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

FILEID**DBGLEVEL3

```
DDDDDDDD    BBBBBBBB      GGGGGGGG  LL          EEEEEEEEEE  VV      VV  EEEEEEEEEE  LL            333333
DDDDDDDD    BBBBBBBB      GGGGGGGG  LL          EEEEEEEEEE  VV      VV  EEEEEEEEEE  LL            333333
DD      DD  BB      BB  GG          LL          EE          VV      VV  EE          LL          33      33
DD      DD  BB      BB  GG          LL          EE          VV      VV  EE          LL          33      33
DD      DD  BB      BB  GG          LL          EE          VV      VV  EE          LL                  33
DD      DD  BB      BB  GG          LL          EE          VV      VV  EE          LL                  33
DD      DD  BBBBBBBB    GG          LL          EEEEEEEE    VV      VV  EEEEEEEE    LL                33
DD      DD  BBBBBBBB    GG          LL          EEEEEEEE    VV      VV  EEEEEEEE    LL                33
DD      DD  BB      BB  GG  GGGGGG  LL          EE          VV      VV  EE          LL                  33
DD      DD  BB      BB  GG  GGGGGG  LL          EE          VV      VV  EE          LL                  33
DD      DD  BB      BB  GG      GG  LL          EE            VV  VV    EE          LL          33      33      ....
DD      DD  BB      BB  GG      GG  LL          EE            VV  VV    EE          LL          33      33      ....
DDDDDDDD    BBBBBBBB      GGGGGG    LLLLLLLLLL  EEEEEEEEEE      VV      EEEEEEEEEE  LLLLLLLLLL    333333        ....
DDDDDDDD    BBBBBBBB      GGGGGG    LLLLLLLLLL  EEEEEEEEEE      VV      EEEEEEEEEE  LLLLLLLLLL    333333        ....

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 DBGLEVEL3 (IDENT = 'V04-000') =
;     2          0002  0
;     3          0003  1 BEGIN
;     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 ! WRITTEN BY
;    29          0029  1 !       John Francis    August, 1982
;    30          0030  1 !
;    31          0031  1 ! MODULE FUNCTION
;    32          0032  1 !       This module contains the DEBUG kernel code for performing the
;    33          0033  1 !       EVALUATE, EXAMINE and DEPOSIT commands.
;    34          0034  1 !
;    35          0035  1
;    36          0036  1 REQUIRE 'SRC$:DBGPROLOG.REQ';
;    37          0170  1
;    38          0171  1 LIBRARY 'LIB$:DBGGEN.L32';
;    39          0172  1
;    40          0173  1 FORWARD ROUTINE
;    41          0174  1     DBG$COLLECT: NOVALUE,
;    42          0175  1     DEPOSIT_HANDLER,
;    43          0176  1     DBG$DEPOSIT: NOVALUE,
;    44          0177  1     DBG$EVALUATE: NOVALUE,
;    45          0178  1     DBG$EXAMINE: NOVALUE,
;    46          0179  1     DBG$NEXTLOC,
;    47          0180  1     DBG$PREVLOC,
;    48          0181  1     MODIFY_PRIMARY,
;    49          0182  1     PRIMARY_ORDER,
;    50          0183  1     CHECK_TEXT_DESCRIPTOR,
;    51          0184  1     FIX_UP_LENGTH;
```

```
;    53          0185  1 EXTERNAL
;    54          0186  1     DBG$GL_CURRENT_PRIMARY,          ! Pointer to the primary being processed
;    55          0187  1     DBG$REG_VALUES: VECTOR[,LONG],   ! Vector of user register values in the
;    56          0188  1     DBG$GL_CURLOC_VMSDESC,           ! Override type for %CURLOC
;    57          0189  1     DBG$GL_DEPOSIT_TOKEN,            ! Assignment operator token
;    58          0190  1     DBG$GL_IDENTITY_TOKEN,           ! Identity operator token
;    59          0191  1     DBG$GL_DFLTTYP,                  ! Default type from "SET TYPE"
;    60          0192  1     DBG$GW_DFLTLENG: WORD,           ! Length of default data-type
;    61          0193  1     DBG$GL_SIGN_FLAG;                ! Print '+' before signed variable
;    62          0194  1
;    63          0195  1 EXTERNAL ROUTINE
;    64          0196  1     DBG$BUILD_PRIMARY_SUBNODE: NOVALUE,  ! Add new primary sub-node
;    65          0197  1     DBG$DATA_LENGTH,                 ! Get length of a data-item
;    66          0198  1     DBG$DO_MAPPING,                  ! Perform "type mapping"
;    67          0199  1     DBG$EVAL_LANG_OPERATOR,          ! Evaluate operator expressions
;    68          0200  1     DBG$FLUSHBUF: NOVALUE,           ! Initialize a new print line
;    69          0201  1     DBG$GET_TEMPMEM,                 ! Allocate temporary memory
;    70          0202  1     DBG$IS_IT_ENTRY,                 ! Check for CALL entry-mask address
;    71          0203  1     DBG$INS_DECODE,                  ! Get length of instruction
;    72          0204  1     DBG$MAKE_VAL_DESC,               ! Construct value descriptor
;    73          0205  1     DBG$NGET_PAGES,                  ! Construct page list
;    74          0206  1     DBG$PC_TO_LINE_LOOKUP,           ! Get line & statement number
;    75          0207  1     DBG$PC_TO_SYMID,                 ! Look up address in SAT
;    76          0208  1     DBG$PRIM_TO_VAL,                 ! Obtain value of data-item
;    77          0209  1     DBG$PRINT: NOVALUE,              ! Formats an output line.
;    78          0210  1     DBG$PRINT_AGGREGATE : NOVALUE,   ! Print array or record
;    79          0211  1     DBG$PRINT_FIELD_REF : NOVALUE,   ! Print <p,s,e> information
;    80          0212  1     DBG$PRINT_IDENTIFIER,            ! Print name of data-item
;    81          0213  1     DBG$PRINT_VALUE: NOVALUE,        ! Print value in a given radix
;    82          0214  1     DBG$PRINT_VALUE_AS_INTEGER: NOVALUE,  ! Print absolute address
;    83          0215  1     DBG$PUSH_TEMPMEM,                ! Save temporary memory state
;    84          0216  1     DBG$POP_TEMPMEM: NOVALUE,        ! Restore      "       "     "
;    85          0217  1     DBG$NEWLINE: NOVALUE,            ! Outputs the output buffer.
;    86          0218  1     DBG$SAVE_LOC: NOVALUE,           ! Save dot
;    87          0219  1     DBG$SAVE_VAL: NOVALUE,           ! Save backslash
;    88          0220  1     DBG$SET_PAGE_PROT,               ! Set page protections
;    89          0221  1     DBG$SRC_TYPE_PC_SOURCE: NOVALUE,     ! Type source text
;    90          0222  1     DBG$STA_ADDRESS_TO_REGDESCR,     ! Translate address to reg descr
;    91          0223  1     DBG$STA_REGISTER_NAME,           ! Obtain reg name from reg descr
;    92          0224  1     DBG$STA_SETREGISTERS: NOVALUE,   ! Store context register values
;    93          0225  1     DBG$STA_SETCONTEXT: NOVALUE,     ! Set up context correctly
;    94          0226  1     DBG$STA_SYMKIND: NOVALUE,        ! Get KIND of data item
;    95          0227  1     DBG$STA_SYMNAME: NOVALUE,        ! Get NAME of data item
;    96          0228  1     DBG$STA_SYMSIZE: NOVALUE,        ! Get SIZE of data item
;    97          0229  1     DBG$STA_SYMTYPE: NOVALUE,        ! Get TYPE of data item
;    98          0230  1     DBG$STA_TYP_RECORD: NOVALUE,     ! Get symbol table information
;    99          0231  1     DBG$STA_VARIANT_SELECT,          ! Get entry from variant set
;   100          0232  1     DBG$TYPEID_FOR_ATOMIC,           ! Make dummy RST entry
;   101          0233  1     DBG$UPDATE_WATCHPOINTS: NOVALUE,     ! Update watched values after DEPOSIT
;   102          0234  1     LIB$SIGNAL;                      ! Signal an error
;   103          0235  1
;   104          0236  1 LITERAL
;   105          0237  1
;   106          0238  1     ! Verb codes for the EVALUATE command.
;   107          0239  1     !
;   108          0240  1     EVALUATE                    = 1,     ! EVALUATE verb code
;   109          0241  1     EVALUATE_ADDR               = 2,     ! EVALUATE/ADDRESS verb code
```

```
;  110          0242  1      EVALUATE_COND                = 3,     ! EVALUATE/CONDITION verb code
;  111          0243  1
;  112          0244  1
;  113          0245  1
;  114          0246  1      ! Verb codes for the EXAMINE command.
;  115          0247  1      !
;  116          0248  1      EXAMINE                      = 1,     ! EXAMINE verb code
;  117          0249  1      EXAMINE_INSTRUCTION          = 2,     ! EXAMINE/INSTRUCTION verb code
;  118          0250  1      EXAMINE_REGISTER             = 3,     ! EXAMINE register verb code
;  119          0251  1      EXAMINE_SOURCE               = 4,     ! EXAMINE/SOURCE verb code
;  120          0252  1      EXAMINE_CONDITION_VALUE      = 5,     ! EXAMINE/CONDITION verb code
;  121          0253  1      EXAMINE_PSL                  = 6,     ! EXAMINE the PSL verb code
;  122          0254  1      EXAMINE_PSW                  = 7;     ! EXAMINE the PSW verb code
;  123          0255  1
;  124          0256  1  OWN
;  125          0257  1      PAGE_LIST;                            ! Pointer to list of pages whose protec-
;  126          0258  1                                            !      tion we may have changed
```

```
;  128          0259  1 GLOBAL ROUTINE DBG$CHANGE_DTYPE(PRM_DESC, NEW_TYPE, NEW_SIZE) =
;  129          0260  1 !
;  130          0261  1 ! FUNCTION
;  131          0262  1 !       ------------------------------
;  132          0263  1 !
;  133          0264  1 ! INPUTS
;  134          0265  1 !       ------------------------------
;  135          0266  1 !
;  136          0267  1 ! OUTPUTS
;  137          0268  1 !       ------------------------------
;  138          0269  1 !
;  139          0270  1
;  140          0271  2     BEGIN
;  141          0272  2
;  142          0273  2     MAP
;  143          0274  2         PRM_DESC: REF DBG$PRIMARY;      ! Pointer to Primary Descriptor
;  144          0275  2
;  145          0276  2     LOCAL
;  146          0277  2         ADDR,                           ! The current instruction address
;  147          0278  2         SIZE,                           ! Size of V-Value Descriptor header
;  148          0279  2         VAL_DESC: REF DBG$VALDESC;      ! Pointer to Value Descriptor
;  149          0280  2
;  150          0281  2
;  151          0282  2
;  152          0283  2     ! Determine what kind of descriptor we have.
;  153          0284  2     !
;  154          0285  2     SELECTONE .PRM_DESC[DBG$B_DHDR_TYPE] OF
;  155          0286  2         SET
;  156          0287  2
;  157          0288  2
;  158          0289  2         ! Handle Primary Descriptors.
;  159          0290  2         !
;  160          0291  2         [DBG$K_PRIMARY_DESC]:
;  161          0292  2             DBG$PRIM_TO_VAL(.PRM_DESC,DBG$K_V_VALUE_DESC,VAL_DESC);
;  162          0293  2
;  163          0294  2
;  164          0295  2         ! Handle Volatile Value Descriptors.
;  165          0296  2         !
;  166          0297  2         [DBG$K_V_VALUE_DESC]:
;  167          0298  3             BEGIN
;  168          0299  3             SIZE = .PRM_DESC[DBG$W_DHDR_LENGTH];
;  169          0300  3             VAL_DESC = DBG$GET_TEMPMEM(.SIZE + (%UPVAL - 1))/%UPVAL);
;  170          0301  3             CH$MOVE(.SIZE,.PRM_DESC,.VAL_DESC);
;  171          0302  3             IF .PRM_DESC[DBG$L_DHDR_SYMIDO] NEQ 0
;  172          0303  3             THEN
;  173          0304  3                 DBG$STA_SETCONTEXT(.PRM_DESC[DBG$L_DHDR_SYMIDO]);
;  174          0305  3
;  175          0306  2             END;
;  176          0307  2
;  177          0308  2
;  178          0309  2         ! Any other descriptor type is invalid so we signal an internal
;  179          0310  2         ! DEBUG error.
;  180          0311  2         !
;  181          0312  2         [OTHERWISE]:
;  182          0313  2             SIGNAL(DBG$_ILLTYPE);
;  183          0314  2
;  184          0315  2         TES;
```

```
;  185          0316  2
;  186          0317  2
;  187          0318  2      ! If the type is DBG$K_NOTYPE, meaning type instruction, we return now.
;  188          0319  2      !
;  189          0320  2      IF .NEW_TYPE EQL DBG$K_NOTYPE THEN RETURN .VAL_DESC;
;  190          0321  2
;  191          0322  2
;  192          0323  2      ! If we get here then we are overriding the type information.  In this
;  193          0324  2      ! case, set the FCODE to "descriptor".  Also set the "override" flag.
;  194          0325  2      !
;  195          0326  2      VAL_DESC[DBG$B_DHDR_FCODE] = RST$K_TYPE_DESCR;
;  196          0327  2      VAL_DESC[DBG$V_DHDR_OVERRIDE] = TRUE;
;  197          0328  2      SELECTONE .NEW_TYPE OF
;  198          0329  2          SET
;  199          0330  2
;  200          0331  2
;  201          0332  2          ! Handle the /ASCIZ, /ASCIC, and /ASCIW qualifiers.  These refer to the
;  202          0333  2          ! zero-terminated and counted ASCII string types.
;  203          0334  2          !
;  204          0335  2          [DSC$K_DTYPE_AZ,
;  205          0336  2           DSC$K_DTYPE_AC,
;  206          0337  2           DSC$K_DTYPE_VT]:
;  207          0338  3              BEGIN
;  208          0339  4              IF (.VAL_DESC[DBG$B_VALUE_CLASS] EQL DSC$K_CLASS_UBS)
;  209          0340  3              THEN
;  210          0341  3                  SIGNAL(DBG$_UNALIGNED);
;  211          0342  3
;  212          0343  3              VAL_DESC[DBG$B_VALUE_CLASS]  = DSC$K_CLASS_VS;
;  213          0344  3              VAL_DESC[DBG$B_VALUE_DTYPE]  = .NEW_TYPE;
;  214          0345  3              VAL_DESC[DBG$W_VALUE_LENGTH] =
;  215          0346  3                                  FIX_UP_LENGTH(VAL_DESC[DBG$A_VALUE_VMSDESC]);
;  216          0347  2              END;
;  217          0348  2
;  218          0349  2
;  219          0350  2          ! Handle the /ASCID qualifier (ASCII string via its descriptor).
;  220          0351  2          !
;  221          0352  2          [DBG$K_DTYPE_AD]:
;  222          0353  3              BEGIN
;  223          0354  3              IF NOT CHECK_TEXT_DESCRIPTOR(.VAL_DESC)
;  224          0355  3              THEN
;  225          0356  3                  SIGNAL(DBG$_DESCNOTSET);
;  226          0357  3
;  227          0358  2              END;
;  228          0359  2
;  229          0360  2
;  230          0361  2          ! Handle the plain ASCII text string data type (the /ASCII qualifier).
;  231          0362  2          !
;  232          0363  2          [DSC$K_DTYPE_T]:
;  233          0364  3              BEGIN
;  234          0365  3              IF .NEW_SIZE NEQ 0
;  235          0366  3              THEN
;  236          0367  3                  VAL_DESC[DBG$W_VALUE_LENGTH] = .NEW_SIZE
;  237          0368  3
;  238          0369  3              ELSE
;  239          0370  3                  VAL_DESC[DBG$W_VALUE_LENGTH] = DBG$DATA_LENGTH(
;  240          0371  3                                      VAL_DESC[DBG$A_VALUE_VMSDESC])/%BPUNIT;
;  241          0372  3
```

```
;  242          0373  3            VAL_DESC[DBG$B_VALUE_CLASS]  = DSC$K_CLASS_Z;
;  243          0374  3            VAL_DESC[DBG$B_VALUE_DTYPE]  = DSC$K_DTYPE_T;
;  244          0375  2            END;
;  245          0376  2
;  246          0377  2
;  247          0378  2        ! Handle the /INSTRUCTION qualifier.
;  248          0379  2        !
;  249          0380  2        [DSC$K_DTYPE_ZI]:
;  250          0381  3            BEGIN
;  251          0382  3            VAL_DESC[DBG$B_VALUE_CLASS]  = DSC$K_CLASS_Z;
;  252          0383  3            ADDR = .VAL_DESC[DBG$L_VALUE_POINTER];
;  253          0384  3            IF DBG$IS_IT_ENTRY(.ADDR)
;  254          0385  3            THEN
;  255          0386  4                BEGIN
;  256          0387  4                VAL_DESC[DBG$B_VALUE_DTYPE]  = DSC$K_DTYPE_ZEM;
;  257          0388  4                VAL_DESC[DBG$W_VALUE_LENGTH] = 2;
;  258          0389  4                END
;  259          0390  4
;  260          0391  3            ELSE
;  261          0392  4                BEGIN
;  262          0393  4                VAL_DESC[DBG$B_VALUE_DTYPE]  = DSC$K_DTYPE_ZI;
;  263          0394  4                VAL_DESC[DBG$W_VALUE_LENGTH] =
;  264          0395  4                                    DBG$INS_DECODE(.ADDR, FALSE, FALSE) - .ADDR;
;  265          0396  3                END;
;  266          0397  3
;  267          0398  2            END;
;  268          0399  2
;  269          0400  2
;  270          0401  2        ! Handle the /PACKED qualifier.
;  271          0402  2        !
;  272          0403  2        [DSC$K_DTYPE_P]:
;  273          0404  3            BEGIN
;  274          0405  3            VAL_DESC[DBG$B_VALUE_CLASS]  = DSC$K_CLASS_Z;
;  275          0406  3            VAL_DESC[DBG$B_VALUE_DTYPE]  = .NEW_TYPE;
;  276          0407  3            IF .NEW_SIZE NEQ %X'0000FFFF'
;  277          0408  3            THEN
;  278          0409  3                VAL_DESC[DBG$W_VALUE_LENGTH] = .NEW_SIZE
;  279          0410  3
;  280          0411  3            ELSE
;  281          0412  4                BEGIN
;  282          0413  4
;  283          0414  4                FIELD
;  284          0415  4                    PACKED_FIELDS =
;  285          0416  4                        SET
;  286          0417  4                        SIGN_NIBBLE = [0,0,4,0]
;  287          0418  4                        TES;
;  288          0419  4                BIND
;  289          0420  4                    PACKED_DATA = .VAL_DESC[DBG$L_VALUE_POINTER]:
;  290          0421  4                                  BLOCKVECTOR[16,1,BYTE] FIELD(PACKED_FIELDS);
;  291          0422  4
;  292          0423  4                INCR I FROM 0 TO 15 DO
;  293          0424  5                    BEGIN
;  294          0425  5                    IF .PACKED_DATA[.I, SIGN_NIBBLE] GTR 9
;  295          0426  5                    THEN
;  296          0427  6                        BEGIN
;  297          0428  6                        VAL_DESC[DBG$W_VALUE_LENGTH] = (.I*2) + 1;
;  298          0429  6                        EXITLOOP;
```

```
;  299          0430  5                        END;
;  300          0431  5
;  301          0432  4                    END;
;  302          0433  4
;  303          0434  3                END;
;  304          0435  3
;  305          0436  2            END;
;  306          0437  2
;  307          0438  2
;  308          0439  2        ! Handle any other data type.
;  309          0440  2        !
;  310          0441  2        [OTHERWISE]:
;  311          0442  3            BEGIN
;  312          0443  3            VAL_DESC[DBG$B_VALUE_CLASS]  = DSC$K_CLASS_Z;
;  313          0444  3            VAL_DESC[DBG$B_VALUE_DTYPE]  = .NEW_TYPE;
;  314          0445  3            VAL_DESC[DBG$W_VALUE_LENGTH] = .NEW_SIZE;
;  315          0446  2            END;
;  316          0447  2
;  317          0448  2        TES;
;  318          0449  2
;  319          0450  2    RETURN .VAL_DESC;
;  320          0451  1    END;


                                                .TITLE  DBGLEVEL3
                                                .IDENT  \V04-000\

                                                .PSECT  DBG$OWN,NOEXE,  PIC,2

                                        00000 PAGE_LIST:
                                                .BLKB   4

                                                .EXTRN  DBG$GL_CURRENT_PRIMARY
                                                .EXTRN  DBG$REG_VALUES, DBG$GL_CURLOC_VMSDESC
                                                .EXTRN  DBG$GL_DEPOSIT_TOKEN
                                                .EXTRN  DBG$GL_IDENTITY_TOKEN
                                                .EXTRN  DBG$GL_DFLTTYP, DBG$GW_DFLTLENG
                                                .EXTRN  DBG$GL_SIGN_FLAG
                                                .EXTRN  DBG$BUILD_PRIMARY_SUBNODE
                                                .EXTRN  DBG$DATA_LENGTH
                                                .EXTRN  DBG$DO_MAPPING, DBG$EVAL_LANG_OPERATOR
                                                .EXTRN  DBG$FLUSHBUF, DBG$GET_TEMPMEM
                                                .EXTRN  DBG$IS_IT_ENTRY
                                                .EXTRN  DBG$INS_DECODE, DBG$MAKE_VAL_DESC
                                                .EXTRN  DBG$NGET_PAGES, DBG$PC_TO_LINE_LOOKUP
                                                .EXTRN  DBG$PC_TO_SYMID
                                                .EXTRN  DBG$PRIM_TO_VAL
                                                .EXTRN  DBG$PRINT, DBG$PRINT_AGGREGATE
                                                .EXTRN  DBG$PRINT_FIELD_REF
                                                .EXTRN  DBG$PRINT_IDENTIFIER
                                                .EXTRN  DBG$PRINT_VALUE
                                                .EXTRN  DBG$PRINT_VALUE_AS_INTEGER
                                                .EXTRN  DBG$PUSH_TEMPMEM
                                                .EXTRN  DBG$POP_TEMPMEM
                                                .EXTRN  DBG$NEWLINE, DBG$SAVE_LOC
                                                .EXTRN  DBG$SAVE_VAL, DBG$SET_PAGE_PROT
                                                .EXTRN  DBG$SRC_TYPE_PC_SOURCE
```

```
                                .EXTRN  DBG$STA_ADDRESS_TO_REGDESCR
                                .EXTRN  DBG$STA_REGISTER_NAME
                                .EXTRN  DBG$STA_SETREGISTERS
                                .EXTRN  DBG$STA_SETCONTEXT
                                .EXTRN  DBG$STA_SYMKIND
                                .EXTRN  DBG$STA_SYMNAME
                                .EXTRN  DBG$STA_SYMSIZE
                                .EXTRN  DBG$STA_SYMTYPE
                                .EXTRN  DBG$STA_TYP_RECORD
                                .EXTRN  DBG$STA_VARIANT_SELECT
                                .EXTRN  DBG$TYPEID_FOR_ATOMIC
                                .EXTRN  DBG$UPDATE_WATCHPOINTS
                                .EXTRN  LIB$SIGNAL

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

                        00FC 00000      .ENTRY  DBG$CHANGE_DTYPE, Save R2,R3,R4,R5,R6,R7    ; 0259
            57 00000000G  00    9E 00002    MOVAB   LIB$SIGNAL, R7                          ;
               5E         04    C2 00009    SUBL2   #4, SP                                  ;
               56   04    AC    D0 0000C    MOVL    PRM_DESC, R6                            ; 0285
         79    8F   02    A6    91 00010    CMPB    2(R6), #121                             ; 0291
                          11    12 00015    BNEQ    1$                                      ;
                          5E    DD 00017    PUSHL   SP                                      ; 0292
               7E   83    8F    9A 00019    MOVZBL  #131, -(SP)                             ;
                          56    DD 0001D    PUSHL   R6                                      ;
 00000000G     00         03    FB 0001F    CALLS   #3, DBG$PRIM_TO_VAL                     ;
                          3B    11 00026    BRB     3$                                      ;
         83    8F   02    A6    91 00028 1$:  CMPB  2(R6), #131                           ; 0297
                          2B    12 0002D    BNEQ    2$                                      ;
               52         66    3C 0002F    MOVZWL  (R6), SIZE                              ; 0299
               50   03    A2    9E 00032    MOVAB   3(R2), R0                               ; 0300
      7E       50         04    C7 00036    DIVL3   #4, R0, -(SP)                           ;
 00000000G     00         01    FB 0003A    CALLS   #1, DBG$GET_TEMPMEM                     ;
               6E         50    D0 00041    MOVL    R0, VAL_DESC                            ;
00    BE       66         52    28 00044    MOVC3   SIZE, (R6), @VAL_DESC                   ; 0301
                    0C    A6    D5 00049    TSTL    12(R6)                                  ; 0302
                          15    13 0004C    BEQL    3$                                      ;
                    0C    A6    DD 0004E    PUSHL   12(R6)                                  ; 0304
 00000000G     00         01    FB 00051    CALLS   #1, DBG$STA_SETCONTEXT                  ;
                          09    11 00058    BRB     3$                                      ;
                                                                                            ; 0285
            000287D8      8F    DD 0005A 2$:  PUSHL #165848                               ; 0313
               67         01    FB 00060    CALLS   #1, LIB$SIGNAL                          ;
               54   08    AC    D0 00063 3$:  MOVL  NEW_TYPE, R4                          ; 0320
 00000080      8F         54    D1 00067    CMPL    R4, #128                                ;
                          04    12 0006E    BNEQ    4$                                      ;
               50         6E    D0 00070    MOVL    VAL_DESC, R0                            ;
                          04    00073       RET                                             ;
               52         6E    D0 00074 4$:  MOVL  VAL_DESC, R2                          ; 0326
         06    A2         03    90 00077    MOVB    #3, 6(R2)                               ;
         04    A2   80    8F    88 0007B    BISB2   #128, 4(R2)                             ; 0327
               25         54    D1 00080    CMPL    R4, #37                                 ; 0335
                          2C    19 00083    BLSS    6$                                      ;
               27         54    D1 00085    CMPL    R4, #39                                 ;
                          27    14 00088    BGTR    6$                                      ;
               53   14    A2    9E 0008A    MOVAB   20(R2), R3                              ; 0339
               0D   03    A3    91 0008E    CMPB    3(R3), #13                              ;
                          09    12 00092    BNEQ    5$                                      ;
```

```
                                00028D08   8F   DD 00094          PUSHL   #167176                                          ; 0341
                                     67    01   FB 0009A          CALLS   #1, LIB$SIGNAL
                                       03  A3   0B 90 0009D 5$:    MOVB    #11, 3(R3)                                      ; 0343
                                       02  A3   54 90 000A1       MOVB    R4, 2(R3)                                       ; 0344
                                           53   DD 000A5          PUSHL   R3                                              ; 0346
                                    0000V  CF   01   FB 000A7     CALLS   #1, FIX_UP_LENGTH
                                           63   50   B0 000AC     MOVW    R0, (R3)
                                                7C   11 000AF     BRB     12$                                             ; 0328
                                           38   54   D1 000B1 6$:  CMPL    R4, #56                                         ; 0352
                                                15   12 000B4     BNEQ    7$
                                                52   DD 000B6     PUSHL   R2                                              ; 0354
                                    0000V  CF   01   FB 000B8     CALLS   #1, CHECK_TEXT_DESCRIPTOR
                                           6D   50   E8 000BD     BLBS    R0, 12$
                                                00028F50   8F   DD 000C0  PUSHL   #167760                                  ; 0356
                                           67   01   FB 000C6     CALLS   #1, LIB$SIGNAL
                                                62   11 000C9     BRB     12$                                             ; 0328
                                           0E   54   D1 000CB 7$:  CMPL    R4, #14                                         ; 0363
                                                25   12 000CE     BNEQ    10$
                                           53       14  A2 9E 000D0  MOVAB   20(R2), R3                                    ; 0367
                                                    0C  AC D5 000D4  TSTL    NEW_SIZE                                        ; 0365
                                                06   13 000D7     BEQL    8$
                                           63       0C  AC B0 000D9  MOVW    NEW_SIZE, (R3)                                  ; 0367
                                                10   11 000DD     BRB     9$
                                                53   DD 000DF 8$:  PUSHL   R3                                              ; 0371
                                00000000G  00   01   FB 000E1     CALLS   #1, DBG$DATA_LENGTH
                                51         50   08   C7 000E8     DIVL3   #8, R0, R1
                                           63   51   B0 000EC     MOVW    R1, (R3)
                                     02    A3   0E   B0 000EF 9$:  MOVW    #14, 2(R3)                                      ; 0374
                                                72   11 000F3     BRB     18$                                             ; 0328
                                           16   54   D1 000F5 10$: CMPL    R4, #22                                         ; 0380
                                                35   12 000F8     BNEQ    13$
                                           53       14  A2 9E 000FA  MOVAB   20(R2), R3                                    ; 0382
                                                    03  A3 94 000FE  CLRB    3(R3)
                                           55       18  A2 D0 00101  MOVL    24(R2), ADDR                                    ; 0383
                                                55   DD 00105     PUSHL   ADDR                                            ; 0384
                                00000000G  00   01   FB 00107     CALLS   #1, DBG$IS_IT_ENTRY
                                           09   50   E9 0010E     BLBC    R0, 11$
                                     02    A3   17   90 00111     MOVB    #23, 2(R3)                                      ; 0387
                                           63   02   B0 00115     MOVW    #2, (R3)                                        ; 0388
                                                4D   11 00118     BRB     18$                                             ; 0384
                                     02    A3   16   90 0011A 11$: MOVB    #22, 2(R3)                                      ; 0393
                                                7E   7C 0011E     CLRQ    -(SP)                                           ; 0395
                                                55   DD 00120     PUSHL   ADDR
                                00000000G  00   03   FB 00122     CALLS   #3, DBG$INS_DECODE
                                     63         50   55   A3 00129  SUBW3   ADDR, R0, (R3)
                                                38   11 0012D 12$: BRB     18$                                             ; 0328
                                           15   54   D1 0012F 13$: CMPL    R4, #21                                         ; 0403
                                                2A   12 00132     BNEQ    16$
                                           54       16  A2 9B 00134  MOVZBW  R4, 22(R2)                                    ; 0406
                                00000FFFF  8F   0C   AC D1 00138     CMPL    NEW_SIZE, #65535                                ; 0407
                                                20   12 00140     BNEQ    17$
                                                50   D4 00142     CLRL    I                                               ; 0423
09         18 B240                   04   00   ED 00144 14$: CMPZV   #0, #4, @24(R2)[I], #9                               ; 0425
                                           51   50   01   78 0014D  ASHL    #1, I, R1                                       ; 0428
                                     14    A2   51   01   A1 00151  ADDW3   #1, R1, 20(R2)
                                                0F   11 00156     BRB     18$                                             ; 0427
```

```
                E8              50              0F  F3 00158 15$:    AOBLEQ  #15, I, 14$                ; 0423
                                                09  11 0015C        BRB     18$                         ; 0328
                        16      A2              54  98 0015E 16$:    MOVZBW  R4, 22(R2)                  ; 0444
                        14      A2      0C      AC  B0 00162 17$:    MOVW    NEW_SIZE, 20(R2)            ; 0445
                                50              52  D0 00167 18$:    MOVL    R2, R0                      ; 0450
                                                    04 0016A        RET                                 ; 0451

; Routine Size:  363 bytes,    Routine Base:  DBG$CODE + 0000
```

```
;  322          0452  1 GLOBAL ROUTINE DBG$COLLECT(PRM_DESC) : NOVALUE =
;  323          0453  1 !
;  324          0454  1 !  FUNCTION
;  325          0455  1 !       ------------------------------
;  326          0456  1 !
;  327          0457  1 !  INPUTS
;  328          0458  1 !       ------------------------------
;  329          0459  1 !
;  330          0460  1 !  OUTPUTS
;  331          0461  1 !       ------------------------------
;  332          0462  1 !
;  333          0463  1
;  334          0464  2     BEGIN
;  335          0465  2
;  336          0466  2     MAP
;  337          0467  2         PRM_DESC: REF DBG$PRIMARY;      ! Pointer to Primary Descriptor
;  338          0468  2
;  339          0469  2     BUILTIN
;  340          0470  2         REMQUE;                         ! Remove queue entry from list
;  341          0471  2
;  342          0472  2     LOCAL
;  343          0473  2         XXXXXXX;                        !<<-------------------------------------
;  344          0474  2
;  345          0475  2
;  346          0476  2
;  347          0477  2     ! --------------
;  348          0478  2     !
;  349          0479  2     IF (.PRM_DESC NEQA 0) THEN
;  350          0480  2      IF (.PRM_DESC[DBG$B_DHDR_TYPE] EQL DBG$K_PRIMARY_DESC) THEN
;  351          0481  2       IF .PRM_DESC[DBG$V_DHDR_AGGR] THEN
;  352          0482  3         BEGIN
;  353          0483  3         LOCAL SUB_NODE : REF DBG$PRIM_NODE;
;  354          0484  3
;  355          0485  3         SUB_NODE = .PRM_DESC[DBG$L_PRIM_BLINK];
;  356          0486  3
;  357          0487  4         IF (.SUB_NODE[DBG$B_PNODE_FCODE] EQL RST$K_TYPE_ARRAY)
;  358          0488  3                                 AND
;  359          0489  4            (.SUB_NODE[DBG$B_PNARR_DTYPE] EQL DSC$K_DTYPE_T)
;  360          0490  3                                 AND
;  361          0491  4            (.SUB_NODE[DBG$W_PNARR_LENGTH] EQL 1)
;  362          0492  3         THEN
;  363          0493  4             BEGIN
;  364          0494  4             BIND S_VECTOR = SUB_NODE[DBG$A_PNARR_SVECTOR] : DBG$PRIM_NODE_SUBS;
;  365          0495  4             LOCAL DIMS,SIZE,BASE,TYPEID,SYMID;
;  366          0496  4             DIMS = .SUB_NODE[DBG$B_PNARR_DIMCNT] - 1;
;  367          0497  4             IF .S_VECTOR[.DIMS,DBG$L_PNSUB_STRIDE] NEQ 1 THEN RETURN;
;  368          0498  4             IF .S_VECTOR[.DIMS,DBG$L_PNSUB_TYPEID] NEQ 0 THEN RETURN;
;  369          0499  4
;  370          0500  4             BASE =  .S_VECTOR[.DIMS,DBG$L_PNSUB_LBOUND];
;  371          0501  4             SIZE = (.S_VECTOR[.DIMS,DBG$L_PNSUB_UBOUND] - .BASE) + 1;
;  372          0502  4             PRM_DESC[DBG$W_PRIM_OFFSET] = .BASE;
;  373          0503  4             PRM_DESC[DBG$W_PRIM_LENGTH] = .SIZE;
;  374          0504  4             PRM_DESC[DBG$V_DHDR_SUBREF] = TRUE;
;  375          0505  4             PRM_DESC[DBG$V_DHDR_TMPREF] = TRUE;
;  376          0506  4             TYPEID = DBG$TYPEID_FOR_ATOMIC(DSC$K_DTYPE_T,.SIZE*%BPUNIT,FALSE);
;  377          0507  4             IF .DIMS GTR 0
;  378          0508  4             THEN
```

```
;  379          0509  5                BEGIN
;  380          0510  5                SUB_NODE[DBG$B_PNARR_DIMCNT]   = .DIMS;
;  381          0511  5                SUB_NODE[DBG$L_PNARR_CELLTYPE] = .TYPEID;
;  382          0512  5                END
;  383          0513  4            ELSE
;  384          0514  5                BEGIN
;  385          0515  5                SYMID = .SUB_NODE[DBG$L_PNODE_SYMID];
;  386          0516  5                REMQUE(.SUB_NODE,SUB_NODE);
;  387          0517  5                DBG$BUILD_PRIMARY_SUBNODE(.PRM_DESC,RST$K_DATA,.SYMID,
;  388          0518  5                                         RST$K_TYPE_ATOMIC,.TYPEID,0);
;  389          0519  5                PRM_DESC[DBG$V_DHDR_AGGR] = FALSE;
;  390          0520  5                SUB_NODE = .PRM_DESC[DBG$L_PRIM_BLINK];
;  391          0521  5                SUB_NODE[DBG$L_PNODE_RELOC] = -.BASE;
;  392          0522  4                END;
;  393          0523  3            END;
;  394          0524  2        END;
;  395          0525  1    END;                        ! End of dbg$collect


                                  001C 00000          .ENTRY  DBG$COLLECT, Save R2,R3,R4                      ; 0452
                51        04    AC  D0 00002         MOVL    PRM_DESC, R1                                    ; 0479
                                01  12 00006         BNEQ    1$                                              ;
                                    04 00008         RET                                                     ;
             79 8F        02    A1  91 00009 1$:     CMPB    2(R1), #121                                     ; 0480
                                34  12 0000E         BNEQ    3$                                              ;
                01        04    A1  E8 00010         BLBS    4(R1), 2$                                       ; 0481
                                    04 00014         RET                                                     ;
                52        18    A1  D0 00015 2$:     MOVL    24(R1), SUB_NODE                                ; 0485
                01        09    A2  91 00019         CMPB    9(SUB_NODE), #1                                 ; 0487
                                25  12 0001D         BNEQ    3$                                              ;
                0E        1A    A2  91 0001F         CMPB    26(SUB_NODE), #14                               ; 0489
                                1F  12 00023         BNEQ    3$                                              ;
                01        1C    A2  B1 00025         CMPW    28(SUB_NODE), #1                                ; 0491
                                19  12 00029         BNEQ    3$                                              ;
                53        1B    A2  9A 0002B         MOVZBL  27(SUB_NODE), DIMS                              ; 0496
                                53  D7 0002F         DECL    DIMS                                            ;
          50                    53  C5 00031         MULL3   #20, DIMS, R0                                   ; 0497
                                2C A240 9F 00035     PUSHAB  44(SUB_NODE)[R0]                                ;
                01                9E D1 00039        CMPL    @(SP)+, #1                                      ;
                                  6D 12 0003C        BNEQ    5$                                              ;
                                38 A240 9F 0003E     PUSHAB  56(SUB_NODE)[R0]                                ; 0498
                                  9E D5 00042        TSTL    @(SP)+                                          ;
                                  65 12 00044 3$:    BNEQ    5$                                              ;
                                30 A240 9F 00046     PUSHAB  48(SUB_NODE)[R0]                                ; 0500
                54                9E D0 0004A        MOVL    @(SP)+, BASE                                    ;
                                34 A240 9F 0004D     PUSHAB  52(SUB_NODE)[R0]                                ; 0501
          50                    54  C3 00051         SUBL3   BASE, @(SP)+, R0                                ;
                                50  D6 00055         INCL    SIZE                                            ;
             10 A1              54  B0 00057         MOVW    BASE, 16(R1)                                    ; 0502
             12 A1              50  B0 0005B         MOVW    SIZE, 18(R1)                                    ; 0503
             04 A1        0102  8F  A8 0005F         BISW2   #258, 4(R1)                                     ; 0505
                                7E  D4 00065         CLRL    -(SP)                                           ; 0506
          7E                    03  78 00067         ASHL    #3, SIZE, -(SP)                                 ;
                                0E  DD 0006B         PUSHL   #14                                             ;
```

```
                00000000G  00          03  FB 0006D          CALLS   #3, DBG$TYPEID_FOR_ATOMIC
                                       53  D5 00074          TSTL    DIMS                                ; 0507
                                       09  15 00076          BLEQ    4$                                  ;
                      1B   A2          53  90 00078          MOVB    DIMS, 27(SUB_NODE)                  ; 0510
                      24   A2          50  D0 0007C          MOVL    TYPEID, 36(SUB_NODE)                ; 0511
                                       04  00080            RET                                         ; 0507
                           51      10   A2  D0 00081  4$:     MOVL    16(SUB_NODE), SYMID                 ; 0515
                           52          62  0F 00085          REMQUE  (SUB_NODE), SUB_NODE                ; 0516
                                       7E  D4 00088          CLRL    -(SP)                               ; 0517
                                       50  DD 0008A          PUSHL   TYPEID                              ; 0518
                                       02  DD 0008C          PUSHL   #2                                  ; 0517
                                       51  DD 0008E          PUSHL   SYMID                               ;
                                       06  DD 00090          PUSHL   #6                                  ;
                           53      04   AC  D0 00092          MOVL    PRM_DESC, R3                        ;
                                       53  DD 00096          PUSHL   R3                                  ;
                00000000G  00          06  FB 00098          CALLS   #6, DBG$BUILD_PRIMARY_SUBNODE       ;
                      04   A3          01  8A 0009F          BICB2   #1, 4(R3)                           ; 0519
                           52      18   A3  D0 000A3          MOVL    24(R3), SUB_NODE                    ; 0520
                      14   A2          54  CE 000A7          MNEGL   BASE, 20(SUB_NODE)                  ; 0521
                                       04  000AB  5$:        RET                                         ; 0525

; Routine Size:  172 bytes,    Routine Base:  DBG$CODE + 016B
```

```
;  397          0526  1 ROUTINE DEPOSIT_HANDLER(SIGNAL_ARGS: REF BLOCK[,BYTE]) =
;  398          0527  1 !
;  399          0528  1 ! FUNCTION
;  400          0529  1 !       This routine is the handler for errors that are signalled during
;  401          0530  1 !       the processing of a DEPOSIT command. A handler is necessary so
;  402          0531  1 !       that we can restore page protections that we may have changed.
;  403          0532  1 !
;  404          0533  1 ! INPUTS
;  405          0534  1 !       NONE
;  406          0535  1 !
;  407          0536  1 ! OUTPUTS
;  408          0537  1 !       ------------------------------
;  409          0538  1 !
;  410          0539  1
;  411          0540  2     BEGIN
;  412          0541  2
;  413          0542  2     LOCAL
;  414          0543  2         MESSAGE_VECT;
;  415          0544  2
;  416          0545  2     ! If we get here the second time around (on the unwind from the
;  417          0546  2     ! final handler) then resignal the exception. Do not free up
;  418          0547  2     ! the page list again.
;  419          0548  2     !
;  420          0549  2     IF .SIGNAL_ARGS[CHF$L_SIG_NAME] EQL SS$_UNWIND
;  421          0550  2     THEN
;  422          0551  2         RETURN SS$_RESIGNAL;
;  423          0552  2
;  424          0553  2     IF .PAGE_LIST NEQ 0
;  425          0554  2     THEN
;  426          0555  2         DBG$SET_PAGE_PROT(PAGE_LIST,TRUE,MESSAGE_VECT);
;  427          0556  2
;  428          0557  2     RETURN SS$_RESIGNAL;
;  429          0558  1     END;


                                0004 00000 DEPOSIT_HANDLER:
                                           .WORD   Save R2                                  ; 0526
                 52 00000000'   EF  9E 00002        MOVAB   PAGE_LIST, R2                    ;
                 5E             04  C2 00009        SUBL2   #4, SP                           ;
                 50       04    AC  D0 0000C        MOVL    SIGNAL_ARGS, R0                  ; 0549
00000920         8F       04    A0  D1 00010        CMPL    4(R0), #2336                     ;
                                11  13 00018        BEQL    1$                               ;
                                62  D5 0001A        TSTL    PAGE_LIST                        ; 0553
                                0D  13 0001C        BEQL    1$                               ;
                                5E  DD 0001E        PUSHL   SP                               ; 0555
                                01  DD 00020        PUSHL   #1                               ;
                                52  DD 00022        PUSHL   R2                               ;
00000000G       00               03  FB 00024        CALLS   #3, DBG$SET_PAGE_PROT            ;
                 50     0918    8F  3C 0002B 1$:     MOVZWL  #2328, R0                        ; 0557
                                04  00030        RET                                      ; 0558

; Routine Size:  49 bytes,    Routine Base:  DBG$CODE + 0217
```

```
;  431          0559  1 GLOBAL ROUTINE DBG$DEPOSIT(VERB_NODE : REF DBG$VERB_NODE) : NOVALUE =
;  432          0560  1 !
;  433          0561  1 ! FUNCTION
;  434          0562  1 !       This routine accepts as input the command execution tree constructed
;  435          0563  1 !       by the parse network and performs the semantic actions corresponding to
;  436          0564  1 !       the parsed DEPOSIT command. If the command cannot be executed, a message
;  437          0565  1 !       argument vector is constructed and returned.
;  438          0566  1 !
;  439          0567  1 !       Upon entrance to this routine, the command has been classified as plain
;  440          0568  1 !       DEPOSIT or Instruction DEPOSIT, and all default and override types have
;  441          0569  1 !       been set up in the adverb nodes.
;  442          0570  1 !
;  443          0571  1 !       There should be two noun nodes. The first is the target of the deposit
;  444          0572  1 !       while the second represents the source (either a value descriptor or a
;  445          0573  1 !       pointer to a counted string for instruction DEPOSITS).
;  446          0574  1 !
;  447          0575  1 ! INPUTS
;  448          0576  1 !       VERB_NODE       - A longword containing the address of the verb (head)
;  449          0577  1 !                         node of the command execution tree
;  450          0578  1 !
;  451          0579  1 ! OUTPUTS
;  452          0580  1 !       ------------------------------
;  453          0581  1 !
;  454          0582  1
;  455          0583  2     BEGIN
;  456          0584  2
;  457          0585  2
;  458          0586  2     ROUTINE TEXT_LENGTH(VAL_DESC : REF DBG$VALDESC) =
;  459          0587  3         BEGIN
;  460          0588  3         LOCAL LENGTH;
;  461          0589  3         SELECTONE .VAL_DESC[DBG$B_VALUE_DTYPE] OF
;  462          0590  3             SET
;  463          0591  3             [DSC$K_DTYPE_T]:    LENGTH = .VAL_DESC[DBG$W_VALUE_LENGTH];
;  464          0592  3             [DSC$K_DTYPE_VT]:   LENGTH = .(.VAL_DESC[DBG$L_VALUE_POINTER])<0,16,0>;
;  465          0593  3             [DSC$K_DTYPE_AC]:   LENGTH = .(.VAL_DESC[DBG$L_VALUE_POINTER])<0, 8,0>;
;  466          0594  3             [OTHERWISE]:        SIGNAL(DBG$_ILLTYPE);
;  467          0595  3             TES;
;  468          0596  3         RETURN .LENGTH;
;  469          0597  2         END;


                                0004 00000 TEXT_LENGTH:
                                           .WORD   Save R2                                          ; 0586
                           50   04   AC  D0 00002          MOVL    VAL_DESC, R0                     ; 0589
                           51   16   A0  9A 00006          MOVZBL  22(R0), R1                       ;
                           0E        51  91 0000A          CMPB    R1, #14                          ; 0591
                                     06  12 0000D          BNEQ    1$                               ;
                           52   14   A0  3C 0000F          MOVZWL  20(R0), LENGTH                   ;
                                     23  11 00013          BRB     4$                               ;
                           25        51  91 00015 1$:      CMPB    R1, #37                          ; 0592
                                     06  12 00018          BNEQ    2$                               ;
                           52   18   B0  3C 0001A          MOVZWL  @24(R0), LENGTH                  ;
                                     18  11 0001E          BRB     4$                               ;
                           26        51  91 00020 2$:      CMPB    R1, #38                          ; 0593
```

```
                                 06  12 00023         BNEQ    3$                                      ;
                          52      18  B0  9A 00025   MOVZBL  @24(R0), LENGTH                          ;
                                      0D  11 00029   BRB     4$                                      ;
                             000287D8 8F  DD 0002B 3$: PUSHL   #165848                                 ; 0594
                 00000000G   00       01  FB 00031   CALLS   #1, LIB$SIGNAL                          ;
                          50          52  D0 00038 4$: MOVL    LENGTH, R0                             ; 0596
                                      04  0003B       RET                                            ; 0597

; Routine Size:  60 bytes,    Routine Base:  DBG$CODE + 0248

;  470          0598  2
;  471          0599  2
;  472          0600  2     LOCAL
;  473          0601  2         SOURCE_NN       : REF DBG$NOUN_NODE,            ! Source of deposit
;  474          0602  2         TARGET_NN       : REF DBG$NOUN_NODE,            ! Target of deposit
;  475          0603  2         TYPE_NODE       : REF DBG$ADVERB_NODE,          ! Command qualifier
;  476          0604  2         PRIM_DESC       : REF DBG$PRIMARY,
;  477          0605  2         ADDR_DESC       : REF DBG$VALDESC,
;  478          0606  2         DATA_DESC       : REF DBG$VALDESC,
;  479          0607  2         MESSAGE_VECT;                                   ! Error message vector
;  480          0608  2
;  481          0609  2     BUILTIN CALLG;
;  482          0610  2
;  483          0611  2     ENABLE DEPOSIT_HANDLER;
;  484          0612  2
;  485          0613  2     TARGET_NN = .VERB_NODE[DBG$L_VERB_OBJECT_PTR];
;  486          0614  2     SOURCE_NN = .TARGET_NN[DBG$L_NOUN_LINK];
;  487          0615  2     PRIM_DESC = .TARGET_NN[DBG$L_NOUN_VALUE];
;  488          0616  2     DATA_DESC = .SOURCE_NN[DBG$L_NOUN_VALUE];
;  489          0617  2     PAGE_LIST = 0;
;  490          0618  2
;  491          0619  2     ! Convert both the source and the target to value descriptors.
;  492          0620  2     ! eval_lang_operator is used to convert the source because it
;  493          0621  2     ! is sensitive to any language-specific rules for converting
;  494          0622  2     ! primaries to values (e.g., in BLISS we do primary->address,
;  495          0623  2     ! in other languages we do primary->value).
;  496          0624  2     !
;  497          0625  2     IF .DATA_DESC[DBG$B_DHDR_TYPE] EQL DBG$K_PRIMARY_DESC
;  498          0626  2     THEN
;  499          0627  2         DATA_DESC = DBG$EVAL_LANG_OPERATOR(DBG$GL_IDENTITY_TOKEN,
;  500          0628  2                                            .DATA_DESC, 0);
;  501          0629  2     DBG$PRIM_TO_VAL(.PRIM_DESC,DBG$K_V_VALUE_DESC,ADDR_DESC);
;  502          0630  2
;  503          0631  2     IF (TYPE_NODE = .VERB_NODE[DBG$L_VERB_ADVERB_PTR]) EQLA 0
;  504          0632  2       THEN DBG$SAVE_LOC(.PRIM_DESC)
;  505          0633  2       ELSE
;  506          0634  3         BEGIN
;  507          0635  3         LOCAL OVERRIDE_TYPE,OVERRIDE_SIZE;
;  508          0636  3         ADDR_DESC[DBG$B_DHDR_FCODE] = RST$K_TYPE_DESCR;
;  509          0637  3         ADDR_DESC[DBG$V_DHDR_OVERRIDE] = TRUE;
;  510          0638  3         OVERRIDE_TYPE = .TYPE_NODE[DBG$B_ADVERB_LITERAL];
;  511          0639  3         OVERRIDE_SIZE = .TYPE_NODE[DBG$L_ADVERB_VALUE];
;  512          0640  3         SELECTONE .OVERRIDE_TYPE OF
;  513          0641  3             SET
;  514          0642  3             [DSC$K_DTYPE_AZ,DSC$K_DTYPE_AC,DSC$K_DTYPE_VT]:
;  515          0643  4                 BEGIN
```

```
;  516          0644  4              IF .ADDR_DESC[DSC$B_CLASS] EQL DSC$K_CLASS_UBS THEN SIGNAL(DBG$_UNALIGNED);
;  517          0645  4              ADDR_DESC[DBG$B_VALUE_CLASS]  = DSC$K_CLASS_VS;
;  518          0646  4              ADDR_DESC[DBG$B_VALUE_DTYPE]  = .OVERRIDE_TYPE;
;  519          0647  4              ADDR_DESC[DBG$W_VALUE_LENGTH] = TEXT_LENGTH(.DATA_DESC);
;  520          0648  3              END;
;  521          0649  3
;  522          0650  3          [DBG$K_DTYPE_AD]:
;  523          0651  3              IF NOT CHECK_TEXT_DESCRIPTOR(.ADDR_DESC) THEN SIGNAL(DBG$_DESCNOTSET);
;  524          0652  3
;  525          0653  3          [OTHERWISE]:
;  526          0654  4              BEGIN
;  527          0655  5              IF (.ADDR_DESC[DBG$B_VALUE_CLASS] EQL DSC$K_CLASS_UBS)
;  528          0656  4              THEN
;  529          0657  5                  BEGIN
;  530          0658  6                  IF (.OVERRIDE_SIZE GTR 32)
;  531          0659  6                  OR (.OVERRIDE_TYPE EQL DSC$K_DTYPE_ZI)
;  532          0660  6                  OR (.OVERRIDE_TYPE EQL DSC$K_DTYPE_T)
;  533          0661  5                  THEN
;  534          0662  5                      SIGNAL(DBG$_UNALIGNED);
;  535          0663  5                  END
;  536          0664  4              ELSE
;  537          0665  4                  ADDR_DESC[DBG$B_VALUE_CLASS]  =  DSC$K_CLASS_Z;
;  538          0666  4
;  539          0667  5              IF (.OVERRIDE_TYPE EQL DSC$K_DTYPE_T)
;  540          0668  4                                 AND
;  541          0669  5                 (.OVERRIDE_SIZE EQL 0)
;  542          0670  4              THEN OVERRIDE_SIZE = TEXT_LENGTH(.DATA_DESC);
;  543          0671  4
;  544          0672  4              ADDR_DESC[DBG$B_VALUE_DTYPE]  = .OVERRIDE_TYPE;
;  545          0673  4              IF .OVERRIDE_TYPE EQL DSC$K_DTYPE_ZI
;  546          0674  4              THEN
;  547          0675  5                  BEGIN
;  548          0676  5                  LOCAL
;  549          0677  5                      ADDR;
;  550          0678  5
;  551          0679  5                  ADDR = .ADDR_DESC[DBG$L_VALUE_POINTER];
;  552          0680  5                  ADDR_DESC[DBG$W_VALUE_LENGTH] =
;  553          0681  5                              DBG$INS_DECODE(.ADDR, FALSE, FALSE ) - .ADDR;
;  554          0682  5                  END
;  555          0683  5
;  556          0684  4              ELSE
;  557          0685  4                  ADDR_DESC[DBG$W_VALUE_LENGTH] = .OVERRIDE_SIZE;
;  558          0686  4
;  559          0687  3              END;
;  560          0688  3
;  561          0689  3          TES;
;  562          0690  3          DBG$SAVE_LOC(.PRIM_DESC,ADDR_DESC[DBG$A_VALUE_VMSDESC]);
;  563          0691  2          END;
;  564          0692  2
;  565          0693  2      DBG$SAVE_VAL(.DATA_DESC);
;  566          0694  2
;  567          0695  2      IF NOT DBG$NGET_PAGES(.PRIM_DESC,PAGE_LIST,MESSAGE_VECT)
;  568          0696  2       OR NOT DBG$SET_PAGE_PROT(PAGE_LIST,FALSE,MESSAGE_VECT)
;  569          0697  2        THEN
;  570          0698  3          BEGIN
;  571          0699  3          PAGE_LIST = 0;
;  572          0700  3          CALLG(.MESSAGE_VECT,LIB$SIGNAL);
```

```
;  573          0701  2          END;
;  574          0702  2
;  575          0703  2      DBG$EVAL_LANG_OPERATOR(DBG$GL_DEPOSIT_TOKEN,.DATA_DESC,.ADDR_DESC);
;  576          0704  2
;  577          0705  2      IF NOT DBG$SET_PAGE_PROT(PAGE_LIST,TRUE,MESSAGE_VECT)
;  578          0706  2      THEN
;  579          0707  3          BEGIN
;  580          0708  3          PAGE_LIST = 0;
;  581          0709  3          CALLG(.MESSAGE_VECT,LIB$SIGNAL);
;  582          0710  2          END;
;  583          0711  2
;  584          0712  2      ! Set registers context and return
;  585          0713  2      !
;  586          0714  2      DBG$STA_SETREGISTERS();
;  587          0715  2
;  588          0716  2      ! Update all watch point event entries after the DEPOSIT.
;  589          0717  2      !
;  590          0718  2      DBG$UPDATE_WATCHPOINTS();
;  591          0719  2
;  592          0720  1      END;


                                    0FFC 00000          .ENTRY  DBG$DEPOSIT, Save R2,R3,R4,R5,R6,R7,R8,R9,- ; 0559
                                                                R10,R11                                      ;
                          5B 00000000G   00   9E 00002          MOVAB   DBG$SAVE_LOC, R11                    ;
                          5A 00000000G   00   9E 00009          MOVAB   DBG$EVAL_LANG_OPERATOR, R10          ;
                          59 00000000G   00   9E 00010          MOVAB   LIB$SIGNAL, R9                       ;
                          58 00000000'   EF   9E 00017          MOVAB   PAGE_LIST, R8                        ;
                          5E             08   C2 0001E          SUBL2   #8, SP                               ;
                          6D        016B   CF   DE 00021          MOVAL   16$, (FP)                            ; 0583
                          52          04   AC   D0 00026          MOVL    VERB_NODE, R2                        ; 0613
                          50          08   A2   D0 0002A          MOVL    8(R2), TARGET_NN                     ;
                          51          08   A0   D0 0002E          MOVL    8(TARGET_NN), SOURCE_NN              ; 0614
                          57               60   D0 00032          MOVL    (TARGET_NN), PRIM_DESC               ; 0615
                          56               61   D0 00035          MOVL    (SOURCE_NN), DATA_DESC               ; 0616
                                           68   D4 00038          CLRL    PAGE_LIST                            ; 0617
                       79 8F           02   A6   91 0003A          CMPB    2(DATA_DESC), #121                   ; 0625
                                           10   12 0003F          BNEQ    1$                                   ;
                                           7E   D4 00041          CLRL    -(SP)                                ; 0627
                                           56   DD 00043          PUSHL   DATA_DESC                            ; 0628
                             00000000G     00   9F 00045          PUSHAB  DBG$GL_IDENTITY_TOKEN                ; 0627
                                6A         03   FB 0004B          CALLS   #3, DBG$EVAL_LANG_OPERATOR           ;
                                56         50   D0 0004E          MOVL    R0, DATA_DESC                        ;
                                           5E   DD 00051 1$:      PUSHL   SP                                   ; 0629
                                7E         83   8F   9A 00053          MOVZBL  #131, -(SP)                          ;
                                           57   DD 00057          PUSHL   PRIM_DESC                            ;
                    00000000G   00             03   FB 00059          CALLS   #3, DBG$PRIM_TO_VAL                  ;
                                53         04   A2   D0 00060          MOVL    4(R2), TYPE_NODE                     ; 0631
                                           08   12 00064          BNEQ    2$                                   ;
                                           57   DD 00066          PUSHL   PRIM_DESC                            ; 0632
                                6B              01   FB 00068          CALLS   #1, DBG$SAVE_LOC                     ;
                                           00BE 31   0006B          BRW     12$                                  ;
                                52              6E   D0 0006E 2$:      MOVL    ADDR_DESC, R2                        ; 0636
                             06 A2              03   90 00071          MOVB    #3, 6(R2)                            ;
```

```
                04   A2        80   8F  88 00075          BISB2   #128, 4(R2)                             ; 0637
                     54             63  9A 0007A          MOVZBL  (TYPE_NODE), OVERRIDE_TYPE              ; 0638
                     55        04   A3  D0 0007D          MOVL    4(TYPE_NODE), OVERRIDE_SIZE             ; 0639
                     25             54  D1 00081          CMPL    OVERRIDE_TYPE, #37                      ; 0642
                                    29  19 00084          BLSS    4$                                      ;
                     27             54  D1 00086          CMPL    OVERRIDE_TYPE, #39                      ;
                                    24  14 00089          BGTR    4$                                      ;
                     0D        0C   A2  91 0008B          CMPB    12(R2), #13                             ; 0644
                                    09  12 0008F          BNEQ    3$                                      ;
                          00028D08  8F  DD 00091          PUSHL   #167176                                 ;
                     69             01  FB 00097          CALLS   #1, LIB$SIGNAL                          ;
                17   A2             0B  90 0009A 3$:      MOVB    #11, 23(R2)                             ; 0645
                16   A2             54  90 0009E          MOVB    OVERRIDE_TYPE, 22(R2)                   ; 0646
                                    56  DD 000A2          PUSHL   DATA_DESC                               ; 0647
              FF1B   CF             01  FB 000A4          CALLS   #1, TEXT_LENGTH                         ;
                14   A2             50  B0 000A9          MOVW    R0, 20(R2)                              ;
                                    75  11 000AD          BRB     11$                                     ; 0640
                     38             54  D1 000AF 4$:      CMPL    OVERRIDE_TYPE, #56                      ; 0650
                                    15  12 000B2          BNEQ    5$                                      ;
                                    52  DD 000B4          PUSHL   R2                                      ; 0651
              0000V  CF             01  FB 000B6          CALLS   #1, CHECK_TEXT_DESCRIPTOR               ;
                     66             50  E8 000BB          BLBS    R0, 11$                                 ;
                          00028F50  8F  DD 000BE          PUSHL   #167760                                 ;
                     69             01  FB 000C4          CALLS   #1, LIB$SIGNAL                          ;
                                    58  11 000C7          BRB     11$                                     ;
                     53        14   A2  9E 000C9 5$:      MOVAB   20(R2), R3                              ; 0655
                     0D        03   A3  91 000CD          CMPB    3(R3), #13                              ;
                                    1A  12 000D1          BNEQ    7$                                      ;
                     20             55  D1 000D3          CMPL    OVERRIDE_SIZE, #32                      ; 0658
                                    0A  14 000D6          BGTR    6$                                      ;
                     16             54  D1 000D8          CMPL    OVERRIDE_TYPE, #22                      ; 0659
                                    05  13 000DB          BEQL    6$                                      ;
                     0E             54  D1 000DD          CMPL    OVERRIDE_TYPE, #14                      ; 0660
                                    0E  12 000E0          BNEQ    8$                                      ;
                          00028D08  8F  DD 000E2 6$:      PUSHL   #167176                                 ; 0662
                     69             01  FB 000E8          CALLS   #1, LIB$SIGNAL                          ;
                                    03  11 000EB          BRB     8$                                      ; 0655
                           03   A3  94 000ED 7$:      CLRB    3(R3)                                   ; 0665
                     0E             54  D1 000F0 8$:      CMPL    OVERRIDE_TYPE, #14                      ; 0667
                                    0E  12 000F3          BNEQ    9$                                      ;
                                    55  D5 000F5          TSTL    OVERRIDE_SIZE                           ; 0669
                                    0A  12 000F7          BNEQ    9$                                      ;
                                    56  DD 000F9          PUSHL   DATA_DESC                               ; 0670
                   FEC4   CF             01  FB 000FB          CALLS   #1, TEXT_LENGTH                         ;
                          55             50  D0 00100          MOVL    R0, OVERRIDE_SIZE               ;
                     02   A3             54  90 00103 9$:      MOVB    OVERRIDE_TYPE, 2(R3)                    ; 0672
                     16             54  D1 00107          CMPL    OVERRIDE_TYPE, #22                      ; 0673
                                    15  12 0010A          BNEQ    10$                                     ;
                     54        18   A2  D0 0010C          MOVL    24(R2), ADDR                            ; 0679
                                    7E  7C 00110          CLRQ    -(SP)                                   ; 0681
                                    54  DD 00112          PUSHL   ADDR                                    ;
                00000000G   00             03  FB 00114          CALLS   #3, DBG$INS_DECODE                      ;
                63          50             54  A3 0011B          SUBW3   ADDR, R0, (R3)                  ;
                                    03  11 0011F          BRB     11$                                     ; 0673
                     63             55  B0 00121 10$:      MOVW    OVERRIDE_SIZE, (R3)                     ; 0685
                           14   A2  9F 00124 11$:      PUSHAB  20(R2)                                  ; 0690
                                    57  DD 00127          PUSHL   PRIM_DESC                               ;
```

```
                    6B              02  FB 00129           CALLS   #2, DBG$SAVE_LOC                        ;
                                    56  DD 0012C 12$:      PUSHL   DATA_DESC                               ; 0693
00000000G           00              01  FB 0012E           CALLS   #1, DBG$SAVE_VAL
                          04        AE  9F 00135           PUSHAB  MESSAGE_VECT                            ; 0695
                        0180        8F  BB 00138           PUSHR   #^M<R7,R8>
00000000G           00              03  FB 0013C           CALLS   #3, DBG$NGET_PAGES                      ;
                    11              50  E9 00143           BLBC    R0, 13$                                 ;
                          04        AE  9F 00146           PUSHAB  MESSAGE_VECT                            ; 0696
                                    7E  D4 00149           CLRL    -(SP)                                   ;
                                    58  DD 0014B           PUSHL   R8                                      ;
00000000G           00              03  FB 0014D           CALLS   #3, DBG$SET_PAGE_PROT                   ;
                    06              50  E8 00154           BLBS    R0, 14$                                 ;
                                    68  D4 00157 13$:      CLRL    PAGE_LIST                               ; 0699
                    69        04    BE  FA 00159           CALLG   @MESSAGE_VECT, LIB$SIGNAL               ; 0700
                                    6E  DD 0015D 14$:      PUSHL   ADDR_DESC                               ; 0703
                                    56  DD 0015F           PUSHL   DATA_DESC                               ;
                  00000000G         00  9F 00161           PUSHAB  DBG$GL_DEPOSIT_TOKEN                    ;
                    6A              03  FB 00167           CALLS   #3, DBG$EVAL_LANG_OPERATOR              ;
                          04        AE  9F 0016A           PUSHAB  MESSAGE_VECT                            ; 0705
                                    01  DD 0016D           PUSHL   #1                                      ;
                                    58  DD 0016F           PUSHL   R8                                      ;
00000000G           00              03  FB 00171           CALLS   #3, DBG$SET_PAGE_PROT                   ;
                    06              50  E8 00178           BLBS    R0, 15$                                 ;
                                    68  D4 0017B           CLRL    PAGE_LIST                               ; 0708
                    69        04    BE  FA 0017D           CALLG   @MESSAGE_VECT, LIB$SIGNAL               ; 0709
00000000G           00              00  FB 00181 15$:      CALLS   #0, DBG$STA_SETREGISTERS                ; 0714
00000000G           00              00  FB 00188           CALLS   #0, DBG$UPDATE_WATCHPOINTS              ; 0718
                                      04 0018F             RET                                             ; 0720
                                  0000 00190 16$:          .WORD   Save nothing                            ; 0583
                                    7E  D4 00192           CLRL    -(SP)                                   ;
                                    5E  DD 00194           PUSHL   SP                                      ;
                          7E        04  AC 7D 00196        MOVQ    4(AP), -(SP)                            ;
                      FDF4  CF          03  FB 0019A       CALLS   #3, DEPOSIT_HANDLER                     ;
                                        04 0019F           RET                                             ;

; Routine Size:  416 bytes,    Routine Base:  DBG$CODE + 0284
```

```
;  594          0721  1 GLOBAL ROUTINE DBG$EVALUATE(VERB_NODE): NOVALUE =
;  595          0722  1 !
;  596          0723  1 ! FUNCTION
;  597          0724  1 !       This routine is the command execution network for the EVALUATE command.
;  598          0725  1 !       Various semantic actions are performed which correspond to the arguments
;  599          0726  1 !       and operands of the parsed input string.
;  600          0727  1 !
;  601          0728  1 !       EVALUATE sets last val '\', EVALUTATE/ADDRESS sets '.', current loc.
;  602          0729  1 !
;  603          0730  1 ! INPUTS
;  604          0731  1 !       VERB_NODE               - A longword containing the address of the head
;  605          0732  1 !                                 node in the command execution tree
;  606          0733  1 !
;  607          0734  1 ! OUTPUTS
;  608          0735  1 !       NONE
;  609          0736  1 !
;  610          0737  1
;  611          0738  2     BEGIN
;  612          0739  2
;  613          0740  2     MAP
;  614          0741  2         VERB_NODE: REF DBG$VERB_NODE;    ! Pointer to the input Verb Node
;  615          0742  2
;  616          0743  2     LOCAL
;  617          0744  2         RADIX,
;  618          0745  2         NOUN_NODE: REF DBG$NOUN_NODE,
;  619          0746  2         BASE_NODE: REF DBG$ADVERB_NODE,
;  620          0747  2         PRM_DESC: REF DBG$PRIMARY,
;  621          0748  2         VAL_DESC: REF DBG$VALDESC;
;  622          0749  2
;  623          0750  2
;  624          0751  2
;  625          0752  2
;  626          0753  2     ! Flush the current print buffer.  Then pick up the first Noun Node pointer,
;  627          0754  2     ! the Adverb Node pointer, and the radix setting for this command.
;  628          0755  2     !
;  629          0756  2     DBG$FLUSHBUF();
;  630          0757  2     NOUN_NODE = .VERB_NODE [DBG$L_VERB_OBJECT_PTR];
;  631          0758  2     BASE_NODE = .VERB_NODE [DBG$L_VERB_ADVERB_PTR];
;  632          0759  2     IF .BASE_NODE EQLA 0
;  633          0760  2     THEN
;  634          0761  2         RADIX = DBG$K_DEFAULT
;  635          0762  2     ELSE
;  636          0763  2         RADIX = .BASE_NODE[DBG$B_ADVERB_LITERAL];
;  637          0764  2
;  638          0765  2
;  639          0766  2     ! Loop through all the Noun Nodes to process each expression on the
;  640          0767  2     ! EVALUATE command.
;  641          0768  2     !
;  642          0769  2     WHILE .NOUN_NODE NEQ 0 DO
;  643          0770  3         BEGIN
;  644          0771  3         PRM_DESC = .NOUN_NODE[DBG$L_NOUN_VALUE];
;  645          0772  3         DBG$COLLECT(.PRM_DESC);
;  646          0773  3
;  647          0774  3
;  648          0775  3         ! Case on the kind of EVALUATE command Verb Node we have as determined
;  649          0776  3         ! by the command qualifiers.
;  650          0777  3         !
```

```
;  651     0778  3          CASE .VERB_NODE[DBG$B_VERB_COMPOSITE] FROM EVALUATE TO EVALUATE_COND OF
;  652     0779  3              SET
;  653     0780  3
;  654     0781  3
;  655     0782  3              ! Handle the plain EVALUATE and the EVALUATE/CONDITION_VALUE com-
;  656     0783  3              ! mands.
;  657     0784  3              !
;  658     0785  3              [EVALUATE,
;  659     0786  3               EVALUATE_COND]:
;  660     0787  4                  BEGIN
;  661     0788  4                  IF .PRM_DESC[DBG$V_DHDR_AGGR] THEN SIGNAL(DBG$_NOVALUE);
;  662     0789  4                  IF .VERB_NODE[DBG$B_VERB_COMPOSITE] EQL EVALUATE_COND
;  663     0790  4                  THEN
;  664     0791  4                      PRM_DESC[DBG$V_DHDR_FORMAT] = 1
;  665     0792  4
;  666     0793  4                  ELSE IF .RADIX NEQ DBG$K_DEFAULT
;  667     0794  4                  THEN
;  668     0795  4                      PRM_DESC[DBG$V_DHDR_FORMAT] = 0;
;  669     0796  4
;  670     0797  4                  DBG$PRINT_VALUE(.PRM_DESC,.RADIX, .DBG$GL_SIGN_FLAG);
;  671     0798  3                  END;
;  672     0799  3
;  673     0800  3
;  674     0801  3              ! Handle the EVALUATE/ADDRESS command.
;  675     0802  3              !
;  676     0803  3              [EVALUATE_ADDR]:
;  677     0804  4                  BEGIN
;  678     0805  4                  LOCAL
;  679     0806  4                      NAMEPTR,
;  680     0807  4                      REGDESCR,
;  681     0808  4                      VMS_DESC: DBG$STG_DESC;
;  682     0809  4
;  683     0810  4                  DBG$SAVE_LOC(.PRM_DESC);
;  684     0811  4                  DBG$PRIM_TO_VAL(.PRM_DESC,DBG$K_V_VALUE_DESC,VAL_DESC);
;  685     0812  4
;  686     0813  4
;  687     0814  4                  ! Check whether the address is in the register save area.
;  688     0815  4                  !
;  689     0816  4                  REGDESCR = DBG$STA_ADDRESS_TO_REGDESCR(.VAL_DESC[DBG$L_VALUE_POINTER]);
;  690     0817  4                  IF .REGDESCR NEQ 0 THEN
;  691     0818  5                    BEGIN
;  692     0819  5                    NAMEPTR = DBG$STA_REGISTER_NAME(.REGDESCR);
;  693     0820  5                    DBG$PRINT(UPLIT BYTE(%ASCIC '!AC'), .NAMEPTR);
;  694     0821  5                    END
;  695     0822  5
;  696     0823  4                  ELSE
;  697     0824  5                    BEGIN
;  698     0825  5                    VMS_DESC[DSC$B_CLASS] = DSC$K_CLASS_Z;
;  699     0826  5                    VMS_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_LU;
;  700     0827  5                    VMS_DESC[DSC$W_LENGTH] = 4;
;  701     0828  5                    VMS_DESC[DSC$A_POINTER] = VAL_DESC[DBG$L_VALUE_POINTER];
;  702     0829  5                    DBG$PRINT_VALUE_AS_INTEGER(VMS_DESC,.RADIX);
;  703     0830  4                    END;
;  704     0831  4
;  705     0832  4
;  706     0833  4                  ! If the address is a bit_field then also print the <p,s,e>.
;  707     0834  4                  !
```

```
;  708          0835  4              DBG$PRINT_FIELD_REF(.VAL_DESC,TRUE);
;  709          0836  4              END;
;  710          0837  4
;  711          0838  4
;  712          0839  4          ! Any other kind of Verb Node should never occur.  If it does, we
;  713          0840  4          ! signal an internal DEBUG coding error.
;  714          0841  4          !
;  715          0842  4          [INRANGE,OUTRANGE]:
;  716          0843  4              $DBG_ERROR('DBGLEVEL3\EVALUATE');
;  717          0844  4
;  718          0845  4          TES;
;  719          0846  4
;  720          0847  4
;  721          0848  4      ! Close out the current print line, link to the next Noun Node on the
;  722          0849  4      ! Noun Node list, and loop.
;  723          0850  4      !
;  724          0851  4      DBG$NEWLINE();
;  725          0852  4      NOUN_NODE = .NOUN_NODE[DBG$L_NOUN_LINK];
;  726          0853  4
;  727          0854  4      END;                            ! End of WHILE loop over expressions
;  728          0855  4
;  729          0856  4
;  730          0857  4  ! The EVALUATE command is processed.  Now return.
;  731          0858  4  !
;  732          0859  4  RETURN;
;  733          0860  4
;  734          0861  1  END;


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

                                                43  41  21  03  00000 P.AAA:  .ASCII  <3>\!AC\
4C  41  56  45  5C  33  4C  45  56  45  4C      47  42  44  12  00004 P.AAB:  .ASCII  <18>\DBGLEVEL3\<92>\EVALUATE\
                                                45  54  41  55  00013

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

                                                    0GFC 00000          .ENTRY  DBG$EVALUATE, Save R2,R3,R4,R5,R6,R7    ; 0721
                                        57 00000000G    00  9E 00002          MOVAB   LIB$SIGNAL, R7                  ;
                                        5E              10  C2 00009          SUBL2   #16, SP                         ;
                          00000000G     00              00  FB 0000C          CALLS   #0, DBG$FLUSHBUF                ; 0756
                                        54      04      AC  D0 00013          MOVL    VERB_NODE, R4                   ; 0757
                                        56      08      A4  D0 00017          MOVL    8(R4), NOUN_NODE                ;
                                        50      04      A4  D0 0001B          MOVL    4(R4), BASE_NODE                ; 0758
                                                        05  12 0001F          BNEQ    1$                              ; 0759
                                        55              01  D0 00021          MOVL    #1, RADIX                       ; 0761
                                                        03  11 00024          BRB     2$                              ;
                                        55              60  9A 00026 1$:      MOVZBL  (BASE_NODE), RADIX              ; 0763
                                                        56  D5 00029 2$:      TSTL    NOUN_NODE                       ; 0769
                                                        01  12 0002B          BNEQ    3$                              ;
                                                            04 0002D          RET                                     ;
                                        52              66  D0 0002E 3$:      MOVL    (NOUN_NODE), PRM_DESC           ; 0771
                                                        52  DD 00031          PUSHL   PRM_DESC                        ; 0772
                              FDOF      CF              01  FB 00033          CALLS   #1, DBG$COLLECT                 ;
```

```
                    02          01      01   A4  8F 00038          CASEB   1(R4), #1, #2                                   ; 0778
                  0019        004F          0019    0003D 4$:       .WORD   5$-4$,-
                                                                            10$-4$,-
                                                                            5$-4$
                                      00000000'  EF  9F 00043          PUSHAB  P.AAB                                   ; 0843
                                                 01  DD 00049          PUSHL   #1
                                      00028362   8F  DD 0004B          PUSHL   #164706
                                          67         03  FB 00051          CALLS   #3, LIB$SIGNAL
                                                     34  11 00054          BRB     9$
                                          09      04     A2  E9 00056 5$:   BLBC    4(PRM_DESC), 6$                 ; 0788
                                      000287F8   8F  DD 0005A          PUSHL   #165880
                                          67         01  FB 00060          CALLS   #1, LIB$SIGNAL
                                          03      01     A4  91 00063 6$:   CMPB    1(R4), #3                       ; 0789
                                                     08  12 00067          BNEQ    7$
                05  A2                  04          04         01  F0 00069          INSV    #1, #4, #4, 5(PRM_DESC)     ; 0791
                                                     0A  11 0006F          BRB     8$
                                          01         55  D1 00071 7$:   CMPL    RADIX, #1                       ; 0793
                                                     05  13 00074          BEQL    8$
                                                 05  A2     F0  8F  8A 00076          BICB2   #240, 5(PRM_DESC)       ; 0795
                                      00000000G  00  DD 0007B 8$:   PUSHL   DBG$GL_SIGN_FLAG                ; 0797
                                                 24  BB 00081          PUSHR   #^M<R2,R5>
                                  00000000G      00         03  FB 00083          CALLS   #3, DBG$PRINT_VALUE
                                                     67  11 0008A 9$:   BRB     13$                             ; 0778
                                                     52  DD 0008C 10$:  PUSHL   PRM_DESC                        ; 0810
                                  00000000G      00         01  FB 0008E          CALLS   #1, DBG$SAVE_LOC
                                                     5E  DD 00095          PUSHL   SP                              ; 0811
                                          7E         83  8F  9A 00097          MOVZBL  #131, -(SP)
                                                     52  DD 0009B          PUSHL   PRM_DESC
                                  00000000G      00         03  FB 0009D          CALLS   #3, DBG$PRIM_TO_VAL
                                          53         6E  D0 000A4          MOVL    VAL_DESC, R3                    ; 0816
                                                 18  A3  DD 000A7          PUSHL   24(R3)
                                  00000000G      00         01  FB 000AA          CALLS   #1, DBG$STA_ADDRESS_TO_REGDESCR
                                                     50  D5 000B1          TSTL    REGDESCR                        ; 0817
                                                     1A  13 000B3          BEQL    11$
                                                     50  DD 000B5          PUSHL   REGDESCR                        ; 0819
                                  00000000G      00         01  FB 000B7          CALLS   #1, DBG$STA_REGISTER_NAME
                                                     50  DD 000BE          PUSHL   NAMEPTR                         ; 0820
                                      00000000'  EF  9F 000C0          PUSHAB  P.AAA
                                  00000000G      00         02  FB 000C6          CALLS   #2, DBG$PRINT
                                                     19  11 000CD          BRB     12$                             ; 0817
                                      04  AE 00040004   8F  D0 000CF 11$:   MOVL    #262148, VMS_DESC               ; 0827
                                      08  AE         18  A3  9E 000D7          MOVAB   24(R3), VMS_DESC+4              ; 0828
                                                     55  DD 000DC          PUSHL   RADIX                           ; 0829
                                                 08  AE  9F 000DE          PUSHAB  VMS_DESC
                                  00000000G      00         02  FB 000E1          CALLS   #2, DBG$PRINT_VALUE_AS_INTEGER
                                                     01  DD 000E8 12$:   PUSHL   #1                             ; 0835
                                                     53  DD 000EA          PUSHL   R3
                                  00000000G      00         02  FB 000EC          CALLS   #2, DBG$PRINT_FIELD_REF
                                  00000000G      00         00  FB 000F3 13$:   CALLS   #0, DBG$NEWLINE                ; 0851
                                          56         08  A6  D0 000FA          MOVL    8(NOUN_NODE), NOUN_NODE         ; 0852
                                                 FF28    31  000FE          BRW     2$                              ; 0769
                                                     04  00101          RET                                     ; 0861

; Routine Size:  258 bytes,    Routine Base:  DBG$CODE + 0424
```

```
;  736          0862  1 GLOBAL ROUTINE DBG$EXAMINE(VERB_NODE: REF DBG$VERB_NODE): NOVALUE =
;  737          0863  1 !
;  738          0864  1 ! FUNCTION
;  739          0865  1 !       This routine performs the action associated with EXAMINE xxx.
;  740          0866  1 !       We always get three adverb nodes linked to the verb node. See the
;  741          0867  1 !       routine header for DBG$NPARSE_EXAMINE in DBGNEXMNE.B32 for details.
;  742          0868  1 !
;  743          0869  1 ! INPUTS
;  744          0870  1 !       VERB_NODE - A longword containing the address of the command
;  745          0871  1 !                   execution tree verb (head) node.
;  746          0872  1 !
;  747          0873  1 ! OUTPUTS
;  748          0874  1 !       NONE
;  749          0875  1 !
;  750          0876  1
;  751          0877  2     BEGIN
;  752          0878  2
;  753          0879  2     LOCAL
;  754          0880  2         NOUN_NODE       : REF DBG$NOUN_NODE,
;  755          0881  2         TYPE_NODE       : REF DBG$ADVERB_NODE,
;  756          0882  2         BASE_NODE       : REF DBG$ADVERB_NODE,
;  757          0883  2         MODE_NODE       : REF DBG$ADVERB_NODE,
;  758          0884  2         PRM_DESC        : REF DBG$PRIMARY,
;  759          0885  2         END_DESC        : REF DBG$PRIMARY,
;  760          0886  2         VAL_DESC        : REF DBG$VALDESC,
;  761          0887  2         NEW_SIZE        : WORD,
;  762          0888  2         NEW_TYPE        : BYTE,
;  763          0889  2         RADIX           : BYTE,
;  764          0890  2         FORMAT_ONE      : BYTE,
;  765          0891  2         FORMAT_TWO      : BYTE;
;  766          0892  2
;  767          0893  2     NOUN_NODE = .VERB_NODE[DBG$L_VERB_OBJECT_PTR];
;  768          0894  2     TYPE_NODE = .VERB_NODE[DBG$L_VERB_ADVERB_PTR];
;  769          0895  2     BASE_NODE = .TYPE_NODE[DBG$L_ADVERB_LINK];
;  770          0896  2     MODE_NODE = .BASE_NODE[DBG$L_ADVERB_LINK];
;  771          0897  2
;  772          0898  2     SELECTONE .VERB_NODE[DBG$B_VERB_COMPOSITE] OF
;  773          0899  2         SET
;  774          0900  2         [EXAMINE]:
;  775          0901  3             BEGIN
;  776          0902  3             NEW_TYPE    = .TYPE_NODE[DBG$B_ADVERB_LITERAL];
;  777          0903  3             NEW_SIZE    = .TYPE_NODE[DBG$L_ADVERB_VALUE];
;  778          0904  3             RADIX       = .BASE_NODE[DBG$B_ADVERB_LITERAL];
;  779          0905  3             FORMAT_ONE = 0;
;  780          0906  2             END;
;  781          0907  2
;  782          0908  2         [EXAMINE_SOURCE]:       0;
;  783          0909  2
;  784          0910  2         [EXAMINE_CONDITION_VALUE]:
;  785          0911  3             BEGIN
;  786          0912  3             NEW_TYPE    = DSC$K_DTYPE_LU;
;  787          0913  3             NEW_SIZE    = 4;
;  788          0914  3             RADIX       = DBG$K_DEFAULT;
;  789          0915  3             FORMAT_ONE = 1;
;  790          0916  2             END;
;  791          0917  2
;  792          0918  2         [EXAMINE_PSL]:
```

```
;  793          0919  3              BEGIN
;  794          0920  3              NEW_TYPE    = DSC$K_DTYPE_LU;
;  795          0921  3              NEW_SIZE    = 4;
;  796          0922  3              RADIX       = DBG$K_DEFAULT;
;  797          0923  3              FORMAT_ONE = 2;
;  798          0924  3              END;
;  799          0925  2
;  800          0926  2          [EXAMINE_PSW]:
;  801          0927  2              BEGIN
;  802          0928  2              NEW_TYPE    = DSC$K_DTYPE_WU;
;  803          0929  2              NEW_SIZE    = 2;
;  804          0930  2              RADIX       = DBG$K_DEFAULT;
;  805          0931  2              FORMAT_ONE = 3;
;  806          0932  2              END;
;  807          0933  2
;  808          0934  2
;  809          0935  2          ! Any other kind of the Verb Node is invalid, so we signal an internal
;  810          0936  2          ! DEBUG coding error.
;  811          0937  2          !
;  812          0938  2          [OTHERWISE]:
;  813          0939  2              $DBG_ERROR('DBGLEVEL3\EXAMINE');
;  814          0940  2
;  815          0941  2          TES;
;  816          0942  2
;  817          0943  2
;  818          0944  2      ! ------
;  819          0945  2      !
;  820          0946  3      DO  BEGIN
;  821          0947  3          DBG$FLUSHBUF();
;  822          0948  3
;  823          0949  3          IF .VERB_NODE[DBG$B_VERB_COMPOSITE] EQL EXAMINE_SOURCE
;  824          0950  3          THEN
;  825          0951  4              BEGIN
;  826          0952  4              LOCAL
;  827          0953  4                  VAL_DESC        : REF DBG$VALDESC,
;  828          0954  4                  START_ADDRESS,
;  829          0955  4                  FINAL_ADDRESS;
;  830          0956  4
;  831          0957  4              DBG$PRIM_TO_VAL(.NOUN_NODE[DBG$L_NOUN_VALUE ],DBG$K_V_VALUE_DESC,VAL_DESC);
;  832          0958  4              START_ADDRESS = .VAL_DESC[DBG$L_VALUE_POINTER];
;  833          0959  4              DBG$PRIM_TO_VAL(.NOUN_NODE[DBG$L_NOUN_VALUE2],DBG$K_V_VALUE_DESC,VAL_DESC);
;  834          0960  4              FINAL_ADDRESS = .VAL_DESC[DBG$L_VALUE_POINTER];
;  835          0961  4
;  836          0962  4              ! Output the source. The third parameter indicates that the
;  837          0963  4              ! module name is to be displayed.
;  838          0964  4
;  839          0965  4              DBG$SRC_TYPE_PC_SOURCE(.START_ADDRESS,.FINAL_ADDRESS,TRUE,FALSE);
;  840          0966  4
;  841          0967  4              PRM_DESC = .NOUN_NODE[DBG$L_NOUN_VALUE2];
;  842          0968  4  !           Commented out because screen windown does EXAMINE/SOURCE and
;  843          0969  4  !           we don't want to save dot there.
;  844          0970  4  !
;  845          0971  4  !           DBG$SAVE_LOC(.PRM_DESC);
;  846          0972  4              END             ! EXAMINE/SOURCE
;  847          0973  4
;  848          0974  4
;  849          0975  4          ! ------
```

```
;  850          0976  4          !
;  851          0977  3          ELSE
;  852          0978  4              BEGIN       ! Data Examine
;  853          0979  4              PRM_DESC = .NOUN_NODE[DBG$L_NOUN_VALUE];
;  854          0980  4              END_DESC = .NOUN_NODE[DBG$L_NOUN_VALUE2];
;  855          0981  4              DBG$COLLECT(.PRM_DESC);
;  856          0982  4              DBG$COLLECT(.END_DESC);
;  857          0983  4
;  858          0984  5              IF (.END_DESC NEQ 0) AND (.PRM_DESC NEQ .END_DESC)
;  859          0985  4              THEN
;  860          0986  5                  BEGIN
;  861          0987  5                  !+
;  862          0988  5                  ! We have a ranged examine (EXAMINE <prm>:<end>)
;  863          0989  5                  ! Check for the case where the two endpoints are part
;  864          0990  5                  ! of th same structure. We have to ensure that a number
;  865          0991  5                  ! of conditions are met, e.g., they are both primaries,
;  866          0992  5                  ! they are not aggregates, and so on.
;  867          0993  5                  !-
;  868          0994  6                  IF      (.PRM_DESC[DBG$B_DHDR_TYPE] EQL DBG$K_PRIMARY_DESC)
;  869          0995  5                                  AND
;  870          0996  6                          (.END_DESC[DBG$B_DHDR_TYPE] EQL DBG$K_PRIMARY_DESC)
;  871          0997  5                                  AND
;  872          0998  6                          (.PRM_DESC[DBG$L_DHDR_SYMIDO] EQL .END_DESC[DBG$L_DHDR_SYMIDO])
;  873          0999  5                                  AND
;  874          1000  6                          (.NEW_TYPE EQL DBG$K_NOTYPE)
;  875          1001  5                                  AND
;  876          1002  6                          (NOT .PRM_DESC[DBG$V_DHDR_AGGR])
;  877          1003  5                                  AND
;  878          1004  6                          (NOT .END_DESC[DBG$V_DHDR_AGGR])
;  879          1005  5                                  AND
;  880          1006  6                          (NOT .PRM_DESC[DBG$V_DHDR_SUBREF])
;  881          1007  5                                  AND
;  882          1008  6                          (NOT .END_DESC[DBG$V_DHDR_SUBREF])
;  883          1009  5                  THEN
;  884          1010  6                      BEGIN
;  885          1011  6                      !+
;  886          1012  6                      ! The start and end of the ranged examine appear to be
;  887          1013  6                      ! part of the same aggregate structure. Check that the
;  888          1014  6                      ! start is earlier than the end
;  889          1015  6                      !-
;  890          1016  6                      IF PRIMARY_ORDER(.PRM_DESC,.END_DESC) GTR 0 THEN SIGNAL(DBG$_EXARANGE);
;  891          1017  6                      WHILE TRUE DO
;  892          1018  7                          BEGIN
;  893          1019  7                          LOCAL MARK;
;  894          1020  7                          MARK = DBG$PUSH_TEMPMEM();
;  895          1021  7                          DBG$PRINT_IDENTIFIER(.PRM_DESC);
;  896          1022  7                          DBG$PRINT(UPLIT BYTE(%ASCIC '!AD! '), 1, UPLIT BYTE(':'));
;  897          1023  7                          DBG$PRIM_TO_VAL(.PRM_DESC,DBG$K_VALUE_DESC,VAL_DESC);
;  898          1024  7                          IF .FORMAT_ONE NEQ 0 THEN VAL_DESC[DBG$V_DHDR_FORMAT] = .FORMAT_ONE;
;  899          1025  7                          DBG$PRINT_VALUE(.VAL_DESC,.RADIX, .DBG$GL_SIGN_FLAG);
;  900          1026  7                          DBG$NEWLINE();
;  901          1027  7                          DBG$SAVE_LOC(.PRM_DESC);
;  902          1028  7                          DBG$POP_TEMPMEM(.MARK);
;  903          1029  7                          IF PRIMARY_ORDER(.PRM_DESC,.END_DESC) GEQ 0 THEN EXITLOOP;
;  904          1030  7                          IF NOT MODIFY_PRIMARY(.PRM_DESC,0) THEN EXITLOOP;
;  905          1031  6                          END;
;  906          1032  6                      END
```

```
;  907     1033  5                ELSE
;  908     1034  6                    BEGIN
;  909     1035  6                    !+
;  910     1036  6                    ! The start and end are NOT part of the same aggregate.
;  911     1037  6                    !-
;  912     1038  6                    LOCAL
;  913     1039  6                        MARK,
;  914     1040  6                        LAST_ADDR,
;  915     1041  6                        NEXT_ADDR,
;  916     1042  6                        DESC_TYPE,
;  917     1043  6                        ADDR_DESC       : REF DBG$VALDESC,
;  918     1044  6                        RDESC_ONE       : DBG$REGDESCR,
;  919     1045  6                        RDESC_TWO       : DBG$REGDESCR,
;  920     1046  6                        LENGTH;
;  921     1047  6
;  922     1048  6                    ADDR_DESC = DBG$CHANGE_DTYPE(.END_DESC,.NEW_TYPE,.NEW_SIZE);
;  923     1049  6                    RDESC_ONE = DBG$STA_ADDRESS_TO_REGDESCR(.ADDR_DESC[DBG$L_VALUE_POINTER]);
;  924     1050  6                    LAST_ADDR = .ADDR_DESC[DBG$L_VALUE_POINTER];
;  925     1051  6                    ADDR_DESC = DBG$CHANGE_DTYPE(.PPM_DESC,.NEW_TYPE,.NEW_SIZE);
;  926     1052  6                    RDESC_TWO = DBG$STA_ADDRESS_TO_REGDESCR(.ADDR_DESC[DBG$L_VALUE_POINTER]);
;  927     1053  6                    IF ((.RDESC_ONE XOR .RDESC_TWO) AND %X'FFFF00FC') NEQ 0
;  928     1054  6                    THEN
;  929     1055  6                        SIGNAL(DBG$_EXARANGE);
;  930     1056  6
;  931     1057  6                    IF .LAST_ADDR LSSA .ADDR_DESC[DBG$L_VALUE_POINTER]
;  932     1058  6                    THEN
;  933     1059  6                        SIGNAL(DBG$_EXARANGE);
;  934     1060  5
;  935     1061  7                    IF (.ADDR_DESC[DBG$B_VALUE_CLASS] EQL DSC$K_CLASS_UBS)
;  936     1062  6                    THEN
;  937     1063  6                        SIGNAL(DBG$_ILLTYPE);
;  938     1064  6
;  939     1065  6                    DESC_TYPE = DBG$K_VALUE_DESC;
;  940     1066  6                    IF (.ADDR_DESC[DBG$B_VALUE_DTYPE] EQL DSC$K_DTYPE_ZEM) OR
;  941     1067  7                       (.ADDR_DESC[DBG$B_VALUE_DTYPE] EQL DSC$K_DTYPE_ZI)
;  942     1068  6                    THEN
;  943     1069  6                        DESC_TYPE = DBG$K_V_VALUE_DESC;
;  944     1070  6
;  945     1071  6
;  946     1072  6                    ! ------
;  947     1073  6                    !
;  948     1074  6                    WHILE TRUE DO
;  949     1075  7                        BEGIN
;  950     1076  7                        MARK = DBG$PUSH_TEMPMEM();
;  951     1077  7                        DBG$PRINT_IDENTIFIER(.ADDR_DESC);
;  952     1078  7                        DBG$PRINT(UPLIT BYTE(%ASCIC '!AD! '),1,UPLIT BYTE(':'));
;  953     1079  7                        DBG$PRIM_TO_VAL(.ADDR_DESC,.DESC_TYPE,VAL_DESC);
;  954     1080  7                        IF .FORMAT_ONE NEQ 0 THEN VAL_DESC[DBG$V_DHDR_FORMAT] = .FORMAT_ONE;
;  955     1081  7                        DBG$PRINT_VALUE(.VAL_DESC,.RADIX, .DBG$GL_SIGN_FLAG);
;  956     1082  7                        DBG$NEWLINE();
;  957     1083  7                        DBG$POP_TEMPMEM(.MARK);
;  958     1084  7
;  959     1085  7                        ! Get the increment we will add to the address for
;  960     1086  7                        ! the next line of the ranged examine. If the increment
;  961     1087  7                        ! is zero then signal an informational and get out of the loop.
;  962     1088  7                        !
;  963     1089  7                        LENGTH = (DBG$DATA_LENGTH(ADDR_DESC[DBG$A_VALUE_VMSDESC]) + (%BPUNIT-1))/%BPUNIT;
```

```
;  964          1090  7                    IF .LENGTH EQL 0
;  965          1091  7                    THEN
;  966          1092  8                        BEGIN
;  967          1093  8                        SIGNAL(DBG$_ZEROINCR); ! Informational
;  968          1094  8                        EXITLOOP;
;  969          1095  7                        END;
;  970          1096  7
;  971          1097  7                    NEXT_ADDR = .ADDR_DESC[DBG$L_VALUE_POINTER] + .LENGTH;
;  972          1098  7                    IF .NEXT_ADDR GTRA .LAST_ADDR THEN EXITLOOP;
;  973          1099  7                    ADDR_DESC[DBG$L_VALUE_POINTER] = .NEXT_ADDR;
;  974          1100  9                    IF (.ADDR_DESC[DBG$B_VALUE_DTYPE] EQL DSC$K_DTYPE_ZEM)
;  975          1101  8                     OR (.ADDR_DESC[DBG$B_VALUE_DTYPE] EQL DSC$K_DTYPE_ZI))
;  976          1102  7                      THEN
;  977          1103  8                        BEGIN
;  978          1104  8                        IF DBG$IS_IT_ENTRY(.NEXT_ADDR)
;  979          1105  8                          THEN ADDR_DESC[DBG$B_VALUE_DTYPE]  = DSC$K_DTYPE_ZEM
;  980          1106  8                          ELSE ADDR_DESC[DBG$B_VALUE_DTYPE]  = DSC$K_DTYPE_ZI;
;  981          1107  8                        ADDR_DESC[DBG$W_VALUE_LENGTH] = DBG$INS_DECODE(.NEXT_ADDR,FALSE) - .NEXT_ADDR;
;  982          1108  8                        END
;  983          1109  7                      ELSE
;  984          1110  7                        ADDR_DESC[DBG$W_VALUE_LENGTH] = FIX_UP_LENGTH(ADDR_DESC[DBG$A_VALUE_VMSDESC]);
;  985          1111  6                    END;
;  986          1112  6                DBG$SAVE_LOC(.ADDR_DESC);
;  987          1113  5                END;
;  988          1114  5            END
;  989          1115  4        ELSE
;  990          1116  5            BEGIN
;  991          1117  5
;  992          1118  5
;  993          1119  5            ! In the case where prm_desc is a volatile value descriptor
;  994          1120  5            ! representing an absolute address, the print_identifier
;  995          1121  5            ! will attempt to symbolize this address to a primary. If
;  996          1122  5            ! it succeeds, it will return the newly-constructed primary..
;  997          1123  5            ! In all other cases, it just returns the descriptor we pass
;  998          1124  5            ! into it, unchanged.
;  999          1125  5            !
; 1000          1126  5            PRM_DESC = DBG$PRINT_IDENTIFIER(.PRM_DESC);
; 1001          1127  5            DBG$SAVE_LOC(.PRM_DESC);
; 1002          1128  5            IF .NEW_TYPE EQL DBG$K_NOTYPE AND .PRM_DESC[DBG$V_DHDR_AGGR]
; 1003          1129  5            THEN
; 1004          1130  5                DBG$PRINT_AGGREGATE(.PRM_DESC,.RADIX)
; 1005          1131  5            ELSE
; 1006          1132  6                BEGIN
; 1007          1133  6                DBG$PRINT(UPLIT BYTE(%ASCIC '!AD! '), 1, UPLIT BYTE(':'));
; 1008          1134  6                VAL_DESC = DBG$CHANGE_DTYPE(.PRM_DESC,.NEW_TYPE,.NEW_SIZE);
; 1009          1135  6                FORMAT_TWO = .FORMAT_ONE;
; 1010          1136  6                IF .NEW_TYPE NEQ DBG$K_NOTYPE
; 1011          1137  6                THEN
; 1012          1138  6                    DBG$SAVE_LOC(.PRM_DESC,VAL_DESC[DBG$A_VALUE_VMSDESC])
; 1013          1139  7                ELSE IF (.FORMAT_ONE EQL 0) AND (.RADIX EQL DBG$K_DEFAULT)
; 1014          1140  7                       AND (.VAL_DESC[DBG$B_VALUE_CLASS] NEQ DSC$K_CLASS_UBS)
; 1015          1141  7                        AND (.VAL_DESC[DBG$L_VALUE_POINTER] EQLA DBG$REG_VALUES[16])
; 1016          1142  6                          THEN FORMAT_TWO = 2;
; 1017          1143  6
; 1018          1144  6                IF .VAL_DESC[DBG$B_VALUE_DTYPE] NEQ DSC$K_DTYPE_ZI
; 1019          1145  6                THEN
; 1020          1146  7                    BEGIN
```

```
;  1021          1147  7                        DBG$PRIM_TO_VAL(.VAL_DESC,DBG$K_VALUE_DESC,VAL_DESC);
;  1022          1148  7                        DBG$DO_MAPPING(.VAL_DESC);
;  1023          1149  6                        END;
;  1024          1150  6
;  1025          1151  6                    IF .FORMAT_TWO NEQ 0 THEN VAL_DESC[DBG$V_DHDR_FORMAT] = .FORMAT_TWO;
;  1026          1152  6                    DBG$PRINT_VALUE(.VAL_DESC,.RADIX,.DBG$GL_SIGN_FLAG);
;  1027          1153  6                    DBG$NEWLINE();
;  1028          1154  5                    END;
;  1029          1155  4                END;
;  1030          1156  3            END;
;  1031          1157  2        END UNTIL (NOUN_NODE = .NOUN_NODE[DBG$L_NOUN_LINK]) EQL 0;
;  1032          1158  2
;  1033          1159  2    RETURN STS$K_SUCCESS;
;  1034          1160  1    END;


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

4D  41  58  45  5C  33  4C  45  56  45  4C  47  42  44  11  00017 P.AAC:  .ASCII  <17>\DBGLEVEL3\<92>\EXAMINE\                ;
                                        45  4E  49  00026                                                               ;
                                5F  21  44  41  21  05  00029 P.AAD:  .ASCII  <5>\!AD!_\                                 ;
                                                    3A  0002F P.AAE:  .ASCII  \:\                                       ;
                                5F  21  44  41  21  05  00030 P.AAF:  .ASCII  <5>\!AD!_\                                 ;
                                                    3A  00036 P.AAG:  .ASCII  \:\                                       ;
                                5F  21  44  41  21  05  00037 P.AAH:  .ASCII  <5>\!AD!_\                                 ;
                                                    3A  0003D P.AAI:  .ASCII  \:\                                       ;


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

                                    0FFC 00000          .ENTRY  DBG$EXAMINE, Save R2,R3,R4,R5,R6,R7,R8,R9,-             ; 0862
                                                                R10,R11                                                 ;
5E              20  C2  00002       SUBL2   #32, SP                                                                 ;
50      04      AC  D0  00005       MOVL    VERB_NODE, R0                                                           ; 0893
55      08      A0  D0  00009       MOVL    8(R0), NOUN_NODE                                                        ;
51      04      A0  D0  0000D       MOVL    4(R0), TYPE_NODE                                                        ; 0894
52      08      A1  D0  00011       MOVL    8(TYPE_NODE), BASE_NODE                                                 ; 0895
53      08      A2  D0  00015       MOVL    8(BASE_NODE), MODE_NODE                                                 ; 0896
5B      01      A0  9A  00019       MOVZBL  1(R0), R11                                                              ; 0898
01              5B  91  0001D       CMPB    R11, #1                                                                 ; 0900
                0E  12  00020       BNEQ    1$                                                                      ;
59              61  90  00022       MOVB    (TYPE_NODE), NEW_TYPE                                                   ; 0902
6E      04      A1  B0  00025       MOVW    4(TYPE_NODE), NEW_SIZE                                                  ; 0903
5A              62  90  00029       MOVB    (BASE_NODE), RADIX                                                      ; 0904
                58  94  0002C       CLRB    FORMAT_ONE                                                              ; 0905
                53  11  0002E       BRB     5$                                                                      ; 0898
04              5B  91  00030 1$:   CMPB    R11, #4                                                                 ; 0908
                4E  13  00033       BEQL    5$                                                                      ;
05              5B  91  00035       CMPB    R11, #5                                                                 ; 0910
                0E  12  00038       BNEQ    2$                                                                      ;
59              04  90  0003A       MOVB    #4, NEW_TYPE                                                            ; 0912
6E              04  B0  0003D       MOVW    #4, NEW_SIZE                                                            ; 0913
5A              01  90  00040       MOVB    #1, RADIX                                                               ; 0914
58              01  90  00043       MOVB    #1, FORMAT_ONE                                                          ; 0915
                3B  11  00046       BRB     5$                                                                      ; 0898
```

```
                    06              5B  91 00048 2$:      CMPB    R11, #6                         ; 0918
                                    0E  12 0004B           BNEQ    3$
                    59              04  90 0004D           MOVB    #4, NEW_TYPE                    ; 0920
                    6E              04  B0 00050           MOVW    #4, NEW_SIZE                    ; 0921
                    5A              01  90 00053           MOVB    #1, RADIX                       ; 0922
                    58              02  90 00056           MOVB    #2, FORMAT_ONE                  ; 0923
                                    28  11 00059           BRB     5$                              ; 0898
                    07              5B  91 0005B 3$:       CMPB    R11, #7                         ; 0926
                                    0E  12 0005E           BNEQ    4$
                    59              03  90 00060           MOVB    #3, NEW_TYPE                    ; 0928
                    6E              02  B0 00063           MOVW    #2, NEW_SIZE                    ; 0929
                    5A              01  90 00066           MOVB    #1, RADIX                       ; 0930
                    58              03  90 00069           MOVB    #3, FORMAT_ONE                  ; 0931
                                    15  11 0006C           BRB     5$                              ; 0898
                        00000000'   EF  9F 0006E 4$:       PUSHAB  P.AAC                           ; 0939
                                    01  DD 00074           PUSHL   #1
                        00028362    8F  DD 00076           PUSHL   #164706
              00000000G 00          03  FB 0007C           CALLS   #3, LIB$SIGNAL
              00000000G 00          00  FB 00083 5$:       CALLS   #0, DBG$FLUSHBUF                ; 0947
                    04              5B  91 0008A           CMPB    R11, #4                         ; 0949
                                    46  12 0008D           BNEQ    6$
                            18      AE  9F 0008F           PUSHAB  VAL_DESC                        ; 0957
                    7E      83      8F  9A 00092           MOVZBL  #131, -(SP)
                                    65  DD 00096           PUSHL   (NOUN_NODE)
              00000000G 00          03  FB 00098           CALLS   #3, DBG$PRIM_TO_VAL
                    50      18      AE  D0 0009F           MOVL    VAL_DESC, R0                    ; 0958
                    52      18      A0  D0 000A3           MOVL    24(R0), START_ADDRESS
                            18      AE  9F 000A7           PUSHAB  VAL_DESC                        ; 0959
                    7E      83      8F  9A 000AA           MOVZBL  #131, -(SP)
                            0C      A5  DD 000AE           PUSHL   12(NOUN_NODE)
              00000000G 00          03  FB 000B1           CALLS   #3, DBG$PRIM_TO_VAL
                    50      18      AE  D0 000B8           MOVL    VAL_DESC, R0                    ; 0960
                    50      18      A0  D0 000BC           MOVL    24(R0), FINAL_ADDRESS
                    7E              01  7D 000C0           MOVQ    #1, -(SP)                       ; 0965
                                    50  DD 000C3           PUSHL   FINAL_ADDRESS
                                    52  DD 000C5           PUSHL   START_ADDRESS
              00000000G 00          04  FB 000C7           CALLS   #4, DBG$SRC_TYPE_PC_SOURCE
                    53      0C      A5  D0 000CE           MOVL    12(NOUN_NODE), PRM_DESC         ; 0967
                                0035D   31 000D2           BRW     36$                             ; 0949
                    53              65  D0 000D5 6$:       MOVL    (NOUN_NODE), PRM_DESC           ; 0979
                    56      0C      A5  D0 000D8           MOVL    12(NOUN_NODE), END_DESC         ; 0980
                                    53  DD 000DC           PUSHL   PRM_DESC                        ; 0981
                  FB62      CF          01  FB 000DE           CALLS   #1, DBG$COLLECT
                                    56  DD 000E3           PUSHL   END_DESC                        ; 0982
                  FB5B      CF          01  FB 000E5           CALLS   #1, DBG$COLLECT
                                    56  D5 000EA           TSTL    END_DESC                        ; 0984
                                    03  13 000EC           BEQL    7$
                    56              53  D1 000EE           CMPL    PRM_DESC, END_DESC
                                    03  12 000F1 7$:       BNEQ    8$
                                02058   31 000F3           BRW     29$
                    79      8F      02  A3 91 000F6 8$:    CMPB    2(PRM_DESC), #121               ; 0994
                                    18  12 000FB           BNEQ    9$
                    79      8F      02  A6 91 000FD        CMPB    2(END_DESC), #121               ; 0996
                                    11  12 00102           BNEQ    9$
                    0C      A6      0C  A3 D1 00104        CMPL    12(PRM_DESC), 12(END_DESC)      ; 0998
                                    0A  12 00109           BNEQ    9$
                    80      8F          59 91 0010B        CMPB    NEW_TYPE, #128                  ; 1000
```

```
                                  04  12 0010F         BNEQ    9$
                    03        04  A3  E9 00111         BLBC    4(PRM_DESC), 10$                         ; 1002
                                0089  31 00115 9$:     BRW     15$
                    F9        04  A6  E8 00118 10$:    BLBS    4(END_DESC), 9$                          ; 1004
          F4      04  A3          01  E0 0011C         BBS     #1, 4(PRM_DESC), 9$                      ; 1006
          EF      04  A6          01  E0 00121         BBS     #1, 4(END_DESC), 9$                      ; 1008
                            0048  8F  BB 00126         PUSHR   #^M<R3,R6>                               ; 1016
               0000V  CF          02  FB 0012A         CALLS   #2, PRIMARY_ORDER
                                  50  D5 0012F         TSTL    R0
                                  0D  15 00131         BLEQ    11$
                        00028190  8F  DD 00133         PUSHL   #164240
           00000000G  00          01  FB 00139         CALLS   #1, LIB$SIGNAL
           00000000G  00          00  FB 00140 11$:    CALLS   #0, DBG$PUSH_TEMPMEM                     ; 1020
                      52          50  D0 00147         MOVL    R0, MARK
                                  53  DD 0014A         PUSHL   PRM_DESC                                 ; 1021
           00000000G  00          01  FB 0014C         CALLS   #1, DBG$PRINT_IDENTIFIER
                       00000000'  EF  9F 00153         PUSHAB  P.AAE                                    ; 1022
                                  01  DD 00159         PUSHL   #1
                       00000000'  EF  9F 0015B         PUSHAB  P.AAD
           00000000G  00          03  FB 00161         CALLS   #3, DBG$PRINT
                              1C  AE  9F 00168         PUSHAB  VAL_DESC                                 ; 1023
                      7E      7A  8F  9A 0016B         MOVZBL  #122, -(SP)
                                  53  DD 0016F         PUSHL   PRM_DESC
           00000000G  00          03  FB 00171         CALLS   #3, DBG$PRIM_TO_VAL
                                  58  95 00178         TSTB    FORMAT_ONE                               ; 1024
                                  0A  13 0017A         BEQL    12$
                      50      1C  AE  D0 0017C         MOVL    VAL_DESC, R0
  05   A0         04  04          58  F0 00180         INSV    FORMAT_ONE, #4, #4, 5(R0)
                       00000000G  00  DD 00186 12$:    PUSHL   DBG$GL_SIGN_FLAG                         ; 1025
                      7E          5A  9A 0018C         MOVZBL  RADIX, -(SP)
                              24  AE  DD 0018F         PUSHL   VAL_DESC
           00000000G  00          03  FB 00192         CALLS   #3, DBG$PRINT_VALUE
           00000000G  00          00  FB 00199         CALLS   #0, DBG$NEWLINE                          ; 1026
                                  53  DD 001A0         PUSHL   PRM_DESC                                 ; 1027
           00000000G  00          01  FB 001A2         CALLS   #1, DBG$SAVE_LOC
                                  52  DD 001A9         PUSHL   MARK                                     ; 1028
           00000000G  00          01  FB 001AB         CALLS   #1, DBG$POP_TEMPMEM
                            0048  8F  BB 001B2         PUSHR   #^M<R3,R6>                               ; 1029
               0000V  CF          02  FB 001B6         CALLS   #2, PRIMARY_ORDER
                                  50  D5 001BB         TSTL    R0
                                  03  19 001BD         BLSS    14$
                                0270  31 001BF 13$:    BRW     36$
                                  7E  D4 001C2 14$:    CLRL    -(SP)                                    ; 1030
                                  53  DD 001C4         PUSHL   PRM_DESC
               0000V  CF          02  FB 001C6         CALLS   #2, MODIFY_PRIMARY
                      F1          50  E9 001CB         BLBC    R0, 13$
                                FF6F  31 001CE         BRW     11$
                      7E          6E  3C 001D1 15$:    MOVZWL  NEW_SIZE, -(SP)                          ; 1048
                      7E          59  9A 001D4         MOVZBL  NEW_TYPE, -(SP)
                                  56  DD 001D7         PUSHL   END_DESC
                F8FC  CF          03  FB 001D9         CALLS   #3, DBG$CHANGE_DTYPE
                      52          50  D0 001DE         MOVL    R0, ADDR_DESC
                              18  A2  DD 001E1         PUSHL   24(ADDR_DESC)                            ; 1049
           00000000G  00          01  FB 001E4         CALLS   #1, DBG$STA_ADDRESS_TO_REGDESCR
                      54          50  D0 001EB         MOVL    R0, RDESC_ONE
                  0C  AE      18  A2  D0 001EE         MOVL    24(ADDR_DESC), LAST_ADDR                 ; 1050
                      7E          6E  3C 001F3         MOVZWL  NEW_SIZE, -(SP)                          ; 1051
```

```
                7E                  59  9A 001F6          MOVZBL  NEW_TYPE, -(SP)                                 ;
                                    53  DD 001F9          PUSHL   PRM_DESC                                        ;
            F8DA    CF              03  FB 001FB          CALLS   #3, DBG$CHANGE_DTYPE                            ;
                    52              50  D0 00200          MOVL    R0, ADDR_DESC                                   ;
                                18  A2  DD 00203          PUSHL   24(ADDR_DESC)                                   ; 1052
    00000000G       00              01  FB 00206          CALLS   #1, DBG$STA_ADDRESS_TO_REGDESCR                 ;
                    50              54  CC 0020D          XORL2   RDESC_ONE, R0                                   ; 1053
    FFFF00FC        8F              50  D3 00210          BITL    R0, #-65284                                     ;
                                    0D  13 00217          BEQL    16$                                             ;
                        00028190    8F  DD 00219          PUSHL   #164240                                         ; 1055
    00000000G       00              01  FB 0021F          CALLS   #1, LIB$SIGNAL                                  ;
           18   A2          0C      AE  D1 00226 16$:     CMPL    LAST_ADDR, 24(ADDR_DESC)                        ; 1057
                                    0D  1E 0022B          BGEQU   17$                                             ;
                        00028190    8F  DD 0022D          PUSHL   #164240                                         ; 1059
    00000000G       00              01  FB 00233          CALLS   #1, LIB$SIGNAL                                  ;
                    54          14  A2  9E 0023A 17$:     MOVAB   20(ADDR_DESC), R4                               ; 1061
                    0D          03  A4  91 0023E          CMPB    3(R4), #13                                      ;
                                    0D  12 00242          BNEQ    18$                                             ;
                        000287D8    8F  DD 00244          PUSHL   #165848                                         ; 1063
    00000000G       00              01  FB 0024A          CALLS   #1, LIB$SIGNAL                                  ;
           08   AE          7A      8F  9A 00251 18$:     MOVZBL  #122, DESC_TYPE                                 ; 1065
                    17          02  A4  91 00256          CMPB    2(R4), #23                                      ; 1066
                                    06  13 0025A          BEQL    19$                                             ;
                    16          02  A4  91 0025C          CMPB    2(R4), #22                                      ; 1067
                                    05  12 00260          BNEQ    20$                                             ;
           08   AE          83      8F  9A 00262 19$:     MOVZBL  #131, DESC_TYPE                                 ; 1069
    00000000G       00              00  FB 00267 20$:     CALLS   #0, DBG$PUSH_TEMPMEM                            ; 1076
           14   AE                  50  D0 0026E          MOVL    R0, MARK                                        ;
                                    52  DD 00272          PUSHL   ADDR_DESC                                       ; 1077
    00000000G       00              01  FB 00274          CALLS   #1, DBG$PRINT_IDENTIFIER                        ;
                        00000000'   EF  9F 0027B          PUSHAB  P.AAG                                           ; 1078
                                    01  DD 00281          PUSHL   #1                                              ;
                        00000000'   EF  9F 00283          PUSHAB  P.AAF                                           ;
                    00000000G       03  FB 00289          CALLS   #3, DBG$PRINT                                   ;
                                1C  AE  9F 00290          PUSHAB  VAL_DESC                                        ; 1079
                                0C  AE  DD 00293          PUSHL   DESC_TYPE                                       ;
                                    52  DD 00296          PUSHL   ADDR_DESC                                       ;
                    00000000G       03  FB 00298          CALLS   #3, DBG$PRIM_TO_VAL                             ;
                                    58  95 0029F          TSTB    FORMAT_ONE                                      ; 1080
                                    0A  13 002A1          BEQL    21$                                             ;
                                50  1C  AE  D0 002A3          MOVL    VAL_DESC, R0                                    ;
    05  A0              04          04  58  F0 002A7          INSV    FORMAT_ONE, #4, #4, 5(R0)                       ;
                            00000000G   00  DD 002AD 21$:     PUSHL   DBG$GL_SIGN_FLAG                                ; 1081
                                7E          5A  9A 002B3          MOVZBL  RADIX, -(SP)                                    ;
                                24  AE  DD 002B6          PUSHL   VAL_DESC                                        ;
                    00000000G       03  FB 002B9          CALLS   #3, DBG$PRINT_VALUE                             ;
                    00000000G       00  FB 002C0          CALLS   #0, DBG$NEWLINE                                 ; 1082
                                14  AE  DD 002C7          PUSHL   MARK                                            ; 1083
                    00000000G       01  FB 002CA          CALLS   #1, DBG$POP_TEMPMEM                             ;
                                    54  DD 002D1          PUSHL   R4                                              ; 1089
                    00000000G       01  FB 002D3          CALLS   #1, DBG$DATA_LENGTH                             ;
                                    50  07  C0 002DA          ADDL2   #7, R0                                          ;
                04  AE              50  08  C7 002DD          DIVL3   #8, R0, LENGTH                                  ;
                                    0F  12 002E2          BNEQ    22$                                             ; 1090
                        000287B3    8F  DD 002E4          PUSHL   #165811                                         ; 1093
                    00000000G       01  FB 002EA          CALLS   #1, LIB$SIGNAL                                  ;
                                    50  11 002F1          BRB     28$                                             ; 1092
```

```
                57 18 A2      04 AE C1 002F3 22$:   ADDL3   LENGTH, 24(ADDR_DESC), NEXT_ADDR          ;1097
                   0C AE         57 D1 002F9        CMPL    NEXT_ADDR, LAST_ADDR                      ;1098
                                 44 1A 002FD        BGTRU   28$
                   18 A2         57 D0 002FF        MOVL    NEXT_ADDR, 24(ADDR_DESC)                  ;1099
                      17      02 A4 91 00303        CMPB    2(R4), #23                                ;1100
                                 06 13 00307        BEQL    23$
                      16      02 A4 91 00309        CMPB    2(R4), #22                                ;1101
                                 27 12 0030D        BNEQ    26$
                                 57 DD 0030F 23$:   PUSHL   NEXT_ADDR                                 ;1104
          00000000G   00         01 FB 00311        CALLS   #1, DBG$IS_IT_ENTRY
                      06         50 E9 00318        BLBC    R0, 24$
                   02 A4         17 90 0031B        MOVB    #23, 2(R4)                                ;1105
                                 04 11 0031F        BRB     25$
                   02 A4         16 90 00321 24$:   MOVB    #22, 2(R4)                                ;1106
                                 7E D4 00325 25$:   CLRL    -(SP)                                     ;1107
                                 57 DD 00327        PUSHL   NEXT_ADDR
          00000000G   00         02 FB 00329        CALLS   #2, DBG$INS_DECODE
             64       50         57 A3 00330        SUBW3   NEXT_ADDR, R0, (R4)
                                 0A 11 00334        BRB     27$                                       ;1100
                                 54 DD 00336 26$:   PUSHL   R4                                        ;1110
                0000V CF         01 FB 00338        CALLS   #1, FIX_UP_LENGTH
                      64         50 B0 0033D        MOVW    R0, (R4)
                                FF24 31 00340 27$:   BRW     20$                                       ;1074
                                 52 DD 00343 28$:   PUSHL   ADDR_DESC                                 ;1112
          00000000G   00         01 FB 00345        CALLS   #1, DBG$SAVE_LOC
                                 2B 11 0034C        BRB     30$                                       ;0984
                                 53 DD 0034E 29$:   PUSHL   PRM_DESC                                  ;1126
          00000000G   00         01 FB 00350        CALLS   #1, DBG$PRINT_IDENTIFIER
                      53         50 D0 00357        MOVL    R0, PRM_DESC
                                 53 DD 0035A        PUSHL   PRM_DESC                                  ;1127
          00000000G   00         01 FB 0035C        CALLS   #1, DBG$SAVE_LOC
                   80 8F         59 91 00363        CMPB    NEW_TYPE, #128                            ;1128
                                 13 12 00367        BNEQ    31$
                      0F      04 A3 E9 00369        BLBC    4(PRM_DESC), 31$
                      7E         5A 9A 0036D        MOVZBL  RADIX, -(SP)                              ;1130
                                 53 DD 00370        PUSHL   PRM_DESC
          00000000G   00         02 FB 00372        CALLS   #2, DBG$PRINT_AGGREGATE
                              00B6 31 00379 30$:    BRW     36$
                00000000'        EF 9F 0037C 31$:   PUSHAB  P.AAI                                     ;1133
                                 01 DD 00382        PUSHL   #1
                00000000'        EF 9F 00384        PUSHAB  P.AAH
          00000000G   00         03 FB 0038A        CALLS   #3, DBG$PRINT
                      7E         6E 3C 00391        MOVZWL  NEW_SIZE, -(SP)                           ;1134
                      7E         59 9A 00394        MOVZBL  NEW_TYPE, -(SP)
                                 53 DD 00397        PUSHL   PRM_DESC
                F73C CF          03 FB 00399        CALLS   #3, DBG$CHANGE_DTYPE
                  1C AE          50 D0 0039E        MOVL    R0, VAL_DESC
                  10 AE          58 90 003A2        MOVB    FORMAT_ONE, FORMAT_TWO                    ;1135
                  80 8F          59 91 003A6        CMPB    NEW_TYPE, #128                            ;1136
                                 10 13 003AA        BEQL    32$
             7E   1C AE          14 C1 003AC        ADDL3   #20, VAL_DESC, -(SP)                      ;1138
                                 53 DD 003B1        PUSHL   PRM_DESC
          00000000G   00         02 FB 003B3        CALLS   #2, DBG$SAVE_LOC
                                 28 11 003BA        BRB     33$
                                 58 95 003BC 32$:   TSTB    FORMAT_ONE                                ;1139
                                 24 12 003BE        BNEQ    33$
                      01         5A 91 003C0        CMPB    RADIX, #1
```

```
                                   1F  12 003C3          BNEQ    33$                                ;
                     50        1C  AE  D0 003C5          MOVL    VAL_DESC, R0                       ; 1140
                     0D        17  A0  91 003C9          CMPB    23(R0), #13                        ;
                                   15  13 003CD          BEQL    33$                                ;
                     50        1C  AE  D0 003CF          MOVL    VAL_DESC, R0                       ; 1141
                     51 00000000G  00  9E 003D3          MOVAB   DBG$REG_VALUES+64, R1              ;
                     51        18  A0  D1 003DA          CMPL    24(R0), R1                         ;
                                   04  12 003DE          BNEQ    33$                                ;
                10   AE            02  90 003E0          MOVB    #2, FORMAT_TWO                     ; 1142
                     50        1C  AE  D0 003E4 33$:     MOVL    VAL_DESC, R0                       ; 1144
                     16        16  A0  91 003E8          CMPB    22(R0), #22                        ;
                                   1A  13 003EC          BEQL    34$                                ;
                               1C  AE  9F 003EE          PUSHAB  VAL_DESC                           ; 1147
                     7E        7A  8F  9A 003F1          MOVZBL  #122, -(SP)                        ;
                                   50  DD 003F5          PUSHL   R0                                 ;
           00000000G 00            03  FB 003F7          CALLS   #3, DBG$PRIM_TO_VAL                ;
                               1C  AE  DD 003FE          PUSHL   VAL_DESC                           ; 1148
           00000000G 00            01  FB 00401          CALLS   #1, DBG$DO_MAPPING                 ;
                               10  AE  95 00408 34$:     TSTB    FORMAT_TWO                         ; 1151
                                   0B  13 0040B          BEQL    35$                                ;
                     50        1C  AE  D0 0040D          MOVL    VAL_DESC, R0                       ;
  05    A0         04  04      10  AE  F0 00411          INSV    FORMAT_TWO, #4, #4, 5(R0)          ;
                        00000000G  00  DD 00418 35$:     PUSHL   DBG$GL_SIGN_FLAG                   ; 1152
                     7E            5A  9A 0041E          MOVZBL  RADIX, -(SP)                       ;
                               24  AE  DD 00421          PUSHL   VAL_DESC                           ;
           00000000G 00            03  FB 00424          CALLS   #3, DBG$PRINT_VALUE                ;
           00000000G 00            00  FB 0042B          CALLS   #0, DBG$NEWLINE                    ; 1153
                     55        08  A5  D0 00432 36$:     MOVL    8(NOUN_NODE), NOUN_NODE            ; 1157
                                   03  13 00436          BEQL    37$                                ;
                                 FC48  31 00438          BRW     5$                                 ;
                                       04 0043B 37$:     RET                                        ; 1160

; Routine Size:  1084 bytes,    Routine Base:  DBG$CODE + 0526
```

```
;1036     1161  1 GLOBAL ROUTINE DBG$NEXTLOC(PRM_DESC) =
;1037     1162  1 !
;1038     1163  1 ! FUNCTION
;1039     1164  1 !       ----------------------------
;1040     1165  1 !
;1041     1166  1 ! INPUTS
;1042     1167  1 !       ----------------------------
;1043     1168  1 !
;1044     1169  1 ! OUTPUTS
;1045     1170  1 !       ----------------------------
;1046     1171  1 !
;1047     1172  1
;1048     1173  2     BEGIN
;1049     1174  2
;1050     1175  2     MAP
;1051     1176  2         PRM_DESC: REF DBG$PRIMARY;      ! Pointer to Primary Descriptor
;1052     1177  2
;1053     1178  2     LOCAL
;1054     1179  2         BYTE_OFFSET,
;1055     1180  2         LENGTH,
;1056     1181  2         REG_DESC: DBG$REGDESCR,
;1057     1182  2         STATUS,
;1058     1183  2         VAL_DESC: REF DBG$VALDESC;
;1059     1184  2
;1060     1185  2
;1061     1186  2     ! ------
;1062     1187  2     !
;1063     1188  2     STATUS = MODIFY_PRIMARY(.PRM_DESC,0);
;1064     1189  2     IF .STATUS THEN RETURN .PRM_DESC;
;1065     1190  2     IF .DBG$GL_CURLOC_VMSDESC NEQ 0
;1066     1191  2     THEN
;1067     1192  2         BEGIN
;1068     1193  3         VAL_DESC = DBG$MAKE_VAL_DESC(.DBG$GL_CURLOC_VMSDESC, DBG$K_V_VALUE_DESC);
;1069     1194  3         VAL_DESC[DBG$B_DHDR_LANG]   = .PRM_DESC[DBG$B_DHDR_LANG];
;1070     1195  3         VAL_DESC[DBG$L_DHDR_SYMIDO] = .PRM_DESC[DBG$L_DHDR_SYMIDO];
;1071     1196  3         END
;1072     1197  3
;1073     1198  2     ELSE
;1074     1199  3         BEGIN
;1075     1200  3         IF .STATUS EQL 2 THEN SIGNAL(DBG$_NOSUCC);
;1076     1201  3         DBG$PRIM_TO_VAL(.PRM_DESC, DBG$K_V_VALUE_DESC, VAL_DESC);
;1077     1202  3         END;
;1078     1203  2
;1079     1204  2
;1080     1205  3     IF (.VAL_DESC[DBG$B_VALUE_CLASS] EQL DSC$K_CLASS_UBS)
;1081     1206  2     THEN
;1082     1207  2         SIGNAL(DBG$_NOSUCC);
;1083     1208  2
;1084     1209  2     LENGTH = (DBG$DATA_LENGTH(VAL_DESC[DBG$A_VALUE_VMSDESC]) - 1)/%BPUNIT + 1;
;1085     1210  2     REG_DESC = DBG$STA_ADDRESS_TO_REGDESCR(.VAL_DESC[DBG$L_VALUE_POINTER]);
;1086     1211  2     IF .REG_DESC NEQ 0
;1087     1212  2     THEN
;1088     1213  3         BEGIN
;1089     1214  3         BYTE_OFFSET = 4*.REG_DESC[DBG$B_REGD_REGNUM]
;1090     1215  3                     + .REG_DESC[DBG$V_REGD_OFFSET]
;1091     1216  3                     + .LENGTH + .DBG$GW_DF[TLENG;
;1092     1217  3         IF (.BYTE_OFFSET GTR 16*%UPVAL) AND
```

```
;1093     1218  4              ((.DBG$GW_DFLTLENG NEQ 2) OR (.DBG$GW_DFLTLENG NEQ 4) OR
;1094     1219  4              (.BYTE_OFFSET NEQ (16*%UPVAL + .DBG$GW_DFLTLENG)))
;1095     1220  3          THEN
;1096     1221  3              SIGNAL(DBG$_NOSUCC);
;1097     1222  3
;1098     1223  2          END;
;1099     1224  2
;1100     1225  2
;1101     1226  2          ! Initialize the Value Descriptor to VMS descriptor class Z (unknown) and
;1102     1227  2          ! set the pointer to the next location to be the current location plus the
;1103     1228  2          ! length of the current object.
;1104     1229  2          !
;1105     1230  2          VAL_DESC[DBG$B_VALUE_CLASS]   = DSC$K_CLASS_Z;
;1106     1231  2          VAL_DESC[DBG$L_VALUE_POINTER] = .VAL_DESC[DBG$L_VALUE_POINTER] + .LENGTH;
;1107     1232  2
;1108     1233  2
;1109     1234  2          ! If the data type is instruction or entry point, determine the type of the
;1110     1235  2          ! next location by seeing if it is an instruction or entry mask.  Also com-
;1111     1236  2          ! pute its length by interpreting the instruction at that location.
;1112     1237  2          !
;1113     1238  2          IF (.VAL_DESC[DBG$B_VALUE_DTYPE] EQL DSC$K_DTYPE_ZI) OR
;1114     1239  3             (.VAL_DESC[DBG$B_VALUE_DTYPE] EQL DSC$K_DTYPE_ZEM)
;1115     1240  2          THEN
;1116     1241  3              BEGIN
;1117     1242  3              IF DBG$IS_IT_ENTRY(.VAL_DESC[DBG$L_VALUE_POINTER])
;1118     1243  3              THEN
;1119     1244  3                  VAL_DESC[DBG$B_VALUE_DTYPE]  = DSC$K_DTYPE_ZEM
;1120     1245  3
;1121     1246  3              ELSE
;1122     1247  3                  VAL_DESC[DBG$B_VALUE_DTYPE]  = DSC$K_DTYPE_ZI;
;1123     1248  3
;1124     1249  3              VAL_DESC[DBG$W_VALUE_LENGTH] =
;1125     1250  3                              DBG$INS_DECODE(.VAL_DESC[DBG$L_VALUE_POINTER], FALSE) -
;1126     1251  3                              .VAL_DESC[DBG$L_VALUE_POINTER];
;1127     1252  3              END
;1128     1253  3
;1129     1254  3
;1130     1255  3          ! The next location is not an instruction or entry mask.  It is thus a
;1131     1256  3          ! data object and we set up the Value Descriptor accordingly.
;1132     1257  3          !
;1133     1258  2          ELSE
;1134     1259  3              BEGIN
;1135     1260  3              VAL_DESC[DBG$B_DHDR_FCODE]    =  RST$K_TYPE_DESCR;
;1136     1261  3              VAL_DESC[DBG$V_DHDR_FORMAT]   =  0;
;1137     1262  3              VAL_DESC[DBG$B_VALUE_CLASS]   =  DSC$K_CLASS_S;
;1138     1263  3              VAL_DESC[DBG$B_VALUE_DTYPE]   = .DBG$GB_DFLTTYP;
;1139     1264  3              VAL_DESC[DBG$W_VALUE_LENGTH] = .DBG$GW_DFLTLENG;
;1140     1265  2              END;
;1141     1266  2
;1142     1267  2
;1143     1268  2          ! Return a pointer to the Value Descriptor for the next location.
;1144     1269  2          !
;1145     1270  2          RETURN .VAL_DESC;
;1146     1271  2
;1147     1272  1          END;
```

```
                                 007C 00000          .ENTRY  DBG$NEXTLOC, Save R2,R3,R4,R5,R6                ; 1161
                         56 00000000G   00    9E 00002      MOVAB   DBG$GW_DFLTLENG, R6
                         55 00000000G   00    9E 00009      MOVAB   LIB$SIGNAL, R5
                         5E             04    C2 00010      SUBL2   #4, SP
                                        7E    D4 00013      CLRL    -(SP)                                   ; 1189
                         52       04    AC    D0 00015      MOVL    PRM_DESC, R2
                                        52    DD 00019      PUSHL   R2
                   0000V CF             02    FB 0001B      CALLS   #2, MODIFY_PRIMARY
                         03             50    E9 00020      BLBC    STATUS, 1$                              ; 1190
                                      010A    31 00023      BRW     12$
                         51 00000000G   00    D0 00026 1$:  MOVL    DBG$GL_CURLOC_VMSDESC, R1               ; 1191
                                        1C    13 0002D      BEQL    2$
                         7E       83    8F    9A 0002F      MOVZBL  #131, -(SP)                             ; 1194
                                        51    DD 00033      PUSHL   R1
             00000000G   00             02    FB 00035      CALLS   #2, DBG$MAKE_VAL_DESC
                         6E             50    D0 0003C      MOVL    R0, VAL_DESC
                   03    A0       03    A2    90 0003F      MOVB    3(R2), 3(R0)                            ; 1195
                   0C    A0       0C    A2    D0 00044      MOVL    12(R2), 12(R0)                          ; 1196
                                        1D    11 00049      BRB     4$                                      ; 1191
                         02             50    D1 0004B 2$:  CMPL    STATUS, #2                              ; 1201
                                        09    12 0004E      BNEQ    3$
                            00028818    8F    DD 00050      PUSHL   #165912
                         65             01    FB 00056      CALLS   #1, LIB$SIGNAL
                                        5E    DD 00059 3$:  PUSHL   SP                                      ; 1202
                         7E       83    8F    9A 0005B      MOVZBL  #131, -(SP)
                                        52    DD 0005F      PUSHL   R2
                   00000000G   00             03    FB 00061      CALLS   #3, DBG$PRIM_TO_VAL
                         52             6E    D0 00068 4$:  MOVL    VAL_DESC, R2                            ; 1205
                         53       14    A2    9E 0006B      MOVAB   20(R2), R3
                         0D       03    A3    91 0006F      CMPB    3(R3), #13
                                        09    12 00073      BNEQ    5$
                            00028818    8F    DD 00075      PUSHL   #165912                                 ; 1207
                         65             01    FB 0007B      CALLS   #1, LIB$SIGNAL
                                        53    DD 0007E 5$:  PUSHL   R3                                      ; 1209
                   00000000G   00             01    FB 00080      CALLS   #1, DBG$DATA_LENGTH
                                              50    D7 00087      DECL    R0
                         50                   08    C6 00089      DIVL2   #8, R0
                         54             01    A0    9E 0008C      MOVAB   1(R0), LENGTH
                                        18    A2    DD 00090      PUSHL   24(R2)                          ; 1210
                   00000000G   00             01    FB 00093      CALLS   #1, DBG$STA_ADDRESS_TO_REGDESCR
                                              50    D5 0009A      TSTL    REG_DESC                        ; 1211
                                              3D    13 0009C      BEQL    7$
             51          50             08          08    EF 0009E      EXTZV   #8, #8, REG_DESC, R1            ; 1214
             50          50             02          00    EF 000A3      EXTZV   #0, #2, REG_DESC, R0            ; 1215
                                        50        6041    DE 000A8      MOVAL   (R0)[R1], R0
                         51             50          54    C1 000AC      ADDL3   LENGTH, R0, R1                  ; 1216
                                        50          66    3C 000B0      MOVZWL  DBG$GW_DFLTLENG, R0
                                        51          50    C0 000B3      ADDL2   R0, BYTE_OFFSET
                   00000040   8F                   51    D1 000B6      CMPL    BYTE_OFFSET, #64                ; 1217
                                        1C          15    13 000BD      BLEQ    7$
                         02             50          B1 000BF      CMPW    R0, #2                          ; 1218
                                        0E          12 000C2      BNEQ    6$
                         04             50          B1 000C4      CMPW    R0, #4
                                        09          12 000C7      BNEQ    6$
```

```
                        50        40    A0  9E 000C9          MOVAB   64(R0), R0                      ; 1219
                        50              51  D1 000CD          CMPL    BYTE_OFFSET, R0                 ;
                                        09  13 000D0          BEQL    7$                              ;
                    00028818            8F  DD 000D2 6$:      PUSHL   #165912                         ; 1221
                        65              01  FB 000D8          CALLS   #1, LIB$SIGNAL                  ;
                                03      A3  94 000DB 7$:      CLRB    3(R3)                           ; 1230
                    18          A2      54  C0 000DE          ADDL2   LENGTH, 24(R2)                  ; 1231
                        16      02      A3  91 000E2          CMPB    2(R3), #22                      ; 1238
                                        06  13 000E6          BEQL    8$                              ;
                        17      02      A3  91 000E8          CMPB    2(R3), #23                      ; 1239
                                        2A  12 000EC          BNEQ    11$                             ;
                                18      A2  DD 000EE 8$:      PUSHL   24(R2)                          ; 1242
                00000000G       00          01  FB 000F1      CALLS   #1, DBG$IS_IT_ENTRY             ;
                                06          50  E9 000F8      BLBC    R0, 9$                          ;
                        02      A3          17  90 000FB      MOVB    #23, 2(R3)                      ; 1244
                                            04  11 000FF      BRB     10$                             ;
                        02      A3          16  90 00101 9$:  MOVB    #22, 2(R3)                      ; 1247
                                            7E  D4 00105 10$:  CLRL    -(SP)                           ; 1250
                                18      A2  DD 00107          PUSHL   24(R2)                          ;
                00000000G       00          02  FB 0010A      CALLS   #2, DBG$INS_DECODE              ;
            63                  50      18  A2  A3 00111      SUBW3   24(R2), R0, (R3)                ; 1251
                                            18  11 00116      BRB     12$                             ; 1238
                        06      A2          03  90 00118 11$:  MOVB    #3, 6(R2)                       ; 1260
                        05      A2      F0  8F  8A 0011C      BICB2   #240, 5(R2)                     ; 1261
                        03      A3          01  90 00121      MOVB    #1, 3(R3)                       ; 1262
                        02      A3  00000000G   00  90 00125  MOVB    DBG$GL_DFLTTYP, 2(R3)           ; 1263
                                63              66  B0 0012D  MOVW    DBG$GW_DFLTLENG, (R3)           ; 1264
                                50              52  D0 00130 12$:  MOVL    R2, R0                          ; 1270
                                                04  00133      RET                                     ; 1272

; Routine Size:  308 bytes,    Routine Base:  DBG$CODE + 0962
```

```
;1149     1273  1 GLOBAL ROUTINE DBG$PREVLOC(PRM_DESC) =
;1150     1274  1 !
;1151     1275  1 ! FUNCTION
;1152     1276  1 !       This routine finds the "previous location", denoted in the command
;1153     1277  1 !       language as %PREVLOC or ^.  It accepts a Primary Descriptor for the
;1154     1278  1 !       current location as input and returns either a Primary Descriptor or
;1155     1279  1 !       a Volatile Value Descriptor for the previous location as output.  If
;1156     1280  1 !       the current location is a structured object of some sort (like an
;1157     1281  1 !       array), MODIFY_PRIMARY is called to find the logical predecessor and
;1158     1282  1 !       the modified Primary Descriptor is returned.  Otherwise, this routine
;1159     1283  1 !       determines the previous instruction location or the previous data
;1160     1284  1 !       location and returns a Volatile Value Descriptor for that location.
;1161     1285  1 !
;1162     1286  1 ! INPUTS
;1163     1287  1 !       PRM_DESC - A pointer to the input Primary Descriptor for the location
;1164     1288  1 !                  whose logical predecessor is to be computed.
;1165     1289  1 !
;1166     1290  1 ! OUTPUTS
;1167     1291  1 !       A pointer to the Primary Descriptor or Volatile Value Descriptor for
;1168     1292  1 !                  the logical predecessor location is returned as this
;1169     1293  1 !                  routine's value.
;1170     1294  1 !
;1171     1295  1
;1172     1296  2     BEGIN
;1173     1297  2
;1174     1298  2     MAP
;1175     1299  2         PRM_DESC: REF DBG$PRIMARY;      ! Pointer to Primary Descriptor
;1176     1300  2
;1177     1301  2     LOCAL
;1178     1302  2         ADDRESS,                        ! Address of the current location - 1
;1179     1303  2         DUMMY,                          ! Dummy routine argument
;1180     1304  2         LENGTH,
;1181     1305  2         LINE,                           ! Line number of the last instruction
;1182     1306  2         NEW_ADDR,                       ! Addess of the current instruction
;1183     1307  2         OLD_ADDR,                       ! Address of previous instruction
;1184     1308  2         PC_BEG,                         ! Beginning PC of current source line
;1185     1309  2         PC_END,                         ! Ending PC of current source line
;1186     1310  2         STATUS,
;1187     1311  2         STMT,                           ! Statement number of last instruction
;1188     1312  2         SYMID: REF RST$ENTRY,           ! The SYMID of the nearest preceding
;1189     1313  2                                         !      symbol (used for instructions)
;1190     1314  2         REG_DESC: DBG$REGDESCR,
;1191     1315  2         VAL_DESC: REF DBG$VALDESC;      ! Pointer to returned Value Descriptor
;1192     1316  2
;1193     1317  2
;1194     1318  2
;1195     1319  2     ! If the input Primary Descriptor describes a structure object, like an
;1196     1320  2     ! array or record, let MODIFY_PRIMARY modify the Primary Descriptor to
;1197     1321  2     ! describe the logical predecessor.  Then return that Primary.
;1198     1322  2     !
;1199     1323  2     STATUS = MODIFY_PRIMARY(.PRM_DESC, 1);
;1200     1324  2     IF .STATUS THEN RETURN .PRM_DESC;
;1201     1325  2
;1202     1326  2
;1203     1327  2     ! If there is a defined "current location" (%CURLOC), then use the Primary
;1204     1328  2     ! or Value Descriptor for that entity to set up a Volatile Value Descriptor
;1205     1329  2     ! for the previous location.
```

```
;  1206          1330  2      !
;  1207          1331  2      IF .DBG$GL_CURLOC_VMSDESC NEQ 0
;  1208          1332  2      THEN
;  1209          1333  2          BEGIN
;  1210          1334  2          VAL_DESC = DBG$MAKE_VAL_DESC(.DBG$GL_CURLOC_VMSDESC, DBG$K_V_VALUE_DESC);
;  1211          1335  2          VAL_DESC[DBG$B_DHDR_LANG]   = .PRM_DESC[DBG$B_DHDR_LANG];
;  1212          1336  2          VAL_DESC[DBG$L_DHDR_SYMIDO] = .PRM_DESC[DBG$L_DHDR_SYMIDO];
;  1213          1337  2          END
;  1214          1338  2
;  1215          1339  2
;  1216          1340  2      ! But if no current location is defined, give an error message or use the
;  1217          1341  2      ! input Primary Descriptor to set up the previous location descriptor.
;  1218          1342  2      !
;  1219          1343  2      ELSE
;  1220          1344  2          BEGIN
;  1221          1345  2          IF .STATUS EQL 2 THEN SIGNAL(DBG$_NOPRED);
;  1222          1346  2          DBG$PRIM_TO_VAL(.PRM_DESC, DBG$K_V_VALUE_DESC, VAL_DESC);
;  1223          1347  2          END;
;  1224          1348  2
;  1225          1349  2
;  1226          1350  2      ! There is no logical successor for an unaligned bit string, so for that
;  1227          1351  2      ! case we signal an error.
;  1228          1352  2      !
;  1229          1353  2      IF .VAL_DESC[DBG$B_VALUE_CLASS] EQL DSC$K_CLASS_UBS
;  1230          1354  2      THEN
;  1231          1355  2          SIGNAL(DBG$_NOPRED);
;  1232          1356  2
;  1233          1357  2
;  1234          1358  2      ! -----
;  1235          1359  2      !
;  1236          1360  2      LENGTH = (DBG$DATA_LENGTH(VAL_DESC[DBG$A_VALUE_VMSDESC]) - 1)/%BPUNIT + 1;
;  1237          1361  2      REG_DESC = DBG$STA_ADDRESS_TO_REGDESCR(.VAL_DESC[DBG$L_VALUE_POINTER]);
;  1238          1362  3      IF (.REG_DESC NEQ 0) AND (.REG_DESC<W_> LSSU (%X'00B4' + .DBG$GW_DFLTLENG))
;  1239          1363  2      THEN
;  1240          1364  2          SIGNAL(DBG$_NOPRED);
;  1241          1365  2
;  1242          1366  2
;  1243          1367  2      ! Initialize the DTYPE of the logical predecessor to be type Z (unknown)
;  1244          1368  2      ! and assume its address is one byte before the current location.  This
;  1245          1369  2      ! may get changed below if appropriate.
;  1246          1370  2      !
;  1247          1371  2      VAL_DESC[DBG$B_VALUE_CLASS] = DSC$K_CLASS_Z;
;  1248          1372  2      ADDRESS = .VAL_DESC[DBG$L_VALUE_POINTER] = 1;
;  1249          1373  2
;  1250          1374  2
;  1251          1375  2      ! If the type of the current object is instruction or entry mask, try to
;  1252          1376  2      ! locate the previous instruction.
;  1253          1377  2      !
;  1254          1378  2      IF (.VAL_DESC[DBG$B_VALUE_DTYPE] EQL DSC$K_DTYPE_ZI) OR
;  1255          1379  3         (.VAL_DESC[DBG$B_VALUE_DTYPE] EQL DSC$K_DTYPE_ZEM)
;  1256          1380  2      THEN
;  1257          1381  3          BEGIN
;  1258          1382  3          OLD_ADDR = .ADDRESS;
;  1259          1383  3
;  1260          1384  3
;  1261          1385  3          ! First try to symbolize the current location - 1 byte (the contents of
;  1262          1386  3          ! ADDRESS) to find the nearest routine, block, or label preceding the
```

```
;1263     1387  3          ! current instruction.  If no such symbol is found, we just leave the
;1264     1388  3          ! address as the current instruction address - 1; there is no way we can
;1265     1389  3          ! locate the true previous instruction.
;1266     1390  3          !
;1267     1391  3          STATUS = DBG$PC_TO_SYMID(.ADDRESS, SYMID);
;1268     1392  4          IF .STATUS AND .SYMID NEQ 0)
;1269     1393  3          THEN
;1270     1394  4              BEGIN
;1271     1395  4
;1272     1396  4
;1273     1397  4              ! A symbol preceding the current instruction was found.  If this
;1274     1398  4              ! is an instruction symbol (a routine, block, or label), save its
;1275     1399  4              ! address for the forward scan to the desired instruction.  However,
;1276     1400  4              ! if a line number preceding the current instruction can be found,
;1277     1401  4              ! use that address instead for a shorter forward scan.
;1278     1402  4              !
;1279     1403  4              IF (.SYMID[RST$B_KIND] EQL RST$K_ROUTINE) OR
;1280     1404  4                 (.SYMID[RST$B_KIND] EQL RST$K_BLOCK)   OR
;1281     1405  5                 (.SYMID[RST$B_KIND] EQL RST$K_LABEL)
;1282     1406  4              THEN
;1283     1407  5                  BEGIN
;1284     1408  5                  OLD_ADDR = .SYMID[RST$L_STARTADDR];
;1285     1409  5                  DUMMY = .SYMID;
;1286     1410  5                  IF DBG$PC_TO_LINE_LOOKUP(.ADDRESS,
;1287     1411  5                                           LINE, STMT, PC_BEG, PC_END, DUMMY)
;1288     1412  5                  THEN
;1289     1413  5                      OLD_ADDR = .PC_BEG;
;1290     1414  5
;1291     1415  4                  END;
;1292     1416  4
;1293     1417  3              END;                          ! End of code if we found a symbolization
;1294     1418  3
;1295     1419  3
;1296     1420  3          ! We now some address where to start the forward scan that looks for
;1297     1421  3          ! the previous instruction.  Scan forward from that address until the
;1298     1422  3          ! desired previous instruction is found.
;1299     1423  3          !
;1300     1424  3          WHILE TRUE DO
;1301     1425  4              BEGIN
;1302     1426  4              NEW_ADDR = DBG$INS_DECODE(.OLD_ADDR, FALSE);
;1303     1427  4              IF .NEW_ADDR GTRA .ADDRESS THEN EXITLOOP;
;1304     1428  4              OLD_ADDR = .NEW_ADDR;
;1305     1429  4              END;
;1306     1430  3
;1307     1431  3
;1308     1432  3          ! Fill the address and length of the found previous instruction into
;1309     1433  3          ! the Value Descriptor.  Also determine if this location is an entry
;1310     1434  3          ! mask--if so, set the DTYPE to be ZEM instead of ZI.
;1311     1435  3          !
;1312     1436  3          VAL_DESC[DBG$L_VALUE_POINTER] = .OLD_ADDR;
;1313     1437  3          VAL_DESC[DBG$W_VALUE_LENGTH]  = .NEW_ADDR - .OLD_ADDR;
;1314     1438  3          VAL_DESC[DBG$B_VALUE_DTYPE] = DSC$K_DTYPE_ZI;
;1315     1439  3          IF DBG$IS_IT_ENTRY(.OLD_ADDR)
;1316     1440  3          THEN
;1317     1441  3              VAL_DESC[DBG$B_VALUE_DTYPE] = DSC$K_DTYPE_ZEM;
;1318     1442  3
;1319     1443  3          END                               ! End of code for previous instruction
```

```
;  1320          1444  3
;  1321          1445  3
;  1322          1446  3      ! The current location is not an instruction location.  We therefore set
;  1323          1447  3      ! up the Value Descriptor for a data object.
;  1324          1448  3      !
;  1325          1449  2      ELSE
;  1326          1450  3          BEGIN
;  1327          1451  3          VAL_DESC[DBG$B_DHDR_FCODE]    =  RST$K_TYPE_DESCR;
;  1328          1452  3          VAL_DESC[DBG$V_DHDR_FORMAT]   =  0;
;  1329          1453  3          VAL_DESC[DBG$B_VALUE_CLASS]   =  DSC$K_CLASS_S;
;  1330          1454  3          VAL_DESC[DBG$B_VALUE_DTYPE]   = .DBG$GL_DFLTTYP;
;  1331          1455  3          VAL_DESC[DBG$W_VALUE_LENGTH]  = .DBG$GW_DFLTLENG;
;  1332          1456  3          VAL_DESC[DBG$L_VALUE_POINTER] = .VAL_DESC[DBG$L_VALUE_POINTER] - .LENGTH;
;  1333          1457  2          END;
;  1334          1458  2
;  1335          1459  2
;  1336          1460  2      ! Return a pointer to the Volatile Value Descriptor for the previous
;  1337          1461  2      ! location.
;  1338          1462  2      !
;  1339          1463  2      RETURN .VAL_DESC;
;  1340          1464  2
;  1341          1465  1      END;


                                    03FC 00000          .ENTRY  DBG$PREVLOC, Save R2,R3,R4,R5,R6,R7,R8,R9    ; 1273
                59 00000000G   00    9E 00002          MOVAB   DBG$GW_DFLTLENG, R9                          ;
                58 00000000G   00    9E 00009          MOVAB   LIB$SIGNAL, R8                               ;
                5E                1C C2 00010          SUBL2   #28, SP                                      ;
                                  01 DD 00013          PUSHL   #1                                           ; 1323
                52        04      AC D0 00015          MOVL    PRM_DESC, R2                                 ;
                                  52 DD 00019          PUSHL   R2                                           ;
          0000V CF                02 FB 0001B          CALLS   #2, MODIFY_PRIMARY                           ;
                57                50 D0 00020          MOVL    R0, STATUS                                   ;
                03                57 E9 00023          BLBC    STATUS, 1$                                   ; 1324
                              014D 31 00026          BRW     12$                                          ;
                50 00000000G   00    D0 00029 1$:     MOVL    DBG$GL_CURLOC_VMSDESC, R0                    ; 1331
                                  1C 13 00030          BEQL    2$                                           ;
                7E        83      8F 9A 00032          MOVZBL  #131, -(SP)                                  ; 1334
                                  50 DD 00036          PUSHL   R0                                           ;
    00000000G   00                02 FB 00038          CALLS   #2, DBG$MAKE_VAL_DESC                        ;
                6E                50 D0 0003F          MOVL    R0, VAL_DESC                                 ;
          03    A0        03      A2 90 00042          MOVB    3(R2), 3(R0)                                 ; 1335
          0C    A0        0C      A2 D0 00047          MOVL    12(R2), 12(R0)                               ; 1336
                                  1D 11 0004C          BRB     4$                                           ; 1331
                02                57 D1 0004E 2$:      CMPL    STATUS, #2                                   ; 1345
                                  09 12 00051          BNEQ    3$                                           ;
                    00028810      8F DD 00053          PUSHL   #165904                                      ;
                68                01 FB 00059          CALLS   #1, LIB$SIGNAL                               ;
                                  5E DD 0005C 3$:      PUSHL   SP                                           ; 1346
                7E        83      8F 9A 0005E          MOVZBL  #131, -(SP)                                  ;
                                  52 DD 00062          PUSHL   R2                                           ;
    00000000G   00                03 FB 00064          CALLS   #3, DBG$PRIM_TO_VAL                          ;
                52                6E D0 0006B 4$:      MOVL    VAL_DESC, R2                                 ; 1353
                54        14      A2 9E 0006E          MOVAB   20(R2), R4                                   ;
```

```
                                    0D       03  A4  91 00072      CMPB    3(R4), #13                     ;
                                                 09  12 00076      BNEQ    5$                             ;
                                     00028810    8F  DD 00078      PUSHL   #165904                        ; 1355
                                         68      01  FB 0007E      CALLS   #1, LIB$SIGNAL                 ;
                                                 54  DD 00081 5$:  PUSHL   R4                             ; 1360
                          00000000G  00          01  FB 00083      CALLS   #1, DBG$DATA_LENGTH            ;
                                                 50  D7 0008A      DECL    R0                             ;
                                     50          08  C6 0008C      DIVL2   #8, R0                         ;
                                     56      01  A0  9E 0008F      MOVAB   1(R0), LENGTH                  ;
                                             18  A2  DD 00093      PUSHL   24(R2)                         ; 1361
                          00000000G  00          01  FB 00096      CALLS   #1, DBG$STA_ADDRESS_TO_REGDESCR ;
                                                 50  D5 0009D      TSTL    REG_DESC                       ; 1362
                                                 18  13 0009F      BEQL    6$                             ;
                                     51          69  3C 000A1      MOVZWL  DBG$GW_DFLTLENG, R1            ;
                                     51    00B4  C1  9E 000A4      MOVAB   180(R1), R1                    ;
          51              50         10          00  ED 000A9      CMPZV   #0, #16, REG_DESC, R1          ;
                                                 09  1E 000AE      BGEQU   6$                             ;
                                     00028810    8F  DD 000B0      PUSHL   #165904                        ; 1364
                                         68      01  FB 000B6      CALLS   #1, LIB$SIGNAL                 ;
                                             03  A4  94 000B9 6$:  CLRB    3(R4)                          ; 1371
                          55         18  A2          01  C3 000BC  SUBL3   #1, 24(R2), ADDRESS            ; 1372
                                     16      02  A4  91 000C1      CMPB    2(R4), #22                     ; 1378
                                                 09  13 000C5      BEQL    7$                             ;
                                     17      02  A4  91 000C7      CMPB    2(R4), #23                     ; 1379
                                                 03  13 000CB      BEQL    7$                             ;
                                               008A  31 000CD      BRW     11$                            ;
                                     53          55  D0 000D0 7$:  MOVL    ADDRESS, OLD_ADDR              ; 1382
                                             04  AE  9F 000D3      PUSHAB  SYMID                          ; 1391
                                                 55  DD 000D6      PUSHL   ADDRESS                        ;
                          00000000G  00          02  FB 000D8      CALLS   #2, DBG$PC_TO_SYMID            ;
                                     57          50  D0 000DF      MOVL    R0, STATUS                     ;
                                     42          57  E9 000E2      BLBC    STATUS, 9$                     ; 1392
                                             04  AE  D5 000E5      TSTL    SYMID                          ;
                                                 3D  13 000E8      BEQL    9$                             ;
                                     50      04  AE  D0 000EA      MOVL    SYMID, R0                      ; 1403
                                     02      14  A0  91 000EE      CMPB    20(R0), #2                     ;
                                                 0C  13 000F2      BEQL    8$                             ;
                                     03      14  A0  91 000F4      CMPB    20(R0), #3                     ; 1404
                                                 06  13 000F8      BEQL    8$                             ;
                                     04      14  A0  91 000FA      CMPB    20(R0), #4                     ; 1405
                                                 27  12 000FE      BNEQ    9$                             ;
                                     53      18  A0  D0 00100 8$:  MOVL    24(R0), OLD_ADDR               ; 1408
                                 08  AE          50  D0 00104      MOVL    R0, DUMMY                      ; 1409
                                             08  AE  9F 00108      PUSHAB  DUMMY                          ; 1410
                                             10  AE  9F 0010B      PUSHAB  PC_END                         ;
                                             18  AE  9F 0010E      PUSHAB  PC_BEG                         ;
                                             20  AE  9F 00111      PUSHAB  STMT                           ;
                                             28  AE  9F 00114      PUSHAB  LINE                           ;
                                                 55  DD 00117      PUSHL   ADDRESS                        ;
                          00000000G  00          06  FB 00119      CALLS   #6, DBG$PC_TO_LINE_LOOKUP      ;
                                     04          50  E9 00120      BLBC    R0, 9$                         ;
                                     53      10  AE  D0 00123      MOVL    PC_BEG, OLD_ADDR               ; 1413
                                                 7E  D4 00127 9$:  CLRL    -(SP)                          ; 1426
                                                 53  DD 00129      PUSHL   OLD_ADDR                       ;
                          00000000G  00          02  FB 0012B      CALLS   #2, DBG$INS_DECODE             ;
                                     55          50  D1 00132      CMPL    NEW_ADDR, ADDRESS              ; 1427
                                                 05  1A 00135      BGTRU   10$                            ;
```

```
                    53              50  D0 00137          MOVL    NEW_ADDR, OLD_ADDR                      ; 1428
                                    EB  11 0013A          BRB     9$                                      ; 1424
                18  A2              53  D0 0013C 10$:     MOVL    OLD_ADDR, 24(R2)                        ; 1436
             64     50              53  A3 00140          SUBW3   OLD_ADDR, NEW_ADDR, (R4)                ; 1437
                02  A4              16  90 00144          MOVB    #22, 2(R4)                              ; 1438
                                    53  DD 00148          PUSHL   OLD_ADDR                                ; 1439
        00000000G   00              01  FB 0014A          CALLS   #1, DBG$IS_IT_ENTRY                     ;
                    22              50  E9 00151          BLBC    R0, 12$                                 ;
                02  A4              17  90 00154          MOVB    #23, 2(R4)                              ; 1441
                                    1C  11 00158          BRB     12$                                     ; 1378
                06  A2              03  90 0015A 11$:     MOVB    #3, 6(R2)                               ; 1451
                05  A2          F0  8F  8A 0015E          BICB2   #240, 5(R2)                             ; 1452
                03  A4              01  90 00163          MOVB    #1, 3(R4)                               ; 1453
                02  A4 00000000G   00  90 00167          MOVB    DBG$GL_DFLTTYP, 2(R4)                   ; 1454
                    64              69  B0 0016F          MOVW    DBG$GW_DFLTLENG, (R4)                   ; 1455
                18  A2              56  C2 00172          SUBL2   LENGTH, 24(R2)                          ; 1456
                    50              52  D0 00176 12$:     MOVL    R2, R0                                  ; 1463
                                        04 00179          RET                                             ; 1465

; Routine Size:  378 bytes,    Routine Base:  DBG$CODE + 0A96
```

```
;  1343          1466  1 ROUTINE MODIFY_PRIMARY(PRM_DESC: REF DBG$PRIMARY, DIRECTION) =
;  1344          1467  1 !
;  1345          1468  1 ! FUNCTION
;  1346          1469  1 !       This routine takes a Primary Descriptor and attempts to find
;  1347          1470  1 !       the logical successor (DIRECTION=0) or logical predecessor
;  1348          1471  1 !       (DIRECTION=1). For example, the Primary for X(0) might
;  1349          1472  1 !       be modified to be a Primary for X(1).
;  1350          1473  1 !
;  1351          1474  1 ! INPUTS
;  1352          1475  1 !       PRM_DESC - points to a Primary Descriptor
;  1353          1476  1 !       DIRECTION - if 0, we want the logical successor.
;  1354          1477  1 !                   if 1, we want the logical predecessor.
;  1355          1478  1 !
;  1356          1479  1 ! OUTPUTS
;  1357          1480  1 !       1 is returned if this routine is successful, and
;  1358          1481  1 !       the Primary pointed to by PRM_DESC is modified.
;  1359          1482  1 !       If this routine fails, it returns 0 or 2.
;  1360          1483  1 !
;  1361          1484  1
;  1362          1485  2     BEGIN
;  1363          1486  2
;  1364          1487  2     MAP
;  1365          1488  2         PRM_DESC: REF DBG$PRIMARY;      ! Pointer to input Primary Descriptor
;  1366          1489  2
;  1367          1490  2     BUILTIN
;  1368          1491  2         INSQUE,
;  1369          1492  2         REMQUE;
;  1370          1493  2
;  1371          1494  2     LABEL
;  1372          1495  2         PASS,
;  1373          1496  2         SCAN;
;  1374          1497  2
;  1375          1498  2     LOCAL
;  1376          1499  2         COMP_FLAG: BYTE,
;  1377          1500  2         ERROR_STATUS,
;  1378          1501  2         DUMMY,
;  1379          1502  2         ROOT_ADR,
;  1380          1503  2         SUB_NODE: REF DBG$PRIM_NODE,
;  1381          1504  2         SYM_NAME: REF VECTOR[,BYTE];
;  1382          1505  2
;  1383          1506  2
;  1384          1507  2
;  1385          1508  2     ! Save away the "current" primary. This is used by our stack
;  1386          1509  2     ! machine code in RSTACCESS to evaluate the "push inner record
;  1387          1510  2     ! address" and "push outer record address" instructions.
;  1388          1511  2     !
;  1389          1512  2     DBG$GL_CURRENT_PRIMARY = .PRM_DESC;
;  1390          1513  2
;  1391          1514  2     ! Give up if the descriptor is not a Primary we can modify
;  1392          1515  2     ! to get logical successor or predecessor.
;  1393          1516  2     !
;  1394          1517  2     IF (.PRM_DESC[DBG$B_DHDR_TYPE] NEQ DBG$K_PRIMARY_DESC) OR
;  1395          1518  3        ((.PRM_DESC[DBG$B_DHDR_KIND] NEQ RST$K_DATA) AND
;  1396          1519  2         (.PRM_DESC[DBG$B_DHDR_KIND] NEQ RST$K_TYPCOMP)) OR
;  1397          1520  3        (.PRM_DESC[DBG$V_DHDR_SUBREF])
;  1398          1521  2     THEN
;  1399          1522  2         RETURN 0;
```

```
;  1400          1523  2
;  1401          1524  2      ERROR_STATUS = 0;
;  1402          1525  2
;  1403          1526  2      WHILE TRUE DO
;  1404          1527  3  PASS:    BEGIN
;  1405          1528  3
;  1406          1529  3
;  1407          1530  3          ! This loop steps to the last/next component in the data aggregate.
;  1408          1531  3          ! In most cases this loop is executed exactly once. However to deal
;  1409          1532  3          ! properly with variant records it is sometimes necessary to repeat
;  1410          1533  3          ! this entire process until we find a valid component.  Examples of
;  1411          1534  3          ! invalid components are :
;  1412          1535  3          !       An anonymous (zero-length) PASCAL Tag Field
;  1413          1536  3          !       A VARIANT SET with no TAG, or an out-of-range Tag Value
;  1414          1537  3          !
;  1415          1538  3          ROOT_ADR = SUB_NODE = PRM_DESC[DBG$A_PRIM_FLINK];
;  1416          1539  4  SCAN:        BEGIN
;  1417          1540  4              !+
;  1418          1541  4              ! Scan the entire primary backwards until we find a group
;  1419          1542  4              ! item (Array, Record or Variant) which has more components
;  1420          1543  4              ! left in it.
;  1421          1544  4              !-
;  1422          1545  4              WHILE (SUB_NODE = .SUB_NODE[DBG$L_PNODE_BLINK]) NEQ .ROOT_ADR
;  1423          1546  4               DO IF .SUB_NODE[DBG$V_PNODE_EVAL] THEN
;  1424          1547  4                SELECTONE .SUB_NODE[DBG$B_PNODE_FCODE] OF
;  1425          1548  4                  SET
;  1426          1549  4                  [RST$K_TYPE_ARRAY]:
;  1427          1550  5                      BEGIN
;  1428          1551  5                      LOCAL S,S_VECT      : REF DBG$PRIM_NODE_SUBS;
;  1429          1552  5
;  1430          1553  5                      ! The following code increments or decrements the
;  1431          1554  5                      ! subscript vector. This must be incremented like
;  1432          1555  5                      ! an odometer in a car; that is, normally we
;  1433          1556  5                      ! just increment the last one, but if that is
;  1434          1557  5                      ! at its upper limit then set it back to the lower
;  1435          1558  5                      ! limit and attempt to increment the next one, and
;  1436          1559  5                      ! so on. We also have to take into account arrays
;  1437          1560  5                      ! that are stored in column order (first subscript
;  1438          1561  5                      ! varies fastest).
;  1439          1562  5                      !
;  1440          1563  5                      ERROR_STATUS = 2;
;  1441          1564  5                      S_VECT = SUB_NODE[DBG$A_PNARR_SVECTOR];
;  1442          1565  5                      INCR DIMENSION FROM 1 TO .SUB_NODE[DBG$B_PNARR_DIMCNT] DO
;  1443          1566  6                          BEGIN
;  1444          1567  7                          S = (IF .SUB_NODE[DBG$V_PNARR_COLUMN]
;  1445          1568  7                                  THEN .DIMENSION - 1
;  1446          1569  6                                  ELSE .SUB_NODE[DBG$B_PNARR_DIMCNT] - .DIMENSION);
;  1447          1570  6
;  1448          1571  6                          IF .DIRECTION EQL 0             ! 0 = NEXTLOC, 1 = PREVLOC
;  1449          1572  6                          THEN
;  1450          1573  6
;  1451          1574  6                              ! Logical successor.
;  1452          1575  6                              !
;  1453          1576  7                              BEGIN
;  1454          1577  7
;  1455          1578  7                              ! Check for being at the upper bound.
;  1456          1579  7                              !
```

```
;  1457          1580  7
;  1458          1581  7
;  1459          1582  8
;  1460          1583  8
;  1461          1584  8
;  1462          1585  8
;  1463          1586  8
;  1464          1587  8
;  1465          1588  8
;  1466          1589  8
;  1467          1590  8
;  1468          1591  8
;  1469          1592  8
;  1470          1593  8
;  1471          1594  8
;  1472          1595  8
;  1473          1596  8
;  1474          1597  8
;  1475          1598  8
;  1476          1599  8
;  1477          1600  8
;  1478          1601  8
;  1479          1602  8
;  1480          1603  8
;  1481          1604  8
;  1482          1605  8
;  1483          1606  8
;  1484          1607  8
;  1485          1608  9
;  1486          1609  8
;  1487          1610  9
;  1488          1611  9
;  1489          1612  9
;  1490          1613  9
;  1491          1614  9
;  1492          1615  8
;  1493          1616  8
;  1494          1617  8
;  1495          1618  8
;  1496          1619  8
;  1497          1620  8
;  1498          1621  7
;  1499          1622  8
;  1500          1623  8
;  1501          1624  8
;  1502          1625  8
;  1503          1626  8
;  1504          1627  8
;  1505          1628  7
;  1506          1629  7
;  1507          1630  6
;  1508          1631  6
;  1509          1632  6
;  1510          1633  6
;  1511          1634  7
;  1512          1635  7
;  1513          1636  7

IF .S_VECT[.S,DBG$L_PNSUB_SVALUE] EQL .S_VECT[.S,DBG$L_PNSUB_UBOUND]
THEN
    BEGIN

    ! If we have no more dimensions and we
    ! are at the top subnode (i.e., there are
    ! no "higher" levels at which we can
    ! increment something) then go ahead
    ! and increment it, (but giving a warning
    ! that we are at the upper bound).
    ! For example, if X is a one-dimensional
    ! array from 1 to 3, and we want the logical
    ! successor of X(3), we'll go ahead and return
    ! X(4) but we'll give an informational saying
    ! you have walked past the upper bound.
    ! But if X were 2-dimensional, say 1:3 by 1:3,
    ! and you want the successor of X(1,3),
    ! then return X(2,1) and not X(1,4).
    ! Or if X were a record of arrays,
    ! and X.A(3) was the upper bound, then
    ! you would want to go to the next
    ! record component, say X.B, instead of
    ! going to X.A(4).
    ! That is the reason for the checks for
    ! DIMENSION EQL DIMCNT and BLINK EQL
    ! ROOT_ADR.
    !

    IF (.DIMENSION EQL .SUB_NODE[DBG$B_PNARR_DIMCNT]) AND
       (.SUB_NODE[DBG$L_PNODE_BLINK] EQL .ROOT_ADR)
    THEN
        BEGIN
        SIGNAL(DBG$_SUBSCRNG, 3, UPLIT BYTE(%ASCIC 'upper'), .DIMENSION, .S_VECT
        S_VECT[.S,DBG$L_PNSUB_SVALUE]  =  .S_VECT[.S,DBG$L_PNSUB_SVALUE] + 1;
        LEAVE SCAN;
        END
    ELSE

        ! Set back to lower bound.
        !
        S_VECT[.S,DBG$L_PNSUB_SVALUE]  =  .S_VECT[.S,DBG$L_PNSUB_LBOUND]
    END
ELSE
    BEGIN

    ! Increment and leave loop.
    !
    S_VECT[.S,DBG$L_PNSUB_SVALUE]  =  .S_VECT[.S,DBG$L_PNSUB_SVALUE] + 1;
    LEAVE SCAN;
    END;
END
ELSE

    ! Logical predecessor.
    !
    BEGIN
    IF .S_VECT[.S,DBG$L_PNSUB_SVALUE] EQL .S_VECT[.S,DBG$L_PNSUB_LBOUND]
    THEN
```

```
;  1514          1637  8
;  1515          1638  8
;  1516          1639  8
;  1517          1640  8
;  1518          1641  8
;  1519          1642  8
;  1520          1643  8
;  1521          1644  8
;  1522          1645  8
;  1523          1646  8
;  1524          1647  8
;  1525          1648  8
;  1526          1649  8
;  1527          1650  8
;  1528          1651  8
;  1529          1652  8
;  1530          1653  8
;  1531          1654  9
;  1532          1655  8
;  1533          1656  9
;  1534          1657  9
;  1535          1658  9
;  1536          1659  9
;  1537          1660  9
;  1538          1661  8
;  1539          1662  8
;  1540          1663  8
;  1541          1664  8
;  1542          1665  8
;  1543          1666  8
;  1544          1667  7
;  1545          1668  8
;  1546          1669  8
;  1547          1670  8
;  1548          1671  8
;  1549          1672  8
;  1550          1673  8
;  1551          1674  7
;  1552          1675  6
;  1553          1676  5
;  1554          1677  4
;  1555          1678  4
;  1556          1679  4
;  1557          1680  5
;  1558          1681  5
;  1559          1682  5
;  1560          1683  5
;  1561          1684  5
;  1562          1685  5
;  1563          1686  5
;  1564          1687  6
;  1565          1688  6
;  1566          1689  6
;  1567          1690  6
;  1568          1691  6
;  1569          1692  6
;  1570          1693  6
```

```
BEGIN

    ! If we have no more dimensions then go ahead
    ! and decrement it, (but giving a warning
    ! that we are at the lower bound).
    ! For example, if X is a one-dimensional
    ! array from 1 to 3, and we want the logical
    ! predecessor of X(1), we'll go ahead and return
    ! X(0) but we'll give an informational saying
    ! you have walked past the upper bound.
    ! But if X were 2-dimensional, say 1:3 by 1:3,
    ! and you want the predecessor of X(3,1),
    ! then return X(2,3) and not X(3,0).
    ! That is the reason for this check for
    ! DIMENSION EQL DIMCNT.
    !

    IF (.DIMENSION EQL .SUB_NODE[DBG$B_PNARR_DIMCNT]) AND
       (.SUB_NODE[DBG$L_PNODE_BLINK] EQL .ROOT_ADR)
    THEN
        BEGIN
        SIGNAL(DBG$_SUBSCRNG, 3, UPLIT BYTE(%ASCIC 'lower'), .DIMENSION, .S_VECT
        S_VECT[.S,DBG$L_PNSUB_SVALUE]  =  .S_VECT[.S,DBG$L_PNSUB_SVALUE] - 1;
        LEAVE SCAN;
        END
    ELSE

        ! Set back to upper bound.
        !
        S_VECT[.S,DBG$L_PNSUB_SVALUE]  =  .S_VECT[.S,DBG$L_PNSUB_UBOUND]
        END
    ELSE
        BEGIN

        ! Decrement and leave loop.
        !
        S_VECT[.S,DBG$L_PNSUB_SVALUE]  =  .S_VECT[.S,DBG$L_PNSUB_SVALUE] - 1;
        LEAVE SCAN;
        END;
    END;
END;

[RST$K_TYPE_RECORD,RST$K_TYPE_VARIANT]:
    BEGIN
    ERROR_STATUS = 2;
    IF .DIRECTION EQL 0          ! 0 = NEXTLOC, 1 = PREVLOC
    THEN

        ! Logical successor.
        !
        BEGIN

        ! If we can go to the next component, do so and exit
        ! the loop.
        !
        IF .SUB_NODE[DBG$W_PNREC_INDEX] LSSU .SUB_NODE[DBG$W_PNREC_NCOMPS]
        THEN
```

```
;  1571     1694  7                    BEGIN
;  1572     1695  7                    SUB_NODE[DBG$W_PNREC_INDEX] = .SUB_NODE[DBG$W_PNREC_INDEX] + 1;
;  1573     1696  7                    LEAVE SCAN;
;  1574     1697  6                    END;
;  1575     1698  6                END
;  1576     1699  5            ELSE
;  1577     1700  5
;  1578     1701  5                ! Logical predecessor.
;  1579     1702  5                !
;  1580     1703  6                BEGIN
;  1581     1704  6
;  1582     1705  6                ! If we can get to the previous component, do so
;  1583     1706  6                ! and exit the loop.
;  1584     1707  6                !
;  1585     1708  6                IF .SUB_NODE[DBG$W_PNREC_INDEX] GTRU 1
;  1586     1709  6                THEN
;  1587     1710  7                    BEGIN
;  1588     1711  7                    SUB_NODE[DBG$W_PNREC_INDEX] = .SUB_NODE[DBG$W_PNREC_INDEX] - 1;
;  1589     1712  7                    LEAVE SCAN;
;  1590     1713  7                    END
;  1591     1714  5                END;
;  1592     1715  4            END;
;  1593     1716  4        [OTHERWISE]:0;
;  1594     1717  4        TES;
;  1595     1718  4
;  1596     1719  4        ! If we fall through to here without succeeding in incrementing
;  1597     1720  4        ! or decrementing anything, then error status will still be
;  1598     1721  4        ! 0 or 2 and we return it, indicating we did not succeed.
;  1599     1722  4        !
;  1600     1723  4        RETURN .ERROR_STATUS;
;  1601     1724  3        END;                        ! End of block scan
;  1602     1725  3
;  1603     1726  3    !+
;  1604     1727  3    ! The following test is a special case so that if the last item examined
;  1605     1728  3    ! was an array element we just examine the next (or previous) element of
;  1606     1729  3    ! the array, even if this is an aggregate (e.g. a record).  In all other
;  1607     1730  3    ! cases we will step down to an individual component of the aggregate.
;  1608     1731  3    !-
;  1609     1732  4    IF (.SUB_NODE[DBG$L_PNODE_FLINK] EQLA .PRM_DESC[DBG$L_PRIM_BLINK])
;  1610     1733  3                                AND
;  1611     1734  4       (.SUB_NODE[DBG$B_PNODE_FCODE] EQL RST$K_TYPE_ARRAY)
;  1612     1735  4
;  1613     1736  4      THEN RETURN 1;
;  1614     1737  4
;  1615     1738  4
;  1616     1739  4    ! We have found the composite entry we are going to modify. Strip off
;  1617     1740  4    ! all subsequent primary sub-nodes, and then add new sub_nodes to get
;  1618     1741  4    ! a primary which describes a single data item.
;  1619     1742  4    !
;  1620     1743  4    WHILE .SUB_NODE[DBG$L_PNODE_FLINK] NEQA .ROOT_ADR DO
;  1621     1744  4        REMQUE(.SUB_NODE[DBG$L_PNODE_FLINK],DUMMY);
;  1622     1745  4
;  1623     1746  4    IF .PRM_DESC[DBG$V_DHDR_TMPREF] THEN
;  1624     1747  4        BEGIN
;  1625     1748  4        PRM_DESC[DBG$V_DHDR_TMPREF] = FALSE;
;  1626     1749  4        PRM_DESC[DBG$V_DHDR_SUBREF] = FALSE;
;  1627     1750  4        PRM_DESC[DBG$W_PRIM_OFFSET] = 0;
```

```
;  1628          1751  4              PRM_DESC[DBG$W_PRIM_LENGTH] = 0;
;  1629          1752  3              END;
;  1630          1753  3
;  1631          1754  3          COMP_FLAG = FALSE;
;  1632          1755  3
;  1633          1756  3          WHILE TRUE DO
;  1634          1757  4              BEGIN
;  1635          1758  4              LOCAL SYMID,TYPEID,FCODE,KIND;
;  1636          1759  4              SELECTONE .SUB_NODE[DBG$B_PNODE_FCODE] OF
;  1637          1760  4                  SET
;  1638          1761  4                  [RST$K_TYPE_ARRAY]:
;  1639          1762  5                      BEGIN
;  1640          1763  5                      IF .COMP_FLAG THEN
;  1641          1764  6                          BEGIN
;  1642          1765  6                          LOCAL S_VECT : REF DBG$PRIM_NODE_SUBS;
;  1643          1766  6                          S_VECT = SUB_NODE[DBG$A_PNARR_SVECTOR];
;  1644          1767  6                          DECR S FROM .SUB_NODE[DBG$B_PNARR_DIMCNT]-1 TO 0 DO
;  1645          1768  6                            S_VECT[.S,DBG$L_PNSUB_SVA[UE]  =  .S_VECT[.S,DBG$L_PNSUB_UBOUND];
;  1646          1769  5                          END;
;  1647          1770  5                      SYMID = 0;
;  1648          1771  5                      KIND = RST$K_DATA;
;  1649          1772  5                      DBG$STA_SYMTYPE(.SUB_NODE[DBG$L_PNARR_CELLTYPE],FCODE,TYPEID);
;  1650          1773  4                      END;
;  1651          1774  4
;  1652          1775  4                  [RST$K_TYPE_RECORD,RST$K_TYPE_VARIANT]:
;  1653          1776  5                      BEGIN
;  1654          1777  5                      LOCAL N_COMPS,S_VECT : REF VECTOR[,LONG];
;  1655          1778  5                      IF .SUB_NODE[DBG$B_PNODE_FCODE] EQL RST$K_TYPE_RECORD
;  1656          1779  5                        THEN DBG$STA_TYP_RECORD(.SUB_NODE[DBG$L_PNODE_TYPEID],N_COMPS,S_VECT,DUMMY)
;  1657          1780  5                        ELSE
;  1658          1781  6                          BEGIN
;  1659          1782  6                          !+
;  1660          1783  6                          ! We have a variant set.  If the TYPEID field is non-zero this is
;  1661          1784  6                          ! a new variant sub node (added by us on the last pass), and so we
;  1662          1785  6                          ! need to obtain the current value of the TAG field. A zero-length
;  1663          1786  6                          ! tag field (or an illegal tag value) cause this entire variant to
;  1664          1787  6                          ! be ignored.
;  1665          1788  6                          !
;  1666          1789  6                          IF (TYPEID = .SUB_NODE[DBG$L_PNODE_TYPEID]) NEQ 0 THEN
;  1667          1790  7                              BEGIN
;  1668          1791  7                              LOCAL TAG,MARK,TYPE,CODE,
;  1669          1792  7                                    VAL_DESC : REF DBG$VALDESC,
;  1670          1793  7                                    VARIANT  : REF RST$VAR_ENTRY;
;  1671          1794  7                              MAP   TYPEID   : REF RST$ENTRY;
;  1672          1795  7
;  1673          1796  7                              REMQUE(.SUB_NODE,SUB_NODE);
;  1674          1797  7                              IF (TAG  = .TYPEID[RST$L_VARTAGPTR]) EQL 0 THEN LEAVE PASS;
;  1675          1798  7                              DBG$STA_SYMNAME(.TAG,SYM_NAME);
;  1676          1799  7                              IF .SYM_NAME[0] EQL 0 THEN LEAVE PASS;
;  1677          1800  7                              !+
;  1678          1801  7                              ! We now need to obtain the actual tag value. This is done
;  1679          1802  7                              ! by stripping off the VARIANT sub-node,  and adding a new
;  1680          1803  7                              ! primary sub-node describing the Tag field.  We then call
;  1681          1804  7                              ! dbg$prim_to_val to get a Debug Value descriptor, extract
;  1682          1805  7                              ! the tag value, and restore the state of the primary.
;  1683          1806  7                              !-
;  1684          1807  7                              MARK = DBG$PUSH_TEMPMEM();
```

```
;  1685          1808  7                    DBG$STA_SYMTYPE(.TAG,CODE,TYPE);
;  1686          1809  7                    DBG$BUILD_PRIMARY_SUBNODE(.PRM_DESC,RST$K_DATA,,.TAG,,.CODE,,.TYPE,0);
;  1687          1810  7                    DBG$PRIM_TO_VAL(.PRM_DESC,DBG$K_VALUE_DESC,VAL_DESC);
;  1688          1811  7                    TAG = .VAL_DESC[DBG$L_VALUE_VALUE0];
;  1689          1812  7                    REMQUE(.PRM_DESC[DBG$L_PRIM_BLINK],DUMMY);
;  1690          1813  7                    DBG$POP_TEMPMEM(.MARK);
;  1691          1814  7                    IF (VARIANT = DBG$STA_VARIANT_SELECT(.TAG,,.TYPEID)) EQL 0 THEN LEAVE PASS;
;  1692          1815  7                    SUB_NODE[DBG$V_PNVAR_VALID]   = TRUE;
;  1693          1816  7                    SUB_NODE[DBG$W_PNVAR_INDEX]   = 1;
;  1694          1817  7                    SUB_NODE[DBG$L_PNODE_TYPEID]  = 0;
;  1695          1818  7                    SUB_NODE[DBG$L_PNVAR_TAGID]   =  .TYPEID[RST$L_VARTAGPTR];
;  1696          1819  7                    SUB_NODE[DBG$W_PNVAR_NCOMPS]  = .VARIANT[RST$L_VAR_COMPCNT];
;  1697          1820  7                    SUB_NODE[DBG$L_PNVAR_COMPLST] =  VARIANT[RST$A_VAR_COMPLST];
;  1698          1821  7                    SUB_NODE[DBG$L_PNVAR_DSTPTR]  = .VARIANT[RST$L_VAR_DSTPTR];
;  1699          1822  7                    INSQUE(.SUB_NODE,,.PRM_DESC[DBG$L_PRIM_BLINK]);
;  1700          1823  6                    END;
;  1701          1824  6                N_COMPS = .SUB_NODE[DBG$W_PNVAR_NCOMPS];
;  1702          1825  6                S_VECT  = .SUB_NODE[DBG$L_PNVAR_COMPLST];
;  1703          1826  5                END;
;  1704          1827  5            IF .COMP_FLAG THEN SUB_NODE[DBG$W_PNREC_INDEX] = .N_COMPS;
;  1705          1828  5            SYMID = .S_VECT[.SUB_NODE[DBG$W_PNREC_INDEX]-1];
;  1706          1829  5            DBG$STA_SYMKIND(.SYMID,KIND);
;  1707          1830  5            IF .KIND EQL RST$K_VARIANT
;  1708          1831  5              THEN
;  1709          1832  6                BEGIN
;  1710          1833  6                FCODE  = RST$K_TYPE_VARIANT;
;  1711          1834  6                TYPEID = .SYMID;
;  1712          1835  6                SYMID  = 0;
;  1713          1836  6                END
;  1714          1837  5              ELSE
;  1715          1838  5                DBG$STA_SYMTYPE(.SYMID,FCODE,TYPEID);
;  1716          1839  4            END;
;  1717          1840  4
;  1718          1841  4                [OTHERWISE]:
;  1719          1842  4                    EXITLOOP;
;  1720          1843  4                TES;
;  1721          1844  4            SUB_NODE[DBG$V_PNODE_EVAL] = TRUE;
;  1722          1845  4            DBG$BUILD_PRIMARY_SUBNODE(.PRM_DESC,,.KIND,,.SYMID,,.FCODE,,.TYPEID, 0);
;  1723          1846  4            SUB_NODE = .PRM_DESC[DBG$L_PRIM_BLINK];
;  1724          1847  4            COMP_FLAG = .DIRECTION;
;  1725          1848  3            END;
;  1726          1849  3
;  1727          1850  3        IF .SUB_NODE[DBG$L_PNODE_SYMID] EQL 0 THEN EXITLOOP;
;  1728          1851  3        DBG$STA_SYMNAME(.SUB_NODE[DBG$L_PNODE_SYMID],SYM_NAME);
;  1729          1852  3        IF .SYM_NAME[0] NEQ 0 THEN EXITLOOP;
;  1730          1853  2        END;
;  1731          1854  2
;  1732          1855  2    RETURN 1;
;  1733          1856  1    END;                          ! End of modify_primary


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

                              72  65  70  70  75  05  0003E P.AAJ:  .ASCII  <5>\upper\                                ;
                              72  65  77  6F  6C  05  00044 P.AAK:  .ASCII  <5>\lower\                                ;
```

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

                                            0FFC 00000 MODIFY_PRIMARY:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1466
                5B 00000000G    00    9E 00002        MOVAB   LIB$SIGNAL, R11                             ;
                5E                    28 C2 00009       SUBL2   #40, SP                                   ;
                50        04    AC    D0 0000C        MOVL    PRM_DESC, R0                                ; 1512
     00000000G  00              50    D0 00010        MOVL    R0, DBG$GL_CURRENT_PRIMARY                  ;
             79 8F         02   A0    91 00017        CMPB    2(R0), #12T                                 ; 1517
                               11    12 0001C        BNEQ    2$                                          ;
                06         07   A0    91 0001E        CMPB    7(R0), #6                                   ; 1518
                               06    13 00022        BEQL    1$                                          ;
                0A         07   A0    91 00024        CMPB    7(R0), #10                                  ; 1519
                               05    12 00028        BNEQ    2$                                          ;
          03        04    A0          01 E1 0002A 1$:   BBC     #1, 4(R0), 3$                             ; 1520
                            02DC      31 0002F 2$:    BRW     45$                                         ;
                              59      D4 00032 3$:    CLRL    ERROR_STATUS                                ; 1524
          53        04    AC          14 C1 00034 4$:   ADDL3   #20, PRM_DESC, SUB_NODE                   ; 1538
                58                    53 D0 00039     MOVL    SUB_NODE, ROOT_ADR                          ;
                53        04    A3    D0 0003C 5$:    MOVL    4(SUB_NODE), SUB_NODE                       ; 1545
                58                    53 D1 00040     CMPL    SUB_NODE, ROOT_ADR                          ;
                                      03 12 00043     BNEQ    6$                                          ;
                                  00E3  31 00045        BRW     22$                                       ;
                F0        0A    A3    E9 00048 6$:    BLBC    10(SUB_NODE), 5$                            ; 1546
                50        09    A3    9A 0004C        MOVZBL  9(SUB_NODE), R0                             ; 1547
                01                    50 91 00050     CMPB    R0, #T                                      ; 1549
                                      03 13 00053     BEQL    7$                                          ;
                                  00AA  31 00055        BRW     19$                                       ;
                59                    02 D0 00058 7$:   MOVL    #2, ERROR_STATUS                            ; 1563
                52        28    A3    9E 0005B        MOVAB   40(R3), S_VECT                              ; 1564
                57        1B    A3    9A 0005F        MOVZBL  27(SUB_NODE), R7                            ; 1565
                                      54 D4 00063     CLRL    DIMENSION                                   ; 1580
                                  0091  31 00065        BRW     17$                                       ;
          06        0A    A3          01 E1 00068 8$:   BBC     #1, 10(SUB_NODE), 9$                      ; 1567
                55        FF    A4    9E 0006D        MOVAB   -1(R4), S                                   ; 1568
                                      04 11 00071     BRB     10$                                         ;
          55                  57      54 C3 00073 9$:   SUBL3   DIMENSION, R7, S                          ; 1569
          50                  55      14 C5 00077 10$:  MULL3   #20, S, R0                                ; 1580
          56                  52      50 C1 0007B     ADDL3   R0, S_VECT, R6                              ;
                                08    AC    D5 0007F        TSTL    DIRECTION                             ; 1571
                                      39 12 00082     BNEQ    13$                                         ;
                                0C A042     9F 00084        PUSHAB  12(R0)[S_VECT]                        ; 1580
                             9E           66 D1 00088     CMPL    (R6), @(SP)+                            ;
                                      2C 12 0008B     BNEQ    12$                                         ;
                             57           54 D1 0008D     CMPL    DIMENSION, R7                           ; 1607
                                      21 12 00090     BNEQ    11$                                         ;
                             58   04    A3    D1 00092        CMPL    4(SUB_NODE), ROOT_ADR               ; 1608
                                      1B 12 00096     BNEQ    11$                                         ;
                                0C A042     9F 00098        PUSHAB  12(R0)[S_VECT]                        ; 1611
                                   9E       DD 0009C        PUSHL   @(SP)+                                ;
                                   54       DD 0009E        PUSHL   DIMENSION                             ;
                      00000000'    EF       9F 000A0        PUSHAB  P.AAJ                                 ;
                                      03    DD 000A6        PUSHL   #3                                    ;
                      000287AB        8F    DD 000A8        PUSHL   #165803                               ;
                             6B           05 FB 000AE        CALLS   #5, LIB$SIGNAL                        ;
```

```
                                   06  11 000B1       BRB     12$                                   ; 1612
                           08 A042 9F  000B3 11$:     PUSHAB  8(R0)[S_VECT]                         ; 1619
                                   37  11 000B7       BRB     15$                                   ;
                                   66  D6 000B9 12$:  INCL    (R6)                                  ; 1626
                                   72  11 000BB       BRB     23$                                   ; 1627
                           08 A042 9F  000BD 13$:     PUSHAB  8(R0)[S_VECT]                         ; 1635
                   9E              66  D1 000C1       CMPL    (R6), @(SP)+                          ;
                                   2F  12 000C4       BNEQ    16$                                   ;
                   57              54  D1 000C6       CMPL    DIMENSION, R7                         ; 1653
                                   21  12 000C9       BNEQ    14$                                   ;
                   58         04   A3  D1 000CB       CMPL    4(SUB_NODE), ROOT_ADR                 ; 1654
                                   1B  12 000CF       BNEQ    14$                                   ;
                           0C A042 9F  000D1          PUSHAB  12(R0)[S_VECT]                        ; 1657
                                   9E  DD 000D5       PUSHL   @(SP)+                                ;
                                   54  DD 000D7       PUSHL   DIMENSION                             ;
                      00000000'    EF  9F 000D9       PUSHAB  P.AAK                                 ;
                                   03  DD 000DF       PUSHL   #3                                    ;
                      000287AB     8F  DD 000E1       PUSHL   #165803                               ;
                   6B              05  FB 000E7       CALLS   #5, LIB$SIGNAL                        ;
                                   09  11 000EA       BRB     16$                                   ; 1658
                           0C A042 9F  000EC 14$:     PUSHAB  12(R0)[S_VECT]                        ; 1665
                   66              9E  D0 000F0 15$:  MOVL    @(SP)+, (R6)                          ;
                                   04  11 000F3       BRB     17$                                   ; 1637
                                   66  D7 000F5 16$:  DECL    (R6)                                  ; 1672
                                   36  11 000F7       BRB     23$                                   ; 1673
  FF69          54 01              57  F1 000F9 17$:  ACBL    R7, #1, DIMENSION, 8$                 ; 1565
                                 FF3A  31 000FF 18$:  BRW     5$                                    ; 1547
                   07              50  91 00102 19$:  CMPB    R0, #7                                ; 1679
                                   05  13 00105       BEQL    20$                                   ;
                   13              50  91 00107       CMPB    R0, #19                               ;
                                   F3  12 0010A       BNEQ    18$                                   ;
                   59              02  D0 0010C 20$:  MOVL    #2, ERROR_STATUS                      ; 1681
                   50         18   A3  9E 0010F       MOVAB   24(SUB_NODE), R0                      ; 1692
                              08   AC  D5 00113       TSTL    DIRECTION                             ; 1682
                                   0A  12 00116       BNEQ    21$                                   ;
               02  A0              60  B1 00118       CMPW    (R0), 2(R0)                           ; 1692
                                   E1  1E 0011C       BGEQU   18$                                   ;
                                   60  B6 0011E       INCW    (R0)                                  ; 1695
                                   0D  11 00120       BRB     23$                                   ; 1696
                   01              60  B1 00122 21$:  CMPW    (R0), #1                              ; 1708
                                   D8  1B 00125       BLEQU   18$                                   ;
                                   60  B7 00127       DECW    (R0)                                  ; 1711
                                   04  11 00129       BRB     23$                                   ; 1712
                   50              59  D0 0012B 22$:  MOVL    ERROR_STATUS, R0                      ; 1723
                                       04 0012E       RET                                           ;
                   50         04   AC  D0 0012F 23$:  MOVL    PRM_DESC, R0                          ; 1732
               18  A0              63  D1 00133       CMPL    (SUB_NODE), 24(R0)                    ;
                                   09  12 00137       BNEQ    24$                                   ;
                   01         09   A3  91 00139       CMPB    9(SUB_NODE), #1                       ; 1734
                                   03  12 0013D       BNEQ    24$                                   ;
                                 01C8  31 0013F       BRW     44$                                   ;
                   58              63  D1 00142 24$:  CMPL    (SUB_NODE), ROOT_ADR                  ; 1743
                                   06  13 00145       BEQL    25$                                   ;
                   6E         00   B3  0F 00147       REMQUE  @0(SUB_NODE), DUMMY                   ; 1744
                                   F5  11 0014B       BRB     24$                                   ;
                   50         04   AC  D0 0014D 25$:  MOVL    PRM_DESC, R0                          ; 1746
                   09         05   A0  E9 00151       BLBC    5(R0), 26$                            ;
```

```
                                04   A0      0102   8F   AA 00155           BICW2   #258, 4(R0)                                 ; 1748
                                      10      A0   D4 00158               CLRL    16(R0)                                      ; 1750
                                               5A   94 0015E 26$:          CLRB    COMP_FLAG                                   ; 1754
                                         50    09   A3   9A 00160 27$:     MOVZBL  9(SUB_NODE), R0                             ; 1759
                                         01         50   91 00164          CMPB    R0, #1                                      ; 1761
                                                    30   12 00167          BNEQ    31$
                                         1B         5A   E9 00169          BLBC    COMP_FLAG, 30$                              ; 1763
                                         52    28   A3   9E 0016C          MOVAB   40(R3), S_VECT                              ; 1766
                                         50    1B   A3   9A 00170          MOVZBL  27(SUB_NODE), S                             ; 1768
                                                    0E   11 00174          BRB     29$
                51                       50         14   C5 00176 28$:     MULL3   #20, S, R1                                  ;
                                               6142   9F 0017A            PUSHAB  (R1)[S_VECT]                                ;
                                            0C A142   9F 0017D            PUSHAB  12(R1)[S_VECT]                              ;
                                         9E         9E   D0 00181          MOVL    @(SP)+, @(SP)+                              ;
                                         EF         50   F4 00184 29$:     SOBGEQ  S, 28$                                      ;
                                                    55   D4 00187 30$:     CLRL    SYMID                                       ; 1770
                                04   AE            06   D0 00189          MOVL    #6, KIND                                    ; 1771
                                            1C    AE   9F 0018D            PUSHAB  TYPEID                                      ; 1772
                                            24    AE   9F 00190            PUSHAB  FCODE                                       ;
                                            24    A3   DD 00193            PUSHL   36(SUB_NODE)                                ;
                                               0127   31 00196            BRW     41$                                         ;
                                         07         50   91 00199 31$:     CMPB    R0, #7                                      ; 1775
                                                    08   13 0019C          BEQL    32$
                                         13         50   91 0019E          CMPB    R0, #19                                     ;
                                                    03   13 001A1          BEQL    32$                                         ;
                                               014A   31 001A3            BRW     43$                                         ;
                                         07         50   91 001A6 32$:     CMPB    R0, #7                                      ; 1778
                                                    15   12 001A9          BNEQ    33$
                                                    5E   DD 001AB          PUSHL   SP                                          ; 1779
                                            08    AE   9F 001AD            PUSHAB  S_VECT                                      ;
                                            10    AE   9F 001B0            PUSHAB  N_COMPS                                     ;
                                            0C    A3   DD 001B3            PUSHL   12(SUB_NODE)                                ;
00000000G                       00                 04   FB 001B6          CALLS   #4, DBG$STA_TYP_RECORD                      ;
                                               00C5   31 001BD            BRW     38$
                                1C      AE      0C   A3   D0 001C0 33$:     MOVL    12(SUB_NODE), TYPEID                        ; 1789
                                                    03   12 001C5          BNEQ    34$
                                               00B1   31 001C7            BRW     37$
                                         53         63   0F 001CA 34$:     REMQUE  (SUB_NODE), SUB_NODE                        ; 1796
                                         50    1C   AE   D0 001CD          MOVL    TYPEID, R0                                  ; 1797
                                         54    10   A0   D0 001D1          MOVL    16(R0), TAG
                                                    79   13 001D5          BEQL    35$
                                            24    AE   9F 001D7            PUSHAB  SYM_NAME                                    ; 1798
                                                    54   DD 001DA          PUSHL   TAG
00000000G                       00                 02   FB 001DC          CALLS   #2, DBG$STA_SYMNAME
                                            24    BE   95 001E3            TSTB    @SYM_NAME                                   ; 1799
                                                    68   13 001E6          BEQL    35$
00000000G                       00                 00   FB 001E8          CALLS   #0, DBG$PUSH_TEMPMEM                        ; 1807
                                         57         50   D0 001EF          MOVL    R0, MARK
                                            0C    AE   9F 001F2            PUSHAB  TYPE                                        ; 1808
                                            14    AE   9F 001F5            PUSHAB  CODE
                                                    54   DD 001F8          PUSHL   TAG
00000000G                       00                 03   FB 001FA          CALLS   #3, DBG$STA_SYMTYPE
                                                    7E   D4 00201          CLRL    -(SP)                                       ; 1809
                                            10    AE   DD 00203            PUSHL   TYPE
                                            18    AE   DD 00206            PUSHL   CODE
                                                    54   DD 00209          PUSHL   TAG
                                                    06   DD 0020B          PUSHL   #6
```

```
                52      04  AC  D0 0020D          MOVL    PRM_DESC, R2                                    ;
                            52  DD 00211          PUSHL   R2                                              ;
00000000G       00          06  FB 00213          CALLS   #6, DBG$BUILD_PRIMARY_SUBNODE                   ;
                        14  AE  9F 0021A          PUSHAB  VAL_DESC                                        ; 1810
                7E      7A  8F  9A 0021D          MOVZBL  #122, -(SP)                                     ;
                            52  DD 00221          PUSHL   R2                                              ;
00000000G       00          03  FB 00223          CALLS   #3, DBG$PRIM_TO_VAL                             ;
                50      14  AE  D0 0022A          MOVL    VAL_DESC, R0                                    ; 1811
                54      20  A0  D0 0022E          MOVL    32(R0), TAG                                     ;
                6E      18  B2  0F 00232          REMQUE  @24(R2), DUMMY                                  ; 1812
                            57  DD 00236          PUSHL   MARK                                            ; 1813
00000000G       00          01  FB 00238          CALLS   #1, DBG$POP_TEMPMEM                             ;
                52      1C  AE  D0 0023F          MOVL    TYPEID, R2                                      ; 1814
                            52  DD 00243          PUSHL   R2                                              ;
                            54  DD 00245          PUSHL   TAG                                             ;
00000000G       00          02  FB 00247          CALLS   #2, DBG$STA_VARIANT_SELECT                      ;
                            50  D5 0024E          TSTL    VARIANT                                         ;
                            03  12 00250 35$:     BNEQ    36$                                             ;
                        FDDF    31 00252          BRW     4$                                              ;
          0A    A3              10  88 00255 36$:  BISB2   #16, 10(SUB_NODE)                              ; 1815
          18    A3              01  B0 00259       MOVW    #1, 24(SUB_NODE)                                ; 1816
                        0C  A3  D4 0025D          CLRL    12(SUB_NODE)                                    ; 1817
          1C    A3      10  A2  D0 00260          MOVL    16(R2), 28(SUB_NODE)                            ; 1818
          1A    A3      04  A0  B0 00265          MOVW    4(VARIANT), 26(SUB_NODE)                        ; 1819
          20    A3      08  A0  9E 0026A          MOVAB   8(R0), 32(SUB_NODE)                             ; 1820
          24    A3          60  D0 0026F          MOVL    (VARIANT), 36(SUB_NODE)                         ; 1821
                50      04  AC  D0 00273          MOVL    PRM_DESC, R0                                    ; 1822
          18    B0              63  0E 00277      INSQUE  (SUB_NODE), @24(R0)                             ;
          08    AE      1A  A3  3C 0027B 37$:     MOVZWL  26(SUB_NODE), N_COMPS                           ; 1824
          04    AE      20  A3  D0 00280          MOVL    32(SUB_NODE), S_VECT                            ; 1825
                05              5A  E9 00285 38$:  BLBC    COMP_FLAG, 39$                                  ; 1827
          18    A3      08  AE  B0 00288          MOVW    N_COMPS, 24(SUB_NODE)                           ;
                50      18  A3  3C 0028D 39$:     MOVZWL  24(SUB_NODE), R0                                ; 1828
                50      04 BE40 DE 00291          MOVAL   @S_VECT[R0], R0                                 ;
                55      FC  A0  D0 00296          MOVL    -4(R0), SYMID                                   ;
                        18  AE  9F 0029A          PUSHAB  KIND                                            ; 1829
                            55  DD 0029D          PUSHL   SYMID                                           ;
00000000G       00          02  FB 0029F          CALLS   #2, DBG$STA_SYMKIND                             ;
                0B      18  AE  D1 002A6          CMPL    KIND, #11                                       ; 1830
                            0C  12 002AA          BNEQ    40$                                             ;
          20    AE              13  D0 002AC      MOVL    #19, FCODE                                      ; 1833
          1C    AE              55  D0 002B0      MOVL    SYMID, TYPEID                                   ; 1834
                            55  D4 002B4          CLRL    SYMID                                           ; 1835
                            0F  11 002B6          BRB     42$                                             ; 1830
                        1C  AE  9F 002B8 40$:     PUSHAB  TYPEID                                          ; 1838
                        24  AE  9F 002BB          PUSHAB  FCODE                                           ;
                            55  DD 002BE          PUSHL   SYMID                                           ;
00000000G       00          03  FB 002C0 41$:     CALLS   #3, DBG$STA_SYMTYP                              ;
          0A    A3              01  88 002C7 42$:  BISB2   #1, 10(SUB_NODE)                              ; 1844
                            7E  D4 002CB          CLRL    -(SP)                                           ; 1845
                        20  AE  DD 002CD          PUSHL   TYPEID                                          ;
                        28  AE  DD 002D0          PUSHL   FCODE                                           ;
                            55  DD 002D3          PUSHL   SYMID                                           ;
                        28  AE  DD 002D5          PUSHL   KIND                                            ;
                52      04  AC  D0 002D8          MOVL    PRM_DESC, R2                                    ;
                            52  DD 002DC          PUSHL   R2                                              ;
00000000G       00          06  FB 002DE          CALLS   #6, DBG$BUILD_PRIMARY_SUBNODE                   ;
```

```
                53      18   A2  D0 002E5          MOVL    24(R2), SUB_NODE                        ; 1846
                5A      08   AC  90 002E9          MOVB    DIRECTION, COMP_FLAG                    ; 1847
                          FE70  31 002ED          BRW     27$                                     ; 1756
                        10   A3  D5 002F0 43$:     TSTL    16(SUB_NODE)                            ; 1850
                             15  13 002F3          BEQL    44$                                     ;
                        24   AE  9F 002F5          PUSHAB  SYM_NAME                                ; 1851
                        10   A3  DD 002F8          PUSHL   16(SUB_NODE)                            ;
00000000G       00           02  FB 002FB          CALLS   #2, DBG$STA_SYMNAME                     ;
                        24   BE  95 00302          TSTB    @SYM_NAME                               ; 1852
                             03  12 00305          BNEQ    44$                                     ;
                          FD2A  31 00307          BRW     4$                                      ;
                50           01  D0 0030A 44$:     MOVL    #1, R0                                  ; 1855
                                 04 0030D          RET                                             ;
                             50  D4 0030E 45$:     CLRL    R0                                      ; 1856
                                 04 00310          RET                                             ;

; Routine Size:  785 bytes,    Routine Base:  DBG$CODE + 0C10
```

```
;1735    1857  1 ROUTINE PRIMARY_ORDER(PRIM_1: REF DBG$PRIMARY, PRIM_2: REF DBG$PRIMARY) =
;1736    1858  1 !
;1737    1859  1 ! FUNCTION
;1738    1860  1 !       ----------------------------
;1739    1861  1 !
;1740    1862  1 ! INPUTS
;1741    1863  1 !       ----------------------------
;1742    1864  1 !
;1743    1865  1 ! OUTPUTS
;1744    1866  1 !       ----------------------------
;1745    1867  1 !
;1746    1868  1
;1747    1869  2     BEGIN
;1748    1870  2     LOCAL
;1749    1871  2         NODE_1  : REF DBG$PRIM_NODE,
;1750    1872  2         NODE_2  : REF DBG$PRIM_NODE,
;1751    1873  2         VALUE_1,
;1752    1874  2         VALUE_2;
;1753    1875  2
;1754    1876  2     NODE_1 = .PRIM_1[DBG$L_PRIM_FLINK];
;1755    1877  2     NODE_2 = .PRIM_2[DBG$L_PRIM_FLINK];
;1756    1878  2
;1757    1879  2     WHILE TRUE DO
;1758    1880  3         BEGIN
;1759    1881  3
;1760    1882  3         SELECTONE .NODE_1[DBG$B_PNODE_FCODE] OF
;1761    1883  3             SET
;1762    1884  3             [RST$K_TYPE_RECORD,RST$K_TYPE_VARIANT]:
;1763    1885  4                 BEGIN
;1764    1886  4                 IF .NODE_1[DBG$B_PNODE_FCODE] EQL RST$K_TYPE_VARIANT THEN
;1765    1887  4                   IF .NODE_1[DBG$L_PNVAR_DSTPTR] NEQ .NODE_2[DBG$L_PNVAR_DSTPTR]
;1766    1888  4                     THEN SIGNAL(DBG$_EXARANGE);
;1767    1889  4                 VALUE_1 = .NODE_1[DBG$W_PNREC_INDEX];
;1768    1890  4                 VALUE_2 = .NODE_2[DBG$W_PNREC_INDEX];
;1769    1891  4                 IF .VALUE_1 LSS .VALUE_2 THEN RETURN -1;
;1770    1892  4                 IF .VALUE_1 GTR .VALUE_2 THEN RETURN +1;
;1771    1893  3                 END;
;1772    1894  3
;1773    1895  3             [RST$K_TYPE_ARRAY]:
;1774    1896  4                 BEGIN
;1775    1897  4                 LOCAL
;1776    1898  4                     SUBS_1      : REF DBG$PRIM_NODE_SUBS,
;1777    1899  4                     SUBS_2      : REF DBG$PRIM_NODE_SUBS;
;1778    1900  4
;1779    1901  4                 IF .NODE_1[DBG$B_PNARR_SUBCNT] NEQ .NODE_2[DBG$B_PNARR_SUBCNT]
;1780    1902  4                   THEN SIGNAL(DBG$_EXARANGE);
;1781    1903  4
;1782    1904  4                 SUBS_1 = NODE_1[DBG$A_PNARR_SVECTOR];
;1783    1905  4                 SUBS_2 = NODE_2[DBG$A_PNARR_SVECTOR];
;1784    1906  4
;1785    1907  4                 INCR DIMENSION FROM 0 TO .NODE_1[DBG$B_PNARR_DIMCNT]-1 DO
;1786    1908  5                     BEGIN
;1787    1909  5                     LOCAL D;
;1788    1910  6                     D = (IF .NODE_1[DBG$V_PNARR_COLUMN]
;1789    1911  6                             THEN .NODE_1[DBG$B_PNARR_DIMCNT] - .DIMENSION - 1
;1790    1912  5                             ELSE .DIMENSION);
;1791    1913  5
```

```
;  1792          1914  5                    VALUE_1 = .SUBS_1[.D,DBG$L_PNSUB_SVALUE] - .SUBS_1[.D,DBG$L_PNSUB_LBOUND];
;  1793          1915  5                    VALUE_2 = .SUBS_2[.D,DBG$L_PNSUB_SVALUE] - .SUBS_2[.D,DBG$L_PNSUB_LBOUND];
;  1794          1916  5                    IF .VALUE_1 LSS .VALUE_2 THEN RETURN -1;
;  1795          1917  5                    IF .VALUE_1 GTR .VALUE_2 THEN RETURN +1;
;  1796          1918  4                    END;
;  1797          1919  3                END;
;  1798          1920  3
;  1799          1921  3                [OTHERWISE]:
;  1800          1922  3                    EXITLOOP;
;  1801          1923  3                TES;
;  1802          1924  3
;  1803          1925  3            NODE_1 = .NODE_1[DBG$L_PNODE_FLINK];
;  1804          1926  3            NODE_2 = .NODE_2[DBG$L_PNODE_FLINK];
;  1805          1927  2            END;
;  1806          1928  2        RETURN 0;
;  1807          1929  1        END;                          ! End of primary_order


                                03FC 00000 PRIMARY_ORDER:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                    ; 1857
                59 00000000G    00  9E 00002           MOVAB   LIB$SIGNAL, R9                      ;
                50         04   AC  D0 00009           MOVL    PRIM_1, R0                          ; 1876
                52         14   A0  D0 0000D           MOVL    20(R0), NODE_1                      ;
                50         08   AC  D0 00011           MOVL    PRIM_2, R0                          ; 1877
                53         14   A0  D0 00015           MOVL    20(R0), NODE_2                      ;
                50         09   A2  9A 00019 1$:       MOVZBL  9(NODE_1), R0                       ; 1882
                07              50  91 0001D           CMPB    R0, #7                              ; 1884
                                05  13 00020           BEQL    2$                                  ;
                13              50  91 00022           CMPB    R0, #19                             ;
                                26  12 00025           BNEQ    4$                                  ;
                13              50  91 00027 2$:       CMPB    R0, #19                             ; 1886
                                10  12 0002A           BNEQ    3$                                  ;
                24   A3         24  A2  D1 0002C           CMPL    36(NODE_1), 36(NODE_2)          ; 1887
                                09  13 00031           BEQL    3$                                  ;
                     00028190   8F  DD 00033           PUSHL   #164240                             ; 1888
                69              01  FB 00039           CALLS   #1, LIB$SIGNAL                      ;
                57         18   A2  3C 0003C 3$:       MOVZWL  24(NODE_1), VALUE_1                 ; 1889
                56         18   A3  3C 00040           MOVZWL  24(NODE_2), VALUE_2                 ; 1890
                56              57  D1 00044           CMPL    VALUE_1, VALUE_2                    ; 1891
                                5B  19 00047           BLSS    9$                                  ;
                                67  15 00049           BLEQ    13$                                 ; 1892
                                5D  11 0004B           BRB     11$                                 ;
                01              50  91 0004D 4$:       CMPB    R0, #1                              ; 1895
                                69  12 00050           BNEQ    14$                                 ;
                1F   A3         1F  A2  91 00052           CMPB    31(NODE_1), 31(NODE_2)          ; 1901
                                09  13 00057           BEQL    5$                                  ;
                     00028190   8F  DD 00059           PUSHL   #164240                             ; 1902
                69              01  FB 0005F           CALLS   #1  LIB$SIGNAL                      ;
                51         28   A2  9E 00062 5$:       MOVAB   40(R2), SUBS_1                      ; 1904
                50         28   A3  9E 00066           MOVAB   40(R3), SUBS_2                      ; 1905
                58         1B   A2  9A 0006A           MOVZBL  27(NODE_1), R8                      ; 1907
                55              01  CE 0006E           MNEGL   #1, DIMENSION                       ; 1915
                                3B  11 00071           BRB     12$                                 ;
          0B    0A   A2             01  E1 00073 6$:       BBC     #1, 10(NODE_1), 7$              ; 1910
```

```
                54      1B   A2  9A 00078         MOVZBL  27(NODE_1), R4                  ; 1911
                54           55  C2 0007C         SUBL2   DIMENSION, R4                   ;
                             54  D7 0007F         DECL    D                               ;
                             03  11 00082         BRB     8$                              ;
                54           55  D0 00083 7$:     MOVL    DIMENSION, D                    ; 1912
                54           14  C4 00086 8$:     MULL2   #20, R4                         ; 1914
                          6441   9F 00089         PUSHAB  (R4)[SUBS_1]                    ;
                       08 A441   9F 0008C         PUSHAB  8(R4)[SUBS_1]                   ;
                57      9E        9E  C3 00090    SUBL3   @(SP)+, @(SP)+, VALUE_1         ;
                          6440   9F 00094         PUSHAB  (R4)[SUBS_2]                    ; 1915
                       08 A440   9F 00097         PUSHAB  8(R4)[SUBS_2]                   ;
                56      9E        9E  C3 0009B    SUBL3   @(SP)+, @(SP)+, VALUE_2         ;
                56           57  D1 0009F         CMPL    VALUE_1, VALUE_2                ; 1916
                             04  18 000A2         BGEQ    10$                             ;
                50           01  CE 000A4 9$:     MNEGL   #1, R0                          ;
                                 04 000A7         RET                                     ;
                             04  15 000A8 10$:     BLEQ    12$                             ; 1917
                50           01  D0 000AA 11$:     MOVL    #1, R0                          ;
                                 04 000AD         RET                                     ;
                C1      55       58  F2 000AE 12$:     AOBLSS  R8, DIMENSION, 6$       ; 1907
                52           62  D0 000B2 13$:     MOVL    (NODE_1), NODE_1                ; 1925
                53           63  D0 000B5         MOVL    (NODE_2), NODE_2                ; 1926
                          FF5E   31 000B8         BRW     1$                              ; 1879
                             50  D4 000BB 14$:     CLRL    R0                              ; 1928
                                 04 000BD         RET                                     ; 1929

; Routine Size:  190 bytes,    Routine Base:  DBG$CODE + 0F21
```

```
;  1809          1930  1 ROUTINE CHECK_TEXT_DESCRIPTOR(VAL_DESC: REF DBG$VALDESC) =
;  1810          1931  1 !
;  1811          1932  1 ! FUNCTION
;  1812          1933  1 !       ----------------------------
;  1813          1934  1 !
;  1814          1935  1 ! INPUTS
;  1815          1936  1 !       ----------------------------
;  1816          1937  1 !
;  1817          1938  1 ! OUTPUTS
;  1818          1939  1 !       ----------------------------
;  1819          1940  1 !
;  1820          1941  1
;  1821          1942  2     BEGIN
;  1822          1943  2
;  1823          1944  2     BUILTIN
;  1824          1945  2         PROBER;
;  1825          1946  2
;  1826          1947  2     BIND VMS_DESC = VAL_DESC[DBG$A_VALUE_VMSDESC]: DBG$STG_DESC;
;  1827          1948  2
;  1828          1949  2
;  1829          1950  2
;  1830          1951  2     IF (.VAL_DESC[DBG$B_VALUE_CLASS] EQL DSC$K_CLASS_UBS) THEN SIGNAL(DBG$_UNALIGNED);
;  1831          1952  2
;  1832          1953  2     IF NOT PROBER(%REF(0),%REF(8) ,.VAL_DESC[DBG$L_VALUE_POINTER])
;  1833          1954  2       THEN SIGNAL(DBG$_NOACCESSR,1,.VAL_DESC[DBG$L_VALUE_POINTER]);
;  1834          1955  2
;  1835          1956  2     CH$MOVE(8,.VAL_DESC[DBG$L_VALUE_POINTER],VAL_DESC[DBG$A_VALUE_VMSDESC]);
;  1836          1957  2
;  1837          1958  3     IF (.VMS_DESC[DSC$B_DTYPE] EQL 0) AND (.VMS_DESC[DSC$B_CLASS] EQL 0)
;  1838          1959  2     THEN
;  1839          1960  3         BEGIN
;  1840          1961  3         VMS_DESC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  1841          1962  3         VMS_DESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  1842          1963  3         RETURN TRUE;
;  1843          1964  2         END;
;  1844          1965  2
;  1845          1966  3     IF  (.VMS_DESC[DSC$B_CLASS] EQL DSC$K_CLASS_D)
;  1846          1967  2      THEN VMS_DESC[DSC$B_CLASS]  =  DSC$K_CLASS_S;
;  1847          1968  2
;  1848          1969  3     IF  (.VMS_DESC[DSC$B_CLASS] EQL DSC$K_CLASS_VS)
;  1849          1970  3     AND (.VMS_DESC[DSC$B_DTYPE] EQL DSC$K_DTYPE_T)
;  1850          1971  2      THEN VMS_DESC[DSC$B_DTYPE]  =  DSC$K_DTYPE_VT;
;  1851          1972  2
;  1852          1973  3     IF  (.VMS_DESC[DSC$B_CLASS] NEQ DSC$K_CLASS_VS)
;  1853          1974  2     AND (.VMS_DESC[DSC$B_CLASS] NEQ DSC$K_CLASS_S)  THEN RETURN FALSE;
;  1854          1975  2
;  1855          1976  2     SELECTONE .VMS_DESC[DSC$B_DTYPE] OF
;  1856          1977  2         SET
;  1857          1978  2         [DSC$K_DTYPE_T] :       VMS_DESC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  1858          1979  2
;  1859          1980  2         [DSC$K_DTYPE_AZ,
;  1860          1981  2          DSC$K_DTYPE_AC,
;  1861          1982  2          DSC$K_DTYPE_VT]:       VMS_DESC[DSC$B_CLASS] = DSC$K_CLASS_VS;
;  1862          1983  2
;  1863          1984  2         [OTHERWISE]:            RETURN FALSE;
;  1864          1985  2
;  1865          1986  2         TES;
```

```
; 1866        1987  2
; 1867        1988  2      RETURN TRUE;
; 1868        1989  1      END;                          ! End of routine check_text_descriptor


                                           00FC 00000 CHECK_TEXT_DESCRIPTOR:
                                                            .WORD   Save R2,R3,R4,R5,R6,R7                        ; 1930
                57 00000000G  00   9E 00002                 MOVAB   LIB$SIGNAL, R7                                ;
                52        04  AC   D0 00009                 MOVL    VAL_DESC, R2                                  ; 1947
                56        14  A2   9E 0000D                 MOVAB   20(R2), R6                                    ;
                0D        03  A6   91 00011                 CMPB    3(R6), #13                                    ; 1951
                              09   12 00015                 BNEQ    1$                                            ;
                   00028D08   8F   DD 00017                 PUSHL   #167176                                       ;
                67            01   FB 0001D                 CALLS   #1, LIB$SIGNAL                                ;
      18   B2   08            00   0C 00020 1$:             PROBER  #0, #8, @24(R2)                               ; 1953
                              0E   12 00025                 BNEQ    2$                                            ;
                          18  A2   DD 00027                 PUSHL   24(R2)                                        ; 1954
                              01   DD 0002A                 PUSHL   #1                                            ;
                   00028228   8F   DD 0002C                 PUSHL   #164392                                       ;
                67            03   FB 00032                 CALLS   #3, LIB$SIGNAL                                ;
           66        18   B2  08   28 00035 2$:             MOVC3   #8, @24(R2), (R6)                             ; 1956
                51        02  A6   9E 0003A                 MOVAB   2(R6), R1                                     ; 1958
                              61   95 0003E                 TSTB    (R1)                                          ;
                              0E   12 00040                 BNEQ    3$                                            ;
                          03  A6   95 00042                 TSTB    3(R6)                                         ;
                              09   12 00045                 BNEQ    3$                                            ;
                61            0E   90 00047                 MOVB    #14, (R1)                                     ; 1961
           03   A6            01   90 0004A                 MOVB    #1, 3(R6)                                     ; 1962
                              3A   11 0004E                 BRB     8$                                            ; 1963
                50        03  A6   9E 00050 3$:             MOVAB   3(R6), R0                                     ; 1966
                02            60   91 00054                 CMPB    (R0), #2                                      ;
                              03   12 00057                 BNEQ    4$                                            ;
                60            01   90 00059                 MOVB    #1, (R0)                                      ; 1967
                0B            60   91 0005C 4$:             CMPB    (R0), #11                                     ; 1969
                              08   12 0005F                 BNEQ    5$                                            ;
                0E            61   91 00061                 CMPB    (R1), #14                                     ; 1970
                              03   12 00064                 BNEQ    5$                                            ;
                61            25   90 00066                 MOVB    #37, (R1)                                     ; 1971
                0B            60   91 00069 5$:             CMPB    (R0), #11                                     ; 1973
                              05   13 0006C                 BEQL    6$                                            ;
                01            60   91 0006E                 CMPB    (R0), #1                                      ; 1974
                              1B   12 00071                 BNEQ    9$                                            ;
                0E            61   91 00073 6$:             CMPB    (R1), #14                                     ; 1978
                              05   12 00076                 BNEQ    7$                                            ;
                60            01   90 00078                 MOVB    #1, (R0)                                      ;
                              0D   11 0007B                 BRB     8$                                            ;
                25            61   91 0007D 7$:             CMPB    (R1), #37                                     ; 1980
                              0C   1F 00080                 BLSSU   9$                                            ;
                27            61   91 00082                 CMPB    (R1), #39                                     ;
                              07   1A 00085                 BGTRU   9$                                            ;
                60            0B   90 00087                 MOVB    #11, (R0)                                     ; 1982
                50            01   D0 0008A 8$:             MOVL    #1, R0                                        ; 1988
                                   04 0008D                 RET                                                   ;
                              50   D4 0008E 9$:             CLRL    R0                                            ; 1989
```

```
                04 00090        RET                                             ;

; Routine Size:  145 bytes,    Routine Base:  DBG$CODE + 0FDF
```

```
;  1870          1990  1 ROUTINE FIX_UP_LENGTH(VMS_DESC: REF DBG$STG_DESC) =
;  1871          1991  1 !
;  1872          1992  1 ! FUNCTION
;  1873          1993  1 !       ----------------------------
;  1874          1994  1 !
;  1875          1995  1 ! INPUTS
;  1876          1996  1 !       ----------------------------
;  1877          1997  1 !
;  1878          1998  1 ! OUTPUTS
;  1879          1999  1 !       ----------------------------
;  1880          2000  1 !
;  1881          2001  1
;  1882          2002  2     BEGIN
;  1883          2003  2
;  1884          2004  2     LOCAL
;  1885          2005  2         SIZE,
;  1886          2006  2         BASE: REF BLOCK[,BYTE];
;  1887          2007  2
;  1888          2008  2
;  1889          2009  2
;  1890          2010  2     BASE = .VMS_DESC[DSC$A_POINTER];
;  1891          2011  2     SELECTONE .VMS_DESC[DSC$B_DTYPE] OF
;  1892          2012  2         SET
;  1893          2013  2
;  1894          2014  2         [DSC$K_DTYPE_VT]:
;  1895          2015  3             BEGIN
;  1896          2016  3             BUILTIN PROBER;
;  1897          2017  3             IF NOT PROBER(%REF(0),%REF(2) ,.BASE)
;  1898          2018  3             THEN SIGNAL(DBG$_NOACCESSR,1,.BASE);
;  1899          2019  3             SIZE = .BASE[0,0,16,0];
;  1900          2020  2             END;
;  1901          2021  2
;  1902          2022  2         [DSC$K_DTYPE_AC]:
;  1903          2023  3             BEGIN
;  1904          2024  3             BUILTIN PROBER;
;  1905          2025  3             IF NOT PROBER(%REF(0),%REF(1) ,.BASE)
;  1906          2026  3             THEN SIGNAL(DBG$_NOACCESSR,1,.BASE);
;  1907          2027  3             SIZE = .BASE[0,0, 8,0];
;  1908          2028  2             END;
;  1909          2029  2
;  1910          2030  2         [DSC$K_DTYPE_AZ]:
;  1911          2031  3             BEGIN
;  1912          2032  3             BUILTIN LOCC,PROBER;
;  1913          2033  3             LOCAL ADDR;
;  1914          2034  3             IF NOT PROBER(%REF(0),%REF(4) ,.BASE)
;  1915          2035  3             THEN SIGNAL(DBG$_NOACCESSR,1,.BASE);
;  1916          2036  3             LOCC(%REF(0), %REF(2048), .BASE; ,ADDR);
;  1917          2037  3             SIZE = .ADDR - .BASE;
;  1918          2038  2             END;
;  1919          2039  2
;  1920          2040  2         [OTHERWISE]:
;  1921          2041  2             SIZE = .VMS_DESC[DSC$W_LENGTH];
;  1922          2042  2
;  1923          2043  2         TES;
;  1924          2044  2
;  1925          2045  2     RETURN .SIZE;
;  1926          2046  2
```

```
; 1927          2047  1     END;                    ! End of routine fix_up_length

                                        001C 00000 FIX_UP_LENGTH:
                                                   .WORD   Save R2,R3,R4                ; 1990
                                  54 00000000G  00  9E 00002          MOVAB   LIB$SIGNAL, R4
                                  50       04   AC  D0 00009          MOVL    VMS_DESC, R0        ; 2010
                                  52       04   A0  D0 0000D          MOVL    4(R0), BASE
                                  51       02   A0  9A 00011          MOVZBL  2(R0), R1           ; 2011
                                  25            51  91 00015          CMPB    R1, #37             ; 2014
                                                18  12 00018          BNEQ    2$
                              62        02      00  0C 0001A          PROBER  #0, #2, (BASE)      ; 2017
                                                0D  12 0001E          BNEQ    1$
                                                52  DD 00020          PUSHL   BASE                ; 2018
                                                01  DD 00022          PUSHL   #1
                                     00028228   8F  DD 00024          PUSHL   #164392
                                  64            03  FB 0002A          CALLS   #3, LIB$SIGNAL
                                  53            62  3C 0002D 1$:      MOVZWL  (BASE), SIZE        ; 2019
                                                44  11 00030          BRB     7$                  ; 2011
                                  26            51  91 00032 2$:      CMPB    R1, #38             ; 2022
                                                18  12 00035          BNEQ    4$
                              62        01      00  0C 00037          PROBER  #0, #1, (BASE)      ; 2025
                                                0D  12 0003B          BNEQ    3$
                                                52  DD 0003D          PUSHL   BASE                ; 2026
                                                01  DD 0003F          PUSHL   #1
                                     00028228   8F  DD 00041          PUSHL   #164392
                                  64            03  FB 00047          CALLS   #3, LIB$SIGNAL
                                  53            62  9A 0004A 3$:      MOVZBL  (BASE), SIZE        ; 2027
                                                27  11 0004D          BRB     7$                  ; 2011
                                  27            51  91 0004F 4$:      CMPB    R1, #39             ; 2030
                                                1F  12 00052          BNEQ    6$
                              62        04      00  0C 00054          PROBER  #0, #4, (BASE)      ; 2034
                                                0D  12 00058          BNEQ    5$
                                                52  DD 0005A          PUSHL   BASE                ; 2035
                                                01  DD 0005C          PUSHL   #1
                                     00028228   8F  DD 0005E          PUSHL   #164392
                                  64            03  FB 00064          CALLS   #3, LIB$SIGNAL
                              62        0800    8F      00  3A 00067 5$:      LOCC    #0, #2048, (BASE)   ; 2036
                              53            51      52  C3 0006D          SUBL3   BASE, ADDR, SIZE    ; 2037
                                                    03  11 00071          BRB     7$                  ; 2011
                                  53            60  3C 00073 6$:      MOVZWL  (R0), SIZE          ; 2041
                                  50            53  D0 00076 7$:      MOVL    SIZE, R0            ; 2045
                                                04  00079          RET                         ; 2047

; Routine Size:  122 bytes,    Routine Base:  DBG$CODE + 1070

; 1928          2048  1
; 1929          2049  0 END ELUDOM


                                                   .EXTRN  LIB$SIGNAL
```

```
;                                  PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  DBG$OWN                          4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  DBG$CODE                      4330  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(0)
;  DBG$PLIT                        74  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         20          0       1000       00:01.9
;  _$255$DUA28:[DEBUG.OBJ]STRUCDEF.L32;1     32          1          3          7       00:00.1
;  _$255$DUA28:[DEBUG.OBJ]DBGLIB.L32;1     1545        190         12         97       00:02.0
;  _$255$DUA28:[DEBUG.OBJ]DSTRECRDS.L32;1
;                                           418          2          0         31       00:00.3
;  _$255$DUA28:[DEBUG.OBJ]DBGMSG.L32;1      386         11          2         22       00:00.3
;  _$255$DUA28:[DEBUG.OBJ]DBGGEN.L32;1      150          0          0         12       00:00.3


;                                  COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:DBGLEVEL3/OBJ=OBJ$:DBGLEVEL3 MSRC$:DBGLEVEL3/UPDATE=(ENH$:DBGLEVEL3)

; Size:          4330 code + 78 data bytes
; Run Time:         01:11.8
; Elapsed Time:     03:41.8
; Lines/CPU Min:     1713
; Lexemes/CPU-Min: 14910
; Memory Used:  362 pages
; Compilation Complete
```

DBGLEVEL3
LIS

DBGLIB
LIS