```
LLL                 IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRRRRRRRRRR
LLL                 IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRRRRRRRRRR
LLL                 IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRRRRRRRRRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR         RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR         RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR         RRR
LLL                    III        NNNNNN      NNN  KKK       KKK    EEE               RRR         RRR
LLL                    III        NNNNNN      NNN  KKK       KKK    EEE               RRR         RRR
LLL                    III        NNNNNN      NNN  KKK       KKK    EEE               RRR         RRR
LLL                    III        NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEE       RRRRRRRRRRR
LLL                    III        NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEE       RRRRRRRRRRR
LLL                    III        NNN   NNN   NNN  KKKKKKKKK        EEEEEEEEEEE       RRRRRRRRRRR
LLL                    III        NNN      NNNNNN  KKK       KKK    EEE               RRR   RRR
LLL                    III        NNN      NNNNNN  KKK       KKK    EEE               RRR   RRR
LLL                    III        NNN      NNNNNN  KKK       KKK    EEE               RRR   RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR      RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR      RRR
LLL                    III        NNN         NNN  KKK         KKK  EEE               RRR      RRR
LLLLLLLLLLLLLLLL    IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRR         RRR
LLLLLLLLLLLLLLLL    IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRR         RRR
LLLLLLLLLLLLLLLL    IIIIIIIII     NNN         NNN  KKK         KKK  EEEEEEEEEEEEEEE   RRR         RRR
```

FILEID**LNKERROUT

```
LL          NN      NN  KK      KK  EEEEEEEEEE  RRRRRRRR    RRRRRRRR      000000    UU      UU  TTTTTTTTTT
LL          NN      NN  KK      KK  EEEEEEEEEE  RRRRRRRR    RRRRRRRR      000000    UU      UU  TTTTTTTTTT
LL          NN      NN  KK      KK  EE          RR      RR  RR      RR  00      00  UU      UU      TT
LL          NN      NN  KK      KK  EE          RR      RR  RR      RR  00      00  UU      UU      TT
LL          NNNN    NN  KK    KK    EE          RR      RR  RR      RR  00      00  UU      UU      TT
LL          NNNN    NN  KK    KK    EE          RR      RR  RR      RR  00      00  UU      UU      TT
LL          NN  NN  NN  KKKKKK      EEEEEEEE    RRRRRRRR    RRRRRRRR    00      00  UU      UU      TT
LL          NN  NN  NN  KKKKKK      EEEEEEEE    RRRRRRRR    RRRRRRRR    00      00  UU      UU      TT
LL          NN    NNNN  KK    KK    EE          RR  RR      RR  RR      00      00  UU      UU      TT
LL          NN    NNNN  KK    KK    EE          RR  RR      RR  RR      00      00  UU      UU      TT
LL          NN      NN  KK      KK  EE          RR    RR    RR    RR    00      00  UU      UU      TT      ....
LL          NN      NN  KK      KK  EE          RR    RR    RR    RR    00      00  UU      UU      TT      ....
LLLLLLLLLL  NN      NN  KK      KK  EEEEEEEEEE  RR      RR  RR      RR    000000    UUUUUUUUUU      TT      ....
LLLLLLLLLL  NN      NN  KK      KK  EEEEEEEEEE  RR      RR  RR      RR    000000    UUUUUUUUUU      TT      ....

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

```
;     1          0001  0 MODULE LNK_HANDLER (
;     2          0002  0                 IDENT ='V04-000',
;     3          0003  0                 ADDRESSING_MODE(EXTERNAL=GENERAL,
;     4          0004  0                                 NONEXTERNAL=LONG_RELATIVE)
;     5          0005  0                 ) =
;     6          0006  0
;     7          0007  1 BEGIN
;     8          0008  1 !
;     9          0009  1 !
;    10          0010  1 !*****************************************************************************
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    13          0013  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    14          0014  1 !*  ALL RIGHTS RESERVED.                                                      *
;    15          0015  1 !*                                                                           *
;    16          0016  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    17          0017  1 !*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE           *
;    18          0018  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER      *
;    19          0019  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
;    20          0020  1 !*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY        *
;    21          0021  1 !*  TRANSFERRED.                                                              *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
;    24          0024  1 !*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
;    25          0025  1 !*  CORPORATION.                                                              *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS      *
;    28          0028  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    29          0029  1 !*                                                                           *
;    30          0030  1 !*                                                                           *
;    31          0031  1 !*****************************************************************************
;    32          0032  1 !
;    33          0033  1 !
;    34          0034  1 !
;    35          0035  1 !
;    36          0036  1 !
;    37          0037  1 !++
;    38          0038  1 !
;    39          0039  1 ! MODULE: LNK_HANDLER
;    40          0040  1 !
;    41          0041  1 ! FACILITY: LINKER
;    42          0042  1 !
;    43          0043  1 ! ABSTRACT: ERROR HANDLER FOR THE LINKER
;    44          0044  1 !
;    45          0045  1 ! HISTORY:
;    46          0046  1 !
;    47          0047  1 !       VERSION: X01.00
;    48          0048  1 !
;    49          0049  1 !       AUTHOR: T.J. PORTER 30-DEC-76
;    50          0050  1 !
;    51          0051  1 ! MODIFIED BY:
;    52          0052  1 !
;    53          0053  1 !       V03-002 BLS0031         Benn Schreiber          19-Jan-1981
;    54          0054  1 !               Print Pass 1 error messages in map in pass 2
;    55          0055  1 !
;    56          0056  1 !       V03-001 BLS0007         Benn Schreiber,         3-Jun-1980
;    57          0057  1 !               Rewrite and convert to MDL data structures.
```

```
;   58          0058  1 !--
```

```
;    60          0059  1
;    61          0060  1 !
;    62          0061  1 !++
;    63          0062  1 !
;    64          0063  1 ! FUNCTIONAL DESCRIPTION:
;    65          0064  1 !
;    66          0065  1 ! This module processes errors for the linker.
;    67          0066  1 !--
;    68          0067  1 !
;    69          0068  1 LIBRARY
;    70          0069  1         'STARLETL32';                   ! VMS AND RMS INTERFACES
;    71          0070  1 LIBRARY
;    72          0071  1         'DATBAS';                       ! DATA BASE
;    73          0072  1 REQUIRE
;    74          0073  1         'PREFIX';                       ! GENERAL MACROS ETC
;    75          0188  1
;    76          0189  1 EXTERNAL
;    77          0190  1         LNK$GB_PASS : BYTE,                             ! PASS 1 OR 2
;    78          0191  1         LNK$GL_CUROMD : REF BLOCK[,BYTE],               ! POINTER TO CURRENT OBJECT MODULE DESCRIPTOR
;    79          0192  1         LNK$GL_CTLMSK : BLOCK[,BYTE];                   ! CONTROL FLAGS
;    80          0193  1
;    81          0194  1 EXTERNAL ROUTINE
;    82          0195  1         LNK$ALLOBLK;                                    ! ALLOCATE VIRTUAL MEMORY
;    83          0196  1
;    84          0197  1 PSECT PLIT = $CODE$;                                    ! MOVE THE PLITS
;    85          0198  1 !
;    86          0199  1 ! $SHR_MESSAGES - a macro which defines facility-specific message codes
;    87          0200  1 !       which are based on the system-wide shared message codes.
;    88          0201  1 !
;    89          0202  1 !       $SHR_MESSAGES( name, code, (msg,severity), ... )
;    90          0203  1 !
;    91          0204  1 !       where:
;    92          0205  1 !           "name" is the name of the facility (e.g., COPY)
;    93          0206  1 !           "code" is the corresponding facility code (e.g., 103)
;    94          0207  1 !           "msg" is the name of the shared message (e.g., BEGIN)
;    95          0208  1 !           "severity" is the desired message severity (e.g., 1, 0, 2)
;    96          0209  1 !
;    97          0210  1
;    98          0211  1 MACRO
;    99        M 0212  1     $SHR_MESSAGES( FACILITY_NAME, FACILITY_CODE ) =
;   100        M 0213  1         GLOBAL LITERAL
;   101          0214  1         SHR$MSG_IDS( FACILITY_NAME, FACILITY_CODE, %REMAINING ); %,
;   102          0215  1
;   103        M 0216  1     SHR$MSG_IDS( FACILITY_NAME, FACILITY_CODE) [ VALUE ] =
;   104          0217  1         SHR$MSG_CALC( FACILITY_NAME, FACILITY_CODE, %REMOVE(VALUE) ) %,
;   105          0218  1
;   106        M 0219  1     SHR$MSG_CALC( FACILITY_NAME, FACILITY_CODE, MSG_ID, SEVERITY ) =
;   107        M 0220  1         %NAME(FACILITY_NAME,'$_',MSG_ID) = %NAME('SHR$_',MSG_ID) + FACILITY_CODE*65536 +
;   108        M 0221  1             %IF %DECLARED(%NAME('STS$K_',SEVERITY))
;   109        M 0222  1                 %THEN %NAME('STS$K_',SEVERITY)
;   110          0223  1                 %ELSE SEVERITY %FI %;
;   111          0224  1
;   112          0225  1
;   113          0226  1 !
;   114          0227  1 EXTERNAL LITERAL
;   115          0228  1         LINK$_FACILITY,
;   116          0229  1         LIN$_FATALERROR;
```

```
;  117          0230  1
;  118        P 0231  1 $SHR_MESSAGES (LIN, 100,                          ! facility is Link, id is 151
;  119        P 0232  1                (openin, severe),                  ! error opening "file" as input
;  120        P 0233  1                (openout, warning),                ! error opening "file" as output
;  121        P 0234  1                (closein, warning),                ! error closing "file" as input
;  122        P 0235  1                (closeout, warning),               ! error closing "file" as output
;  123        P 0236  1                (readerr, warning),                ! error reading "file"
;  124          0237  1                (writeerr, error));                ! error writing "file"
;  125          0238  1
;  126          0239  1 EXTERNAL ROUTINE
;  127          0240  1         LNK$MAPOUT,                               ! Output to map file
;  128          0241  1         LNK$EXIT;                                 ! Linker exit routine
;  129          0242  1
;  130          0243  1
;  131          0244  1 GLOBAL
;  132          0245  1         LNK$GL_ERRLIST,                           ! Listhead for pass 1 non-module errors
;  133          0246  1         LNK$GL_LASTERR : INITIAL (LNK$GL_ERRLIST),! pointer to last pass 1 non-module error
;  134          0247  1         LNK$GB_MAXERCOD : BYTE INITIAL (1);       ! Maximum error code seen during link
```

```
;  136     0248  1 GLOBAL ROUTINE LNK$ERRMSG(ERRORMSGDESC) =
;  137     0249  1
;  138     0250  1 !---
;  139     0251  1 !
;  140     0252  1 ! This routine is called by SYS$PUTMSG after it has formatted the error
;  141     0253  1 ! message and just before it writes the message to SYS$ERROR or SYS$OUTPUT.
;  142     0254  1 ! The input parameter is a string descriptor for the formatted message.
;  143     0255  1 ! This routine is used to output error messages to the map file.
;  144     0256  1 !
;  145     0257  1 ! INPUTS:
;  146     0258  1 !
;  147     0259  1 !       ERRORMSGDESC    address of string descriptor of error text.
;  148     0260  1 !
;  149     0261  1 ! OUTPUTS:
;  150     0262  1 !
;  151     0263  1 !       error text also output to map file if open.
;  152     0264  1 !---
;  153     0265  1
;  154     0266  2 BEGIN
;  155     0267  2
;  156     0268  2 MAP
;  157     0269  2     ERRORMSGDESC : REF BLOCK[,BYTE];
;  158     0270  2
;  159     0271  2 IF .LNK$GL_CTLMSK[LNK$V_MAPOPN]
;  160     0272  2 THEN LNK$MAPOUT(.ERRORMSGDESC[DSC$A_POINTER],                   ! Put message in map if mapping
;  161     0273  2                  .ERRORMSGDESC[DSC$W_LENGTH]);
;  162     0274  2
;  163     0275  2 IF .LNK$GB_PASS EQL 1                                          ! If this is pass 1
;  164     0276  2     AND .LNK$GL_CTLMSK[LNK$V_MAP]                              !  and a map will be generated
;  165     0277  3 THEN BEGIN
;  166     0278  3     LOCAL
;  167     0279  3         LASTBLOCK : REF BLOCK[,BYTE],
;  168     0280  3         EBLOCK : REF BLOCK[,BYTE];
;  169     0281  3
;  170     0282  3     LNK$ALLOBLK(OEB$C_SIZE+.ERRORMSGDESC[DSC$W_LENGTH],        ! Allocate a text block
;  171     0283  3                 EBLOCK);
;  172     0284  3     EBLOCK[OEB$L_NXTOEB] = 0;                                  ! Zero link to next block
;  173     0285  3     EBLOCK[OEB$W_BYTCNT] = .ERRORMSGDESC[DSC$W_LENGTH];        ! Fill in the error message block
;  174     0286  3     IF .LNK$GL_CUROMD NEQ 0
;  175     0287  4     THEN BEGIN
;  176     0288  4         LASTBLOCK = .LNK$GL_CUROMD[OMD$L_LSTERR];              ! Get pointer to last on list
;  177     0289  4         LASTBLOCK[OEB$L_NXTOEB] = .EBLOCK;                     ! Link new one into list
;  178     0290  4         LNK$GL_CUROMD[OMD$L_LSTERR] = .EBLOCK;                 !  and make this one the new last
;  179     0291  4         END
;  180     0292  4     ELSE BEGIN                                                 ! It's not related to any particular module
;  181     0293  4         LASTBLOCK = .LNK$GL_LASTERR;                           ! Get pointer to last non-module error
;  182     0294  4         LASTBLOCK[OEB$L_NXTOEB] = .EBLOCK;                     ! And link new one in
;  183     0295  4         LNK$GL_LASTERR = .EBLOCK;                              ! Make this one the new last block
;  184     0296  3         END;
;  185     0297  3     CH$MOVE(.ERRORMSGDESC[DSC$W_LENGTH],                       ! Copy the text for printing in pass 2
;  186     0298  3             .ERRORMSGDESC[DSC$A_POINTER],
;  187     0299  3                 EBLOCK[OEB$T_TEXT]);
;  188     0300  2     END;
;  189     0301  2 RETURN TRUE;                                                   ! Tell $PUTMSG to output
;  190     0302  2                                                                !  msg on terminal
;  191     0303  1 END;                                                           ! end of map_outmsg
```

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

                                        .PSECT  $GLOBAL$,NOEXE,2

                                00000 LNK$GL_ERRLIST::
                                        .BLKB   4
                           00000000' 00004 LNK$GL_LASTERR::
                                        .ADDRESS LNK$GL_ERRLIST                         ;
                                 01  00008 LNK$GB_MAXERCOD::
                                        .BYTE   1                                       ;

                                        LIN$_OPENIN==       6557852
                                        LIN$_OPENOUT==      6557856
                                        LIN$_CLOSEIN==      6557776
                                        LIN$_CLOSEOUT==     6557784
                                        LIN$_READERR==      6557872
                                        LIN$_WRITEERR==     6557906
                                        .EXTRN  LNK$GB_PASS, LNK$GL_CUROMD
                                        .EXTRN  LNK$GL_CTLMSK, LNK$ALLOBLK
                                        .EXTRN  LINK$_FACILITY, LIN$_FATALERROR
                                        .EXTRN  LNK$MAPOUT, LNK$EXIT

                                        .PSECT  $CODE$,NOWRT,2

                                  00FC 00000            .ENTRY  LNK$ERRMSG, Save R2,R3,R4,R5,R6,R7      ; 0248
                 57 00000000G   00  9E 00002            MOVAB   LNK$GL_CTLMSK, R7                       ;
                 56 00000000'   EF  9E 00009            MOVAB   LNK$GL_LASTERR, R6                      ;
                 5E                 04  C2 00010        SUBL2   #4, SP                                  ;
           11    67                 05  E1 00013        BBC     #5, LNK$GL_CTLMSK, 1$                   ; 0271
                 50         04      AC  D0 00017        MOVL    ERRORMSGDESC, R0                        ; 0273
                 7E                 60  3C 0001B        MOVZWL  (R0), -(SP)                             ;
                            04      A0  DD 0001E        PUSHL   4(R0)                                   ; 0272
              00000000G     00          FB 00021        CALLS   #2, LNK$MAPOUT                          ;
                            01 00000000G 00 91 00028 1$: CMPB   LNK$GB_PASS, #1                         ; 0275
                                        45  12 0002F    BNEQ    4$                                      ;
           41    67                 04  E1 00031        BBC     #4, LNK$GL_CTLMSK, 4$                   ; 0276
                 5E                 DD  00035          PUSHL   SP                                      ; 0282
                 53         04      AC  D0 00037        MOVL    ERRORMSGDESC, R3                        ;
                 7E                 63  3C 0003B        MOVZWL  (R3), -(SP)                             ;
                 6E                 06  C0 0003E        ADDL2   #6, (SP)                                ;
              00000000G     00          FB 00041        CALLS   #2, LNK$ALLOBLK                         ;
                 52                 6E  D0 00048        MOVL    EBLOCK, R2                              ; 0284
                                    62  D4 0004B        CLRL    (R2)                                    ;
                       04    A2          63  B0 0004D    MOVW    (R3), 4(R2)                             ; 0285
                 50 00000000G   00  D0 00051            MOVL    LNK$GL_CUROMD, R0                       ; 0286
                                    0D  13 00058        BEQL    2$                                      ;
                 51         24      A0  D0 0005A        MOVL    36(R0), LASTBLOCK                       ; 0288
                 61                 52  D0 0005E        MOVL    R2, (LASTBLOCK)                         ; 0289
                       24    A0          52  D0 00061    MOVL    R2, 36(R0)                              ; 0290
                                        09  11 00065    BRB     3$                                      ; 0286
                 51                 66  D0 00067 2$:    MOVL    LNK$GL_LASTERR, LASTBLOCK               ; 0293
                 61                 52  D0 0006A        MOVL    R2, (LASTBLOCK)                         ; 0294
                 66                 52  D0 0006D        MOVL    R2, LNK$GL_LASTERR                      ; 0295
     06    A2          04    B3          63  28 00070 3$: MOVC3   (R3), @4(R3), 6(R2)                 ; 0299
                 50                 01  D0 00076 4$:    MOVL    #1, R0                                  ; 0301
```

```
                                04 00079        RET                                     ; 0303

; Routine Size:  122 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  193          0304  1 GLOBAL ROUTINE LNK$HANDLER (SIGARGS, MECHARGS) =
;  194          0305  2 BEGIN
;  195          0306  2
;  196          0307  2 !---
;  197          0308  2 !
;  198          0309  2 ! This routine is a condition handler called whenever a signal is done by
;  199          0310  2 ! the linker.  It merely remembers the most severe error for an exit status.
;  200          0311  2 !
;  201          0312  2 !--
;  202          0313  2
;  203          0314  2 MAP
;  204          0315  2     SIGARGS : REF BLOCK[,BYTE],
;  205          0316  2     MECHARGS : REF BLOCK[,BYTE];
;  206          0317  2
;  207          0318  2 BIND
;  208          0319  2     SIGNAME = SIGARGS[CHF$L_SIG_NAME] : BLOCK[,BYTE];
;  209          0320  2
;  210          0321  2 !
;  211          0322  2 ! If this is not a linker error then resignal
;  212          0323  2 !
;  213          0324  2 IF .SIGNAME[STS$V_FAC_NO] NEQ LINK$_FACILITY
;  214          0325  2     THEN RETURN SS$_RESIGNAL;
;  215          0326  2
;  216          0327  2 IF NOT .SIGNAME                                      ! If this is an error signal
;  217          0328  4     AND ((.SIGNAME[STS$V_SEVERITY]                    !  and severity is worse than it was
;  218          0329  4         GEQU .LNK$GB_MAXERCOD)
;  219          0330  3         OR .LNK$GB_MAXERCOD)                          ! or we haven't had any errors
;  220          0331  2     THEN LNK$GB_MAXERCOD =                            !  then set new worst error
;  221          0332  2                 .SIGNAME[STS$V_SEVERITY];
;  222          0333  2
;  223          0334  2 SIGARGS[CHF$L_SIG_ARGS] = .SIGARGS[CHF$L_SIG_ARGS] - 2; !Don't count pc/psl
;  224          0335  2
;  225          0336  2 $PUTMSG(MSGVEC=SIGARGS[CHF$L_SIG_ARGS],ACTRTN=LNK$ERRMSG);! output the message
;  226          0337  2
;  227          0338  2 SIGARGS[CHF$L_SIG_ARGS] = .SIGARGS[CHF$L_SIG_ARGS] + 2;
;  228          0339  2 IF .LNK$GB_MAXERCOD EQL STS$K_SEVERE                 ! if this is a severe error
;  229          0340  4 THEN LNK$EXIT((IF .LNK$GB_PASS EQL 0 THEN (.SIGNAME OR STS$M_INHIB_MSG)
;  230          0341  2                         ELSE LIN$_FATALERROR));      !  then exit
;  231          0342  2
;  232          0343  2 RETURN SS$_CONTINUE                                  ! signal to continue
;  233          0344  1 END;


                                                                .EXTRN  SYS$PUTMSG

                                                  000C 00000    .ENTRY  LNK$HANDLER, Save R2,R3                 ; 0304
                                53 00000000'  EF  9E  00002     MOVAB   LNK$GB_MAXERCOD, R3                     ;
                                52        04  AC  D0  00009     MOVL    SIGARGS, R2                             ; 0319
00000000G   8F        06  A2    0C            00  ED  0000D     CMPZV   #0, #12, 6(R2), #LINK$_FACILITY         ; 0324
                                              06  13  00017     BEQL    1$
                                50      0918  8F  3C  00019     MOVZWL  #2328, R0                               ; 0325
                                              04  0001E     RET
                                17        04  A2  E8  0001F 1$: BLBS    4(R2), 3$                               ; 0327
                                50            63  9A  00023     MOVZBL  LNK$GB_MAXERCOD, R0                     ; 0329
            50        04  A2    03            00  ED  00026     CMPZV   #0, #3, 4(R2), R0
                                              03  1E  0002C     BGEQU   2$                                      ;
```

```
                                                09              63  E9 0002E          BLBC    LNK$GB_MAXERCOD, 3$                     ; 0330
            50        04    A2                  03              00  EF 00031 2$:      EXTZV   #0, #3, 4(R2), R0                      ; 0332
                                                63              50  90 00037          MOVB    R0, LNK$GB_MAXERCOD                     ;
                                                62              02  C2 0003A 3$:      SUBL2   #2, (R2)                                ; 0334
                                                                7E  7C 0003D          CLRQ    -(SP)                                   ; 0336
                                                        FF43    CF  9F 0003F          PUSHAB  LNK$ERRMSG                              ;
                                                                52  DD 00043          PUSHL   R2                                      ;
                      00000000G                 00            04  FB 00045          CALLS   #4, SYS$PUTMSG                          ;
                                                62              02  C0 0004C          ADDL2   #2, (R2)                                ; 0338
                                                04              63  91 0004F          CMPB    LNK$GB_MAXERCOD, #4                     ; 0339
                                                                22  12 00052          BNEQ    6$                                      ;
                                                    00000000G   00  95 00054          TSTB    LNK$GB_PASS                             ; 0340
                                                                0D  12 0005A          BNEQ    4$                                      ;
                      52        04    A2  10000000              8F  C9 0005C          BISL3   #268435456, 4(R2), R2                   ;
                                                                52  DD 00065          PUSHL   R2                                      ;
                                                                06  11 00067          BRB     5$                                      ;
                                                    00000000G   8F  DD 00069 4$:      PUSHL   #LIN$_FATALERROR                        ;
                      00000000G                 00            01  FB 0006F 5$:      CALLS   #1, LNK$EXIT                            ;
                                                50              01  D0 00076 6$:      MOVL    #1, R0                                  ; 0343
                                                                    04 00079          RET                                             ; 0344

; Routine Size:  122 bytes,    Routine Base:  $CODE$ + 007A
```

```
;  235          0345  1 GLOBAL ROUTINE LNK$FILNAMDSC (IFAB) =
;  236          0346  2 BEGIN
;  237          0347  2
;  238          0348  2 !--
;  239          0349  2 ! This routine returns the address of a string descriptor for the input fab.
;  240          0350  2 !
;  241          0351  2 ! INPUTS:
;  242          0352  2 !
;  243          0353  2 !       IFAB    is the address of the fab
;  244          0354  2 !
;  245          0355  2 ! ROUTINE VALUE:
;  246          0356  2 !
;  247          0357  2 !       the address of a string descriptor for the file name
;  248          0358  2 !
;  249          0359  2 !---
;  250          0360  2 MAP
;  251          0361  2     IFAB : REF BLOCK[,BYTE];
;  252          0362  2
;  253          0363  2 BIND
;  254          0364  2     NAM = .IFAB [FAB$L_NAM] : BBLOCK;
;  255          0365  2
;  256          0366  2 OWN
;  257          0367  2     FILEDESC : BLOCK[DSC$C_S_BLN,BYTE];                 ! the string descriptor
;  258          0368  2
;  259          0369  2 IF (FILEDESC [DSC$W_LENGTH] = .NAM [NAM$B_RSL]) NEQ 0   ! if resultant name present
;  260          0370  2 THEN FILEDESC [DSC$A_POINTER] = .NAM [NAM$L_RSA]
;  261          0371  2 ELSE IF (FILEDESC [DSC$W_LENGTH] = .NAM [NAM$B_ESL]) NEQ 0 ! if expanded name present
;  262          0372  2     THEN FILEDESC [DSC$A_POINTER] = .NAM [NAM$L_ESA]
;  263          0373  3     ELSE BEGIN
;  264          0374  3         FILEDESC [DSC$W_LENGTH] = .IFAB [FAB$B_FNS];    ! use filename string
;  265          0375  3         FILEDESC [DSC$A_POINTER] = .IFAB [FAB$L_FNA];   !  if all else fails
;  266          0376  2         END;
;  267          0377  2
;  268          0378  2 RETURN FILEDESC
;  269          0379  1 END;                                                    !Of LNK$FILNAMDSC


                                                .PSECT  $OWN$,NOEXE,2

                                    00000 FILEDESC:
                                                .BLKB   8


                                                .PSECT  $CODE$,NOWRT,2

                                      0004 00000        .ENTRY  LNK$FILNAMDSC, Save R2                  ; 0345
                 52 00000000'   EF   9E 00002         MOVAB   FILEDESC, R2                            ;
                 51         04  AC   D0 00009         MOVL    IFAB, R1                                ; 0364
                 50         28  A1   D0 0000D         MOVL    40(R1), R0                              ;
                 62         03  A0   9B 00011         MOVZBW  3(R0), FILEDESC                         ; 0369
                                07   13 00015         BEQL    1$                                      ;
              04 A2         04  A0   D0 00017         MOVL    4(R0), FILEDESC+4                       ; 0370
                                16   11 0001C         BRB     3$                                      ;
                 62         0B  A0   9B 0001E 1$:     MOVZBW  11(R0), FILEDESC                        ; 0371
                                07   13 00022         BEQL    2$                                      ;
```

```
                                04   A2       0C   A0  D0 00024          MOVL    12(R0), FILEDESC+4                      ; 0372
                                                   09  11 00029          BRB     3$                                      ;
                                     62       34   A1  9B 0002B 2$:     MOVZBW  52(R1), FILEDESC                        ; 0374
                                04   A2       2C   A1  D0 0002F          MOVL    44(R1), FILEDESC+4                      ; 0375
                                     50            62  9E 00034 3$:     MOVAB   FILEDESC, R0                            ; 0378
                                                       04 00037          RET                                             ; 0379

; Routine Size:  56 bytes,    Routine Base:  $CODE$ + 00F4

;  270          0380  1
;  271          0381  0 END ELUDOM



;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $GLOBAL$                        9  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        300  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  . ABS .                         0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)
;  $OWN$                           8  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                               Library Statistics
;
;                                    -------- Symbols --------      Pages       Processing
;       File                         Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1   9776       32         0         581        00:01.1
;  _$255$DUA28:[LINKER.OBJ]DATBAS.L32;1  538        7         1          28        00:00.5



;                                   COMMAND QUALIFIERS

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

; Size:         300 code + 17 data bytes
; Run Time:       00:09.1
; Elapsed Time:   00:25.8
; Lines/CPU Min:   2503
; Lexemes/CPU-Min: 20490
; Memory Used:  73 pages
; Compilation Complete
```