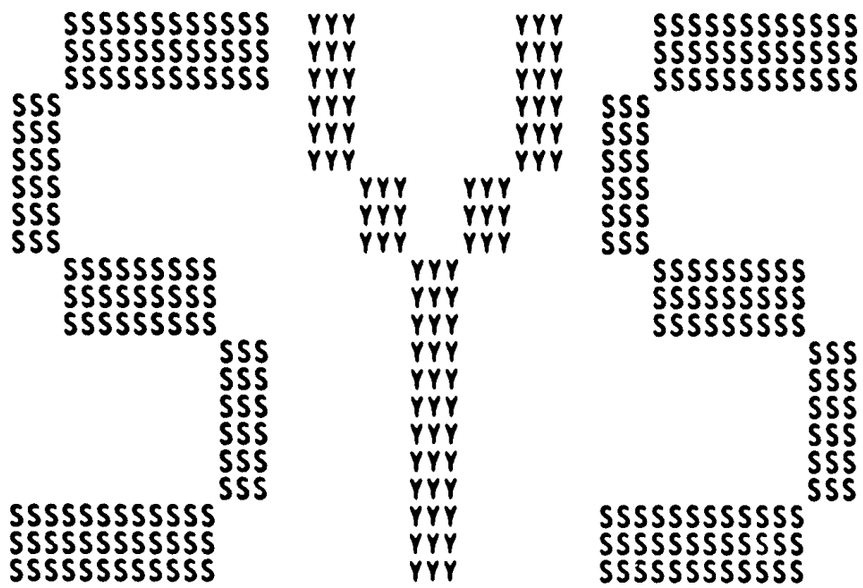

FILEID**SYSLOGNAM

```
  SSSSSSSS  YY      YY    SSSSSSSS  LL            OOOOOO      GGGGGGGG  NN      NN    AAAAAA    MM      MM
  SSSSSSSS  YY      YY    SSSSSSSS  LL            OOOOOO      GGGGGGGG  NN      NN    AAAAAA    MM      MM
SS          YY      YY  SS          LL          OO      OO  GG          NN      NN  AA      AA  MMMM  MMMM
SS          YY      YY  SS          LL          OO      OO  GG          NN      NN  AA      AA  MMMM  MMMM
SS            YY  YY    SS          LL          OO      OO  GG          NNNN    NN  AA      AA  MM  MM  MM
SS            YY  YY    SS          LL          OO      OO  GG          NNNN    NN  AA      AA  MM  MM  MM
  SSSSSS        YY        SSSSSS    LL          OO      OO  GG          NN  NN  NN  AA      AA  MM      MM
  SSSSSS        YY        SSSSSS    LL          OO      OO  GG          NN  NN  NN  AA      AA  MM      MM
        SS      YY              SS  LL          OO      OO  GG  GGGGGG  NN    NNNN  AAAAAAAAAA  MM      MM
        SS      YY              SS  LL          OO      OO  GG  GGGGGG  NN    NNNN  AAAAAAAAAA  MM      MM
        SS      YY              SS  LL          OO      OO  GG      GG  NN      NN  AA      AA  MM      MM      ....
        SS      YY              SS  LL          OO      OO  GG      GG  NN      NN  AA      AA  MM      MM      ....
SSSSSSSS        YY      SSSSSSSS    LLLLLLLLLL    OOOOOO      GGGGGG    NN      NN  AA      AA  MM      MM      ....
SSSSSSSS        YY      SSSSSSSS    LLLLLLLLLL    OOOOOO      GGGGGG    NN      NN  AA      AA  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
```

Table of contents

```
0000     1          .TITLE  SYSLOGNAM - Old System Service Interface to Logical Names
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;**********************************************************************
0000     6  ;*                                                                    *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
0000     9  ;*  ALL RIGHTS RESERVED.                                              *
0000    10  ;*                                                                    *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
0000    16  ;*  TRANSFERRED.                                                       *
0000    17  ;*                                                                    *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0000    20  ;*  CORPORATION.                                                       *
0000    21  ;*                                                                    *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
0000    24  ;*                                                                    *
0000    25  ;*                                                                    *
0000    26  ;**********************************************************************
0000    27  ;
0000    28  ; D. N. CUTLER 10-OCT-76
0000    29  ;
0000    30  ; Old System Services Interfaces to Manipulate Logical Names
0000    31  ;
0000    32  ;       $CRELOG - CREATE    Logical Name
0000    33  ;       $DELLOG - DELETE    Logical Name
0000    34  ;       $TRNLOG - TRANSLATE Logical Name
0000    35  ;
0000    36  ;   The arguments are probed by these routines before they are passed off
0000    37  ;   to the new system services so that the previous access mode is correct.
0000    38  ;   The new system services are call thru the CMKRNL dispatcher so that the
0000    39  ;   previous mode is CMKRNL so that literal data in this code can be accessed.
0000    40  ;
0000    41  ;
0000    42  ; MODIFICATION HISTORY:
0000    43  ;
0000    44  ;       V03-014 MSH0053         Michael S. Harvey       29-May-1984
0000    45  ;               Probe argument lists for $CRELOG, $TRNLOG and $DELLOG to
0000    46  ;               ensure accessibility. Also, make sure that the minimum
0000    47  ;               number of arguments for these services has been specified.
0000    48  ;               This ensures that behavior observed prior to these services
0000    49  ;               becoming caller's mode services is maintained.
0000    50  ;
0000    51  ;       V03-013 TMK0003         Todd M. Katz            15-Nov-1983
0000    52  ;               Fix a bug introduced by TMK0002. The address of $TRNLOG's
0000    53  ;               optional output table parameter must be loaded into R8
0000    54  ;               before R8 is ever referenced. With the TMK0002 change if the
0000    55  ;               logical name started with an underscore (and no translation
0000    56  ;               is attempted), or DSBMSK specified that no table was to be
0000    57  ;               searched, then R8 was never being initialized before it was
```

```
0000     58 ;                referenced. As one might imagine, this lead to unpredictable
0000     59 ;                results.
0000     60 ;
0000     61 ;        V03-012 TMK0002         Todd M. Katz            19-Nov-1983
0000     62 ;                Optimize $TRNLOG in those cases when the caller has not
0000     63 ;                requested that the table in which the translated logical name
0000     64 ;                was found be returned. The caller requested that the table be
0000     65 ;                returned by specifying the optional system service output
0000     66 ;                parameter TABLE. The optimization consists of calling $TRNLNM
0000     67 ;                only once giving as the table name a logical name that
0000     68 ;                equates to the list of one or more tables that the caller wants
0000     69 ;                searched for the target logical name. The logical names that
0000     70 ;                may potentially serve as table names are created at system
0000     71 ;                initialization time. The optional system service input
0000     72 ;                parameter DSBMSK is used to select which logical name is used
0000     73 ;                as the table name. The reason why this optimization is
0000     74 ;                applicable only when TABLE has not been specified is because
0000     75 ;                it is not obvious as to which table the logical name was found
0000     76 ;                when $TRNLNM is presented with a list of tables to search. Thus,
0000     77 ;                the TABLE information could not be appropriately returned.
0000     78 ;
0000     79 ;        V03-011 TMK0001         Todd M. Katz            23-Oct-1983
0000     80 ;                Change the logical name table names that these old services
0000     81 ;                use. Replace LNM$PROCESS with LOG$PROCESS, LNM$GROUP with
0000     82 ;                LOG$GROUP, and LNM$SYSTEM with LOG$SYSTEM. These logical names
0000     83 ;                which serve as table names for all three of the old system
0000     84 ;                services (SYS$CRELOG, SYS$DELLOG, and SYS$TRNLOG) are setup at
0000     85 ;                system initialization time, and maybe changed by users in
0000     86 ;                order for them to make use of the features of the new logical
0000     87 ;                name services without converting existing programs to use the
0000     88 ;                new services.
0000     89 ;
0000     90 ;        V03-010 RAS0175         Ron Schaefer            28-Jul-1983
0000     91 ;                Add support in $CRELOG to strip a leading "_" and set
0000     92 ;                LNM$M_TERMINAL;  remove recognition of LNM$M_CONCEALED
0000     93 ;                from $TRNLOG; and make $TRNLOG recognize PPF-format names.
0000     94 ;
0000     95 ;        V03-009 RAS0167         Ron Schaefer            21-Jul-1983
0000     96 ;                Add special condition handler to $TRNLOG in order to
0000     97 ;                properly process programs with system service failure mode
0000     98 ;                exception enabled.
0000     99 ;
0000    100 ;        V03-008 RAS0171         Ron Schaefer            20-Jul-1983
0000    101 ;                Make the register masks in RAS0170 be constant and
0000    102 ;                not relocatable.
0000    103 ;
0000    104 ;        V03-007 RAS0170         Ron Schaefer            18-Jul-1983
0000    105 ;                Fixed trashed registers from a MOVC3 when prefixing
0000    106 ;                "_"s in $TRNLOG.
0000    107 ;
0000    108 ;        V03-006 RAS0165         Ron Schaefer            5-Jul-1983
0000    109 ;                Eliminate the access mode argument to $TRNLNM from $TRNLOG.
0000    110 ;                Make sure there is a non-kernel mode access mode argument
0000    111 ;                for $CRELOG and $DELLOG.
0000    112 ;
0000    113 ;        V03-005 DMW4059         DMWalp                  23-Jun-1983
0000    114 ;                Change $xxLNM value parameters to be by reference
```

```
0000    115 ;
0000    116 ;       V03-004 DMW4055         DMWalp                  22-Jun-1983
0000    117 ;               Make system services to be mode of caller.  Removed all
0000    118 ;               priv checks, probes and added condition handler setup.
0000    119 ;
0000    120 ;       V03-003 DMW4049         DMWalp                  15-Jun-1983
0000    121 ;               Save R0 during MOVC3
0000    122 ;
0000    123 ;       V03-002 DMW4030         DMWalp                  20-Oct-1982
0000    124 ;               Rewrite to call new system services.
0000    125 ;
0000    126 ;       V03-001 KDM0002         Kathleen D  Morse       28-Jun-1982
0000    127 ;               Added $PRDEF.
0000    128 ;
```

```
                     0000   131 ;
                     0000   132 ; MACRO library calls
                     0000   133 ;
                     0000   134         $CHFDEF                         ; define condition handling
                     0000   135         $CRELOGDEF                      ; define CRELOG symbols
                     0000   136         $CRELNMDEF                      ; define CRELNM arglist
                     0000   137         $DELLOGDEF                      ; define DELLOG symbols
                     0000   138         $DELLNMDEF                      ; define DELLNM arglist
                     0000   139         $TRNLOGDEF                      ; define TRNLOG symbols
                     0000   140         $TRNLNMDEF                      ; define TRNLNM arglist
                     0000   141         $LNMDEF                         ; define new log nam offsets
                     0000   142         $LOGDEF                         ; define log offsets
                     0000   143         $PSLDEF                         ; define PSL offsets
                     0000   144         $SSDEF                          ; define system status values
                     0000   145
                     0000   146 ;
                     0000   147 ; LOCAL SYMBOLS
                     0000   148 ;
                     0000   149 ; argument list offset definitions for CREATE logical name
                     0000   150 ;
00000004             0000   151 TBLFLG   = 4                            ; logical name table number
00000008             0000   152 LOGNAM   = 8                            ; address of logical name string
                     0000   153                                         ;  descriptor
0000000C             0000   154 EQLNAM   = 12                           ; address of equivalence name string
                     0000   155                                         ;  descriptor
00000010             0000   156 CRACMODE = 16                           ; access mode
                     0000   157
                     0000   158 ;
                     0000   159 ; argument list offset definitions for DELETE logical name
                     0000   160 ;
00000004             0000   161 TBLFLG   = 4                            ; logical name table number
00000008             0000   162 LOGNAM   = 8                            ; address of logical name string
                     0000   163                                         ;  descriptor
0000000C             0000   164 DLACMODE = 12                           ; access mode
                     0000   165
                     0000   166 ;
                     0000   167 ; argument list offset definitions for TRANSLATE logical name
                     0000   168 ;
00000004             0000   169 TRLOGNAM = 4                            ; address of logical name string
                     0000   170                                         ;  descriptor
00000008             0000   171 RSLLEN   = 8                            ; address to store length of result
                     0000   172                                         ;  string
0000000C             0000   173 RSLBUF   = 12                           ; address of result buffer descriptor
00000010             0000   174 TABLE    = 16                           ; address to store translation table
                     0000   175                                         ;  number
00000014             0000   176 TRACMODE = 20                           ; address to store assignment access
                     0000   177                                         ;  mode
00000018             0000   178 DSBMSK   = 24                           ; table search disable mask
                     0000   179
                     0000   180 ;
                     0000   181 ; local area storage defination for $CRELNM call
                     0000   182 ;
00000000             0000   183 RET_TRANSIZE = 0                        ; the returned translation size
00000004             0000   184 RET_TRANATTR = 4                        ; the returned translation attributes
00000008             0000   185 RET_ACMODE   = 8                        ; the returned access mode
0000000C             0000   186 IN_ACMODE    = 12                       ; the input access mode
00000010             0000   187 LOCAL_AREA   = 16                       ; the number of byte the above take
```

```
                                0000   188
                       0000001B 0000   189 ESCAPE = 27                          ; ASCII escape char for PPFs
                                0000   190
                       00000000 0000   191         .PSECT  YF$SYSLOGNAM
                                0000   192 ;
                                0000   193 ; Literal string definitions
                                0000   194 ;
                                0000   195 TABLE_NAME_LIST:                     ; array of descriptor addresses
                       00000028' 0000   196         .ADDRESS SYSTEM_TABLE      ;  pointer to system table name
                       0000003A' 0004   197         .ADDRESS GROUP_TABLE       ;  pointer to group table name
                       0000004B' 0008   198         .ADDRESS PROCESS_TABLE     ;  pointer to process table name
                                000C   199
                                000C   200 TRNLOG_TABLE:                       ; array of desc addrs equated to DSBMSK
                       000000B5' 000C   201         .ADDRESS PGS_TABLE         ; DSBMSK = 0 -> PROCESS, GROUP, & SYSTEM
                       0000007A' 0010   202         .ADDRESS PG_TABLE          ; DSBMSK = 1 -> PROCESS & GROUP
                       00000097' 0014   203         .ADDRESS PS_TABLE          ; DSBMSK = 2 -> PROCESS & SYSTEM
                       0000004B' 0018   204         .ADDRESS PROCESS_TABLE     ; DSBMSK = 3 -> PROCESS
                       0000005E' 001C   205         .ADDRESS GS_TABLE          ; DSBMSK = 4 -> GROUP & SYSTEM
                       0000003A' 0020   206         .ADDRESS GROUP_TABLE       ; DSBMSK = 5 -> GROUP
                       00000028' 0024   207         .ADDRESS SYSTEM_TABLE      ; DSBMSK = 6 -> SYSTEM
                                0028   208
                                0028   209 SYSTEM_TABLE:
59 53 24 47 4F 4C 00000030'010E0000' 0028   210         .ASCID  "LOG$SYSTEM"       ; system table name
                   4D 45 54 53  0036
                                003A   211 GROUP_TABLE:
52 47 24 47 4F 4C 00000042'010E0000' 003A   212         .ASCID  "LOG$GROUP"        ; group table name
                      50 55 4F  0048
                                004B   213 PROCESS_TABLE:
52 50 24 47 4F 4C 00000053'010E0000' 004B   214         .ASCID  "LOG$PROCESS"      ; process table name
                53 53 45 43 4F  0059
                                005E   215
                                005E   216 GS_TABLE:                           ; group and system table names
47 4F 4C 4E 52 54 00000066'010E0000' 005E   217         .ASCID  "TRNLOG$_GROUP_SYSTEM"
54 53 59 53 5F 50 55 4F 52 47 5F 24  006C
                         4D 45  0078
                                007A   218
                                007A   219 PG_TABLE:                           ; process and group table names
47 4F 4C 4E 52 54 00000082'010E0000' 007A   220         .ASCID  "TRNLOG$_PROCESS_GROUP"
52 47 5F 53 53 45 43 4F 52 50 5F 24  0088
                      50 55 4F  0094
                                0097   221
                                0097   222 PS_TABLE:                           ; process and system table names
47 4F 4C 4E 52 54 0000009F'010E0000' 0097   223         .ASCID  "TRNLOG$_PROCESS_SYSTEM"
59 53 5F 53 53 45 43 4F 52 50 5F 24  00A5
                   4D 45 54 53  00B1
                                00B5   224
                                00B5   225 PGS_TABLE:                          ; process, group, and system table names
47 4F 4C 4E 52 54 000000BD'010E0000' 00B5   226         .ASCID  "TRNLOG$_PROCESS_GROUP_SYSTEM"
52 47 5F 53 53 45 43 4F 52 50 5F 24  00C3
      4D 45 54 53 59 53 5F 50 55 4F  00CF
                                00D9   227
                                00D9   228 TABLE_MODE:
                          03 01 00D9   229         .BYTE   1, 3                   ; System Table is exec mode
                                00DB   230                                        ; Group Table is user mode
                                00DB   231
                                00DB   232 LNM_TERM:
                       00000200 00DB   233         .LONG   LNM$M_TERMINAL         ; terminal attribute for $CRELNM
```

```
                        00DF    236          .SBTTL  CREATE Logical Name
                        00DF    237 ;+
                        00DF    238 ; EXE$CRELOG - CREATE Logical Name
                        00DF    239 ;
                        00DF    240 ; This service provides the capability to insert a logical name equivalence
                        00DF    241 ; into either the process, group, or system logical name table via the old
                        00DF    242 ; system service interface.
                        00DF    243 ;
                        00DF    244 ; INPUTS:
                        00DF    245 ;
                        00DF    246 ;       TBLFLG (AP)   = logical name table number.
                        00DF    247 ;       LOGNAM (AP)   = address of logical name string descriptor.
                        00DF    248 ;       EQLNAM (AP)   = address of equivalence name string descriptor.
                        00DF    249 ;       CRACMODE (AP) = access mode of logical name to be created.
                        00DF    250 ;
                        00DF    251 ; OUTPUTS:
                        00DF    252 ;
                        00DF    253 ;       R0 low bit clear indicates failure to create logical name table entry.
                        00DF    254 ;               R0 = exception turned in return status
                        00DF    255 ;               R0 = status returned from new sevice
                        00DF    256 ;
                        00DF    257 ;       R0 low bit set indicates successful completion.
                        00DF    258 ;               R0 = status returned from new sevice
                        00DF    259 ;
                        00DF    260 ;-
                        00DF    261
                01DC    00DF    262          .ENTRY  EXE$CRELOG,^M<R2,R3,R4,R6,R7,R8>
                        00E1    263          ;
                        00E1    264          ; Only registers R0-R4 are used, the other registers are saved for
                        00E1    265          ; compatability with old versions of the service
                        00E1    266          ;
                        00E1    267
                        00E1    268          ;
                        00E1    269          ; Return exception as return status
                        00E1    270          ;
                6D  00000000'EF  9E  00E1    271          MOVAB   L^EXE$SIGTORET,(FP)
                        00E8    272
                        00E8    273          ;
                        00E8    274          ; Verify that the argument list is accessible and has the minimum
                        00E8    275          ; number of arguments for successful execution of the service.
                        00E8    276          ;
                        00E8    277          ASSUME CRELOG$_NARGS LT 128
                    04  6C  91  00E8    278          CMPB    (AP),#CRELOG$_NARGS     ; Minimum number of arguments specified?
                      7C  1F  00EB    279          BLSSU   INSARG                  ; If LSSU, no, return error status
                   10 AC  D5  00ED    280          TSTL    <CRELOG$_NARGS*4>(AP)   ; Touch last argument, verify accessibility
                        00F0    281
                        00F0    282          ;
                        00F0    283          ; Calculate access mode and check table number
                        00F0    284          ;
            52  10 AC  02  00  EF  00F0    285          EXTZV   #0,#2,CRACMODE(AP),R2   ; get caller specified access mode
                     004D  30  00F6    286          BSBW    CHECKOUT                ; check table number and privilege
                        00F9    287
                        00F9    288
                        00F9    289          ; Set up itemlist for $CRELNM
                        00F9    290          ;  Default index ( LNM$_INDEX ) of zero
                        00F9    291          ;
                      7E  7C  00F9    292          CLRQ    -(SP)                   ; end of list, return size not needed
```

```
                                                K 10

                    54   D4  00FB   293          CLRL    R4                              ; flag no attributes
               50   0C BC   7D  00FD   294          MOVQ    @EQLNAM(AP),R0                  ; get equiv name descriptor
               61   5F 8F   91  0101   295          CMPB    #^A\_\,(R1)                     ; leading "_"?
                       06   12  0105   296          BNEQ    10$                             ; nope
                       50   D7  0107   297          DECL    R0                              ; reduce size
                       51   D6  0109   298          INCL    R1                              ; advance ptr
                       54   D6  010B   299          INCL    R4                              ; flag terminal attribute
                  7E   50   7D  010D   300 10$:     MOVQ    R0,-(SP)                        ; equivalence name descriptor
               02 AE   02   B0  0110   301          MOVW    #LNM$_STRING,2(SP)              ; stuff the item list code
                    0B 54   E9  0114   302          BLBC    R4,20$                          ; need an attribute item?
                       7E   D4  0117   303          CLRL    -(SP)                           ; no return len
                    BF AF   DF  0119   304          PUSHAL  LNM_TERM                        ; point to terminal attribute
               00030004 8F   DD  011C   305          PUSHL   #<4@16+<LNM$_ATTRIBUTES@16>>    ; item code and size
                  50   5E   D0  0122   306 20$:     MOVL    SP,R0                           ; save the pointer to the list
          51   FED6 CF43   D0  0125   307          MOVL    TABLE_NAME_LIST[R3],R1          ; level of indirection
                       52   DD  012B   308          PUSHL   R2                              ; save access mode for reference
                       04   DD  012D   309          PUSHL   #LNM$M_CRELOG                   ; save attributes for reference
                    52   5E   D0  012F   310          MOVL    SP,R2                           ; save pointer to ACMODE and ATTR
                                 0132   311
                                 0132   312          ;
                                 0132   313          ; Call System Service
                                 0132   314          ;
                                 0132   315 ;        $CRELNM_S  ATTR     = (R2),-
                                 0132   316 ;                   TABNAM   = (R1),-
                                 0132   317 ;                   LOGNAM   = @LOGNAM(AP),-
                                 0132   318 ;                   ACMODE   = 4(R2),-
                                 0132   319 ;                   ITMLST   = (R0)
                       60   DF  0132   320          PUSHAL  (R0)
                                 0134   321          ASSUME  CRELNM$_ITMLST EQ CRELNM$_ACMODE+4
                    04 A2   DF  0134   322          PUSHAL  4(R2)
                                 0137   323          ASSUME  CRELNM$_ACMODE EQ CRELNM$_LOGNAM+4
                    08 BC   DF  0137   324          PUSHAL  @LOGNAM(AP)
                                 013A   325          ASSUME  CRELNM$_LOGNAM EQ CRELNM$_TABNAM+4
                       61   DF  013A   326          PUSHAL  (R1)
                                 013C   327          ASSUME  CRELNM$_TABNAM EQ CRELNM$_ATTR+4
                       62   DF  013C   328          PUSHAL  (R2)
                                 013E   329          ASSUME  CRELNM$_NARGS  EQ 5
    80000000'9F        05   FB  013E   330          CALLS   #CRELNM$_NARGS,@#SYS$CRELNM - P1SYSVECTORS + ^X80000000
                                 0145   331          RET
```

```
                                0146   334          .SBTTL  Check table number, privileges and logical name string
                                0146   335 ;
                                0146   336 ; Check table number, and access mode
                                0146   337 ;
                                0146   338 CHECKOUT:                                ; check table number and access mode
                                0146   339          ;
                                0146   340          ; Logical name table number checkout
                                0146   341          ;
          53    04 AC     9A    0146   342          MOVZBL  TBLFLG(AP),R3           ; get logical name table number
             53    02     D1    014A   343          CMPL    #LOG$C_PROCESS,R3       ; legal table number?
                   08     19    014D   344          BLSS    10$                     ; number greater then 2
                   0C     13    014F   345          BEQL    20$                     ; number was 2 ( process table )
                                0151   346
                                0151   347          ;
                                0151   348          ; Set up access mode for group and system tables
                                0151   349          ;
          52    84 AF43   9A    0151   350          MOVZBL  TABLE_MODE[R3],R2       ; convert table to acmode
                                0156   351                                          ; group is user mode, system is exec
                         05     0156   352          RSB                             ;
                                0157   353
          50    015C 8F   3C    0157   354 10$:     MOVZWL  #SS$_IVLOGTAB,R0        ; set invalid logical name table number
                         04     015C   355          RET                             ;
                                015D   356
                      52     D5 015D   357 20$:     TSTL    R2                      ; was an access mode given?
                      07     12 015F   358          BNEQ    30$                     ; use it if so
                      52     DC 0161   359          MOVPSL  R2                      ; otherwise use mode of caller
       52    52    02    18   EF 0163   360          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R2,R2
                         05     0168   361 30$:     RSB
                                0169   362
                                0169   363
                                0169   364
          50    0114 8F   3C    0169   365 INSARG:  MOVZWL  #SS$_INSFARG,R0         ; Report incorrect arg list count
                         04     016E   366          RET
```

```
                        016F   369          .SBTTL  DELETE Logical Name
                        016F   370 ;+
                        016F   371 ; EXE$DELLOG - DELETE Logical Name
                        016F   372 ;
                        016F   373 ; This service provides the capability to delete a previously created logical
                        016F   374 ; name equivalence from either the process, group, or system logical
                        016F   375 ; name table via the old system service interface.
                        016F   376 ;
                        016F   377 ; INPUTS:
                        016F   378 ;
                        016F   379 ;       TBLFLG (AP)   = logical name table number.
                        016F   380 ;       LOGNAM (AP)   = address of logical name string descriptor. zero
                        016F   381 ;                       implies all.
                        016F   382 ;       DLACMODE (AP) = access mode of logical name to be deleted.
                        016F   383 ;
                        016F   384 ; OUTPUTS:
                        016F   385 ;
                        016F   386 ;       R0 low bit clear indicates failure to delete logical name table entry.
                        016F   387 ;               R0 = exception turned in return status
                        016F   388 ;               R0 = status returned from new sevice
                        016F   389 ;
                        016F   390 ;       R0 low bit set indicates successful completion.
                        016F   391 ;               R0 = status returned from new sevice
                        016F   392 ;
                        016F   393 ;-
                        016F   394
                01FC    016F   395          .ENTRY  EXE$DELLOG,^M<R2,R3,R4,R5,R6,R7,R8>
                        0171   396          ;
                        0171   397          ; Only registers R0-R3 are used, the other registers are saved for
                        0171   398          ; compatability with old versions of the service
                        0171   399          ;
                        0171   400
                        0171   401          ;
                        0171   402          ; Return exception as return status
                        0171   403          ;
                6D  00000000'EF  9E  0171   404          MOVAB   L^EXE$SIGTORET,(FP)
                        0178   405
                        0178   406          ;
                        0178   407          ; Verify that the argument list is accessible and has the minimum
                        0178   408          ; number of arguments for successful execution of the service.
                        0178   409          ;
                        0178   410          ASSUME DELLOG$_NARGS LT 128
                     03  6C  91  0178   411          CMPB    (AP),#DELLOG$_NARGS     ; Minimum number of arguments specified?
                     EC  1F  017B   412          BLSSU   INSARG                  ; If LSSU, no, return error status
                   0C AC  D5  017D   413          TSTL    <DELLOG$_NARGS*4>(AP)   ; Touch last argument, verify accessibility
                        0180   414
                        0180   415          ;
                        0180   416          ; Calculate access mode and check table number
                        0180   417          ;
            52  0C AC  02  00  EF  0180   418          EXTZV   #0,#2,DLACMODE(AP),R2   ; get specified access mode
                     FFBD  30  0186   419          BSBW    CHECKOUT                ; check table number and privileges
                        0189   420
                51  FE72 CF43  D0  0189   421          MOVL    TABLE_NAME_LIST[R3],R1  ; level of indirection
                        018F   422
                     52  DD  018F   423          PUSHL   R2                      ; save access mode for reference
                     52  5E  D0  0191   424          MOVL    SP,R2                   ; save pointer to ACMODE
                        0194   425          ;
```

```
                                0194   426          ; Call System Service
                                0194   427          ;
                                0194   428 ;        $DELLNM_S  TABNAM   = (R1),-
                                0194   429 ;                   LOGNAM   = @LOGNAM(AP),-
                                0194   430 ;                   ACMODE   = (R2)
                          62    DF    0194   431          PUSHAL  (R2)
                                0196   432          ASSUME  DELLNM$_ACMODE EQ DELLNM$_LOGNAM+4
                       08 BC    DF    0196   433          PUSHAL  @LOGNAM(AP)
                                0199   434          ASSUME  DELLNM$_LOGNAM EQ DELLNM$_TABNAM+4
                          61    DF    0199   435          PUSHAL  (R1)
                                019B   436          ASSUME  DELLNM$_NARGS  EQ 3
               80000000'9F      03    FB    019B   437          CALLS   #DELLNM$_NARGS,@#SYS$DELLNM - P1SYSVECTORS + ^X80000000
                                      04    01A2   438          RET
```

```
                     01A3   441          .SBTTL  TRANSLATE Logical Name
                     01A3   442 ;+
                     01A3   443 ; EXE$TRNLOG - TRANSLATE Logical Name
                     01A3   444 ;
                     01A3   445 ; This service provides the capability to translate a logical name string
                     01A3   446 ; to a resultant name string via old system service interface.
                     01A3   447 ;
                     01A3   448 ; INPUTS:
                     01A3   449 ;
                     01A3   450 ;       TRLOGNAM (AP)   address of logical name string descriptor.
                     01A3   451 ;       RSLLEN (AP)    = address to store length of result string.
                     01A3   452 ;       RSLBUF (AP)    = address of resultant string buffer descriptor.
                     01A3   453 ;       TABLE (AP)     = address to store translation table number.
                     01A3   454 ;       TRACMODE (AP)  = address to store assignment access mode.
                     01A3   455 ;       DSBMSK (AP)    = table search disable mask.
                     01A3   456 ;
                     01A3   457 ; OUTPUTS:
                     01A3   458 ;
                     01A3   459 ;       R0 low bit clear indicates failure to translate logical name string.
                     01A3   460 ;               R0 = SS$_RESULTOVF - can not add "_" or ":" to  output buffer,
                     01A3   461 ;                                    the buffer is to small
                     01A3   462 ;               R0 = exception turned in return status
                     01A3   463 ;               R0 = status returned from new sevice
                     01A3   464 ;
                     01A3   465 ;       R0 low bit set indicates successful completion.
                     01A3   466 ;               R0 = status returned from new sevice
                     01A3   467 ;-
                     01A3   468
            01FC     01A3   469          .ENTRY  EXE$TRNLOG,^M<R2,R3,R4,R5,R6,R7,R8>
                     01A5   470
                     01A5   471          ;
                     01A5   472          ; Return exception as return status
                     01A5   473          ;
      6D 00000000'EF  9E     01A5   474          MOVAB   L^EXE$SIGTORET,(FP)
                     01AC   475
                     01AC   476          ;
                     01AC   477          ; Verify that the argument list is accessible and has the minimum
                     01AC   478          ; number of arguments for successful execution of the service.
                     01AC   479          ;
                     01AC   480          ASSUME TRNLOG$_NARGS LT 128
            06   6C   91     01AC   481          CMPB    (AP),#TRNLOG$_NARGS     ; Minimum number of arguments specified?
                 B8   1F     01AF   482          BLSSU   INSARG                  ; If LSSU, no, return error status
          18 AC   D5     01B1   483          TSTL    <TRNLOG$_NARGS*4>(AP)   ; Touch last argument, verify accessibility
                     01B4   484
55  18 AC FFFFFFF8 8F   CB     01B4   485          BICL3   #^C^X07,DSBMSK(AP),R5   ; set table search disable mask
          58   10 AC   D0     01BD   486          MOVL    TABLE(AP),R8            ; pickup address of the table parameter
                     01C1   487
                     01C1   488          ;
                     01C1   489          ; Resultant string checkout
                     01C1   490          ;
          54   0C AC   D0     01C1   491          MOVL    RSLBUF(AP),R4           ; get address of result string buffer
                     01C5   492                                                  ;  descriptor
             53   64   3C     01C5   493          MOVZWL  (R4),R3                 ; get size of result string buffer
          54   04 A4   D0     01C8   494          MOVL    4(R4),R4                ; get address of result string buffer
                     01CC   495
                     01CC   496          ;
                     01CC   497          ; Logical Name Checkout
```

```
                     01CC   498          ;
51   04 AC    D0     01CC   499          MOVL    TRLOGNAM(AP),R1          ; get address of logical name string
     50 61    3C     01D0   500          MOVZWL  (R1),R0                  ; get length of logical name string
51   04 A1    D0     01D3   501          MOVL    4(R1),R1                 ; get address of logical name string
        51    DD     01D7   502          PUSHL   R1                       ; local copy of lognam descriptor
        50    DD     01D9   503          PUSHL   R0                       ;
     52 5E    D0     01DB   504          MOVL    SP,R2                    ; remember the location of descriptor
                     01DE   505
                     01DE   506          ;
                     01DE   507          ; Get local stack storage
                     01DE   508          ;
     5E 10    C2     01DE   509          SUBL2   #LOCAL_AREA,SP
     56 5E    D0     01E1   510          MOVL    SP,R6                    ; get a pointer to our local store
                     01E4   511
                     01E4   512          ;
                     01E4   513          ; Test for no tranlation
                     01E4   514          ;
61   5F 8F    91     01E4   515          CMPB    #^A"_",(R1)              ; logical name start with underscore?
           03 12     01E8   516          BNEQ    5$                       ; continue if no; otherwise,
         0081 31     01EA   517          BRW     NOTRANS                  ; special case code for no translation
                     01ED   518
                     01ED   519          ;
                     01ED   520          ; Build itemlist
                     01ED   521          ;
        7E 7C     01ED   522 5$:       CLRQ    -(SP)                    ; end of list, no return length
     08 A6 DF     01EF   523          PUSHAL  RET_ACMODE(R6)           ; area for returned access mode
00060001 8F DD     01F2   524          PUSHL   #1!<LNM$_ACMODE@16>      ; size of area and item code
        00 DD     01F8   525          PUSHL   #0                       ; no return length
     04 A6 DF     01FA   526          PUSHAL  RET_TRANATTR(R6)         ; area for returned trans attributes
00030004 8F DD     01FD   527          PUSHL   #4!<LNM$_ATTRIBUTES@16>  ; size of area and item code
        66 D4     0203   528          CLRL    RET_TRANSIZE(R6)         ; clear returned longword
        66 DF     0205   529          PUSHAL  RET_TRANSIZE(R6)         ; return length of translation address
        64 DF     0207   530          PUSHAL  (R4)                     ; translation buffer address
        53 DD     0209   531          PUSHL   R3                       ; size of translation buffer
  02 AE 02 B0     020B   532          MOVW    #LNM$_STRING,2(SP)       ; stuff the item list code
        00 DD     020F   533          PUSHL   #0                       ; no return length
        6E DF     0211   534          PUSHAL  (SP)                     ; point to a zeroed longword (0 index)
00010004 8F DD     0213   535          PUSHL   #4!<LNM$_INDEX@16>       ; stuff the item list code
                     0219   536
                     0219   537          ;
                     0219   538          ; If TABLE has not been specified, then use DSBMSK to index into
                     0219   539          ; the list of logical name table names to obtain the address of
                     0219   540          ; the logical name to use as the table name in the single call to
                     0219   541          ; $TRNLNM.
                     0219   542          ;
                     0219   543
07   55  03 00 ED     0219   544          CMPZV   #0,#3,R5,#7              ; are any tables to be searched?
           49 13     021E   545          BEQL    50$                      ; return immediately if no
                     0220   546
   51 FDE7 CF45 D0     0220   547          MOVL    TRNLOG_TABLE[R5],R1      ; assume TABLE has not been specified
           58 D5     0226   548          TSTL    R8                       ; TABLE specified?
           0D 13     0228   549          BEQL    20$                      ; must do table loop if so; otherwise,
                     022A   550                                           ; only one $TRNLNM need be done
                     022A   551
                     022A   552          ;
                     022A   553          ; Loop thru the valid logical name tables.
                     022A   554          ;
```

```
                     022A   555
                     022A   556          MOVZBL  #2,R7
                     022D   557 10$:      BBS     R7,R5,40$
                     0231   558
                     0231   559          MOVL    TABLE_NAME_LIST[R7],R1  ; level of indirection
                     0237   560
                     0237   561          ;
                     0237   562          ; Call system service
                     0237   563          ;
                     0237   564 ;        $TRNLNM_S  TABNAM   = (R1),-
                     0237   565 ;                   LOGNAM   = (R2),-
                     0237   566 ;                   ITMLST   = -<4*13>(R6)    ; 40 is 13 longwords of itemlist
                     0237   567 20$:      PUSHAL  -<4*13>(R6)
                     023A   568          ASSUME  TRNLNM$_ITMLST EQ TRNLNM$_ACMODE+4
                     023A   569          PUSHL   #0
                     023C   570          ASSUME  TRNLNM$_ACMODE EQ TRNLNM$_LOGNAM+4
                     023C   571          PUSHAL  (R2)
                     023E   572          ASSUME  TRNLNM$_LOGNAM EQ TRNLNM$_TABNAM+4
                     023E   573          PUSHAL  (R1)
                     0240   574          ASSUME  TRNLNM$_TABNAM EQ TRNLNM$_ATTR+4
                     0240   575          PUSHL   #0
                     0242   576          ASSUME  TRNLNM$_NARGS  EQ 5
                     0242   577
6D   000002D2'EF     9E   0242   578          MOVAB   LNM$SIGTORET,(FP)       ; special handler
                     0249   579
80000000'9F    05    FB   0249   580          CALLS   #TRNLNM$_NARGS,@#SYS$TRNLNM - P1SYSVECTORS + ^X80000000
                     0250   581
6D   00000000'EF     9E   0250   582          MOVAB   L^EXE$SIGTORET,(FP)
                     0257   583
                     0257   584          ;
                     0257   585          ; Test if it worked ( Should only be SS$_NORMAL )
                     0257   586          ;
          30 50      E8   0257   587          BLBS    R0,CHECK_ATTR
                     025A   588
                     025A   589          ;
                     025A   590          ; If no translation then try next table
                     025A   591          ;                    else return error
                     025A   592          ;
      01BC 8F    50   B1   025A   593          CMPW    R0,#SS$_NOLOGNAM
                01   13   025F   594          BEQL    30$
                04   0261   595          RET
                     0262   596
                     0262   597          ;
                     0262   598          ; try next table if table looping is being done.
                     0262   599          ;
                     0262   600
             58      D5   0262   601 30$:     TSTL    R8                      ; is table looping being done?
                03   13   0264   602          BEQL    50$                     ; done if no table looping
          C4 57      F4   0266   603 40$:     SOBGEQ  R7,10$                  ; otherwise continue with next table
                     0269   604
                     0269   605          ;
                     0269   606          ; Tried all tables with no luck
                     0269   607          ;
          50 62      7D   0269   608 50$:     MOVQ    (R2),R0
                04   11   026C   609          BRB     NOTRANS2                ; special case code for no translation
                     026E   610
                     026E   611          ;
```

```
                                026E   612          ; No translation, stuff the input into the output buffer
                                026E   613          ;
                          50    D7     026E   614 NOTRANS:DECL    R0                          ; get rid of leading "_", decrease size
                          51    D6     0270   615          INCL    R1                          ; bump pointer
                                0272   616 NOTRANS2:
                    53    50    D1     0272   617          CMPL    R0,R3                       ; is the output buffer big enought
                          55    1A     0275   618          BGTRU   BUFOVR                      ; buffer to small, pail out
                    66    50    3C     0277   619          MOVZWL  R0,RET_TRANSIZE(R6)         ; store result string length
                64  61    50    28     027A   620          MOVC3   R0,(R1),(R4)                ; stuff in user buffer
                          57    D4     027E   621          CLRL    R7                          ; zero the table number
                      08  A6    D4     0280   622          CLRL    RET_ACMODE(R6)              ; zero the access mode
                50    0629  8F    3C     0283   623          MOVZWL  #SS$_NOTRAN,R0              ; stuff old error code
                          27    11     0288   624          BRB     RET_VALUES                  ; rejoin for normal exit
                                028A   625
                                028A   626          ;
                                028A   627          ; Check if underscores need to be added
                                028A   628          ;
                                028A   629 CHECK_ATTR:
                                028A   630          ;
                                028A   631          ; If actual size is greater then returned size
                                028A   632          ;
                22 04 A6    09    E1     028A   633          BBC     #LNM$V_TERMINAL,RET_TRANATTR(R6),RET_VALUES
                          51    66    D0     028F   634          MOVL    RET_TRANSIZE(R6),R1         ; get actual size
                                66    D6     0292   635          INCL    RET_TRANSIZE(R6)            ; add "_" to count
                          53    51    D1     0294   636          CMPL    R1,R3                       ; will string now fit?
                                33    1E     0297   637          BGEQU   BUFOVR                      ; nope
                          64    1B    B1     0299   638          CMPW    #ESCAPE,(R4)                ; is this a PPF?
                                06    12     029C   639          BNEQ    60$                         ; nope
                          54    04    C0     029E   640          ADDL2   #4,R4                       ; insert "_" after PPF info
                          51    04    C2     02A1   641          SUBL2   #4,R1
                                11    BB     02A4   642 60$:     PUSHR   #^M<R0,R4>                  ; save status and start addr
             01 A4    64    51    28     02A6   643          MOVC3   R1,(R4),1(R4)               ; shift right 1 character
                                11    BA     02AB   644          POPR    #^M<R0,R4>                  ; restore registers
                   64    5F  8F    90     02AD   645          MOVB    #^A"_",(R4)                 ; stuff underscore
                                      02B1   646
                                      02B1   647          ;
                                      02B1   648          ; Return the translation size, table number and access mode
                                      02B1   649          ;
                                      02B1   650 RET_VALUES:
                          51    08  AC    D0     02B1   651          MOVL    RSLLEN(AP),R1               ; get address to store result length
                                03    13     02B5   652          BEQL    110$                        ; if not specified
                             61   66    B0     02B7   653          MOVW    RET_TRANSIZE(R6),(R1)       ; store result string length
                                58    D5     02BA   654 110$:    TSTL    R8                          ; is table number to be returned?
                                03    13     02BC   655          BEQL    120$                        ; if not specified
                             68   57    90     02BE   656          MOVB    R7,(R8)                     ; store translation table number
                          51    14  AC    D0     02C1   657 120$:    MOVL    TRACMODE(AP),R1             ; get address to store assignment
                                      02C5   658                                                     ;  access mode
                                04    13     02C5   659          BEQL    130$                        ; if not specified
                          61    08  A6    90     02C7   660          MOVB    RET_ACMODE(R6),(R1)         ; store assignment access mode
                                04    02CB   661 130$:    RET                                 ;
                                      02CC   662
                                      02CC   663          ;
                                      02CC   664          ; Buffer is too small
                                      02CC   665          ;
                50    0214  8F    3C     02CC   666 BUFOVR: MOVZWL  #SS$_RESULTOVF,R0           ; buffer specified too small
                                04    02D1   667          RET                                 ;
```

```
                                02D2   669          .SBTTL  LNM$SIGTORET - TURN SS$_NOLOGNAM EXCEPTION INTO RETURN STATUS
                                02D2   670 ;++
                                02D2   671 ; FUNCTIONAL DESCRIPTION:
                                02D2   672 ;
                                02D2   673 ;       THIS IS A CONDITION HANDLER THAT TURNS A SS$_NOLOGNAM
                                02D2   674 ;       EXCEPTION IN THE A LOWER
                                02D2   675 ;       FRAME INTO A RETURN FROM THE LOWER FRAME WITH THE EXCEPTION NAME
                                02D2   676 ;       AS THE STATUS.  ALL OTHERE EXCEPTIONS
                                02D2   677 ;       ARE RESIGNALLED.  UNWINDS ARE IGNORED.
                                02D2   678 ;
                                02D2   679 ; INPUT PARAMETERS:
                                02D2   680 ;       00(AP) = NUMBER OF CONDITION ARGUMENTS.
                                02D2   681 ;       04(AP) = ADDRESS OF SIGNAL ARGUMENT LIST.
                                02D2   682 ;       08(AP) = ADDRESS OF MECHANISM ARGUMENT LIST.
                                02D2   683 ;
                                02D2   684 ; OUTPUT PARAMETERS:
                                02D2   685 ;       R0 - COMPLETION STATUS CODE
                                02D2   686 ;               SS$_RESIGNAL - ALWAYS
                                02D2   687 ;
                                02D2   688
                                02D2   689 LNM$SIGTORET:
                         0000   02D2   690          .WORD   0
                                02D4   691
                                02D4   692          ASSUME  CHF$L_MCHARGLST,EQ,CHF$L_SIGARGLST+4
                                02D4   693
                   50   04 AC   7D   02D4   694          MOVQ    CHF$L_SIGARGLST(AP),R0  ; GET ADDRESS OF SIGNAL ARGUMENT LIST
          04 A0    000001BC 8F   D1   02D8   695          CMPL    #SS$_NOLOGNAM,CHF$L_SIG_NAME(R0) ; MAGIC ERROR?
                             1B   13   02E0   696          BEQL    10$                     ; BRANCH TO EXIT IF NOT
          04 A0    0000045C 8F   D1   02E2   697          CMPL    #SS$_SSFAIL,CHF$L_SIG_NAME(R0) ; OR SSFAIL?
                             22   12   02EA   698          BNEQ    50$                     ; BRANCH TO EXIT IF NOT
          08 A0    000001BC 8F   D1   02EC   699          CMPL    #SS$_NOLOGNAM,CHF$L_SIG_ARG1(R0) ; MAGIC ERROR?
                             18   12   02F4   700          BNEQ    50$                     ; BRANCH TO EXIT IF NOT
                0C A1    08 A0   D0   02F6   701          MOVL    CHF$L_SIG_ARG1(R0),CHF$L_MCH_SAVR0(R1) ; SET RETURN STATUS
                             05   11   02FB   702          BRB     20$
                0C A1    04 A0   D0   02FD   703 10$:     MOVL    CHF$L_SIG_NAME(R0),CHF$L_MCH_SAVR0(R1) ; SET RETURN STATUS
                             7E   D4   0302   704 20$:     CLRL    -(SP)                   ; CLEAR NEW PC ARGUMENT
                          08 A1   9F   0304   705          PUSHAB  CHF$L_MCH_DEPTH(R1)     ; NUMBER OF FRAMES TO UNWIND
             00000000'GF      02   FB   0307   706          CALLS   #2,G^SYS$UNWIND         ; UNWIND TO ESTABLISHER
                50    0918 8F   3C   030E   707 50$:     MOVZWL  #SS$_RESIGNAL,R0        ; RETURN RESIGNAL STATUS
                                04   0313   708          RET                             ;
                                     0314   709
                                     0314   710          .END
```

```
$$ARGS                  =  00000005
$$T1                    =  00000018
BUFOVR                     000002CC  R      02
CHECKOUT                   00000146  R      02
CHECK_ATTR                 0000028A  R      02
CHF$L_MCHARGLST         =  00000008
CHF$L_MCH_DEPTH         =  00000008
CHF$L_MCH_SAVRO         =  0000000C
CHF$L_SIGARGLST         =  00000004
CHF$L_SIG_ARG1          =  00000008
CHF$L_SIG_NAME          =  00000004
CRACMODE                =  00000010
CRELNM$_ACMODE          =  00000010
CRELNM$_ATTR            =  00000004
CRELNM$_ITMLST          =  00000014
CRELNM$_LOGNAM          =  0000000C
CRELNM$_NARGS           =  00000005
CRELNM$_TABNAM          =  00000008
CRELOG$_ACMODE          =  00000010
CRELOG$_EQLNAM          =  0000000C
CRELOG$_LOGNAM          =  00000008
CRELOG$_NARGS           =  00000004
CRELOG$_TBLFLG          =  00000004
DELLNM$_ACMODE          =  0000000C
DELLNM$_LOGNAM          =  00000008
DELLNM$_NARGS           =  000000C3
DELLNM$_TABNAM          =  00000004
DELLOG$_ACMODE          =  0000000C
DELLOG$_LOGNAM          =  00000008
DELLOG$_NARGS           =  00000003
DELLOG$_TBLFLG          =  00000004
DLACMODE                =  0000000C
DSBMSK                  =  00000018
EQLNAM                  =  0000000C
ESCAPE                  =  0000001B
EXE$CRELOG                 000000DF  RG     02
EXE$DELLOG                 0000016F  RG     02
EXE$SIGTORET               ********    X    02
EXE$TRNLOG                 000001A3  RG     02
GROUP_TABLE                0000003A  R      02
GS_TABLE                   0000005E  R      02
INSARG                     00000169  R      02
IN_ACMODE               =  0000000C
LNM$M_CRELOG            =  00000004
LNM$M_TERMINAL          =  00000200
LNM$SIGTORET               000002D2  R      02
LNM$V_TERMINAL          =  00000009
LNM$_ACMODE             =  00000006
LNM$_ATTRIBUTES         =  00000003
LNM$_INDEX              =  00000001
LNM$_STRING             =  00000002
LNM_TERM                   000000DB  R      02
LOCAL_AREA              =  00000010
LOG$C_PROCESS           =  00000002
LOGNAM                  =  00000008
NOTRANS                    0000026E  R      02
NOTRANS2                   00000272  R      02
P1SYSVECTORS               ********    X    02
PGS_TABLE                  000000B5  R      02
PG_TABLE                   0000007A  R      02
PROCESS_TABLE              0000004B  R      02
PSL$S_CURMOD            =  00000002
PSL$V_CURMOD            =  00000018
PS_TABLE                   00000097  R      02
RET_ACMODE              =  00000008
RET_TRANATTR            =  00000004
RET_TRANSIZE            =  00000000
RET_VALUES                 000002B1  R      02
RSLBUF                  =  0000000C
RSLLEN                  =  00000008
SS$_INSFARG             =  00000114
SS$_IVLOGTAB            =  0000015C
SS$_NOLOGNAM            =  000001BC
SS$_NOTRAN              =  00000629
SS$_RESIGNAL            =  00000918
SS$_RESULTOVF           =  00000214
SS$_SSFAIL              =  0000045C
SYS$CRELNM                 ********    X    02
SYS$DELLNM                 ********    X    02
SYS$TRNLNM                 ********    X    02
SYS$UNWIND                 ********    X    02
SYSTEM_TABLE               00000028  R      02
TABLE                   =  00000010
TABLE_MODE                 000000D9  R      02
TABLE_NAME_LIST            00000000  R      02
TBLFLG                  =  00000004
TRACMODE                =  00000014
TRLOGNAM                =  00000004
TRNLNM$_ACMODE          =  00000010
TRNLNM$_ATTR            =  00000004
TRNLNM$_ITMLST          =  00000014
TRNLNM$_LOGNAM          =  0000000C
TRNLNM$_NARGS           =  00000005
TRNLNM$_TABNAM          =  00000008
TRNLOG$_ACMODE          =  00000014
TRNLOG$_DSBMSK          =  00000018
TRNLOG$_LOGNAM          =  00000004
TRNLOG$_NARGS           =  00000006
TRNLOG$_RSLBUF          =  0000000C
TRNLOG$_RSLLEN          =  00000008
TRNLOG$_TABLE           =  00000010
TRNLOG_TABLE               0000000C  R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
YF$SYSLOGNAM	00000314 (788.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	36	00:00:00.09	00:00:00.26
Command processing	129	00:00:00.56	00:00:01.33
Pass 1	260	00:00:06.78	00:00:15.05
Symbol table sort	0	00:00:00.88	00:00:01.61
Pass 2	135	00:00:01.94	00:00:04.19
Symbol table output	13	00:00:00.09	00:00:00.11
Psect synopsis output	2	00:00:00.02	00:00:00.05
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	577	00:00:10.37	00:00:22.60

The working set limit was 1500 pages.
39221 bytes (77 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 588 non-local and 18 local symbols.
710 source lines were read in Pass 1, producing 22 object records in Pass 2.
20 pages of virtual memory were used to define 19 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	1
_255DUA28:[SYSLIB]STARLET.MLB;2	15
TOTALS (all libraries)	16

623 GETS were required to define 16 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:SYSLOGNAM/OBJ=OBJ$:SYSLOGNAM MSRC$:SYSLOGNAM/UPDATE=(ENH$:SYSLOGNAM)+EXECML$/LIB
```

0386 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SYSPARAM
LIS

SYSLOGNAM
LIS

SYSMTACC
LIS

SYSIMGSTA
LIS

SYSLNM
LIS

SYSLOAVEC
LIS

SYSLKWSET
LIS

SYSMAILBX
LIS