```
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
```

```
LL            NN       NN  KK      KK  DDDDDDDD    YY      YY  NN       NN  MM       MM  EEEEEEEEEE  MM       MM
LL            NN       NN  KK      KK  DDDDDDDD    YY      YY  NN       NN  MM       MM  EEEEEEEEEE  MM       MM
LL            NN       NN  KK      KK  DD      DD  YY      YY  NN       NN  MMMM   MMMM  EE          MMMM   MMMM
LL            NN       NN  KK      KK  DD      DD  YY      YY  NN       NN  MMMM   MMMM  EE          MMMM   MMMM
LL            NNNN     NN  KK    KK    DD      DD    YY  YY    NNNN     NN  MM  MM  MM  EE          MM  MM  MM
LL            NNNN     NN  KK    KK    DD      DD    YY  YY    NNNN     NN  MM  MM  MM  EE          MM  MM  MM
LL            NN  NN   NN  KKKKKK      DD      DD      YY      NN  NN   NN  MM      MM  EEEEEEEE    MM      MM
LL            NN  NN   NN  KKKKKK      DD      DD      YY      NN  NN   NN  MM      MM  EEEEEEEE    MM      MM
LL            NN    NNNN  KK    KK    DD      DD      YY      NN    NNNN  MM      MM  EE          MM      MM
LL            NN    NNNN  KK    KK    DD      DD      YY      NN    NNNN  MM      MM  EE          MM      MM
LL            NN      NN  KK      KK  DD      DD      YY      NN      NN  MM      MM  EE          MM      MM
LL            NN      NN  KK      KK  DD      DD      YY      NN      NN  MM      MM  EE          MM      MM
LLLLLLLLLL    NN      NN  KK      KK  DDDDDDDD        YY      NN      NN  MM      MM  EEEEEEEEEE  MM      MM
LLLLLLLLLL    NN      NN  KK      KK  DDDDDDDD        YY      NN      NN  MM      MM  EEEEEEEEEE  MM      MM

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_DYNMEM
;  2          0002  0                 (IDENT = 'V04-000'
;  3          0003  0                 ,ADDRESSING_MODE
;  4          0004  0                         (EXTERNAL = GENERAL)
;  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_DYNMEM
; 40          0040  1 !
; 41          0041  1 ! FACILITY: LINKER
; 42          0042  1 !
; 43          0043  1 ! ABSTRACT:  DYNAMIC MEMORY ALLOCATION AND DEALLOCATION
; 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 14-JAN-77
; 50          0050  1 !
; 51          0051  1 ! MODIFIED BY:
; 52          0052  1 !
; 53          0053  1 !       V03-003 ADE0002         Alan D. Eldridge        30-Jul-1284
; 54          0054  1 !               Fix LNK$REQUESTMEM to return LIN$_EXPAGQUO or LIN$_MEMFUL
; 55          0055  1 !               rather than false.
; 56          0056  1 !
; 57          0057  1 !       V03-002 ADE0001         Alan D. Eldridge        24-Jul-1984
```

```
;    58          0058  1 !               Differentiate page file quota problems from lack of virtual
;    59          0059  1 !               address space when reporting errors.
;    60          0060  1 !
;    61          0061  1 !       V03-001 JWT0038         Jim Teague              23-Jun-1982
;    62          0062  1 !               Clean up INFO#212 errors.  Also add routine LNK$REQUESTMEM
;    63          0063  1 !               which will make $EXPREG calls with better
;    64          0064  1 !               error handling.
;    65          0065  1 !
;    66          0066  1 !       V03-001 BLS0007         Benn Schreiber,         3-Jun-1980
;    67          0067  1 !               Convert to MDL data structures.
;    68          0068  1 !--
```

```
;    70          0069  1 !
;    71          0070  1 !++
;    72          0071  1 !
;    73          0072  1 ! FUNCTIONAL DESCRIPTION:
;    74          0073  1 !
;    75          0074  1 !       THIS MODULE CONTAINS ALL THE DYNAMIC MEMORY ALLOCATION
;    76          0075  1 !       AND DE-ALLOCATION LOGIC OF THE LINKER.  A SINGLY
;    77          0076  1 !       LINKED LIST OF FREE BLOCKS OF MEMORY IS MAINTAINED
;    78          0077  1 !       (LISTHEAD IS LNK$GL_MEMLHD) AND MEMORY IS ALLOCATED
;    79          0078  1 !       BY FIRST FIT.  SHOULD THERE BE NO AVAILABLE MEMORY BLOCK
;    80          0079  1 !       OF REQUIRED SIZE, THE ALLOCATION ROUTINE EXPANDS
;    81          0080  1 !       THE PROGRAM REGION BY THE NUMBER OF PAGES EQUAL TO
;    82          0081  1 !       LNK$C_MEMEXP, LINKS THIS ON THE END OF THE FREE
;    83          0082  1 !       LIST AND ALLOCATES THE REQUIRED MEMORY FROM THAT NEW
;    84          0083  1 !       BLOCK.  THE FREE MEMORY LIST IS THEREFORE INITIALIZED
;    85          0084  1 !       ON FIRST ALLOCATION CALL.  MEMORY IS ALWAYS ALLOCATED
;    86          0085  1 !       IN EIGHT BYTE QUANTA, WITH A MAXIMUM OF LNK$C_MEMEXP*512
;    87          0086  1 !       BYTES.  DEALLOCATION EFFECTS COMPACTION WHENEVER POSSIBLE.
;    88          0087  1 !
;    89          0088  1 ! CALLING SEQUENCES:
;    90          0089  1 !       LNK$ALLOBLK(BLOCKSIZE,BLOCKADDR)
;    91          0090  1 !       LNK$DEALBLK(BLOCKSIZE,BLOCKADDR)
;    92          0091  1 !               WHERE:
;    93          0092  1 !                 BLOCKSIZE = NUMBER OF BYTES TO BE (DE)ALLOCATED.
;    94          0093  1 !                 BLOCKADDR = ADDRESS OF CELL FOR THE ADDRESS OF
;    95          0094  1 !                             THE BLOCK ALLOCATED OR TO BE DEALLOCATED.
;    96          0095  1 ! ERROR CONDITIONS:
;    97          0096  1 !
;    98          0097  1 !       1.  BLOCKSIZE < OR = 0 OR > LNK$C_MEMEXP*512 (CODE = 0,10)
;    99          0098  1 !
;   100          0099  1 !       2.  FAILURE TO EXPAND THE PROGRAM REGION ISSUES A MESSAGE
;   101          0100  1 !           THAT MEMORY IS FULL AND THE LINKER ABORTS.
;   102          0101  1 !
;   103          0102  1 !       3.  ANY PART OF A BLOCK TO BE DEALLOCATED IS:
;   104          0103  1 !               (I) WITHIN A FREE BLOCK (CODE = 2,13)
;   105          0104  1 !               (II) BEYOND TOP OF PROGRAM REGION (CODE = 11)
;   106          0105  1 !               (III) LOWER THAN THAN LOWEST BLOCK EVER ALLOCATED (CODE = 12)
;   107          0106  1 !
;   108          0107  1 !       IN CASES 1 AND 3 A FATAL (''BUG'') MESSAGE IS ISSUED AND
;   109          0108  1 !               THE LINKER TERMINATES.
;   110          0109  1 !
;   111          0110  1 !--
;   112          0111  1 !
;   113          0112  1 LIBRARY
;   114          0113  1         'LIBL32';
;   115          0114  1
;   116          0115  1 REQUIRE
;   117          0116  1         'PREFIX';
;   118          0231  1 !
;   119          0232  1 FORWARD ROUTINE
;   120          0233  1         LNK$REQUESTMEM,
;   121          0234  1         ALLOCATE ,                              ! ALLOCATION ROUTINE
;   122          0235  1         DEALLOCATE;                             ! DEALLOCATION ROUTINE
;   123          0236  1 GLOBAL
;   124          0237  1         LNK$GL_MEMLHD   : VECTOR[2]
;   125          0238  1                           INITIAL (0,0),        ! FREE MEMORY LISTHEAD
;   126          0239  1         LNK$GL_MINADDR : INITIAL(CONTROL_REGION); ! LOWEST ADDRESS EVER ALLOCATED
```

```
;  127          0240  1
;  128          0241  1 EXTERNAL LITERAL
;  129          0242  1          LIN$_EXPAGQUO,                  ! RAN OUT OF PAGE FILE QUOTA
;  130          0243  1          LIN$_MEMBUG,                    ! ILLEGAL BLOCK SIZE MESSAGE
;  131          0244  1          LIN$_MEMFUL;                    ! RAN OUT OF VM MESSAGE
;  132          0245  1
;  133          0246  1 GLOBAL LITERAL
;  134          0247  1          LNK$C_MEMEXP = 128;             ! NO. OF PAGES TO EXTEND PROGRAM REGION
;  135          0248  1
;  136          0249  1 LITERAL
;  137          0250  1          MAXBLOCKSIZE = LNK$C_MEMEXP*512;        ! MAXIMUM ALLOCATION SIZE
;  138          0251  1
;  139          0252  1 OWN
;  140          0253  1          ERRORCODE       : BYTE,         ! ERROR CODE FOR FAILURE MESSAGE
;  141          0254  1          NEWBLOCK        : REF VECTOR [2],       ! CURRENT BLOCK POINTER
;  142          0255  1          NEXTBLOCK       : REF VECTOR [2],       ! NEXT BLOCK POINTER
;  143          0256  1          LASTBLOCK       : REF VECTOR [2];       ! PREVIOUS BLOCK POINTER
;  144          0257  1
```

```
;  146          0258  1 GLOBAL ROUTINE LNK$ALLOBLK (SIZE, BLOCKADDR) : NOVALUE =
;  147          0259  1 !
;  148          0260  1 !       ALLOCATE A BLOCK FROM THE FREE
;  149          0261  1 !       MEMORY LIST.
;  150          0262  1 !
;  151          0263  2 BEGIN
;  152          0264  2
;  153          0265  2 LOCAL
;  154          0266  2         BLOCKSIZE;
;  155          0267  2
;  156          0268  2 ERRORCODE = 0;                                  ! INITIALIZE ERROR CODE
;  157          0269  2
;  158          0270  2 BLOCKSIZE = (.SIZE + 7) AND ( NOT 7);           ! ROUND UP TO MULTIPLE OF 8 BYTES
;  159          0271  2
;  160          0272  2 IF .BLOCKSIZE EQL 0                             ! CHECK LEGAL BLOCK
;  161          0273  2 OR .BLOCKSIZE GTRU MAXBLOCKSIZE                 ! SIZE WAS REQUESTED
;  162          0274  3 OR NOT  (ERRORCODE = .ERRORCODE + 1;            ! SET NEW ERROR CODE
;  163          0275  3          ALLOCATE (.BLOCKSIZE, .BLOCKADDR)      ! GO ALLOCATE
;  164          0276  3         )
;  165          0277  2 THEN SIGNAL_STOP(LIN$_MEMBUG, 3                 ! ISSUE ERROR MESSAGE AND
;  166          0278  2                  ,.BLOCKSIZE,.BLOCKADDR
;  167          0279  2                  ,.ERRORCODE                    ! TERMINATE IF FAILURE
;  168          0280  2                  );
;  169          0281  2 RETURN;                                         ! OTHERWISE RETURN
;  170          0282  1 END;                                            ! OF LNK$ALLOBLK ROUTINE


                                        .TITLE  LNK_DYNMEM
                                        .IDENT  \V04-000\

                                        .PSECT  $OWN$,NOEXE,2

                                00000 ERRORCODE:
                                        .BLKB   1
                                00001           .BLKB   3
                                00004 NEWBLOCK:
                                        .BLKB   4
                                00008 NEXTBLOCK:
                                        .BLKB   4
                                0000C LASTBLOCK:
                                        .BLKB   4

                                        .PSECT  $GLOBAL$,NOEXE,2

                    00000000    00000 LNK$GL_MEMLHD::
                                        .LONG   0, 0                                    ;
                    40000000    00008 LNK$GL_MINADDR::
                                        .LONG   1073741824                              ;

                                LNK$C_MEMEXP==      128
                                        .EXTRN  LIN$_EXPAGQUO, LIN$_MEMBUG
                                        .EXTRN  LIN$_MEMFUL

                                        .PSECT  $CODE$,NOWRT,2

                                000C 00000      .ENTRY  LNK$ALLOBLK, Save R2,R3                 ; 0258
53      0000'   CF   9E 00002          MOVAB   ERRORCODE, R3                                   ;
```

```
                                         63  94 00007          CLRB    ERRORCODE                        ; 0268
        50        04  AC                 07  C1 00009          ADDL3   #7, SIZE, R0                     ; 0270
        52            50                 07  CB 0000E          BICL3   #7, R0, BLOCKSIZE                ;
                                         18  13 00012          BEQL    1$                               ; 0272
           00010000   8F                 52  D1 00014          CMPL    BLOCKSIZE, #65536                ; 0273
                                         0F  1A 0001B          BGTRU   1$                               ;
                                         63  96 0001D          INCB    ERRORCODE                        ; 0274
                                  08     AC  DD 0001F          PUSHL   BLOCKADDR                        ; 0275
                                         52  DD 00022          PUSHL   BLOCKSIZE                        ;
                 0000V  CF               02  FB 00024          CALLS   #2, ALLOCATE                     ;
                        17               50  E8 00029          BLBS    R0, 2$                           ;
                        7E               63  9A 0002C 1$:      MOVZBL  ERRORCODE, -(SP)                 ; 0279
                                  08     AC  DD 0002F          PUSHL   BLOCKADDR                        ; 0278
                                         52  DD 00032          PUSHL   BLOCKSIZE                        ;
                                         03  DD 00034          PUSHL   #3                               ; 0277
                             00000000G   8F  DD 00036          PUSHL   #LIN$_MEMBUG                     ;
           00000000G  00                 05  FB 0003C          CALLS   #5, LIB$STOP                     ;
                                             04 00043 2$:      RET                                      ; 0282

; Routine Size:  68 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  172     0283  1 ROUTINE ALLOCATE (SIZE, ADDRESS) =
;  173     0284  1 !
;  174     0285  1 !       ROUTINE TO DO ACTUAL ALLOCATION AND PROGRAM
;  175     0286  1 !       REGION EXPANSION
;  176     0287  1 !
;  177     0288  2 BEGIN
;  178     0289  2 LOCAL
;  179     0290  2         STATUS,
;  180     0291  2         MEMLIMITS : VECTOR [2];
;  181     0292  2
;  182     0293  2 LASTBLOCK = LNK$GL_MEMLHD [0];                          ! INITIALLY AT TOP OF FREE LIST
;  183     0294  2
;  184     0295  2 WHILE (NEWBLOCK = .LASTBLOCK [0]) NEQ 0                 ! FOLLOW DOWN FREE LIST
;  185     0296  3 DO  BEGIN
;  186     0297  3     IF .NEWBLOCK [1] EQL .SIZE                          ! LOOK FOR SUITABLE FREE BLOCK
;  187     0298  4     THEN BEGIN                                          ! EXACT SIZE MATCH
;  188     0299  4          LASTBLOCK [0] = .NEWBLOCK [0];                 ! SO LAST POINTS WHERE THIS ONE POINTED
;  189     0300  4          IF (.ADDRESS = NEWBLOCK [0]) LSSU .LNK$GL_MINADDR ! NOW RECORD LOWEST
;  190     0301  4          THEN LNK$GL_MINADDR = NEWBLOCK [0];            ! ALLOCATED ADDRESS
;  191     0302  4          RETURN TRUE;                                   ! AND WE ARE DONE
;  192     0303  4          END
;  193     0304  3     ELSE IF .NEWBLOCK [1] GTRU .SIZE                    ! OR ONE LARGER THAN REQUESTED
;  194     0305  4          THEN BEGIN
;  195     0306  4               NEXTBLOCK     =  NEWBLOCK [0] + .SIZE;   ! IN WHICH CASE THERE IS A NEW
;  196     0307  4               NEXTBLOCK [0] = .NEWBLOCK [0];
;  197     0308  4               NEXTBLOCK [1] = .NEWBLOCK [1] - .SIZE;   ! NEXT BLOCK (THE PART REMAINING
;  198     0309  4               LASTBLOCK [0] =  NEXTBLOCK [0];          ! AFTER TAKING REQUESTED BLOCK OFF
;  199     0310  4
;  200     0311  4               IF (.ADDRESS = NEWBLOCK[0]) LSSU .LNK$GL_MINADDR  ! NOW RECORD LOWEST
;  201     0312  4               THEN LNK$GL_MINADDR = NEWBLOCK [0];              ! ALLOCATED ADDRESS
;  202     0313  4
;  203     0314  4               RETURN TRUE;                             ! AND WE ARE DONE
;  204     0315  4               END
;  205     0316  3          ELSE LASTBLOCK = NEWBLOCK[0];                 ! WHEN NOT SUITABLE THIS BLOCK BECOMES PREVIOUS BLOC
;  206     0317  2          END;                                          ! OF WHILE LOOP
;  207     0318  2 !
;  208     0319  2 !       AT THIS POINT WE HAVE REACHED THE END OF THE FREE
;  209     0320  2 !       MEMORY LIST WITHOUT FINDING A BLOCK OF REQUIRED SIZE.
;  210     0321  2 !       THUS, WE EXPAND THE ADDRESS SPACE AND ATTEMPT TO
;  211     0322  2 !       ALLOCATE FROM ADDITIONAL VIRTUAL MEMORY.
;  212     0323  2 !
;  213     0324  3 IF (STATUS = LNK$REQUESTMEM (LNK$C_MEMEXP, MEMLIMITS))  ! SUCCESSFULLY EXPANDED PROGRAM REGION
;  214     0325  3 THEN BEGIN                                              ! DEALLOCATE NEW SPACE TO END OF
;  215     0326  3
;  216     0327  3      IF NOT DEALLOCATE (MAXBLOCKSIZE, .MEMLIMITS [0], LASTBLOCK [0])
;  217     0328  3      THEN RETURN FALSE;
;  218     0329  3
;  219     0330  3      IF NOT ALLOCATE (.SIZE, .ADDRESS)                  ! FREE LIST THEN ALLOCATE FROM IT
;  220     0331  3      THEN RETURN FALSE;
;  221     0332  3
;  222     0333  3      RETURN TRUE;
;  223     0334  3      END
;  224     0335  2 ELSE SIGNAL_STOP (.STATUS);                             ! IS FATAL
;  225     0336  2
;  226     0337  2 RETURN FALSE;                                           ! DUMMY TO ELIMINATE INFO#212
;  227     0338  1 END;                                                    ! OF ALLOCATE ROUTINE
```

```
                                                001C 00000 ALLOCATE:
                                                           .WORD   Save R2,R3,R4                                   ; 0283
                                    54      0000'  CF  9E 00002    MOVAB   LASTBLOCK, R4                           ;
                                    5E             08  C2 00007    SUBL2   #8, SP                                  ;
                                    64      0000'  CF  9E 0000A    MOVAB   LNK$GL_MEMLHD, LASTBLOCK                ; 0293
                                    53        04   AC  D0 0000F    MOVL    SIZE, R3                                ; 0297
                                    52             64  D0 00013 1$: MOVL    LASTBLOCK, R2                           ; 0295
                                F8  A4             62  D0 00016    MOVL    (R2), NEWBLOCK                          ;
                                               3D  13 0001A    BEQL    5$                                      ;
                                    51        F8   A4  D0 0001C    MOVL    NEWBLOCK, R1                            ; 0297
                                    53        04   A1  D1 00020    CMPL    4(R1), R3                               ;
                                               05  12 00024    BNEQ    2$                                      ;
                                    62             61  D0 00026    MOVL    (R1), (R2)                              ; 0299
                                               17  11 00029    BRB     3$                                      ; 0300
                                               27  1B 0002B 2$: BLEQU   4$                                      ; 0304
                            FC  A4              51  53  C1 0002D    ADDL3   R3, R1, NEXTBLOCK                       ; 0306
                                    50        FC   A4  D0 00032    MOVL    NEXTBLOCK, R0                           ; 0307
                                    60             61  D0 00036    MOVL    (R1), (R0)                              ;
                            04  A0      04      A1  53  C3 00039    SUBL3   R3, 4(R1), 4(R0)                        ; 0308
                                    62             50  D0 0003F    MOVL    R0, (R2)                                ; 0309
                                  08        BC   51  D0 00042 3$: MOVL    R1, @ADDRESS                            ; 0311
                                0000'       CF  51  D1 00046    CMPL    R1, LNK$GL_MINADDR                      ;
                                               3A  1E 0004B    BGEQU   6$                                      ;
                                0000'       CF  51  D0 0004D    MOVL    R1, LNK$GL_MINADDR                      ; 0312
                                               33  11 00052    BRB     6$                                      ; 0314
                                    64             51  D0 00054 4$: MOVL    R1, LASTBLOCK                           ; 0316
                                               BA  11 00057    BRB     1$                                      ; 0295
                                               5E  DD 00059 5$: PUSHL   SP                                      ; 0324
                                    7E        80   8F  9A 0005B    MOVZBL  #128, -(SP)                             ;
                                0000V       CF  02  FB 0005F    CALLS   #2, LNK$REQUESTMEM                      ;
                                    24             50  E9 00064    BLBC    STATUS, 7$                              ;
                                               64  DD 00067    PUSHL   LASTBLOCK                               ; 0327
                                          04   AE  DD 00069    PUSHL   MEMLIMITS                               ;
                                  00010000   8F  DD 0006C    PUSHL   #65536                                  ;
                                0000V       CF  03  FB 00072    CALLS   #3, DEALLOCATE                          ;
                                    1A             50  E9 00077    BLBC    R0, 8$                                  ;
                                          08   AC  DD 0007A    PUSHL   ADDRESS                                 ; 0330
                                               53  DD 0007D    PUSHL   R3                                      ;
                                FF7C        CF  02  FB 0007F    CALLS   #2, ALLOCATE                            ;
                                    0D             50  E9 00084    BLBC    R0, 8$                                  ;
                                    50             01  D0 00087 6$: MOVL    #1, R0                                  ; 0333
                                                   04  0008A    RET                                             ;
                                               50  DD 0008B 7$: PUSHL   STATUS                                  ; 0335
                            00000000G       00      01  FB 0008D    CALLS   #1, LIB$STOP                            ;
                                               50  D4 00094 8$: CLRL    R0                                      ; 0338
                                                   04  00096    RET                                             ;

; Routine Size:  151 bytes,    Routine Base:  $CODE$ + 0044
```

```
;  229          0339  1 GLOBAL ROUTINE LNK$REQUESTMEM ( PAGES, RANGE ) =
;  230          0340  1 !
;  231          0341  1 !       THIS ROUTINE MAKES CALLS TO THE SYSTEM SERVICE $EXPREG TO
;  232          0342  1 !       EXPAND THE PROGRAM REGION.  IF THE REQUEST SUCCEEDS, THE
;  233          0343  1 !       RANGE IS RETURNED TO THE CALLING PROGRAM IN THE RANGE VECTOR.
;  234          0344  1 !       IF THE REQUEST FAILS, THE AMOUNT OF MEMORY ACTUALLY GRANTED IS
;  235          0345  1 !       DELETED VIA THE $DELTVA SYSTEM SERVICE, AND THE ROUTINE VALUE
;  236          0346  1 !       IS FALSE.
;  237          0347  1 !
;  238          0348  2 BEGIN
;  239          0349  2 LOCAL STATUS ;
;  240          0350  2
;  241          0351  2 STATUS = $EXPREG (PAGCNT=.PAGES, RETADR=.RANGE);! MAKE  REQUEST
;  242          0352  2
;  243          0353  2 IF NOT .STATUS                                  ! IF PROBLEM
;  244          0354  3 THEN BEGIN                                      ! THEN THROW AWAY WHAT WE WERE
;  245          0355  3      $DELTVA ( INADR=.RANGE );                  ! GRANTED, 'CAUSE IT WASN'T ENOUGH
;  246          0356  3      IF  .STATUS EQL SS$_EXQUOTA                !  AND MAP THE RETURN STATUS
;  247          0357  3      THEN STATUS = LIN$_EXPAGQUO
;  248          0358  3      ELSE STATUS = LIN$_MEMFUL ;
;  249          0359  2      END ;
;  250          0360  2 RETURN .STATUS
;  251          0361  1 END;


                                                .EXTRN  SYS$EXPREG, SYS$DELTVA

                                    0004 00000          .ENTRY  LNK$REQUESTMEM, Save R2                 ; 0339
                                 7E  7C 00002           CLRQ    -(SP)                                   ; 0351
                          7E      04  AC  7D 00004           MOVQ    PAGES, -(SP)                            ;
                00000000G 00          04  FB 00008           CALLS   #4, SYS$EXPREG                          ;
                          52          50  D0 0000F           MOVL    R0, STATUS                              ;
                          21          52  E8 00012           BLBS    STATUS, 2$                              ; 0353
                                      7E  7C 00015           CLRQ    -(SP)                                   ; 0355
                              08      AC  DD 00017           PUSHL   RANGE                                   ;
                00000000G 00          03  FB 0001A           CALLS   #3, SYS$DELTVA                          ;
                          1C          52  D1 00021           CMPL    STATUS, #28                             ; 0356
                                      09  12 00024           BNEQ    1$                                      ;
                          52 00000000G  8F  D0 00026           MOVL    #LIN$_EXPAGQUO, STATUS                  ; 0357
                                      07  11 0002D           BRB     2$                                      ;
                          52 00000000G  8F  D0 0002F 1$:       MOVL    #LIN$_MEMFUL, STATUS                    ; 0358
                          50          52  D0 00036 2$:       MOVL    STATUS, R0                              ; 0360
                                          04 00039           RET                                             ; 0361

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 00DB
```

```
;  253          0362  1 GLOBAL ROUTINE LNK$DEALBLK (SIZE, BLOCKADDR) : NOVALUE =
;  254          0363  1 !
;  255          0364  1 !       ROUTINE TO DEALLOCATE A BLOCK TO THE FREE
;  256          0365  1 !       MEMORY LIST AFTER CHECKING ITS SIZE
;  257          0366  1 !
;  258          0367  2 BEGIN
;  259          0368  2
;  260          0369  2 LOCAL
;  261          0370  2         BLOCKSIZE;
;  262          0371  2
;  263          0372  2 ERRORCODE = 10;                                         ! INITIALIZE ERROR CODE
;  264          0373  2
;  265          0374  2 BLOCKSIZE = (.SIZE + 7) AND ( NOT 7);                   ! ROUND UP TO A MULTIPLE OF 8 BYTES
;  266          0375  2
;  267          0376  2 IF .BLOCKSIZE EQL 0                                     ! CHECK BLOCK SIZE IS
;  268          0377  2 OR .BLOCKSIZE GTRU MAXBLOCKSIZE                         ! LEGAL AND THAT IT LIES
;  269          0378  3 OR (ERRORCODE = .ERRORCODE + 1;                         !
;  270          0379  2    (.BLOCKADDR + .BLOCKSIZE - 1)) GTRU CONTROL_REGION   ! COMPLETELY WITHIN PROGRAM REGION
;  271          0380  2                                                        ! AND IF NOT...
;  272          0381  3 OR (ERRORCODE = .ERRORCODE + 1;                         ! ISSUE FATAL ERROR MESSAGE
;  273          0382  3    .BLOCKADDR LSSU .LNK$GL_MINADDR)                    ! ALSO IF BELOW MINIMUM ALLOCATED ADDRESS
;  274          0383  2 OR NOT DEALLOCATE (.BLOCKSIZE,.BLOCKADDR,LNK$GL_MEMLHD) ! ATTEMPT DEALLOCATION
;  275          0384  2 THEN SIGNAL_STOP (LIN$_MEMBUG,3                        ! ISSUING FATAL ERROR IF FAILURE
;  276          0385  2                   ,.BLOCKSIZE,.BLOCKADDR,.ERRORCODE
;  277          0386  2                   );
;  278          0387  2 RETURN;                                                ! OTHERWISE JUST RETURN
;  279          0388  1 END;

                                000C 00000          .ENTRY  LNK$DEALBLK, Save R2,R3                 ; 0362
                         53     0000' CF  9E 00002  MOVAB   ERRORCODE, R3
                         63           0A  90 00007  MOVB    #10, ERRORCODE                          ; 0372
          50        C4   AC           07  C1 0000A  ADDL3   #7, SIZE, R0                            ; 0374
          52             50           07  CB 0000F  BICL3   #7, R0, BLOCKSIZE
                                      36  13 00013  BEQL    1$                                      ; 0376
              00010000   8F           52  D1 00015  CMPL    BLOCKSIZE, #65536                       ; 0377
                                      2D  1A 0001C  BGTRU   1$
                                      63  96 0001E  INCB    ERRORCODE                               ; 0378
          50             52     08    AC  C1 00020  ADDL3   BLOCKADDR, BLOCKSIZE, R0                ; 0379
                                      50  D7 00025  DECL    R0
              40000000   8F           50  D1 00027  CMPL    R0, #1073741824
                                      1B  1A 0002E  BGTRU   1$
                                      63  96 00030  INCB    ERRORCODE                               ; 0381
                   0000' CF     08    AC  D1 00032  CMPL    BLOCKADDR, LNK$GL_MINADDR               ; 0382
                                      11  1F 00038  BLSSU   1$
                              0000' CF  9F 0003A  PUSHAB  LNK$GL_MEMLHD                           ; 0383
                                 08   AC  DD 0003E  PUSHL   BLOCKADDR
                                      52  DD 00041  PUSHL   BLOCKSIZE
                   0000V      CF        03  FB 00043  CALLS   #3, DEALLOCATE
                              17        50  E8 00048  BLBS    R0, 2$
                              7E        63  9A 0004B  1$:    MOVZBL  ERRORCODE, -(SP)                ; 0385
                                 08   AC  DD 0004E  PUSHL   BLOCKADDR
                                      52  DD 00051  PUSHL   BLOCKSIZE                               ; 0384
                                      03  DD 00053  PUSHL   #3
```

```
                                00000000G  8F  DD 00055          PUSHL   #LIN$_MEMBUG                    ;
                     00000000G  00          05  FB 0005B          CALLS   #5, LIB$STOP                    ;
                                            04 00062 2$:        RET                                     ; 0388

; Routine Size:  99 bytes,    Routine Base:  $CODE$ + 0115
```

```
;  281     0389  1 ROUTINE DEALLOCATE (SIZE, ADDRESS, LISTHEAD) =
;  282     0390  1 !
;  283     0391  1 !       ROUTINE TO PUT A BLOCK ONTO A LIST OF FREE BLOCKS,
;  284     0392  1 !       WITH MAXIMAL COMPACTION
;  285     0393  1 !
;  286     0394  2 BEGIN
;  287     0395  2 LASTBLOCK = .LISTHEAD;                          ! PREVIOUS BLOCK INITIALLY THE LISTHEAD
;  288     0396  2 NEWBLOCK  = .ADDRESS;                           ! CURRENT BLOCK IS TO BE INSERTED
;  289     0397  2
;  290     0398  2 WHILE (NEXTBLOCK = .LASTBLOCK [0]) NEQ 0        ! FOLLOW DOWN FREE LIST TILL
;  291     0399  3 DO  BEGIN                                       ! THE END, OR TILL WE REACH
;  292     0400  3     IF NEWBLOCK [0] LEQU NEXTBLOCK [0]
;  293     0401  4     THEN BEGIN                                  ! THE POSITION FOR INSERTION.
;  294     0402  4          IF NEWBLOCK [0]+.SIZE EQL NEXTBLOCK [0]
;  295     0403  5          THEN   BEGIN                           ! HERE WE COMPACT WITH NEXT BLOCK
;  296     0404  5                 NEWBLOCK [0] = .NEXTBLOCK [0];
;  297     0405  5                 NEWBLOCK [1] = .NEXTBLOCK [1] + .SIZE;
;  298     0406  5                 END
;  299     0407  5          ELSE   BEGIN
;  300     0408  5                 IF NEWBLOCK [0] + .SIZE GTRU NEXTBLOCK [0] ! IF THE BLOCK TO DEALLOCATE
;  301     0409  6                 THEN (ERRORCODE = .ERRORCODE + 1;       ! EXTENDS INTO NEXT FREE BLOCK
;  302     0410  5                       RETURN FALSE);                    ! AND RETURN FAILURE
;  303     0411  5                 NEWBLOCK [0] = NEXTBLOCK [0];           ! ELSE SET POINTER AND SIZE SINCE NO
;  304     0412  5                 NEWBLOCK [1] = .SIZE;                   ! FORWARD COMPACTION NEEDED
;  305     0413  4                 END;
;  306     0414  4          IF NEWBLOCK [0] EQL LASTBLOCK [0]+.LASTBLOCK [1]
;  307     0415  5          THEN   BEGIN                           ! HERE WE COMPACT WITH PREVIOUS
;  308     0416  5                 LASTBLOCK [0] = .NEWBLOCK [0];  ! BLOCK
;  309     0417  5                 LASTBLOCK [1] = .NEWBLOCK [1] +.LASTBLOCK [1];
;  310     0418  5                 END
;  311     0419  4          ELSE                                   ! NO BACKWARD COMPACTION BUT...
;  312     0420  5                 BEGIN                          ! MUST CHECK THAT BLOCK TO
;  313     0421  5                 IF NEWBLOCK [0] LSSU LASTBLOCK [0] + .LASTBLOCK [1]
;  314     0422  5                                                ! DEALLOCATE IS NOT PARTIALLY IN
;  315     0423  6                 THEN (ERRORCODE = .ERRORCODE + 1;       ! PREVIOUS HOLE -- FAILURE IF SO
;  316     0424  5                       RETURN FALSE);
;  317     0425  5                 LASTBLOCK [0] = NEWBLOCK [0];           ! IF OK PREVIOUS POINTS TO NEW ONE.
;  318     0426  4                 END;                                    ! AND WE ARE DONE COMPACTING
;  319     0427  4          RETURN TRUE;                                   ! SO RETURN SUCCESS.
;  320     0428  4          END
;  321     0429  3     ELSE LASTBLOCK = NEXTBLOCK [0];             ! NOT THERE YET SO LAST BLOCK IS ONE JUST TESTED
;  322     0430  2     END;                                        ! OF WHILE LOOP
;  323     0431  2 !
;  324     0432  2 ! THE BLOCK TO DEALLOCATE IS BEYOND LAST HOLE
;  325     0433  2 !
;  326     0434  2 IF NEWBLOCK[0] LSSU LASTBLOCK[0] + .LASTBLOCK[1]        ! BUT IF IT STARTS WITHIN
;  327     0435  3 THEN (ERRORCODE = .ERRORCODE + 1;                       ! THE LAST HOLE - FAIL IT
;  328     0436  3                 RETURN FALSE)
;  329     0437  2 ELSE
;  330     0438  3         BEGIN                                           ! OTHERWISE CHECK FOR COMPACTION
;  331     0439  3         IF NEWBLOCK[0] EQL LASTBLOCK[0] + .LASTBLOCK[1] ! WITH LAST HOLE
;  332     0440  3         THEN LASTBLOCK[1] = .LASTBLOCK[1] + .SIZE       ! AND ADD IN SIZE IF REQUIRED
;  333     0441  3         ELSE
;  334     0442  4                 BEGIN                           ! OTHERWISE JUST
;  335     0443  4                 NEWBLOCK[0] = 0;                ! PUT ON END OF FREE LIST.
;  336     0444  4                 NEWBLOCK[1] =.SIZE;
;  337     0445  4                 LASTBLOCK[0] = NEWBLOCK[0];
```

```
;  338     0446  3                END;                        ! AND ALL DONE
;  339     0447  3          RETURN TRUE;                      ! SO RETURN SUCCESS
;  340     0448  2          END;
;  341     0449  1 END;                        . OF ROUTINE



                                                001C 00000 DEALLOCATE:
                                                           .WORD   Save R2,R3,R4                   ; 0389
                        54      0000'  CF  9E 00002        MOVAB   LASTBLOCK, R4
                        64        0C   AC  D0 00007        MOVL    LISTHEAD, LASTBLOCK             ; 0395
                    F8  A4        08   AC  D0 0000B        MOVL    ADDRESS, NEWBLOCK               ; 0396
                        51        F8   A4  D0 00010        MOVL    NEWBLOCK, R1                    ; 0400
                        50            64  D0 00014 1$:     MOVL    LASTBLOCK, R0                   ; 0398
                    FC  A4            60  D0 00017        MOVL    (R0), NEXTBLOCK
                                      46  13 0001B        BEQL    6$
                        52        FC  A4  D0 0001D        MOVL    NEXTBLOCK, R2                   ; 0400
                        52            51  D1 00021        CMPL    R1, R2
                                      38  1A 00024        BGTRU   5$
                53      51        04  AC  C1 00026        ADDL3   SIZE, R1, R3                    ; 0402
                        52            53  D1 0002B        CMPL    R3, R2
                                      0C  12 0002E        BNEQ    2$
                        61            62  D0 00030        MOVL    (R2), (R1)                      ; 0404
            04  A1      04  A2       04  AC  C1 00033        ADDL3   SIZE, 4(R2), 4(R1)              ; 0405
                                      0A  11 0003A        BRB     3$                              ; 0402
                                      32  1A 0003C 2$:     BGTRU   7$                              ; 0408
                        61            52  D0 0003E        MOVL    R2, (R1)                        ; 0411
                    04  A1        04  AC  D0 00041        MOVL    SIZE, 4(R1)                     ; 0412
                52      50        04  A0  C1 00046 3$:     ADDL3   4(R0), R0, R2                   ; 0414
                        52            51  D1 0004B        CMPL    R1, R2
                                      0A  12 0004E        BNEQ    4$
                        60            61  D0 00050        MOVL    (R1), (R0)                      ; 0416
                    04  A0        04  A1  C0 00053        ADDL2   4(R1), 4(R0)                    ; 0417
                                      2E  11 00058        BRB     11$                             ; 0414
                                      29  1E 0005A 4$:     BGEQU   10$                             ; 0421
                                      12  11 0005C        BRB     7$                              ; 0423
                        64            52  D0 0005E 5$:     MOVL    R2, LASTBLOCK                   ; 0429
                                      B1  11 00061        BRB     1$                              ; 0398
                        50            64  D0 00063 6$:     MOVL    LASTBLOCK, R0                   ; 0434
            52          50        04  A0  C1 00066        ADDL3   4(R0), R0, R2
                        52            51  D1 0006B        CMPL    R1, R2
                                      05  1E 0006E        BGEQU   8$
                            F4    A4  96  00070 7$:     INCB    ERRORCODE                       ; 0435
                                      17  11 00073        BRB     12$                             ; 0438
                                      07  12 00075 8$:     BNEQ    9$                              ; 0439
                    04  A0      04  AC  C0 00077        ADDL2   SIZE, 4(R0)                     ; 0440
                                      0A  11 0007C        BRB     11$
                                      61  D4 0007E 9$:     CLRL    (R1)                            ; 0443
                    04  A1      04  AC  D0 00080        MOVL    SIZE, 4(R1)                     ; 0444
                        60            51  D0 00085 10$:    MOVL    R1, (R0)                        ; 0445
                        50            01  D0 00088 11$:    MOVL    #1, R0                          ; 0447
                                          04  0008B        RET                                     ; 0438
                                      50  D4 0008C 12$:    CLRL    R0                              ; 0449
                                          04  0008E        RET
```

```
; Routine Size:  143 bytes,    Routine Base:  $CODE$ + 0178

;  342          0450  0 END ELUDOM                ! OF MODULE


                                                        .EXTRN  LIB$STOP

;                                PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  $GLOBAL$                        12  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                           16  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                         519  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  .  ABS  .                        0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)


;                              Library Statistics

;                                      -------- Symbols --------      Pages       Processing
;       File                           Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]LIB.L32;1        18619        11         0      1000          00:01.9

;                                   COMMAND QUALIFIERS

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

; Size:          519 code + 28 data bytes
; Run Time:        00:12.7
; Elapsed Time:    00:29.3
; Lines/CPU Min:    2119
; Lexemes/CPU-Min: 15899
; Memory Used:  103 pages
; Compilation Complete
```