ASCII
                                57 DD 00E82         PUSHL   R7
00000000G 00                    03 FB 00E84         CALLS   #3, DBG$NMATCH
          01                    50 D1 00E8B         CMPL    R0, #1
                                59 12 00E8E         BNEQ    184$
          68                    0E D0 00E90         MOVL    #14, (NOUN_NODE)            ;1827
          62                    04 D0 00E93         MOVL    #4, (LENGTH_NODE)           ;1832
                                01 DD 00E96         PUSHL   #1                          ;1836
```

```
                00000000'  EF  9F 00E98          PUSHAB  DBG$CS_COLON
                           57  DD 00E9E          PUSHL   R7
00000000G       00         03  FB 00EA0          CALLS   #3, DBG$NMATCH
                CD         50  E9 00EA7          BLBC    R0, 179$                    ;
                           52  DD 00EAA          PUSHL   LENGTH_NODE                 ; 1843
                           57  DD 00EAC          PUSHL   R7
00000000G       00         02  FB 00EAE          CALLS   #2, DBG$NSAVE_INTEGER
                03         50  E8 00EB5 181$:    BLBS    R0, 182$                    ;
                     0222  31  00EB8            BRW     212$                        ;
                           62  D5 00EBB 182$:    TSTL    (LENGTH_NODE)               ; 1849
                           B8  12 00EBD          BNEQ    179$                        ;
00000000'       EF         01  B0 00EBF          MOVW    #1, ZERO_DESC               ; 1857
00000000'       EF 00000000'  EF  9E 00EC6       MOVAB   P.AEF, ZERO_DESC+4          ; 1858
                00000000'  EF  9F 00ED1          PUSHAB  ZERO_DESC                   ; 1859
                           01  DD 00ED7          PUSHL   #1                          ;
                000281B0   8F  DD 00ED9          PUSHL   #164272                     ;
00000000G       00         03  FB 00EDF 183$:    CALLS   #3, DBG$NMAKE_ARG_VECT      ;
                     01F0  31  00EE6            BRW     211$                        ;
                           05  DD 00EE9 184$:    PUSHL   #5                          ; 1867
                00000000'  EF  9F 00EEB          PUSHAB  DBG$CS_ASCIZ                ;
                           57  DD 00EF1          PUSHL   R7                          ;
00000000G       00         03  FB 00EF3          CALLS   #3, DBG$NMATCH              ;
                53         50  D0 00EFA          MOVL    R0, R3                      ;
                01         53  D1 00EFD          CMPL    R3, #1                      ;
                           16  13 00F00          BEQL    185$                        ;
                           02  DD 00F02          PUSHL   #2                          ; 1868
                00000000'  EF  9F 00F04          PUSHAB  DBG$CS_AZ                   ;
                           57  DD 00F0A          PUSHL   R7                          ;
00000000G       00         03  FB 00F0C          CALLS   #3, DBG$NMATCH              ;
                01         50  D1 00F13          CMPL    R0, #1                      ;
                           06  12 00F16          BNEQ    186$                        ;
                68         27  D0 00F18 185$:    MOVL    #39, (NOUN_NODE)            ; 1870
                     00D9  31  00F1B            BRW     195$                        ; 1871
                           01  DD 00F1E 186$:    PUSHL   #1                          ; 1874
                00000000'  EF  9F 00F20          PUSHAB  DBG$CS_BYTE                 ;
                           57  DD 00F26          PUSHL   R7                          ;
00000000G       00         03  FB 00F28          CALLS   #3, DBG$NMATCH              ;
                01         50  D1 00F2F          CMPL    R0, #1                      ;
                           09  12 00F32          BNEQ    187$                        ;
                68         06  D0 00F34          MOVL    #6, (NOUN_NODE)             ; 1876
                62         01  D0 00F37          MOVL    #1, (LENGTH_NODE)           ; 1877
                     01A4  31  00F3A            BRW     213$                        ; 1779
                           02  DD 00F3D 187$:    PUSHL   #2                          ; 1880
                00000000'  EF  9F 00F3F          PUSHAB  DBG$CS_DATE_TIME            ;
                           57  DD 00F45          PUSHL   R7                          ;
00000000G       00         03  FB 00F47          CALLS   #3, DBG$NMATCH              ;
                01         50  D1 00F4E          CMPL    R0, #1                      ;
                           05  12 00F51          BNEQ    188$                        ;
                68         23  D0 00F53          MOVL    #35, (NOUN_NODE)            ; 1882
                           68  11 00F56          BRB     192$                        ; 1883
                           01  DD 00F58 188$:    PUSHL   #1                          ; 1886
                00000000'  EF  9F 00F5A          PUSHAB  DBG$CS_D_FLOAT              ;
                           57  DD 00F60          PUSHL   R7                          ;
00000000G       00         03  FB 00F62          CALLS   #3, DBG$NMATCH              ;
                01         50  D1 00F69          CMPL    R0, #1                      ;
                           05  12 00F6C          BNEQ    189$                        ;
                68         0B  D0 00F6E          MOVL    #11, (NOUN_NODE)            ; 1888
```

```
                                4D  11  00F71          BRB     192$                                    ; 1889
                                01  DD  00F73  189$:   PUSHL   #1                                      ; 1892
                    00000000'   EF  9F  00F75          PUSHAB  DBG$CS_FLOAT
                                57  DD  00F7B          PUSHL   R7
00000000G   00                  03  FB  00F7D          CALLS   #3, DBG$NMATCH
            53                  50  D0  00F84          MOVL    R0, R3
            01                  53  D1  00F87          CMPL    R3, #1
                                16  13  00F8A          BEQL    190$
                                01  DD  00F8C          PUSHL   #1                                      ; 1893
                    00000000'   EF  9F  00F8E          PUSHAB  DBG$CS_F_FLOAT
                                57  DD  00F94          PUSHL   R7
00000000G   00                  03  FB  00F96          CALLS   #3, DBG$NMATCH
            01                  50  D1  00F9D          CMPL    R0, #1
                                05  12  00FA0          BNEQ    191$
            68                  0A  D0  00FA2  190$:   MOVL    #10, (NOUN_NODE)                        ; 1895
                                6D  11  00FA5          BRB     197$                                    ; 1896
                                01  DD  00FA7  191$:   PUSHL   #1                                      ; 1899
                    00000000'   EF  9F  00FA9          PUSHAB  DBG$CS_G_FLOAT
                                57  DD  00FAF          PUSHL   R7
00000000G   00                  03  FB  00FB1          CALLS   #3, DBG$NMATCH
            01                  50  D1  00FB8          CMPL    R0, #1
                                06  12  00FBB          BNEQ    193$
            68                  1B  D0  00FBD          MOVL    #27, (NOUN_NODE)                        ; 1901
                          008E  31  00FC0  192$:   BRW     201$                                    ; 1902
                                01  DD  00FC3  193$:   PUSHL   #1                                      ; 1905
                    00000000'   EF  9F  00FC5          PUSHAB  DBG$CS_H_FLOAT
                                57  DD  00FCB          PUSHL   R7
00000000G   00                  03  FB  00FCD          CALLS   #3, DBG$NMATCH
            01                  50  D1  00FD4          CMPL    R0, #1
                                05  12  00FD7          BNEQ    194$
            68                  1C  D0  00FD9          MOVL    #28, (NOUN_NODE)                        ; 1907
                                54  11  00FDC          BRB     199$                                    ; 1908
                                01  DD  00FDE  194$:   PUSHL   #1                                      ; 1911
                    00000000'   EF  9F  00FE0          PUSHAB  DBG$CS_INSTRUCTION
                                57  DD  00FE6          PUSHL   R7
00000000G   00                  03  FB  00FE8          CALLS   #3, DBG$NMATCH
            01                  50  D1  00FEF          CMPL    R0, #1
                                07  12  00FF2          BNEQ    196$
            68                  16  D0  00FF4          MOVL    #22, (NOUN_NODE)                        ; 1913
                                62  D4  00FF7  195$:   CLRL    (LENGTH_NODE)                           ; 1914
                                77  11  00FF9          BRB     203$                                    ; 1779
                                01  DD  00FFB  196$:   PUSHL   #1                                      ; 1917
                    00000000'   EF  9F  00FFD          PUSHAB  DBG$CS_LONGWORD
                                57  DD  01003          PUSHL   R7
00000000G   00                  03  FB  01005          CALLS   #3, DBG$NMATCH
            01                  50  D1  0100C          CMPL    R0, #1
                                08  12  0100F          BNEQ    198$
            68                  08  D0  01011          MOVL    #8, (NOUN_NODE)                         ; 1919
            62                  04  D0  01014  197$:   MOVL    #4, (LENGTH_NODE)                       ; 1920
                                59  11  01017          BRB     203$                                    ; 1779
                                01  DD  01019  198$:   PUSHL   #1                                      ; 1923
                    00000000'   EF  9F  0101B          PUSHAB  DBG$CS_OCTAWORD
                                57  DD  01021          PUSHL   R7
00000000G   00                  03  FB  01023          CALLS   #3, DBG$NMATCH
            01                  50  D1  0102A          CMPL    R0, #1
                                09  12  0102D          BNEQ    200$
            68                  1A  D0  0102F          MOVL    #26, (NOUN_NODE)                        ; 1925
```

```
                62              10  DO 01032 199$:  MOVL    #16, (LENGTH_NODE)              ; 1926
                            008D  31 01035         BRW     209$                            ; 1779
                                01  DD 01038 200$:  PUSHL   #1                             ; 1929
                    00000000'   EF  9F 0103A         PUSHAB  DBG$CS_QUADWORD
                                57  DD 01040         PUSHL   R7
00000000G       00              03  FB 01042         CALLS   #3, DBG$NMATCH
                01              50  D1 01049         CMPL    R0, #1
                                08  12 0104C         BNEQ    202$
                68              09  D0 0104E         MOVL    #9, (NOUN_NODE)             ; 1931
                62              08  D0 01051 201$:  MOVL    #8, (LENGTH_NODE)           ; 1932
                                6F  11 01054         BRB     209$                        ; 1779
                                01  DD 01056 202$:  PUSHL   #1                          ; 1935
                    00000000'   EF  9F 01058         PUSHAB  DBG$CS_WORD
                                57  DD 0105E         PUSHL   R7
00000000G       00              03  FB 01060         CALLS   #3, DBG$NMATCH
                01              50  D1 01067         CMPL    R0, #1
                                45  12 0106A         BNEQ    207$
                68              07  D0 0106C         MOVL    #7, (NOUN_NODE)             ; 1937
                62              02  D0 0106F         MOVL    #2, (LENGTH_NODE)           ; 1938
                                6D  11 01072 203$:  BRB     213$                        ; 1779
                                01  DD 01074 204$:  PUSHL   #1                          ; 1959
                    00000000'   EF  9F 01076         PUSHAB  DBG$CS_WATCH
                                57  DD 0107C         PUSHL   R7
00000000G       00              03  FB 0107E         CALLS   #3, DBG$NMATCH
                01              50  D1 01085         CMPL    R0, #1
                                13  12 01088         BNEQ    206$
            01  A9              11  90 0108A         MOVB    #17, 1(R9)                  ; 1961
                        0C      AC  DD 0108E 205$:  PUSHL   MESSAGE_VECT                ; 1962
                      0280      8F  BB 01091         PUSHR   #^M<R7,R9>
00000000G       00              03  FB 01095         CALLS   #3, DBG$EVENT_SYNTAX
                                    04 0109C         RET
                                03  DD 0109D 206$:  PUSHL   #3                          ; 1968
                    00000000'   EF  9F 0109F         PUSHAB  DBG$CS_WINDOW
                                57  DD 010A5         PUSHL   R7
00000000G       00              03  FB 010A7         CALLS   #3, DBG$NMATCH
                01              50  D1 010AE         CMPL    R0, #1
                                03  13 010B1 207$:  BEQL    208$
                            FCF7  31 010B3         BRW     169$
            01  A9              1A  90 010B6 208$:  MOVB    #26, 1(R9)                  ; 1970
                          0280  8F  BB 010BA         PUSHR   #^M<R7,R9>                  ; 1971
00000000G       00              02  FB 010BE         CALLS   #2, DBG$SCR_PARSE_SETWIND_CMD
                                1A  11 010C5 209$:  BRB     213$                        ; 0553
                                57  DD 010C7 210$:  PUSHL   R7                          ; 1984
00000000G       00              01  FB 010C9         CALLS   #1, DBG$NNEXT_WORD
                                50  DD 010D0         PUSHL   R0
00000000G       00              01  FB 010D2         CALLS   #1, DBG$NSYNTAX_ERROR
            0C  BC              50  D0 010D9 211$:  MOVL    R0, @MESSAGE_VECT
                50              04  D0 010DD 212$:  MOVL    #4, R0                      ; 1986
                                    04 010E0         RET
                50              01  D0 010E1 213$:  MOVL    #1, R0                      ; 1994
                                    04 010E4         RET                                 ; 1996

; Routine Size:  4325 bytes,    Routine Base:  DBG$CODE + 0000
```

```
;1867     1997  1 GLOBAL ROUTINE DBG$NEXECUTE_SET (VERB_NODE, MESSAGE_VECT) =
;1868     1998  1 !
;1869     1999  1 ! FUNCTIONAL DESCRIPTION:
;1870     2000  1 !
;1871     2001  1 !       This routine processes the command execution tree corresponding to the parsed
;1872     2002  1 !       SET ... command. Appropriate semantic actions are taken.
;1873     2003  1 !
;1874     2004  1 ! FORMAL PARAMETERS:
;1875     2005  1 !
;1876     2006  1 !       VERB_NODE -             The head node in the command execution tree
;1877     2007  1 !
;1878     2008  1 !       MESSAGE_VECT -          The address of a longword to contain the address
;1879     2009  1 !                               of a message argument vector
;1880     2010  1 !
;1881     2011  1 ! IMPLICIT INPUTS:
;1882     2012  1 !
;1883     2013  1 !       DBG$GL_LOG_BUF -        Version 2 debugger pointer to counted string filespec
;1884     2014  1 !
;1885     2015  1 !       DBG$GL_CONTEXT -           "    "    "     context switch vector for log and output
;1886     2016  1 !
;1887     2017  1 !       DBG$GB_DEF_OUT -           "    "    "     output control vector
;1888     2018  1 !
;1889     2019  1 !       DBG$GL_LOGFAB -            "    "    "     FAB for log file
;1890     2020  1 !
;1891     2021  1 !       DBG$GL_LOGRAB -            "    "    "     RAB  "   "   "
;1892     2022  1 !
;1893     2023  1 ! IMPLICIT OUTPUTS:
;1894     2024  1 !
;1895     2025  1 !       Any of the above implicit inputs may be altered
;1896     2026  1 !
;1897     2027  1 ! ROUTINE VALUE:
;1898     2028  1 !
;1899     2029  1 !       An unsigned integer longword completion code
;1900     2030  1 !
;1901     2031  1 ! COMPLETION CODES:
;1902     2032  1 !
;1903     2033  1 !       STS$K_SEVERE (4)        - Signifies that the command was not executed
;1904     2034  1 !
;1905     2035  1 !       STS$K_SUCCESS (1)       - Signifies that a command was executed.
;1906     2036  1 !
;1907     2037  1 ! SIDE EFFECTS:
;1908     2038  1 !
;1909     2039  1 !       Various semantic actions are performed on data structures. That is, the
;1910     2040  1 !       state of the debugger is altered.
;1911     2041  1 !
;1912     2042  1
;1913     2043  2     BEGIN
;1914     2044  2
;1915     2045  2     MAP
;1916     2046  2         VERB_NODE : REF DBG$VERB_NODE;  ! Pointer to command Verb Node
;1917     2047  2
;1918     2048  2     LOCAL
;1919     2049  2         NOUN_NODE : REF DBG$NOUN_NODE;  ! Pointer to current Noun Node
;1920     2050  2
;1921     2051  2
;1922     2052  2
;1923     2053  2     ! Recover the noun node.
```

```
; 1924          2054  2      !
; 1925          2055  2      NOUN_NODE = .VERB_NODE [DBG$L_VERB_OBJECT_PTR];
; 1926          2056  2
; 1927          2057  2
; 1928          2058  2      ! Transfer to a subnetwork on the basis of the composite verb.
; 1929          2059  2      !
; 1930          2060  2      CASE .VERB_NODE[DBG$B_VERB_COMPOSITE] FROM SET_MINIMUM TO SET_MAXIMUM OF
; 1931          2061  2              SET
; 1932          2062  2
; 1933          2063  2
; 1934          2064  2              ! Execute the SET BREAK command.
; 1935          2065  2              !
; 1936          2066  2              [SET_BREAK]:
; 1937          2067  2                  DBG$EVENT_SEMANTICS (.VERB_NODE, .MESSAGE_VECT);
; 1938          2068  2
; 1939          2069  2
; 1940          2070  2              ! Execute the SET DEVELOPER command by setting the specified
; 1941          2071  2              ! bits in the DBG$GL_DEVELOPER longword.
; 1942          2072  2              !
; 1943          2073  2              [SET_DEVELOPER] :
; 1944          2074  3                  BEGIN
; 1945          2075  3                  IF .VERB_NODE[DBG$L_VERB_OBJECT_PTR] NEQ 0
; 1946          2076  3                  THEN
; 1947          2077  4                      BEGIN
; 1948          2078  4                      NOUN_NODE = .VERB_NODE[DBG$L_VERB_OBJECT_PTR];
; 1949          2079  4                      WHILE TRUE DO
; 1950          2080  5                          BEGIN
; 1951          2081  5                          DBG$GL_DEVELOPER[.NOUN_NODE [DBG$L_NOUN_VALUE]] = TRUE;
; 1952          2082  5                          IF .NOUN_NODE[DBG$L_NOUN_LINK] EQL 0 THEN EXITLOOP;
; 1953          2083  5                          NOUN_NODE = .NOUN_NODE[DBG$L_NOUN_LINK];
; 1954          2084  4                          END;
; 1955          2085  4
; 1956          2086  3                      END;
; 1957          2087  3
; 1958          2088  2                  END;
; 1959          2089  2
; 1960          2090  2
; 1961          2091  2              ! Execute the SET DEFINE command.
; 1962          2092  2              !
; 1963          2093  2              [SET_DEFINE] :
; 1964          2094  3                  BEGIN
; 1965          2095  3                  DBG$SET_DEFINE_LVL (USER_DEF_DEFINE);
; 1966          2096  3                  DBG$GB_DEFINE_PTR[DEFINE_ONLY] = .NOUN_NODE [DBG$L_NOUN_VALUE];
; 1967          2097  2                  END;
; 1968          2098  2
; 1969          2099  2
; 1970          2100  2              ! Execute the SET DISPLAY command.
; 1971          2101  2              !
; 1972          2102  2              [SET_DISPLAY]:
; 1973          2103  2                  DBG$SCR_EXECUTE_DISPLAY_CMD(.VERB_NODE, TRUE);
; 1974          2104  2
; 1975          2105  2
; 1976          2106  2              ! Execute the SET EXCEPTION BREAK command.
; 1977          2107  2              !
; 1978          2108  2              [set_exception_break] :
; 1979          2109  3                  BEGIN
; 1980          2110  3                  dbg$gb_resignal = false;
```

```
;  1981     2111  3              RETURN DBG$EVENT_SEMANTICS (.VERB_NODE, .MESSAGE_VECT);
;  1982     2112  2              END;
;  1983     2113  2
;  1984     2114  2
;  1985     2115  2          ! Execute the SET KEY command.
;  1986     2116  2          !
;  1987     2117  2          [SET_KEY] :
;  1988     2118  2              BEGIN
;  1989     2119  2
;  1990     2120  2              LOCAL
;  1991     2121  2                  OUTPUT_LOG,
;  1992     2122  2                  SET_STATUS,
;  1993     2123  2                  DESC_PTR: REF DBG$STG_DESC;
;  1994     2124  2
;  1995     2125  2              OUTPUT_LOG = .VERB_NODE [DBG$L_VERB_ADVERB_PTR];
;  1996     2126  2
;  1997     2127  2
;  1998     2128  2              ! Make the call to set the status.
;  1999     2129  2              !
;  2000     2130  2              SET_STATUS = SMG$SET_DEFAULT_STATE (DBG$GL_KEY_TABLE_ID,
;  2001     2131  2                                                  .NOUN_NODE [DBG$L_NOUN_VALUE]);
;  2002     2132  2              IF NOT .SET_STATUS THEN SIGNAL(DBG$_SETKEYERR);
;  2003     2133  2              IF .OUTPUT_LOG
;  2004     2134  2              THEN
;  2005     2135  2                  SIGNAL(DBG$_SETKEY, 1, .NOUN_NODE [DBG$L_NOUN_VALUE]);
;  2006     2136  2
;  2007     2137  2              END;
;  2008     2138  2
;  2009     2139  2          [SET_LANGUAGE] :
;  2010     2140  3              BEGIN
;  2011     2141  3              DBG$SET_LANG (0, .NOUN_NODE [DBG$L_NOUN_VALUE]);
;  2012     2142  2              END;
;  2013     2143  2
;  2014     2144  2          [set_log] :
;  2015     2145  3              BEGIN
;  2016     2146  3              dbg$gl_log_buf = .noun_node [dbg$l_noun_value];
;  2017     2147  3              IF NOT dbg$nset_log (.message_vect)
;  2018     2148  3              THEN
;  2019     2149  3                  RETURN sts$k_severe;
;  2020     2150  2              END;
;  2021     2151  2
;  2022     2152  2          [set_margins] :
;  2023     2153  3              BEGIN
;  2024     2154  3              dbg$src_left_margin = .noun_node[dbg$l_noun_value];
;  2025     2155  3              dbg$src_right_margin = .noun_node[dbg$l_noun_value2];
;  2026     2156  2              END;
;  2027     2157  2
;  2028     2158  2          [set_max_source_files] :
;  2029     2159  3              BEGIN
;  2030     2160  3              dbg$src_set_max_files(.noun_node[dbg$l_noun_value]);
;  2031     2161  2              END;
;  2032     2162  2
;  2033     2163  2
;  2034     2164  2          ! Execute the SET MODE command.
;  2035     2165  2          !
;  2036     2166  2          [SET_MODE] :
;  2037     2167  3              BEGIN
```

```
;  2038          2168  3
;  2039          2169  3
;  2040          2170  3                ! Set the mode level to the correct level.
;  2041          2171  3                !
;  2042          2172  3                DBG$SET_MOD_LVL (USER_DEF_MODE);
;  2043          2173  3
;  2044          2174  3
;  2045          2175  3                ! Now fill in the correct mode values.
;  2046          2176  3                !
;  2047          2177  3                WHILE .NOUN_NODE NEQA 0 DO
;  2048          2178  4                    BEGIN
;  2049          2179  4                    CASE .NOUN_NODE [DBG$L_NOUN_VALUE]
;  2050          2180  4                                        FROM MODE_LOWEST TO MODE_HIGHEST OF
;  2051          2181  4                        SET
;  2052          2182  4
;  2053          2183  4                        [MODE_BINARY] :
;  2054          2184  5                            BEGIN
;  2055          2185  5                            DBG$GB_RADIX[DBG$B_RADIX_INPUT] = DBG$K_BINARY;
;  2056          2186  5                            DBG$GB_RADIX[DBG$B_RADIX_OUTPUT] = DBG$R_BINARY;
;  2057          2187  4                            END;
;  2058          2188  4
;  2059          2189  4                        [MODE_OCTAL] :
;  2060          2190  5                            BEGIN
;  2061          2191  5                            DBG$GB_RADIX[DBG$B_RADIX_INPUT] = DBG$K_OCTAL;
;  2062          2192  5                            DBG$GB_RADIX[DBG$B_RADIX_OUTPUT] = DBG$R_OCTAL;
;  2063          2193  4                            END;
;  2064          2194  4
;  2065          2195  4                        [MODE_DECIMAL] :
;  2066          2196  5                            BEGIN
;  2067          2197  5                            DBG$GB_RADIX[DBG$B_RADIX_INPUT] = DBG$K_DECIMAL;
;  2068          2198  5                            DBG$GB_RADIX[DBG$B_RADIX_OUTPUT] = DBG$R_DECIMAL;
;  2069          2199  4                            END;
;  2070          2200  4
;  2071          2201  4                        [MODE_DEFAULT] :
;  2072          2202  5                            BEGIN
;  2073          2203  5                            DBG$GB_RADIX[DBG$B_RADIX_INPUT] =
;  2074          2204  5                                DBG$NGET_TRANS_RADIX(DBG$K_DEFAULT);
;  2075          2205  5                            DBG$GB_RADIX[DBG$B_RADIX_OUTPUT] =
;  2076          2206  5                                DBG$NGET_TRANS_RADIX(DBG$K_DEFAULT);
;  2077          2207  4                            END;
;  2078          2208  4
;  2079          2209  4                        [MODE_HEX] :
;  2080          2210  5                            BEGIN
;  2081          2211  5                            DBG$GB_RADIX[DBG$B_RADIX_INPUT] = DBG$K_HEX;
;  2082          2212  5                            DBG$GB_RADIX[DBG$B_RADIX_OUTPUT] = DBG$R_HEX;
;  2083          2213  4                            END;
;  2084          2214  4
;  2085          2215  4                        [MODE_SYMBLS] :
;  2086          2216  4                            DBG$GB_MOD_PTR [MODE_SYMBOLS] = TRUE;
;  2087          2217  4
;  2088          2218  4                        [MODE_NOSYMBLS] :
;  2089          2219  4                            DBG$GB_MOD_PTR [MODE_SYMBOLS] = FALSE;
;  2090          2220  4
;  2091          2221  4                        [MODE_G_FLOAT]:
;  2092          2222  4                            DBG$GB_MOD_PTR [MODE_G_FLOATS] = TRUE;
;  2093          2223  4
;  2094          2224  4                        [MODE_NOG_FLOAT]:
```

```
;  2095          2225  4                        DBG$GB_MOD_PTR [MODE_G_FLOATS] = FALSE;
;  2096          2226  4
;  2097          2227  4                    [MODE_SCREEN]:
;  2098          2228  4                        DBG$SCR_SCREEN_MODE(TRUE);
;  2099          2229  4
;  2100          2230  4                    [MODE_NOSCREEN]:
;  2101          2231  4                        DBG$SCR_SCREEN_MODE(FALSE);
;  2102          2232  4
;  2103          2233  4                    [MODE_KEYPAD]:
;  2104          2234  5                        BEGIN
;  2105          2235  5                        DBG$GB_KEYPAD_INPUT = TRUE;
;  2106          2236  5                        SMG$SET_KEYPAD_MODE (DBG$GL_KEYBOARD_ID,%REF(1));
;  2107          2237  4                        END;
;  2108          2238  4
;  2109          2239  4                    [MODE_NOKEYPAD]:
;  2110          2240  5                        BEGIN
;  2111          2241  5                        DBG$GB_KEYPAD_INPUT = FALSE;
;  2112          2242  5                        SMG$SET_KEYPAD_MODE (DBG$GL_KEYBOARD_ID,%REF(0));
;  2113          2243  4                        END;
;  2114          2244  4
;  2115          2245  4                    TES;
;  2116          2246  4
;  2117          2247  4
;  2118          2248  4                ! Recover the next Noun Node and loop.
;  2119          2249  4                !
;  2120          2250  4                NOUN_NODE = .NOUN_NODE [DBG$L_NOUN_LINK];
;  2121          2251  4
;  2122          2252  3                END;                ! End of loop over SET MODE keywords
;  2123          2253  3
;  2124          2254  2            END;                    : End of SET MODE command
;  2125          2255  2
;  2126          2256  2
;  2127          2257  2        ! Handle the SET MODULE command.
;  2128          2258  2        !
;  2129          2259  2        [SET_MODULE, SET_MODULE_ALL] :
;  2130          2260  3            BEGIN
;  2131          2261  3
;  2132          2262  3            EXTERNAL ROUTINE
;  2133          2263  3                LIB$SPAWN,
;  2134          2264  3                SYS$WAITFR;
;  2135          2265  3
;  2136          2266  3            LOCAL
;  2137          2267  3                ALLOCATE_FLAG,
;  2138          2268  3                EVNT_FLAG,
;  2139          2269  3                FLAGS,
;  2140          2270  3                NAME_BUFF: REF VECTOR [,BYTE],      ! Module name buffer
;  2141          2271  3                STATUS;
;  2142          2272  3
;  2143          2273  3
;  2144          2274  3            ! Before doing anything, check for /NOWAIT flag.
;  2145          2275  3            ! The ADVERB_POINTER field indicates whether /NOWAIT
;  2146          2276  3            ! was specified and/or whether /ALLOCATE was specified.
;  2147          2277  3            ! The codes are
;  2148          2278  3            !   0 - neither /NOWAIT or /ALLOCATE
;  2149          2279  3            !   1 - /NOWAIT, not /ALLOCATE
;  2150          2280  3            !   2 - not /NOWAIT, /ALLOCATE
;  2151          2281  3            !   3 - both were specified
```

```
;  2152     2282  3                !
;  2153     2283  3                STATUS = FALSE;
;  2154     2284  3                IF .VERB_NODE [DBG$L_VERB_ADVERB_PTR]
;  2155     2285  3                THEN
;  2156     2286  4                    BEGIN
;  2157     2287  4
;  2158     2288  4
;  2159     2289  4                    ! We accomplish SET MODU/NOWAIT by spawning a subprocess
;  2160     2290  4                    ! with the /NOWAIT flag passed to LIB$SPAWN. The
;  2161     2291  4                    ! subprocess can accept DCL commands until the
;  2162     2292  4                    ! SET MODU completes. We pass in an event flag number (0)
;  2163     2293  4                    ! to be set when the subprocess completes.
;  2164     2294  4                    !
;  2165     2295  4                    FLAGS = 1;
;  2166     2296  4                    EVNT_FLAG = 0;
;  2167     2297  4                    STATUS = LIB$SPAWN(0,0,0,FLAGS,0,0,0,EVNT_FLAG);
;  2168     2298  4                    IF NOT .STATUS
;  2169     2299  4                    THEN
;  2170     2300  4                        SIGNAL(DBG$_NOSPAWNI,0,.STATUS);
;  2171     2301  3                    END;
;  2172     2302  3
;  2173     2303  3                ! SET MODULE/ALLOCATE was specified if the 2nd bit of
;  2174     2304  3                ! the ADVERB_PTR field is set.
;  2175     2305  3                !
;  2176     2306  3                ALLOCATE_FLAG = .VERB_NODE[DBG$L_VERB_ADVERB_PTR]/2;
;  2177     2307  3
;  2178     2308  3                ! Loop through the list of modules to set.
;  2179     2309  3                !
;  2180     2310  3                IF .verb_node[dbg$b_verb_composite] EQL set_module_all
;  2181     2311  3                THEN
;  2182     2312  3                    dbg$rst_setmod (0, 0, .allocate_flag)
;  2183     2313  3
;  2184     2314  3                ELSE
;  2185     2315  3                    WHILE .noun_node NEQA 0
;  2186     2316  3                    DO
;  2187     2317  4                        BEGIN
;  2188     2318  4
;  2189     2319  4                        ! Retrieve the name buffer and call the symbol table
;  2190     2320  4                        ! Module names are stored away as counted strings
;  2191     2321  4                        !
;  2192     2322  4                        name_buff = .noun_node [dbg$l_noun_value];
;  2193     2323  4                        IF NOT dbg$rst_setmod (name_buff [1], .name_buff [0],
;  2194     2324  4                                               .allocate_flag)
;  2195     2325  4                        THEN
;  2196     2326  5                            BEGIN
;  2197     2327  5                            .message_vect = dbg$nmake_arg_vect (dbg$_nosuchmodu,
;  2198     2328  5                                                                1,
;  2199     2329  5                                                                name_buff [0]);
;  2200     2330  5                            RETURN sts$k_severe;
;  2201     2331  4                            END;
;  2202     2332  4
;  2203     2333  4                        ! Obtain the next noun node
;  2204     2334  4                        !
;  2205     2335  4                        noun_node = .noun_node [dbg$l_noun_link];
;  2206     2336  3                        END;    ! End of loop
;  2207     2337  3
;  2208     2338  3                ! Wait for spawned subprocess to complete if there was one.
```

```
;  2209          2339  3                !
;  2210          2340  3                IF .status
;  2211          2341  3                THEN
;  2212          2342  4                    BEGIN
;  2213          2343  4                    SIGNAL(DBG$_SETMODU);
;  2214          2344  4                    SYS$WAITFR(.EVNT_FLAG);
;  2215          2345  3                    END;                                                    ;
;  2216          2346  2                END;
;  2217          2347  2
;  2218          2348  2            [set_output] :
;  2219          2349  3                BEGIN
;  2220          2350  3                LITERAL
;  2221          2351  3                    NOUN_LITERAL_LOG            = 1,
;  2222          2352  3                    NOUN_LITERAL_NOLOG          = 2,
;  2223          2353  3                    NOUN_LITERAL_SCREEN         = 3,
;  2224          2354  3                    NOUN_LITERAL_NOSCREEN       = 4,
;  2225          2355  3                    NOUN_LITERAL_TERMINAL       = 5,
;  2226          2356  3                    NOUN_LITERAL_NOTERMINAL     = 6,
;  2227          2357  3                    NOUN_LITERAL_VERIFY         = 7,
;  2228          2358  3                    NOUN_LITERAL_NOVERIFY       = 8;
;  2229          2359  3
;  2230          2360  3                ! Add the user defined output setting
;  2231          2361  3
;  2232          2362  3                WHILE TRUE DO
;  2233          2363  4                    BEGIN
;  2234          2364  4                    CASE .NOUN_NODE[DBG$L_NOUN_VALUE] FROM NOUN_LITERAL_LOG
;  2235          2365  4                                                      TO NOUN_LITERAL_NOVERIFY OF
;  2236          2366  4                        SET
;  2237          2367  4
;  2238          2368  4                        [NOUN_LITERAL_LOG] :
;  2239          2369  5                            BEGIN
;  2240          2370  5                            LOCAL
;  2241          2371  5                                STATUS,
;  2242          2372  5                                B_TYPE;
;  2243          2373  5
;  2244          2374  5                            STATUS = DBG$NSETUP_LOG (B_TYPE);
;  2245          2375  5                            IF NOT .STATUS
;  2246          2376  5                            THEN
;  2247          2377  5                                OUTPUT_RMS_ERROR (.B_TYPE);
;  2248          2378  5
;  2249          2379  5                            DBG$GB_DEF_OUT [OUT_LOG] = TRUE;
;  2250          2380  5                            IF .DBG$GB_DEF_OUT[OUT_SCREEN]
;  2251          2381  5                            THEN
;  2252          2382  5                                DBG$GL_SCREEN_LOG = TRUE;
;  2253          2383  5
;  2254          2384  4                            END;
;  2255          2385  4
;  2256          2386  4                        [NOUN_LITERAL_NOLOG] :
;  2257          2387  5                            BEGIN
;  2258          2388  5                            IF NOT .DBG$GB_DEF_OUT [OUT_TERM]
;  2259          2389  5                            THEN
;  2260          2390  5                                DBG$NOUT_INFO (DBG$_OUTPUTLOST);
;  2261          2391  5
;  2262          2392  5                            DBG$GB_DEF_OUT [OUT_LOG] = FALSE;
;  2263          2393  5                            DBG$GL_SCREEN_LOG = FALSE;
;  2264          2394  4                            END;
;  2265          2395  4
```

```
;  2266          2396  4                [NOUN_LITERAL_SCREEN]:
;  2267          2397  5                    BEGIN
;  2268          2398  5                    DBG$GB_DEF_OUT[OUT_SCREEN] = TRUE;
;  2269          2399  5                    IF .DBG$GB_DEF_OUT[OUT_LOG]
;  2270          2400  5                    THEN
;  2271          2401  5                        DBG$GL_SCREEN_LOG = TRUE;
;  2272          2402  5
;  2273          2403  4                    END;
;  2274          2404  4
;  2275          2405  4                [NOUN_LITERAL_NOSCREEN]:
;  2276          2406  5                    BEGIN
;  2277          2407  5                    DBG$GB_DEF_OUT[OUT_SCREEN] = FALSE;
;  2278          2408  5                    DBG$GL_SCREEN_LOG = FALSE;
;  2279          2409  4                    END;
;  2280          2410  4
;  2281          2411  4                [NOUN_LITERAL_TERMINAL] :
;  2282          2412  5                    BEGIN
;  2283          2413  5                    DBG$GB_DEF_OUT [OUT_TERM] = TRUE;
;  2284          2414  4                    END;
;  2285          2415  4
;  2286          2416  4                [NOUN_LITERAL_NOTERMINAL] :
;  2287          2417  5                    BEGIN
;  2288          2418  5                    IF NOT .DBG$GB_DEF_OUT [OUT_LOG]
;  2289          2419  5                    THEN
;  2290          2420  5                        DBG$NOUT_INFO (DBG$_OUTPUTLOST);
;  2291          2421  5
;  2292          2422  5                    DBG$GB_DEF_OUT [OUT_TERM] = FALSE;
;  2293          2423  4                    END;
;  2294          2424  4
;  2295          2425  4                [NOUN_LITERAL_VERIFY] :
;  2296          2426  4                    DBG$GB_DEF_OUT [OUT_VERIFY] = TRUE;
;  2297          2427  4
;  2298          2428  4                [NOUN_LITERAL_NOVERIFY] :
;  2299          2429  4                    DBG$GB_DEF_OUT [OUT_VERIFY] = FALSE;
;  2300          2430  4
;  2301          2431  4                    TES;
;  2302          2432  4
;  2303          2433  4                ! Check for another object
;  2304          2434  4                !
;  2305          2435  4                IF .NOUN_NODE [DBG$L_NOUN_LINK] EQLA 0
;  2306          2436  4                THEN
;  2307          2437  4                    EXITLOOP;
;  2308          2438  4
;  2309          2439  4                ! Update the noun node
;  2310          2440  4                !
;  2311          2441  4                NOUN_NODE = .NOUN_NODE [DBG$L_NOUN_LINK];
;  2312          2442  4
;  2313          2443  3                END;                ! End of loop over OUTPUT parameters
;  2314          2444  3
;  2315          2445  2            END;
;  2316          2446  2
;  2317          2447  2
;  2318          2448  2        ! Execute the SET RADIX command.
;  2319          2449  2        !
;  2320          2450  2        [SET_RADIX, SET_RADIX_OVERRIDE] :
;  2321          2451  3            BEGIN
;  2322          2452  3            LOCAL
```

```
;  2323          2453  3                RADIX;
;  2324          2454  3
;  2325          2455  3            ! Fill in the correct radix value.
;  2326          2456  3            !
;  2327          2457  3            CASE .NOUN_NODE [DBG$L_NOUN_VALUE]
;  2328          2458  3                            FROM MODE_LOWEST TO MODE_HIGHEST OF
;  2329          2459  3                SET
;  2330          2460  3
;  2331          2461  3                [MODE_BINARY]  : RADIX = DBG$K_BINARY;
;  2332          2462  3
;  2333          2463  3                [MODE_OCTAL]   : RADIX = DBG$K_OCTAL;
;  2334          2464  3
;  2335          2465  3                [MODE_DECIMAL] : RADIX = DBG$K_DECIMAL;
;  2336          2466  3
;  2337          2467  3                [MODE_DEFAULT] : RADIX = DBG$K_DEFAULT;
;  2338          2468  3
;  2339          2469  3                [MODE_HEX]     : RADIX = DBG$K_HEX;
;  2340          2470  3
;  2341          2471  3                [INRANGE, OUTRANGE]:
;  2342          2472  3                    $DBG_ERROR('DBGNSET\DBG$NEXECUTE_SET unexpected radix');
;  2343          2473  3
;  2344          2474  3                TES;
;  2345          2475  3
;  2346          2476  3            ! The VALUE2 field contains:
;  2347          2477  3            !       0 if no /INPUT or /OUTPUT was specified (in this
;  2348          2478  3            !         case we set both radices)
;  2349          2479  3            !       1 if /INPUT was specified
;  2350          2480  3            !       2 if /OUTPUT was specified
;  2351          2481  3            ! The VERB_COMPOSITE field tells us whether /OVERRIDE was
;  2352          2482  3            ! specified.
;  2353          2483  3            ! We use these pieces of information to tell us which
;  2354          2484  3            ! of the three elements of the DBG$GB_RADIX byte vector to
;  2355          2485  3            ! fill in.
;  2356          2486  3            !
;  2357          2487  3            IF .VERB_NODE[DBG$B_VERB_COMPOSITE] EQL SET_RADIX
;  2358          2488  3            THEN
;  2359          2489  4                BEGIN
;  2360          2490  4                IF .RADIX EQL DBG$K_DEFAULT
;  2361          2491  4                THEN
;  2362          2492  4                    RADIX = DBG$NGET_TRANS_RADIX(.RADIX);
;  2363          2493  4
;  2364          2494  4                IF .NOUN_NODE[DBG$L_NOUN_VALUE2] LEQ 1
;  2365          2495  4                THEN
;  2366          2496  4                    DBG$GB_RADIX[DBG$B_RADIX_INPUT] = .RADIX;
;  2367          2497  4
;  2368          2498  4                IF NOT .NOUN_NODE[DBG$L_NOUN_VALUE2]
;  2369          2499  4                THEN
;  2370          2500  4                    DBG$GB_RADIX[DBG$B_RADIX_OUTPUT] = .RADIX;
;  2371          2501  4
;  2372          2502  4                END
;  2373          2503  4
;  2374          2504  3            ELSE IF .VERB_NODE[DBG$B_VERB_COMPOSITE] EQL SET_RADIX_OVERRIDE
;  2375          2505  3            THEN
;  2376          2506  3                DBG$GB_RADIX[DBG$B_RADIX_OUTPUT_OVER] = .RADIX
;  2377          2507  3
;  2378          2508  3            ELSE
;  2379          2509  3                $DBG_ERROR('DBGNSET\DBG$NEXECUTE_SET');
```

```
;  2380          2510  3
;  2381          2511  2                END;                        ! End of SET RADIX command
;  2382          2512  2
;  2383          2513  2
;  2384          2514  2            [set_scope] :
;  2385          2515  3                BEGIN
;  2386          2516  3                LOCAL
;  2387          2517  3                    SCOPE_LIST : REF VECTOR [,LONG],
;  2388          2518  3                    BAD_SCOPE,
;  2389          2519  3                    ERR_INDEX;
;  2390          2520  3
;  2391          2521  3                ! Recover the scope vector
;  2392          2522  3                !
;  2393          2523  3                scope_list = .noun_node [dbg$l_noun_value];
;  2394          2524  3
;  2395          2525  3                ! Set the scopes. 1 was filled into the adverb field if
;  2396          2526  3                ! SET SCOPE/MODULE was specified; 0 otherwise.
;  2397          2527  3                !
;  2398          2528  3                dbg$rst_setscope (.scope_list, err_index,
;  2399          2529  3                                  .verb_node[dbg$l_verb_adverb_ptr]);
;  2400          2530  3
;  2401          2531  3                ! Check for an error
;  2402          2532  3                !
;  2403          2533  3                IF .err_index NEQ 0
;  2404          2534  3                THEN
;  2405          2535  4                    BEGIN
;  2406          2536  4
;  2407          2537  4                    ! Translate the bad scope to a counted string
;  2408          2538  4                    !
;  2409          2539  4                    dbg$npathdesc_to_cs (.scope_list [.err_index], bad_scope);
;  2410          2540  4
;  2411          2541  4                    ! Format an error message
;  2412          2542  4                    !
;  2413          2543  4                    .message_vect = dbg$nmake_arg_vect (dbg$_badscope, 1, .bad_scope);
;  2414          2544  4
;  2415          2545  4                    RETURN sts$k_severe;
;  2416          2546  3                    END;
;  2417          2547  2                END;
;  2418          2548  2
;  2419          2549  2            [set_search] :
;  2420          2550  3                BEGIN
;  2421          2551  3
;  2422          2552  3                LITERAL
;  2423          2553  3                    noun_literal_all = 1,
;  2424          2554  3                    noun_literal_next = 2,
;  2425          2555  3                    noun_literal_string = 3,
;  2426          2556  3                    noun_literal_ident = 4;
;  2427          2557  3
;  2428          2558  3                ! First set the search structure to the user defined level
;  2429          2559  3                !
;  2430          2560  3                dbg$set_search_lvl (user_def_search);
;  2431          2561  3
;  2432          2562  3                ! Add the new user defined settings. Remember the first noun
;  2433          2563  3                ! node has already been recovered.
;  2434          2564  3                !
;  2435          2565  3                WHILE TRUE DO
;  2436          2566  4                    BEGIN
```

```
;  2437     2567  4
;  2438     2568  4                CASE .noun_node [dbg$l_noun_value] FROM noun_literal_all
;  2439     2569  4                                                   TO   noun_literal_ident
;  2440     2570  4                    OF SET
;  2441     2571  4
;  2442     2572  4                    [noun_literal_all] : ! SET SEARCH ALL
;  2443     2573  5                        BEGIN
;  2444     2574  5                        dbg$gb_search_ptr[search_all] = TRUE;
;  2445     2575  4                        END;
;  2446     2576  4
;  2447     2577  4                    [noun_literal_next] : ! SET SEARCH NEXT
;  2448     2578  5                        BEGIN
;  2449     2579  5                        dbg$gb_search_ptr[search_all] = FALSE;
;  2450     2580  4                        END;
;  2451     2581  4
;  2452     2582  4                    [noun_literal_ident] : ! SET SEARCH IDENT
;  2453     2583  5                        BEGIN
;  2454     2584  5                        dbg$gb_search_ptr[search_ident] = TRUE;
;  2455     2585  4                        END;
;  2456     2586  4
;  2457     2587  4                    [noun_literal_string] : ! SET SEARCH STRING
;  2458     2588  5                        BEGIN
;  2459     2589  5                        dbg$gb_search_ptr[search_ident] = FALSE;
;  2460     2590  4                        END;
;  2461     2591  4
;  2462     2592  4                    TES;
;  2463     2593  4
;  2464     2594  4                ! Now check for another switch. If none, exit the loop.
;  2465     2595  4                !
;  2466     2596  4                IF .noun_node [dbg$l_noun_link] EQL 0
;  2467     2597  4                THEN
;  2468     2598  4                    EXITLOOP;
;  2469     2599  4
;  2470     2600  4                ! Update the noun node to the next object
;  2471     2601  4                !
;  2472     2602  4                noun_node = .noun_node [dbg$l_noun_link];
;  2473     2603  4
;  2474     2604  3                END;        ! End of loop
;  2475     2605  3
;  2476     2606  2            END;
;  2477     2607  2
;  2478     2608  2        [set_source] :              ! Set source directory search list
;  2479     2609  3            BEGIN
;  2480     2610  3
;  2481     2611  3            dbg$src_set_source(
;  2482     2612  3                .noun_node[dbg$l_adjective_ptr],
;  2483     2613  3                .noun_node[dbg$l_noun_value]);
;  2484     2614  3
;  2485     2615  2            END;
;  2486     2616  2
;  2487     2617  2
;  2488     2618  2        [set_step] :                ! Set step values
;  2489     2619  3            BEGIN
;  2490     2620  3            RETURN DBG$EVENT_SEMANTICS (.VERB_NODE, .MESSAGE_VECT);
;  2491     2621  2            END;
;  2492     2622  2
;  2493     2623  2
```

```
;  2494          2624  2          ! Execute the SET TASK command.
;  2495          2625  2          !
;  2496          2626  2          [SET_TASK]:
;  2497          2627  2              DBG$NEXECUTE_SET_TASK(.VERB_NODE);
;  2498          2628  2
;  2499          2629  2
;  2500          2630  2          ! Execute the SET TERMINAL command.
;  2501          2631  2          !
;  2502          2632  2          [SET_TERMINAL]:
;  2503          2633  2              DBG$SCR_EXECUTE_SETTERM_CMD(.VERB_NODE);
;  2504          2634  2
;  2505          2635  2
;  2506          2636  2          ! Execute the SET TRACE command.
;  2507          2637  2          !
;  2508          2638  2          [SET_TRACE]:
;  2509          2639  2              DBG$EVENT_SEMANTICS(.VERB_NODE, .MESSAGE_VECT);
;  2510          2640  2
;  2511          2641  2
;  2512          2642  2          [set_type] :
;  2513          2643  3              BEGIN
;  2514          2644  3              LOCAL
;  2515          2645  3                  LENGTH_NODE : REF dbg$noun_node;
;  2516          2646  3
;  2517          2647  3              ! We always have a second noun node (containing the length).
;  2518          2648  3              !
;  2519          2649  3              length_node = .noun_node [dbg$l_noun_link];
;  2520          2650  3
;  2521          2651  3              dbg$gl_dflttyp = .noun_node [dbg$l_noun_value];
;  2522          2652  3              dbg$gw_dfltleng = .length_node [dbg$l_noun_value];
;  2523          2653  2              END;
;  2524          2654  2
;  2525          2655  2          [set_type_override] :
;  2526          2656  3              BEGIN
;  2527          2657  3              LOCAL
;  2528          2658  3                  LENGTH_NODE : REF dbg$noun_node;
;  2529          2659  3
;  2530          2660  3              ! We always have a second noun node (containing the length).
;  2531          2661  3              !
;  2532          2662  3              length_node = .noun_node [dbg$l_noun_link];
;  2533          2663  3
;  2534          2664  3              dbg$gl_gbltyp = .noun_node [dbg$l_noun_value];
;  2535          2665  3              dbg$gw_gbllngth = .length_node [dbg$l_noun_value];
;  2536          2666  2              END;
;  2537          2667  2
;  2538          2668  2
;  2539          2669  2          ! Execute the SET WATCH command.
;  2540          2670  2          !
;  2541          2671  2          [set_watch]:
;  2542          2672  2              DBG$EVENT_SEMANTICS (.VERB_NODE, .MESSAGE_VECT);
;  2543          2673  2
;  2544          2674  2
;  2545          2675  2          ! Execute the SET WINDOW command.
;  2546          2676  2          !
;  2547          2677  2          [SET_WINDOW]:
;  2548          2678  2              DBG$SCR_EXECUTE_SETWIND_CMD(.VERB_NODE);
;  2549          2679  2
;  2550          2680  2
```

```
;  2551          2681  2              ! Any other command CASE index should never occur.  Hence we signal
;  2552          2682  2              ! an internal DEBUG error in this case.
;  2553          2683  2              !
;  2554          2684  2              [INRANGE, OUTRANGE]:
;  2555          2685  2                  $DBG_ERROR('DBGNSET\NEXECUTE_SET');
;  2556          2686  2
;  2557          2687  2              TES;
;  2558          2688  2
;  2559          2689  2
;  2560          2690  2      ! The command has successfully been executed.  Now return.
;  2561          2691  2      !
;  2562          2692  2      RETURN STS$K_SUCCESS;
;  2563          2693  2
;  2564          2694  1      END;


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

45 4E 24 47 42 44 5C 54 45 53 4E 47 42 44 29  00309 P.AEG:   .ASCII  \)DBGNSET\<92>\DBG$NEXECUTE_SET unexpect\   ;
78 65 6E 75 20 54 45 53 5F 45 54 55 43 45 58  00318                                                            ;
                                  74 63 65 70  00327                                                            ;
                     78 69 64 61 72 20 64 65  0032B             .ASCII  \ed radix\                              ;
45 4E 24 47 42 44 5C 54 45 53 4E 47 42 44 18  00333 P.AEH:   .ASCII  <24>\DBGNSET\<92>\DBG$NEXECUTE_SET\       ;
                  54 45 53 5F 45 54 55 43 45  00342                                                            ;
55 43 45 58 45 4E 5C 54 45 53 4E 47 42 44 14  0034C P.AEI:   .ASCII  <20>\DBGNSET\<92>\NEXECUTE_SET\           ;
                     54 45 53 5F 45 54 45 53  0035B                                                            ;

                                                                                .EXTRN  LIB$SPAWN, SYS$WAITFR

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

                                                  0FFC 00000                    .ENTRY  DBG$NEXECUTE_SET, Save R2,R3,R4,R5,R6,R7,-  ; 1997
                                                                                        R8,R9,R10,R11                               ;
                                       5B 00000000G   00  9E 00002              MOVAB   DBG$NGET_TRANS_RADIX, R11                   ;
                                       5A 00000000G   00  9E 00009              MOVAB   DBG$GB_MOD_PTR, R10                         ;
                                       59 00000000G   00  9E 00010              MOVAB   LIB$SIGNAL, R9                              ;
                                       58 00000000G   00  9E 00017              MOVAB   DBG$GL_LOGFAB+52, R8                        ;
                                       57 00000000G   00  9E 0001E              MOVAB   DBG$GB_RADIX, R7                            ;
                                       56 00000000G   00  9E 00025              MOVAB   DBG$GB_DEF_OUT, R6                          ;
                                       5E             18  C2 0002C              SUBL2   #24, SP                                     ;
                                       52       04    AC  D0 0002F              MOVL    VERB_NODE, R2                               ; 2055
                                       53       08    A2  D0 00033              MOVL    8(R2), NOUN_NODE                            ;
                                 1D    01       01    A2  8F 00037              CASEB   1(R2), #1, #29                              ; 2060
00BB                0083  003C             047D       0003C 1$:      .WORD   98$-1$,-                                            ;
01AF                01AF  00FA             00C8       00044                  2$-1$,-                                             ;
0455                0432  03AC             0241       0004C                  8$-1$,-                                             ;
003C                003C  047D             0469       00054                  11$-1$,-                                            ;
00DD                00EE  0424             047D       0005C                  13$-1$,-                                            ;
0076                0061  0045             03EB       00064                  18$-1$,-                                            ;
0321                0321  048B             044A       0006C                  37$-1$,-                                            ;
                          043F             008C       00074                  37$-1$,-                                            ;
                                                                               43$-1$,-                                            ;
                                                                               80$-1$,-                                            ;
                                                                               93$-1$,-                                            ;
                                                                               96$-1$,-                                            ;
                                                                               97$-1$,-                                            ;
```

```
                                                                98$-1$,-
                                                                2$-1$,-
                                                                2$-1$,-
                                                                98$-1$,-
                                                                92$-1$,-
                                                                16$-1$,-
                                                                14$-1$,-
                                                                84$-1$,-
                                                                3$-1$,-
                                                                6$-1$,-
                                                                7$-1$,-
                                                                95$-1$,-
                                                                99$-1$,-
                                                                63$-1$,-
                                                                63$-1$,-
                                                                9$-1$,-
                                                                94$-1$
                    00000000'  EF  9F 00078 2$:     PUSHAB  P.AEI                                   ; 2685
                           0359  31 0007E           BRW     77$
                        08 A2  D5 00081 3$:         TSTL    8(R2)                                   ; 2075
                           7C  13 00084             BEQL    12$
                53      08 A2  D0 00086             MOVL    8(R2), NOUN_NODE                        ; 2078
00 00000000'    EF         63  E2 0008A 4$:         BBSS    (NOUN_NODE), DBG$GL_DEVELOPER, 5$       ; 2081
                        08 A3  D5 00092 5$:         TSTL    8(NOUN_NODE)                            ; 2082
                           6B  13 00095             BEQL    12$
                53      08 A3  D0 00097             MOVL    8(NOUN_NODE), NOUN_NODE                 ; 2083
                           ED  11 0009B             BRB     4$                                      ; 2079
                           01  DD 0009D 6$:         PUSHL   #1                                      ; 2095
   00000000G    00         01  FB 0009F             CALLS   #1, DBG$SET_DEFINE_LVL
                50 00000000G  00  D0 000A6          MOVL    DBG$GB_DEFINE_PTR, R0                   ; 2096
                60            63  90 000AD         MOVB    (NOUN_NODE), (R0)
                              76  11 000B0         BRB     15$                                     ; 2060
                              01  DD 000B2 7$:     PUSHL   #1                                      ; 2103
                              52  DD 000B4         PUSHL   R2
00000000G       00            02  FB 000B6         CALLS   #2, DBG$SCR_EXECUTE_DISPLAY_CMD
                              74  11 000BD         BRB     17$
                    00000000G  00  94 000BF 8$:     CLRB    DBG$GB_RESIGNAL                         ; 2110
                           03A6  31 000C5           BRW     93$                                     ; 2111
                54        04 A2  D0 000C8 9$:     MOVL    4(R2), OUTPUT_LOG                       ; 2125
                              63  DD 000CC         PUSHL   (NOUN_NODE)                             ; 2131
                    00000000G  00  9F 000CE         PUSHAB  DBG$GL_KEY_TABLE_ID                     ; 2130
00000000G       00            02  FB 000D4         CALLS   #2, SMG$SET_DEFAULT_STATE
                09            50  E8 000DB         BLBS    SET_STATUS, 10$                         ; 2132
                    00028150  8F  DD 000DE         PUSHL   #164176
                69            01  FB 000E4         CALLS   #1, LIB$SIGNAL
                49            54  E9 000E7 10$:    BLBC    OUTPUT_LOG, 17$                         ; 2133
                              63  DD 000EA         PUSHL   (NOUN_NODE)                             ; 2135
                              01  DD 000EC         PUSHL   #1
                    000280C3  8F  DD 000EE         PUSHL   #164035
                           02EB  31 000F4           BRW     78$
                              63  DD 000F7 11$:    PUSHL   (NOUN_NODE)                             ; 2141
                              7E  D4 000F9         CLRL    -(SP)
00000000G       00            02  FB 000FB         CALLS   #2, DBG$SET_LANG
                              2F  11 00102 12$:    BRB     17$                                     ; 2060
00000000G       00            63  D0 00104 13$:    MOVL    (NOUN_NODE), DBG$GL_LOG_BUF             ; 2146
                        08    AC  DD 0010B         PUSHL   MESSAGE_VEC                             ; 2147
          0000V CF            01  FB 0010E         CALLS   #1, DBG$NSET_LOG
```

```
                       1D          50 E8 00113       BLBS    R0, 17$                               ;
                                 030A 31 00116       BRW     83$                                   ; 2149
             00000000G 00          63 D0 00119 14$:  MOVL    (NOUN_NODE), DBG$SRC_LEFT_MARGIN      ; 2154
             00000000G 00     0C   A3 D0 00120       MOVL    12(NOUN_NODE), DBG$SRC_RIGHT_MARGIN   ; 2155
                                   09 11 00128 15$:  BRB     17$                                   ; 2060
                                   63 DD 0012A 16$:  PUSHL   (NOUN_NODE)                           ; 2160
             00000000G 00          01 FB 0012C       CALLS   #1, DBG$SRC_SET_MAX_FILES             ;
                                 039A 31 00133 17$:  BRW     100$                                  ; 2060
                                   01 DD 00136 18$:  PUSHL   #1                                    ; 2172
             00000000G 00          01 FB 00138       CALLS   #1, DBG$SET_MOD_LVL                   ;
                                   53 D5 0013F 19$:  TSTL    NOUN_NODE                             ; 2177
                                   F0 13 00141       BEQL    17$                                   ;
              0C           01      63 CF 00143       CASEL   (NOUN_NODE), #1, #12                  ; 2179
   0042     0028         0021    001A    00147 20$:  .WORD   21$-20$,-                             ;
   006B     002F         0052    0049    0014F               22$-20$,-                             ;
   005A     0086         007A    006F    00157               23$-20$,-                             ;
                                 0063    0015F               25$-20$,-                             ;
                                                             26$-20$,-                             ;
                                                             27$-20$,-                             ;
                                                             24$-20$,-                             ;
                                                             30$-20$,-                             ;
                                                             31$-20$,-                             ;
                                                             33$-20$,-                             ;
                                                             34$-20$,-                             ;
                                                             28$-20$,-                             ;
                                                             29$-20$                               ;
                       67   0202   8F B0 00161 21$:  MOVW    #514, DBG$GB_RADIX                    ; 2185
                                   7C 11 00166       BRB     36$                                   ; 2179
                       67   0808   8F B0 00168 22$:  MOVW    #2056, DBG$GB_RADIX                   ; 2191
                                   75 11 0016D       BRB     36$                                   ; 2179
                       67   0A0A   8F B0 0016F 23$:  MOVW    #2570, DBG$GB_RADIX                   ; 2197
                                   6E 11 00174       BRB     36$                                   ; 2179
                                   01 DD 00176 24$:  PUSHL   #1                                    ; 2204
                       6B          01 FB 00178       CALLS   #1, DBG$NGET_TRANS_RADIX              ;
                       67          50 90 0017B       MOVB    R0, DBG$GB_RADIX                      ;
                                   01 DD 0017E       PUSHL   #1                                    ; 2206
                       6B          01 FB 00180       CALLS   #1, DBG$NGET_TRANS_RADIX              ;
                 01    67          50 90 00183       MOVB    R0, DBG$GB_RADIX+1                    ;
                                   5B 11 00187       BRB     36$                                   ; 2179
                       67   1010   8F B0 00189 25$:  MOVW    #4112, DBG$GB_RADIX                   ; 2211
                                   54 11 0018E       BRB     36$                                   ; 2179
                       50          6A D0 00190 26$:  MOVL    DBG$GB_MOD_PTR, R0                    ; 2216
                 02    A0          01 90 00193       MOVB    #1, 2(R0)                             ;
                                   4B 11 00197       BRB     36$                                   ;
                       50          6A D0 00199 27$:  MOVL    DBG$GB_MOD_PTR, R0                    ; 2219
                              02   A0 94 0019C       CLRB    2(R0)                                 ;
                                   43 11 0019F       BRB     36$                                   ;
                       50          6A D0 001A1 28$:  MOVL    DBG$GB_MOD_PTR, R0                    ; 2222
                 09    A0          01 90 001A4       MOVB    #1, 9(R0)                             ;
                                   3A 11 001A8       BRB     36$                                   ;
                       50          6A D0 001AA 29$:  MOVL    DBG$GB_MOD_PTR, R0                    ; 2225
                              09   A0 94 001AD       CLRB    9(R0)                                 ;
                                   32 11 001B0       BRB     36$                                   ;
                                   01 DD 001B2 30$:  PUSHL   #1                                    ; 2228
                                   02 11 001B4       BRB     32$                                   ;
                                   7E D4 001B6 31$:  CLRL    -(SP)                                 ; 2231
             00000000G 00          01 FB 001B8 32$:  CALLS   #1, DBG$SCR_SCREEN_MODE               ;
```

```
                                   23  11 001BF          BRB     36$                                 ;
              00000000G  00        01  90 001C1  33$:    MOVB    #1, DBG$GB_KEYPAD_INPUT             ; 2235
                         6E        01  D0 001C8          MOVL    #1, (SP)                            ; 2236
                                   08  11 001CB          BRB     35$                                 ;
                       00000000G   00  94 001CD  34$:    CLRB    DBG$GB_KEYPAD_INPUT                 ; 2241
                                   6E  D4 001D3          CLRL    (SP)                                ; 2242
                                   5E  DD 001D5  35$:    PUSHL   SP                                  ;
                       00000000G   00  9F 001D7          PUSHAB  DBG$GL_KEYBOARD_ID                  ;
              00000000G  00        02  FB 001DD          CALLS   #2, SMG$SET_KEYPAD_MODE             ;
                         53    08  A3  D0 001E4  36$:    MOVL    8(NOUN_NODE), NOUN_NODE             ; 2250
                                 FF54  31 001E8          BRW     19$                                 ; 2177
                                   55  D4 001EB  37$:    CLRL    STATUS                              ; 2283
                         2F    04  A2  E9 001ED          BLBC    4(R2), 38$                          ; 2284
                    08   AE        01  D0 001F1          MOVL    #1, FLAGS                           ; 2295
                               04  AE  D4 001F5          CLRL    EVNT_FLAG                           ; 2296
                               04  AE  9F 001F8          PUSHAB  EVNT_FLAG                           ; 2297
                                   7E  7C 001FB          CLRQ    -(SP)                               ;
                                   7E  D4 001FD          CLRL    -(SP)                               ;
                               18  AE  9F 001FF          PUSHAB  FLAGS                               ;
                                   7E  7C 00202          CLRQ    -(SP)                               ;
                                   7E  D4 00204          CLRL    -(SP)                               ;
              00000000G  00        08  FB 00206          CALLS   #8, LIB$SPAWN                       ;
                         55        50  D0 0020D          MOVL    R0, STATUS                          ;
                         0D        55  E8 00210          BLBS    STATUS, 38$                         ; 2298
                                   55  DD 00213          PUSHL   STATUS                              ; 2300
                                   7E  D4 00215          CLRL    -(SP)                               ;
                       000286DB    8F  DD 00217          PUSHL   #165595                             ;
                         69        03  FB 0021D          CALLS   #3, LIB$SIGNAL                      ;
         54         04   A2        02  C7 00220  38$:    DIVL3   #2, 4(R2), ALLOCATE_FLAG            ; 2306
                         08    01  A2  91 00225          CMPB    1(R2), #8                           ; 2310
                                   0D  12 00229          BNEQ    39$                                 ;
                                   54  DD 0022B          PUSHL   ALLOCATE_FLAG                       ; 2312
                                   7E  7C 0022D          CLRQ    -(SP)                               ;
              00000000G  00        03  FB 0022F          CALLS   #3, DBG$RST_SETMOD                  ;
                                   2C  11 00236          BRB     41$                                 ;
                                   53  D5 00238  39$:    TSTL    NOUN_NODE                           ; 2315
                                   28  13 0023A          BEQL    41$                                 ;
                         52        63  D0 0023C          MOVL    (NOUN_NODE), NAME_BUFF              ; 2322
                                   54  DD 0023F          PUSHL   ALLOCATE_FLAG                       ; 2324
                         7E        62  9A 00241          MOVZBL  (NAME_BUFF), -(SP)                  ; 2323
                               01  A2  9F 00244          PUSHAB  1(NAME_BUFF)                        ;
              00000000G  00        03  FB 00247          CALLS   #3, DBG$RST_SETMOD                  ;
                         0D        50  E8 0024E          BLBS    R0, 40$                             ;
                                   52  DD 00251          PUSHL   NAME_BUFF                           ; 2329
                                   01  DD 00253          PUSHL   #1                                  ;
                       000281E8    8F  DD 00255          PUSHL   #164328                             ;
                                 01BA  31 0025B          BRW     81$                                 ;
                         53    08  A3  D0 0025E  40$:    MOVL    8(NOUN_NODE), NOUN_NODE             ; 2335
                                   D4  11 00262          BRB     39$                                 ; 2315
                         13        55  E9 00264  41$:    BLBC    STATUS, 42$                         ; 2340
                       000286E3    8F  DD 00267          PUSHL   #165603                             ; 2343
                         69        01  FB 0026D          CALLS   #1, LIB$SIGNAL                      ;
                               04  AE  DD 00270          PUSHL   EVNT_FLAG                           ; 2344
              00000000G  00        01  FB 00273          CALLS   #1, SYS$WAITFR                      ;
                                 0253  31 0027A  42$:    BRW     100$                                ; 2060
         07              01        63  CF 0027D  43$:    CASEL   (NOUN_NODE), #1, #7                 ; 2364
  00A1   0091            007C    0010     00281  44$:    .WORD   45$-44$,-                           ;
```

```
                00CD        00C7        00B2        00AC    00289                   51$-44$,-
                                                                                    53$-44$,-
                                                                                    55$-44$,-
                                                                                    57$-44$,-
                                                                                    58$-44$,-
                                                                                    60$-44$,-
                                                                                    61$-44$
                                                0C  AE  9F 00291 45$:   PUSHAB  B_TYPE                          ; 2374
                                          0000V CF      01  FB 00294      CALLS   #1, DBG$NSETUP_LOG
                                                58      50  E8 00299      BLBS    STATUS, 50$                   ; 2375
                                                        02  DD 0029C      PUSHL   #2                            ; 2377
                                    00000000G   00      01  FB 0029E      CALLS   #1, DBG$GET_TEMPMEM
                                                  00000000G 00  D5 002A5  TSTL    DBG$GL_LOG_BUF
                                                        0A  13 002AB      BEQL    46$
                                                60      68  9B 002AD      MOVZBW  DBG$GL_LOGFAB+52, (MSG_DESC)
                                              04 A0   F8    A8  D0 002B0  MOVL    DBG$GL_LOGFAB+44, 4(MSG_DESC)
                                                        09  11 002B5      BRB     47$
                                                60      01  A8  9B 002B7 46$:    MOVZBW  DBG$GL_LOGFAB+53, (MSG_DESC)
                                              04 A0   FC    A8  D0 002BB  MOVL    DBG$GL_LOGFAB+48, 4(MSG_DESC)
                                                01      0C  AE  D1 002C0 47$:    CMPL    B_TYPE, #1
                                                        0A  12 002C4      BNEQ    48$
                                                52      D4  A8  D0 002C6  MOVL    DBG$GL_LOGFAB+8, STS
                                                51      D8  A8  D0 002CA  MOVL    DBG$GL_LOGFAB+12, STV
                                                        0E  11 002CE      BRB     49$
                                                52 00000000G  00  D0 002D0 48$:   MOVL    DBG$GL_LOGRAB+8, STS
                                                51 00000000G  00  D0 002D7  MOVL    DBG$GL_LOGRAB+12, STV
                                                        51  DD 002DE 49$:   PUSHL   STV
                                                        05  BB 002E0      PUSHR   #^M<R0,R2>
                                                        01  DD 002E2      PUSHL   #1
                                                  000210A0  8F  DD 002E4      PUSHL   #135328
                                    00000000G   00      05  FB 002EA      CALLS   #5, DBG$NMAKE_ARG_VECT
                                                      012B  31 002F1      BRW     82$
                                                66      01  90 002F4 50$:   MOVB    #1, DBG$GB_DEF_OUT            ; 2379
                                                56    03    A6  E? 002F7      BLBC    DBG$GB_DEF_OUT+3, 62$         ; 2380
                                                        1C  11 002FB      BRB     54$                           ; 2382
                                                0D      01  A6  E8 002FD 51$:   BLBS    DBG$GB_DEF_OUT+1, 52$         ; 2388
                                                  00028083  8F  DD 0030?      PUSHL   #16397T                       ; 2390
                                    00000000G   00      01  FB 00307      CALLS   #1, DBG$NOUT_INFO
                                                        66  94 0030E 52$:   CLRB    DBG$GB_DEF_OUT                ; 2392
                                                        13  11 00310      BRB     56$                           ; 2393
                                              03    A6      01  90 00312 53$:   MOVB    #1, DBG$GB_DEF_OUT+3          ; 2398
                                                38      66  E9 00316      BLBC    DBG$GB_DEF_OUT, 62$           ; 2399
                                    00000000G   00      01  D0 00319 54$:   MOVL    #1, DBG$GL_SCREEN_LOG         ; 2401
                                                        2F  11 00320      BRB     62$                           ; 2364
                                                    03  A6  94 00322 55$:   CLRB    DBG$GB_DEF_OUT+3              ; 2407
                                                  00000000G 00  D4 00325 56$:   CLRL    DBG$GL_SCREEN_LOG             ; 2408
                                                        24  11 0032B      BRB     62$                           ; 2364
                                              01    A6      01  90 0032D 57$:   MOVB    #1, DBG$GB_DEF_OUT+1          ; 2413
                                                        1E  11 00331      BRB     62$                           ; 2364
                                                0D      66  E8 00333 58$:   BLBS    DBG$GB_DEF_OUT, 59$           ; 2418
                                                  00028083  8F  DD 00336      PUSHL   #16397T                       ; 2420
                                    00000000G   00      01  FB 0033C      CALLS   #1, DBG$NOUT_INFO
                                                    01  A6  94 00343 59$:   CLRB    DBG$GB_DEF_OUT+1              ; 2422
                                                        09  11 00346      BRB     62$                           ; 2364
                                              02    A6      01  90 00348 60$:   MOVB    #1, DBG$GB_DEF_OUT+2          ; 2426
                                                        03  11 0034C      BRB     62$
                                                    02  A6  94 0034E 61$:   CLRB    DBG$GB_DEF_OUT+2              ; 2429
```

```
                          08  A3  D5 00351 62$:   TSTL    8(NOUN_NODE)                        ; 2435
                              7C  13 00354        BEQL    75$
                    53    08  A3  D0 00356        MOVL    8(NOUN_NODE), NOUN_NODE             ; 2441
                            FF20  31 0035A        BRW     43$                                 ; 2362
          0C        01        63  CF 0035D 63$:   CASEL   (NOUN_NODE), #1, #12                ; 2457
 0041     0037      0032    002D     00361 64$:   .WORD   66$-64$,-
 001A     003C      001A    001A     00369                67$-64$,-
 001A     001A      001A    001A     00371                68$-64$,-
                            001A     00379                70$-64$,-
                                                          65$-64$,-
                                                          65$-64$,-
                                                          69$-64$,-
                                                          65$-64$,-
                                                          65$-64$,-
                                                          65$-64$,-
                                                          65$-64$,-
                                                          65$-64$,-
                                                          65$-64$
                00000000'     EF  9F 0037B 65$:   PUSHAB  P.AEG                               ; 2472
                              01  DD 00381        PUSHL   #1
                00028362      8F  DD 00383        PUSHL   #164706
                    69        03  FB 00389        CALLS   #3, LIB$SIGNAL
                              17  11 0038C        BRB     71$
                    50        02  D0 0038E 66$:   MOVL    #2, RADIX                           ; 2461
                              12  11 00391        BRB     71$
                    50        08  D0 00393 67$:   MOVL    #8, RADIX                           ; 2463
                              0D  11 00396        BRB     71$
                    50        0A  D0 00398 68$:   MOVL    #10, RADIX                          ; 2465
                              08  11 0039B        BRB     71$
                    50        01  D0 0039D 69$:   MOVL    #1, RADIX                           ; 2467
                              03  11 003A0        BRB     71$
                    50        10  D0 003A2 70$:   MOVL    #16, RADIX                          ; 2469
                    1B    01  A2  91 003A5 71$:   CMPB    1(R2), #27                          ; 2487
                              1D  12 003A9        BNEQ    74$
                    01        50  D1 003AB        CMPL    RADIX, #1                           ; 2490
                              05  12 003AE        BNEQ    72$
                              50  DD 003B0        PUSHL   RADIX                               ; 2492
                    6B        01  FB 003B2        CALLS   #1, DBG$NGET_TRANS_RADIX
                    01    0C  A3  D1 003B5 72$:   CMPL    12(NOUN_NODE), #1                   ; 2494
                    03        14     003B9        BGTR    73$
                    67        50  90 003BB        MOVB    RADIX, DBG$GB_RADIX                 ; 2496
                    23    0C  A3  E8 003BE 73$:   BLBS    12(NOUN_NODE), 79$                  ; 2498
               01   A7        50  90 003C2        MOVB    RADIX, DBG$GB_RADIX+1               ; 2500
                              1D  11 003C6        BRB     79$                                 ; 2487
                    1C    01  A2  91 003C8 74$:   CMPB    1(R2), #28                          ; 2504
                              06  12 003CC        BNEQ    76$
               02   A7        50  90 003CE        MOVB    RADIX, DBG$GB_RADIX+2               ; 2506
                              11  11 003D2 75$:   BRB     79$
                00000000'     EF  9F 003D4 76$:   PUSHAB  P.AEH                               ; 2509
                              01  DD 003DA 77$:   PUSHL   #1
                00028362      8F  DD 003DC        PUSHL   #164706
                    69        03  FB 003E2 78$:   CALLS   #3, LIB$SIGNAL
                            00E8  31 003E5 79$:   BRW     100$                                ; 2060
                    54        63  D0 003E8 80$:   MOVL    (NOUN_NODE), SCOPE_LIST             ; 2523
                          04  A2  DD 003EB        PUSHL   4(R2)                               ; 2529
                          14  AE  9F 003EE        PUSHAB  ERR_INDEX                           ; 2528
                              54  DD 003F1        PUSHL   SCOPE_LIST                          ;
```

```
                00000000G  00          03  FB 003F3          CALLS   #3, DBG$RST_SETSCOPE                     ;
                           50       10  AE  D0 003FA          MOVL    ERR_INDEX, R0                           ; 2533
                                        E5  13 003FE          BEQL    79$                                     ;
                                    14  AE  9F 00400          PUSHAB  BAD_SCOPE                               ; 2539
                                  6440  DD  00403          PUSHL   (SCOPE_LIST)[R0]                        ;
                00000000G  00          02  FB 00406          CALLS   #2, DBG$NPATHDESC_TO_CS                 ;
                                    14  AE  DD 0040D          PUSHL   BAD_SCOPE                               ; 2543
                                        01  DD 00410          PUSHL   #1                                      ;
                                 000281D8  8F  DD 00412          PUSHL   #164312                                 ;
                00000000G  00          03  FB 00418 81$:      CALLS   #3, DBG$NMAKE_ARG_VECT                  ;
                        08 BC          50  D0 0041F 82$:      MOVL    R0, @MESSAGE_VECT                       ;
                           50          04  D0 00423 83$:      MOVL    #4, R0                                  ; 2545
                                       04  00426              RET                                             ;
                                       01  DD 00427 84$:      PUSHL   #1                                      ; 2560
                00000000G  00          01  FB 00429          CALLS   #1, DBG$SET_SEARCH_LVL                  ;
                           50 00000000G 00  D0 00430          MOVL    DBG$GB_SEARCH_PTR, R0                   ; 2574
                     03                63  CF 00437 85$:      CASEL   (NOUN_NODE), #1, #3                     ; 2568
          0011       0017       000D        0008      0043B 86$:      .WORD   87$-86$,-                               ;
                                                                              88$-86$,-                       ;
                                                                              90$-86$,-                       ;
                                                                              89$-86$                         ;
                           60              01  90 00443 87$:      MOVB    #1, (R0)                        ; 2574
                                           0D  11 00446          BRB     91$                             ; 2568
                                           60  94 00448 88$:      CLRB    (R0)                            ; 2579
                                           09  11 0044A          BRB     91$                             ; 2568
                        01 A0              01  90 0044C 89$:      MOVB    #1, 1(R0)                       ; 2584
                                           03  11 00450          BRB     91$                             ; 2568
                                    01  A0  94  00452 90$:      CLRB    1(R0)                           ; 2589
                                    08  A3  D5  00455 91$:      TSTL    8(NOUN_NODE)                    ; 2596
                                           76  13 00458          BEQL    100$                            ;
                           53           08  A3  D0 0045A          MOVL    8(NOUN_NODE), NOUN_NODE         ; 2602
                                           D7  11 0045E          BRB     85$                             ; 2565
                                           63  DD 00460 92$:      PUSHL   (NOUN_NODE)                     ; 2613
                                    04  A3  DD 00462          PUSHL   4(NOUN_NODE)                    ; 2612
                00000000G  00          02  FB 00465          CALLS   #2, DBG$SRC_SET_SOURCE                  ;
                                           62  11 0046C          BRB     100$                            ; 2060
                                    08  AC  DD 0046E 93$:      PUSHL   MESSAGE_VECT                    ; 2620
                                           52  DD 00471          PUSHL   R2                              ;
                00000000G  00          02  FB 00473          CALLS   #2, DBG$EVENT_SEMANTICS                 ;
                                       04  0047A              RET                                             ;
                                           52  DD 0047B 94$:      PUSHL   R2                              ; 2627
                00000000G  00          01  FB 0047D          CALLS   #1, DBG$NEXECUTE_SET_TASK               ;
                                           4A  11 00484          BRB     100$                            ;
                                           52  DD 00486 95$:      PUSHL   R2                              ; 2633
                00000000G  00          01  FB 00488          CALLS   #1, DBG$SCR_EXECUTE_SETTERM_CMD         ;
                                           3F  11 0048F          BRB     100$                            ;
                           50           08  A3  D0 00491 96$:      MOVL    8(NOUN_NODE), LENGTH_NODE       ; 2649
                00000000G  00          63  D0 00495          MOVL    (NOUN_NODE), DBG$GL_DFLTTYP             ; 2651
                00000000G  00          60  B0 0049C          MOVW    (LENGTH_NODE), DBG$GW_DFLTLENG          ; 2652
                                           2B  11 004A3          BRB     100$                            ; 2060
                           50           08  A3  D0 004A5 97$:      MOVL    8(NOUN_NODE), LENGTH_NODE       ; 2662
                00000000G  00          63  D0 004A9          MOVL    (NOUN_NODE), DBG$GL_GBLTYP              ; 2664
                00000000G  00          60  B0 004B0          MOVW    (LENGTH_NODE), DBG$GW_GBLLNGTH          ; 2665
                                           17  11 004B7          BRB     100$                            ; 2060
                                    08  AC  DD 004B9 98$:      PUSHL   MESSAGE_VECT                    ; 2672
                                           52  DD 004BC          PUSHL   R2                              ;
                00000000G  00          02  FB 004BE          CALLS   #2, DBG$EVENT_SEMANTICS                 ;
```

```
                                 09  11 004C5          BRB     100$                                    ;
                                 52  DD 004C7 99$:     PUSHL   R2                                      ; 2678
                    00000000G    00  01 FB 004C9       CALLS   #1, DBG$SCR_EXECUTE_SETWIND_CMD         ;
                                 50  01 D0 004D0 100$:  MOVL    #1, R0                                  ; 2692
                                     04 004D3          RET                                             ; 2694

; Routine Size:  1236 bytes,    Routine Base:  DBG$CODE + 10E5
```

```
;  2566          2695  1 GLOBAL ROUTINE DBG$NSET_LOG  (MESSAGE_VECT) =
;  2567          2696  1 !++
;  2568          2697  1 ! FUNCTIONAL DESCRIPTION:
;  2569          2698  1 !       This routine closes any log file that is currently open and opens
;  2570          2699  1 ! the file specified by the user in SET LOG "filespec"
;  2571          2700  1 !
;  2572          2701  1 ! FORMAL PARAMETERS:
;  2573          2702  1 !
;  2574          2703  1 !       message_vect    - The address of a longword to contain the address of a
;  2575          2704  1 !                         message argument vector
;  2576          2705  1 !
;  2577          2706  1 ! IMPLICIT INPUTS:
;  2578          2707  1 !       The RMS-set IFI (internal file identifier field) bit is used to
;  2579          2708  1 ! determine if a previous log file is still open.
;  2580          2709  1 !
;  2581          2710  1 !       dbg$gl_log_buf _        pointer to log filespec buffer
;  2582          2711  1 !
;  2583          2712  1 ! IMPLICIT OUTPUTS:
;  2584          2713  1 !       none
;  2585          2714  1 !
;  2586          2715  1 ! ROUTINE VALUE:
;  2587          2716  1 !
;  2588          2717  1 !       sts$k_severe (4) -      the log could not be set to the specified filespec
;  2589          2718  1 !
;  2590          2719  1 !       sts$k_success (1) -     the log file was set to the user specified filespec
;  2591          2720  1 !
;  2592          2721  1 ! SIDE EFFECTS:
;  2593          2722  1 !       Any previously open log file is closed and a new log file is opened.
;  2594          2723  1 ! Logging is inhibited for the duration of this routine.
;  2595          2724  1 !--
;  2596          2725  1
;  2597          2726  2         BEGIN
;  2598          2727  2
;  2599          2728  2         LOCAL
;  2600          2729  2                 STATUS,
;  2601          2730  2                 OPENED_LOG,
;  2602          2731  2                 FNS : BYTE,
;  2603          2732  2                 FNA,
;  2604          2733  2                 OLD_NAM_PTR,
;  2605          2734  2                 B_TYPE,
;  2606          2735  2                 TEMP_NAM : REF $NAM_DECL,
;  2607          2736  2                 TEMP_FSE : REF VECTOR [,BYTE],
;  2608          2737  2                 TEMP_FSR : REF VECTOR [,BYTE],
;  2609          2738  2                 LOG_TEMP : BYTE;
;  2610          2739  2
;  2611          2740  2         opened_log = FALSE ;
;  2612          2741  2
;  2613          2742  2         ! inhibit logging until we can redirect the log file
;  2614          2743  2
;  2615          2744  2         log_temp = .dbg$gb_def_out [out_log];
;  2616          2745  2         dbg$gb_def_out [out_log] = FALSE;
;  2617          2746  2
;  2618          2747  2         ! see if a log file has been previously opened. If so, close it,
;  2619          2748  2         ! since it must be CLOSEd befor we can do the new $PARSE.
;  2620          2749  2
;  2621          2750  2         IF .dbg$gl_logfab [fab$w_ifi] NEQ 0
;  2622          2751  2         THEN
```

```
;  2623     2752  3          BEGIN
;  2624     2753  3
;  2625     2754  3          opened_log = TRUE;
;  2626     2755  3          fna = .dbg$gl_logfab [fab$l_fna];
;  2627     2756  3          fns = .dbg$gl_logfab [fab$b_fns];
;  2628     2757  3
;  2629     2758  3          status = $CLOSE (FAB = dbg$gl_logfab);
;  2630     2759  3          IF NOT .status
;  2631     2760  3          THEN
;  2632     2761  4                  BEGIN
;  2633     2762  4
;  2634     2763  4                  LOCAL
;  2635     2764  4                          MSG_DESC : REF dbg$stg_desc;
;  2636     2765  4
;  2637     2766  4                  msg_desc = dbg$get_tempmem(2);
;  2638     2767  4                  build_error_desc (msg_desc);
;  2639     2768  4                  .message_vect = dbg$nmake_arg_vect
;  2640     2769  4                                      (shr$_closeout+dbg_fac_code,
;  2641     2770  4                                       1, .msg_desc,
;  2642     2771  4                                       .dbg$gl_logfab[fab$l_sts],
;  2643     2772  4                                       .dbg$gl_logfab[fab$l_stv]);
;  2644     2773  4
;  2645     2774  4                  RETURN sts$k_severe;
;  2646     2775  4
;  2647     2776  3                  END;
;  2648     2777  3
;  2649     2778  3          ! get new NAM block, preserve the original
;  2650     2779  3
;  2651     2780  3                  temp_nam = dbg$get_memory ((nam$c_bln+3) / %UPVAL);
;  2652     2781  3                  $nam_init (nam = .temp_nam);
;  2653     2782  3                  temp_fse = dbg$get_memory ((nam$c_maxrss+3)/%UPVAL);
;  2654     2783  3                  temp_fsr = dbg$get_memory ((nam$c_maxrss+3)/%UPVAL);
;  2655     2784  3                  temp_nam [nam$l_esa] = .temp_fse;
;  2656     2785  3                  temp_nam [nam$l_rsa] = .temp_fsr;
;  2657     2786  3                  temp_nam [nam$b_ess] = nam$c_maxrss;
;  2658     2787  3                  temp_nam [nam$b_rss] = nam$c_maxrss;
;  2659     2788  3                  dbg$gl_logfab [fab$l_nam] = .temp_nam;
;  2660     2789  3                  old_nam_ptr = .dbg$gl_lognam;
;  2661     2790  3                  dbg$gl_lognam = .temp_nam;
;  2662     2791  3
;  2663     2792  2                  END;
;  2664     2793  2
;  2665     2794  2          ! reset the appropriate FAB file process options
;  2666     2795  2
;  2667     2796  2          dbg$gl_logfab [fab$v_cif] = 0;
;  2668     2797  2          dbg$gl_logfab [fab$v_mxv] = 1;
;  2669     2798  2          dbg$gl_logfab [fab$v_nam] = 0;
;  2670     2799  2
;  2671     2800  2          ! Set up file name
;  2672     2801  2
;  2673     2802  2          IF .dbg$gl_log_buf NEQ 0
;  2674     2803  2          THEN
;  2675     2804  3                  BEGIN
;  2676     2805  3
;  2677     2806  3                  MAP
;  2678     2807  3                      DBG$GL_LOG_BUF : REF VECTOR [,BYTE];
;  2679     2808  3
```

```
;  2680          2809  3          dbg$gl_logfab [fab$l_fna] = .dbg$gl_log_buf + 1; ! filename addr starts in 2nd byte
;  2681          2810  3          dbg$gl_logfab [fab$b_fns] = .dbg$gl_log_buf [0];  ! 1st byte is count
;  2682          2811  3
;  2683          2812  3          ! Parse the filespec to see if an explicit version number was
;  2684          2813  3          ! given. If so set the CIF bit, otherwise we must maximize
;  2685          2814  3          ! the version number.
;  2686          2815  3
;  2687          2816  3          status = $PARSE (FAB = dbg$gl_logfab);
;  2688          2817  3          IF NOT .status
;  2689          2818  3          THEN
;  2690          2819  4                  BEGIN
;  2691          2820  4                  LOCAL
;  2692          2821  4                          STS,
;  2693          2822  4                          STV,
;  2694          2823  4                          MSG_DESC : REF dbg$stg_desc;
;  2695          2824  4
;  2696          2825  4                  msg_desc = dbg$get_tempmem (2);
;  2697          2826  4
;  2698          2827  4                  ! save these because restore_nam will reset them
;  2699          2828  4
;  2700          2829  4                  sts = .dbg$gl_logfab [fab$l_sts];
;  2701          2830  4                  stv = .dbg$gl_logfab [fab$l_stv];
;  2702          2831  4                  build_error_desc (msg_desc);
;  2703          2832  4
;  2704          2833  4                  IF .opened_log
;  2705          2834  4                  THEN
;  2706          2835  5                          restore_nam
;  2707          2836  4                  ELSE
;  2708          2837  5                          BEGIN
;  2709          2838  5                          dbg$gl_logfab [fab$l_fna] = 0;
;  2710          2839  5                          dbg$gl_logfab [fab$b_fns] = 0;
;  2711          2840  5                          dbg$gl_log_buf  = 0;
;  2712          2841  4                          END;
;  2713          2842  4
;  2714          2843  4                  .message_vect = dbg$nmake_arg_vect
;  2715          2844  4                                  (shr$_syntax + dbg_fac_code,
;  2716          2845  4                                   1, .msg_desc, .sts, .stv);
;  2717          2846  4
;  2718          2847  4                  RETURN sts$k_severe;
;  2719          2848  4
;  2720          2849  3                  END;
;  2721          2850  3
;  2722          2851  3          IF .dbg$gl_lognam [nam$v_exp_ver]
;  2723          2852  3          THEN
;  2724          2853  4                  BEGIN
;  2725          2854  4                  dbg$gl_logfab [fab$v_cif] = 1;
;  2726          2855  4                  dbg$gl_logfab [fab$v_mxv] = 0;
;  2727          2856  3                  END;
;  2728          2857  3
;  2729          2858  3          dbg$gl_logfab [fab$v_nam] = 1;          ! open by NAM block since
;  2730          2859  2          END;                                    ! we've already parsed the filespec
;  2731          2860  2
;  2732          2861  2  ! now open the new log file
;  2733          2862  2
;  2734          2863  2  status = dbg$nsetup_log (b_type);
;  2735          2864  2  IF NOT .status
;  2736          2865  2  THEN
```

```
;  2737     2866  3          BEGIN
;  2738     2867  3          LOCAL
;  2739     2868  3                  MSG_DESC : REF dbg$stg_desc,
;  2740     2869  3                  STS;
;  2741     2870  3                  STV;
;  2742     2871  3
;  2743     2872  3          msg_desc = dbg$get_tempmem (2);
;  2744     2873  3          build_error_desc (msg_desc);
;  2745     2874  3          IF .b_type EQL 1
;  2746     2875  3          THEN
;  2747     2876  4                  BEGIN
;  2748     2877  4                  sts = .dbg$gl_logfab [fab$l_sts];
;  2749     2878  4                  stv = .dbg$gl_logfab [fab$l_stv];
;  2750     2879  4                  END
;  2751     2880  3          ELSE
;  2752     2881  4                  BEGIN
;  2753     2882  4                  sts = .dbg$gl_lograb [rab$l_sts];
;  2754     2883  4                  stv = .dbg$gl_lograb [rab$l_stv];
;  2755     2884  3                  END;
;  2756     2885  3
;  2757     2886  3          IF .opened_log
;  2758     2887  3          THEN
;  2759     2888  4                  restore_nam
;  2760     2889  3          ELSE
;  2761     2890  4                  BEGIN
;  2762     2891  4                  dbg$gl_logfab [fab$l_fna] = 0;
;  2763     2892  4                  dbg$gl_logfab [fab$b_fns] = 0;
;  2764     2893  4                  dbg$gl_log_buf  = 0;
;  2765     2894  4                  dbg$gl_logfab [fab$v_nam] = 0;
;  2766     2895  3                  END;
;  2767     2896  3
;  2768     2897  3                  .message_vect = dbg$nmake_arg_vect
;  2769     2898  3                                  (shr$_openout+dbg_fac_code, 3,
;  2770     2899  3                                  .msg_desc, .sts, .stv);
;  2771     2900  3
;  2772     2901  3                  RETURN sts$k_severe;
;  2773     2902  3
;  2774     2903  2                  END;
;  2775     2904  2
;  2776     2905  2          ! restore logging status and free temp storage
;  2777     2906  2
;  2778     2907  2          dbg$gb_def_out [out_log] = .log_temp;
;  2779     2908  2          IF .opened_log
;  2780     2909  2          THEN
;  2781     2910  2
;  2782     2911  2                  ! Cleanup the "original" NAM block
;  2783     2912  2
;  2784     2913  3                  BEGIN
;  2785     2914  3                  dbg$rel_memory (.old_nam_ptr);
;  2786     2915  3                  dbg$rel_memory (.dbg$gb_logfse);
;  2787     2916  3                  dbg$rel_memory (.dbg$gb_logfsr);
;  2788     2917  3
;  2789     2918  3                  dbg$gb_logfsr = .temp_fsr;
;  2790     2919  3                  dbg$gb_logfse = .temp_fse;
;  2791     2920  3
;  2792     2921  3                  ! Release the buffer that holds the log file spec. Since this
;  2793     2922  3                  ! is a counted string we have to adjust the address to free
```

```
;  2794          2923  3
;  2795          2924  3                IF .fns NEQ 0
;  2796          2925  3                THEN
;  2797          2926  3                    dbg$rel_memory (.fna-1);
;  2798          2927  2                END;
;  2799          2928  2
;  2800          2929  2        RETURN sts$k_success;
;  2801          2930  2
;  2802          2931  1        END;


                                                .EXTRN  SYS$CLOSE, SYS$PARSE

                                    0FFC 00000  .ENTRY  DBG$NSET_LOG, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 2695
                                                        R10,R11
                                5E          18  C2 00002  SUBL2  #24, SP
                                            5A  D4 00005  CLRL   OPENED_LOG                          ; 2740
                                5B 00000000G    90 00007  MOVB   DBG$GB_DEF_OUT, LOG_TEMP           ; 2744
                                   00000000G    94 0000E  CLRB   DBG$GB_DEF_OUT                     ; 2745
                                   00000000G    B5 00014  TSTW   DBG$GL_LOGFAB+2                    ; 2750
                                            03  12 0001A  BNEQ   1$
                                          00C7  31 0001C  BRW    5$
                                5A          01  D0 0001F  1$:   MOVL   #1, OPENED_LOG               ; 2754
                                57 00000000G    D0 00022  MOVL   DBG$GL_LOGFAB+44, FNA             ; 2755
                                6E 00000000G    90 00029  MOVB   DBG$GL_LOGFAB+52, FNS             ; 2756
                                   00000000G    9F 00030  PUSHAB DBG$GL_LOGFAB                      ; 2758
                        00000000G  00           01  FB 00036  CALLS  #1, SYS$CLOSE
                                59              50  D0 0003D  MOVL   R0, STATUS
                                45              59  E8 00040  BLBS   STATUS, 4$                    ; 2759
                                                02  DD 00043  PUSHL  #2                            ; 2766
                                00000000G   00  01  FB 00045  CALLS  #1, DBG$GET_TEMPMEM
                                    00000000G   00  D5 0004C  TSTL   DBG$GL_LOG_BUF                 ; 2767
                                            11  13 00052  BEQL   2$
                                60 00000000G    00  9B 00054  MOVZBW DBG$GL_LOGFAB+52, (MSG_DESC)
                                04 A0 00000000G 00  D0 0005B  MOVL   DBG$GL_LOGFAB+44, 4(MSG_DESC)
                                            0F  11 00063  BRB    3$
                                60 00000000G    00  9B 00065  2$:   MOVZBW DBG$GL_LOGFAB+53, (MSG_DESC)
                                04 A0 00000000G 00  D0 0006C  MOVL   DBG$GL_LOGFAB+48, 4(MSG_DESC)
                                7E 00000000G    00  7D 00074  3$:   MOVQ   DBG$GL_LOGFAB+8, -(SP)   ; 2771
                                                50  DD 0007B  PUSHL  MSG_DESC                       ; 2770
                                                01  DD 0007D  PUSHL  #1                            ; 2769
                                    00021058    8F  DD 0007F  PUSHL  #135256
                                          0319  31  00085  BRL    34$
                                                18  DD 00088  4$:   PUSHL  #24                     ; 2780
                                00000000G   00  01  FB 0008A  CALLS  #1, DBG$GET_MEMORY
                                56              50  D0 00091  MOVL   R0, TEMP_NAM
0060  8F        00              6E              00  2C 00094  MOVC5  #0, (SP), #0, #96, (TEMP_NAM) ; 2781
                                                66      0009B
                                66      6002    8F  B0 0009C  MOVW   #24578, (TEMP_NAM)
                                7E        40    8F  9A 000A1  MOVZBL #64, -(SP)                   ; 2782
                                00000000G   00  01  FB 000A5  CALLS  #1, DBG$GET_MEMORY
                                      08    AE        50  D0 000AC  MOVL   R0, TEMP_FSE
                                7E        40    8F  9A 000B0  MOVZBL #64, -(SP)                   ; 2783
                                00000000G   00  01  FB 000B4  CALLS  #1, DBG$GET_MEMORY
                                      04    AE        50  D0 000BB  MOVL   R0, TEMP_FSR
                                      0C    A6    08  AE  D0 000BF  MOVL   TEMP_FSE, 12(TEMP_NAM)   ; 2784
```

```
                04   A6        04   AE  D0 000C4          MOVL    TEMP_FSR, 4(TEMP_NAM)                    ; 2785
                0A   A6             01  8E 000C9          MNEGB   #1, 10(TEMP_NAM)                         ; 2786
                02   A6             01  8E 000CD          MNEGB   #1, 2(TEMP_NAM)                          ; 2787
00000000G       00                  56  D0 000D1          MOVL    TEMP_NAM, DBG$GL_LOGFAB+40               ; 2788
                   58 00000000G     00  D0 000D8          MOVL    DBG$GL_LOGNAM, OLD_NAM_PTR               ; 2789
00000000G       00                  56  D0 000DF          MOVL    TEMP_NAM, DBG$GL_LOGNAM                  ; 2790
00000000G       00                  02  8A 000E6 5$:      BICB2   #2, DBG$GL_LOGFAB+7                      ; 2796
00000000G       00                  02  88 000ED          BISB2   #2, DBG$GL_LOGFAB+4                      ; 2797
00000000G       00                  01  8A 000F4          BICB2   #1, DBG$GL_LOGFAB+7                      ; 2798
                   50 00000000G     00  D0 000FB          MOVL    DBG$GL_LOG_BUF, R0                       ; 2802
                                    03  12 00102          BNEQ    6$
                                  016D  31 00104          BRW     21$
00000000G       00        01        A0  9E 00107 6$:      MOVAB   1(R0), DBG$GL_LOGFAB+44                  ; 2809
00000000G       00                  60  90 0010F          MOVB    (R0), DBG$GL_LOGFAB+52                   ; 2810
                   00000000G        00  9F 00116          PUSHAB  DBG$GL_LOGFAB                            ; 2816
00000000G       00                  01  FB 0011C          CALLS   #1, SYS$PARSE
                   59               50  D0 00123          MOVL    R0, STATUS
                   03               59  E9 00126          BLBC    STATUS, 7$                               ; 2817
                                  0128  31 00129          BRW     19$
                                    02  DD 0012C 7$:      PUSHL   #2                                       ; 2825
00000000G       00                  01  FB 0012E          CALLS   #1, DBG$GET_TEMPMEM
                   52               50  D0 00135          MOVL    R0, MSG_DESC
                   55 00000000G     00  D0 00138          MOVL    DBG$GL_LOGFAB+8, STS                     ; 2829
                   54 00000000G     00  D0 0013F          MOVL    DBG$GL_LOGFAB+12, STV                    ; 2830
                      00000000G     00  D5 00146          TSTL    DBG$GL_LOG_BUF                           ; 2831
                                    11  13 0014C          BEQL    8$
                   62 00000000G     00  9B 0014E          MOVZBW  DBG$GL_LOGFAB+52, (MSG_DESC)
                04 A2 00000000G     00  D0 00155          MOVL    DBG$GL_LOGFAB+44, 4(MSG_DESC)
                                    0F  11 0015D          BRB     9$
                   62 00000000G     00  9B 0015F 8$:      MOVZBW  DBG$GL_LOGFAB+53, (MSG_DESC)
                04 A2 00000000G     00  D0 00166          MOVL    DBG$GL_LOGFAB+48, 4(MSG_DESC)
                   03               5A  E8 0016E 9$:      BLBS    OPENED_LOG, 10$                          ; 2833
                                  00BF  31 00171          BRW     17$
                                    56  DD 00174 10$:     PUSHL   TEMP_NAM                                 ; 2834
00000000G       00                  01  FB 00176          CALLS   #1, DBG$REL_MEMORY
                        04          AE  DD 0017D          PUSHL   TEMP_FSR
00000000G       00                  01  FB 00180          CALLS   #1, DBG$REL_MEMORY
                        08          AE  DD 00187          PUSHL   TEMP_FSE
00000000G       00                  01  FB 0018A          CALLS   #1, DBG$REL_MEMORY
00000000G       00                  58  D0 00191          MOVL    OLD_NAM_PTR, DBG$GL_LOGNAM
                                    58  D4 00198          CLRL    OLD_NAM_PTR
00000000G       00 00000000G        00  D0 0019A          MOVL    DBG$GL_LOGNAM, DBG$GL_LOGFAB+40
00000000G       00                  57  D0 001A5          MOVL    FNA, DBG$GL_LOGFAB+44
00000000G       00                  6E  90 001AC          MOVB    FNS, DBG$GL_LOGFAB+52
00000000G       00                  03  88 001B3          BISB2   #3, DBG$GL_LOGFAB+7
00000000G       00                  02  8A 001BA          BICB2   #2, DBG$GL_LOGFAB+4
                        0C          AE  9F 001C1          PUSHAB  TYPE_B
0000V           CF                  01  FB 001C4          CALLS   #1, DBG$NSETUP_LOG
                   5E               50  E8 001C9          BLBS    ERR, 16$
                                    02  DD 001CC          PUSHL   #2
00000000G       00                  01  FB 001CE          CALLS   #1, DBG$GET_TEMPMEM
                      00000000G     00  D5 001D5          TSTL    DBG$GL_LOG_BUF
                                    11  13 001DB          BEQL    11$
                   60 00000000G     00  9B 001DD          MOVZBW  DBG$GL_LOGFAB+52, (MSG_DESC)
                04 A0 00000000G     00  D0 001E4          MOVL    DBG$GL_LOGFAB+44, 4(MSG_DESC)
                                    0F  11 001EC          BRB     12$
                   60 00000000G     00  9B 001EE 11$:     MOVZBW  DBG$GL_LOGFAB+53, (MSG_DESC)
```

```
                04   A0 00000000G  00   D0 001F5          MOVL    DBG$GL_LOGFAB+48, 4(MSG_DESC)                ;
                     01        0C   AE   D1 001FD 12$:     CMPL    TYPE_B, #1                                   ;
                                    10   12 00201 13$:     BNEQ    14$                                          ;
                     53 00000000G  00   D0 00203          MOVL    DBG$GL_LOGFAB+8, STS                         ;
                     51 00000000G  00   D0 0020A          MOVL    DBG$GL_LOGFAB+12, STV                        ;
                                    0E   11 00211          BRB     15$                                          ;
                     53 00000000G  00   D0 00213 14$:     MOVL    DBG$GL_LOGRAB+8, STS                         ;
                     51 00000000G  00   D0 0021A          MOVL    DBG$GL_LOGRAB+12, STV                        ;
                                    51   DD 00221 15$:     PUSHL   STV                                          ;
                                    09   BB 00223          PUSHR   #^M<R0,R3>                                   ;
                                    01   DD 00225          PUSHL   #1                                           ;
                                  0171   31 00227          BRW     33$                                          ;
00000000G       00                  5B   90 0022A 16$:     MOVB    LOG_TEMP, DBG$GB_DEF_OUT                     ;
                                    12   11 00231          BRB     18$                                          ; 2833
                        00000000G   00   D4 00233 17$:     CLRL    DBG$GL_LOGFAB+44                             ; 2838
                        00000000G   00   94 00239          CLRB    DBG$GL_LOGFAB+52                             ; 2839
                        00000000G   00   D4 0023F          CLRL    DBG$GL_LOG_BUF                               ; 2840
                                    54   DD 00245 18$:     PUSHL   STV                                          ; 2845
                                    24   BB 00247          PUSHR   #^M<R2,R5>                                   ;
                                    01   DD 00249          PUSHL   #1                                           ; 2844
                        000210F8    8F   DD 0024B          PUSHL   #135416                                      ;
                                  014D   31 00251          BRW     34$                                          ;
                     50 00000000G  00   D0 00254 19$:     MOVL    DBG$GL_LOGNAM, R0                            ; 2851
                     0E        34   A0   E9 0025B          BLBC    52(R0), 20$                                  ;
00000000G       00                  02   88 0025F          BISB2   #2, DBG$GL_LOGFAB+7                          ; 2854
00000000G       00                  02   8A 00266          BICB2   #2, DBG$GL_LOGFAB+4                          ; 2855
00000000G       00                  01   88 0026D 20$:     BISB2   #1, DBG$GL_LOGFAB+7                          ; 2858
                           10   AE   9F   00274 21$:     PUSHAB  B_TYPE                                       ; 2863
          0000V CF                  01   FB 00277          CALLS   #1, DBG$NSETUP_LOG                           ;
                     59             50   D0 0027C          MOVL    R0, STATUS                                   ;
                     03             59   E9 0027F          BLBC    STATUS, 22$                                  ; 2864
                                  012B   31 00282          BRW     35$                                          ;
                                    02   DD 00285 22$:     PUSHL   #2                                           ; 2872
00000000G       00                  01   FB 00287          CALLS   #1, DBG$GET_TEMPMEM                          ;
                     55             50   D0 0028E          MOVL    R0, MSG_DESC                                 ;
                        00000000G   00   D5 00291          TSTL    DBG$GL_LOG_BUF                               ; 2873
                                    11   13 00297          BEQL    23$                                          ;
                     65 00000000G  00   9B 00299          MOVZBW  DBG$GL_LOGFAB+52, (MSG_DESC)                 ;
                04   A5 00000000G  00   D0 002A0          MOVL    DBG$GL_LOGFAB+44, 4(MSG_DESC)                ;
                                    0F   11 002A8          BRB     24$                                          ;
                     65 00000000G  00   9B 002AA 23$:     MOVZBW  DBG$GL_LOGFAB+53, (MSG_DESC)                 ;
                04   A5 00000000G  00   D0 002B1          MOVL    DBG$GL_LOGFAB+48, 4(MSG_DESC)                ;
                     01        10   AE   D1 002B9 24$:     CMPL    B_TYPE, #1                                   ; 2874
                                    10   12 002BD          BNEQ    25$                                          ;
                     52 00000000G  00   D0 002BF          MOVL    DBG$GL_LOGFAB+8, STS                         ; 2877
                     54 00000000G  00   D0 002C6          MOVL    DBG$GL_LOGFAB+12, STV                        ; 2878
                                    0E   11 002CD          BRB     26$                                          ; 2874
                     52 00000000G  00   D0 002CF 25$:     MOVL    DBG$GL_LOGRAB+8, STS                         ; 2882
                     54 00000000G  00   D0 002D6          MOVL    DBG$GL_LOGRAB+12, STV                        ; 2883
                     03             5A   E8 002DD 26$:     BLBS    OPENED_LOG, 27$                              ; 2886
                                  0099   31 002E0          BRW     31$                                          ;
                                    56   DD 002E3 27$:     PUSHL   TEMP_NAM                                     ; 2887
00000000G       00                  01   FB 002E5          CALLS   #1, DBG$REL_MEMORY                           ;
                           04   AE   DD   002EC          PUSHL   TEMP_FSR                                     ;
00000000G       00                  01   FB 002EF          CALLS   #1, DBG$REL_MEMORY                           ;
                           08   AE   DD   002F6          PUSHL   TEMP_FSE                                     ;
00000000G       00                  01   FB 002F9          CALLS   #1, DBG$REL_MEMORY                           ;
```

```
00000000G  00                 58  D0 00300         MOVL    OLD_NAM_PTR, DBG$GL_LOGNAM          ;
                              58  D4 00307         CLRL    OLD_NAM_PTR                         ;
00000000G  00 00000000G       00  D0 00309         MOVL    DBG$GL_LOGNAM, DBG$GL_LOGFAB+40     ;
00000000G  00                 57  D0 00314         MOVL    FNA, DBG$GL_LOGFAB+44               ;
00000000G  00                 6E  90 0031B         MOVB    FNS, DBG$GL_LOGFAB+52               ;
00000000G  00                 03  88 00322         BISB2   #3, DBG$GL_LOGFAB+7                 ;
00000000G  00                 02  8A 00329         BICB2   #2, DBG$GL_LOGFAB+4                 ;
                        14    AE  9F 00330         PUSHAB  TYPE_B                              ;
    0000V  CF                 01  FB 00333         CALLS   #1, DBG$NSETUP_LOG                  ;
           38                 50  E8 00338         BLBS    ERR, 30$                            ;
                              02  DD 0033B         PUSHL   #2                                  ;
00000000G  00                 01  FB 0033D         CALLS   #1, DBG$GET_TEMPMEM                 ;
              00000000G       00  D5 00344         TSTL    DBG$GL_LOG_BUF                      ;
                              11  13 0034A         BEQL    28$                                 ;
           60 00000000G       00  9B 0034C         MOVZBW  DBG$GL_LOGFAB+52, (MSG_DESC)        ;
       04  A0 00000000G       00  D0 00353         MOVL    DBG$GL_LOGFAB+44, 4(MSG_DESC)       ;
                              0F  11 0035B         BRB     29$                                 ;
           60 00000000G       00  9B 0035D 28$:    MOVZBW  DBG$GL_LOGFAB+53, (MSG_DESC)        ;
       04  A0 00000000G       00  D0 00364         MOVL    DBG$GL_LOGFAB+48, 4(MSG_DESC)       ;
           01           14    AE  D1 0036C 29$:    CMPL    TYPE_B, #1                          ;
                            FE8E  31 00370         BRW     13$                                 ;
00000000G  00                 5B  90 00373 30$:    MOVB    LOG_TEMP, DBG$GB_DEF_OUT            ;
                              19  11 0037A         BRB     32$                                 ; 2886
              00000000G       00  D4 0037C 31$:    CLRL    DBG$GL_LOGFAB+44                    ; 2891
              00000000G       00  94 00382         CLRB    DBG$GL_LOGFAB+52                    ; 2892
              00000000G       00  D4 00388         CLRL    DBG$GL_LOG_BUF                      ; 2893
00000000G  00                 01  8A 0038E         BICB2   #1, DBG$GL_LOGFAB+7                 ; 2894
                              14  BB 00395 32$:    PUSHR   #^M<R2,R4>                          ; 2899
                              55  DD 00397         PUSHL   MSG_DESC                            ;
                              03  DD 00399         PUSHL   #3                                  ; 2898
              000210A0        8F  DD 0039B 33$:    PUSHL   #135328                             ;
00000000G  00                 05  FB 003A1 34$:    CALLS   #5, DBG$NMAKE_ARG_VECT              ;
       04  BC                 50  D0 003A8         MOVL    R0, @MESSAGE_VECT                   ;
           50                 04  D0 003AC         MOVL    #4, R0                              ; 2901
                                  04 003AF         RET                                         ;
00000000G  00                 5B  90 003B0 35$:    MOVB    LOG_TEMP, DBG$GB_DEF_OUT            ; 2907
           41                 5A  E9 003B7         BLBC    OPENED_LOG, 36$                     ; 2908
                              58  DD 003BA         PUSHL   OLD_NAM_PTR                         ; 2914
00000000G  00                 01  FB 003BC         CALLS   #1, DBG$REL_MEMORY                  ;
              00000000G       00  DD 003C3         PUSHL   DBG$GB_LOGFSE                       ; 2915
00000000G  00                 01  FB 003C9         CALLS   #1, DBG$REL_MEMORY                  ;
              00000000G       00  DD 003D0         PUSHL   DBG$GB_LOGFSR                       ; 2916
00000000G  00                 01  FB 003D6         CALLS   #1, DBG$REL_MEMORY                  ;
00000000G  00           04    AE  D0 003DD         MOVL    TEMP_FSR, DBG$GB_LOGFSR             ; 2918
00000000G  00           08    AE  D0 003E5         MOVL    TEMP_FSE, DBG$GB_LOGFSE             ; 2919
                              6E  95 003ED         TSTB    FNS                                 ; 2924
                              0A  13 003EF         BEQL    36$                                 ;
                        FF    A7  9F 003F1         PUSHAB  -1(FNA)                             ; 2926
00000000G  00                 01  FB 003F4         CALLS   #1, DBG$REL_MEMORY                  ;
           50                 01  D0 003FB 36$:    MOVL    #1, R0                              ; 2929
                                  04 003FE         RET                                         ; 2931

; Routine Size:  1023 bytes,    Routine Base:  DBG$CODE + 15B9
```

```
;  2804          2932  1 ROUTINE DBG$NSETUP_LOG (BLOCK_TYPE)  =
;  2805          2933  1 !++
;  2806          2934  1 ! FUNCTIONAL DESCRIPTION:
;  2807          2935  1 !       This routine creates and opens the log file.
;  2808          2936  1 !
;  2809          2937  1 ! FORMAL PARAMETERS:
;  2810          2938  1 !
;  2811          2939  1 !       block_type - indicates whether error to report is for
;  2812          2940  1 !                    FAB ( = 1) or RAB ( = 2)
;  2813          2941  1 !
;  2814          2942  1 ! IMPLICIT INPUTS:
;  2815          2943  1 !
;  2816          2944  1 !       dbg$gl_log_buf -        pointer to log filespec buffer
;  2817          2945  1 !
;  2818          2946  1 ! IMPLICIT OUTPUTS:
;  2819          2947  1 !       The resultant LOG file-spec as resolved by RMS
;  2820          2948  1 !
;  2821          2949  1 ! ROUTINE VALUE:
;  2822          2950  1 !       Returns the RMS status code if there's an error, otherwise returns TRUE
;  2823          2951  1 !
;  2824          2952  1 ! SIDE EFFECTS:
;  2825          2953  1 !       The log file is created and a RAB connected to it.
;  2826          2954  1 !--
;  2827          2955  1
;  2828          2956  2         BEGIN
;  2829          2957  2
;  2830          2958  2         LOCAL
;  2831          2959  2                 STATUS;
;  2832          2960  2         MAP
;  2833          2961  2                 DBG$GL_LOG_BUF : REF VECTOR [,BYTE];
;  2834          2962  2
;  2835          2963  2         ! See if a log file is already open
;  2836          2964  2
;  2837          2965  2         IF .dbg$gl_logfab [fab$w_ifi] NEQ 0
;  2838          2966  2         THEN
;  2839          2967  2                 RETURN sts$k_success;
;  2840          2968  2
;  2841          2969  2         .block_type = 1;
;  2842          2970  2         status = $CREATE(FAB = dbg$gl_logfab);
;  2843          2971  2
;  2844          2972  2         IF NOT .status
;  2845          2973  2         THEN
;  2846          2974  2                 RETURN (.status) ;
;  2847          2975  2
;  2848          2976  2         dbg$gl_lograb [rab$l_fab] = dbg$gl_logfab;
;  2849          2977  2         .block_type = 2;
;  2850          2978  2         status = $CONNECT (RAB = dbg$gl_lograb);
;  2851          2979  2
;  2852          2980  2         IF NOT .status
;  2853          2981  2         THEN
;  2854          2982  2                 RETURN (.status) ;
;  2855          2983  2
;  2856          2984  2
;  2857          2985  2         ! Eventually put out the title line here
;  2858          2986  2
;  2859          2987  2         RETURN true;
;  2860          2988  2
```

```
; 2861          2989  1        END;            ! END OF DBG$NSET_UP_LOG


                                                .EXTRN  SYS$CREATE, SYS$CONNECT

                                        0004 00000 DBG$NSETUP_LOG:
                                                   .WORD   Save R2                                          ; 2932
                          52 00000000G   00  9E 00002          MOVAB   DBG$GL_LOGFAB, R2
                                  02     A2  B5 00009          TSTW    DBG$GL_LOGFAB+2                       ; 2965
                                         2B  12 0000C          BNEQ    1$
                             04    BC      01  D0 0000E          MOVL    #1, @BLOCK_TYPE                       ; 2969
                                         52  DD 00012          PUSHL   R2                                    ; 2970
                  00000000G      00      01  FB 00014          CALLS   #1, SYS$CREATE
                                 1E      50  E9 0001B          BLBC    STATUS, 2$                            ; 2972
                  00000000G      00      62  9E 0001E          MOVAB   DBG$GL_LOGFAB, DBG$GL_LOGRAB+60       ; 2976
                             04    BC      02  D0 00025          MOVL    #2, @BLOCK_TYPE                       ; 2977
                                    00000000G  00  9F 00029     PUSHAB  DBG$GL_LOGRAB                         ; 2978
                  00000000G      00      01  FB 0002F          CALLS   #1, SYS$CONNECT
                                 03      50  E9 00036          BLBC    STATUS, 2$                            ; 2980
                                 50      01  D0 00039 1$:      MOVL    #1, R0                                ; 2987
                                             04 0003C 2$:      RET                                           ; 2989

; Routine Size:  61 bytes,    Routine Base:  DBG$CODE + 19B8
```

```
; 2863          2990  1 ROUTINE DUMMY(X): NOVALUE =
; 2864          2991  1 !
; 2865          2992  1 ! FUNCTION
; 2866          2993  1 !       This is a do-nothing routine that is called from within DBG$NPARSE_SET
; 2867          2994  1 !       in order to foil an incorrect BLISS compiler optimization.
; 2868          2995  1 !
; 2869          2996  2     BEGIN
; 2870          2997  2     0
; 2871          2998  1     END;


                                                0000 00000 DUMMY:  .WORD   Save nothing                          ; 2990
                                                  04 00002         RET                                           ; 2998

; Routine Size:  3 bytes,    Routine Base:  DBG$CODE + 19F5

; 2872          2999  1 END                                 ! End of module
; 2873          3000  0 ELUDOM



                                                                 .EXTRN  LIB$SIGNAL

;                                PSECT SUMMARY

;     Name                            Bytes                           Attributes

;   DBG$GLOBAL                            4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   DBG$PLIT                            865  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(0)
;   DBG$OWN                              12  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   DBG$CODE                           6648  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        62         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        48         3          97        00:02.0
;   _$255$DUA28:[DEBUG.OBJ]DSTRECRDS.L32;1
;                                          418        13         3          31        00:00.3
;   _$255$DUA28:[DEBUG.OBJ]DBGMSG.L32;1    386        17         4          22        00:00.3
;   _$255$DUA28:[DEBUG.OBJ]DBGGEN.L32;1    150        13         8          12        00:00.3
```

```
;                                       COMMAND QUALIFIERS

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

; Size:         6648 code + 881 data bytes
; Run Time:        01:54.5
; Elapsed Time:    05:45.4
; Lines/CPU Min:    1571
; Lexemes/CPU-Min: 12554
; Memory Used:  1089 pages
; Compilation Complete
```

DBGNSET
LIS

DBGNSDATA
LIS

DBGNSEARC
LIS

DBGNTYPE
LIS

DBGNSHOW
LIS

DBGPARSER
LIS