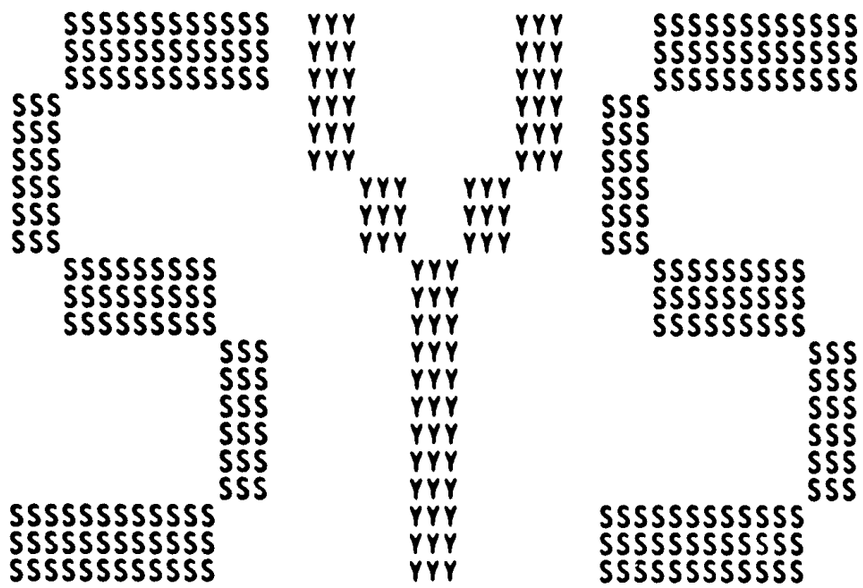

FILEID**SYSRTSLST

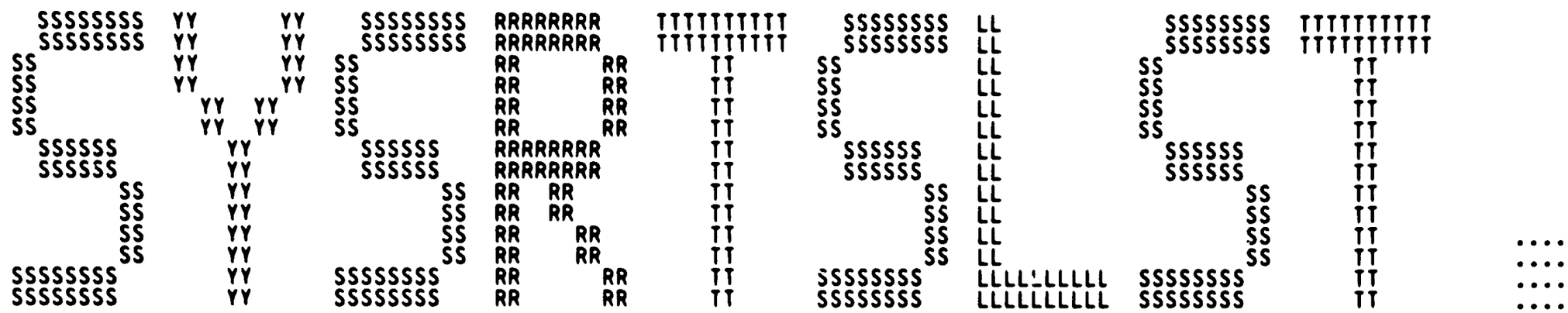

Table of contents

```
0000     1          .TITLE  SYSRTSLST - Rights List Manipulation Services
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
0000    29  ;++
0000    30  ; FACILITY:  VAX/VMS Executive
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;
0000    34  ;       This module contains services to manipulate the system
0000    35  ;       and process rights lists.
0000    36  ;
0000    37  ; ENVIRONMENT:
0000    38  ;
0000    39  ;       VAX/VMS exec, process context
0000    40  ;
0000    41  ;--
0000    42  ;
0000    43  ; AUTHOR:  Andrew C. Goldstein, CREATION DATE:  11-Mar-1983  11:39
0000    44  ;
0000    45  ; MODIFIED BY:
0000    46  ;
0000    47  ;       V03-001 ACG0335         Andrew C. Goldstein,    10-May-1983  19:33
0000    48  ;               Add support for extended rights list
0000    49  ;
0000    50  ;**
0000    51
0000    52  ;
0000    53  ; Define needed system structures
0000    54  ;
0000    55          $ARBDEF                         ; access rights block
0000    56          $DYNDEF                         ; dynamic structure types
0000    57          $PCBDEF                         ; process control block
```

```
                     0000     58          $PRVDEF                         ; privilege bits
                     0000     59          $SSDEF                          ; system status codes
                     0000     60
                     0000     61 ;
                     0000     62 ; Argument list offsets
                     0000     63 ;
00000004             0000     64          PIDADR  = 4                     ; address of PID
00000008             0000     65          PRCNAM  = 8                     ; address of process name desc
0000000C             0000     66          ID      = 12                    ; address of identifier and attrib
00000010             0000     67          NAME    = 16                    ; address of identifier name desc
00000010             0000     68          MODE    = 16                    ; grant/revoke mode for kernel routine
00000014             0000     69          PRVATR  = 20                    ; address for previous attributes
```

```
                        71          .SBTTL  EXE$GRANTID - Grant Identifier to Process
                        72
                        73 ;++
                        74 ;
                        75 ;       EXE$GRANTID - grant identifier to process
                        76 ;       EXE$REVOKID - revoke identifier from process
                        77 ;
                        78 ; FUNCTIONAL DESCRIPTION:
                        79 ;
                        80 ;       This routine grants or revokes the specified identifier
                        81 ;       to or from the specified process. Process name and right
                        82 ;       name are translated to PID and identifier as necessary.
                        83 ;       If a PID of -1 is given, the system rights list is
                        84 ;       operated on.
                        85 ;
                        86 ; CALLING SEQUENCE:
                        87 ;       EXE$GRANTID (PIDADR, PRCNAM, ID, NAME, PRVATR)
                        88 ;       EXE$REVOKID (PIDADR, PRCNAM, ID, NAME, PRVATR)
                        89 ;
                        90 ; INPUT PARAMETERS:
                        91 ;       PIDADR: address of PID of process
                        92 ;       PRCNAM: address of descriptor of process name
                        93 ;       ID:     address of identifier to grant
                        94 ;       NAME:   address of descriptor of identifier name
                        95 ;
                        96 ; IMPLICIT INPUTS:
                        97 ;       NONE
                        98 ;
                        99 ; OUTPUT PARAMETERS:
                       100 ;       PIDADR: address to store resulting PID
                       101 ;       IDADDR: address to store resulting identifier
                       102 ;       PRVATR: previous attributes of superseded or revoked identifier
                       103 ;
                       104 ; IMPLICIT OUTPUTS:
                       105 ;       NONE
                       106 ;
                       107 ; ROUTINE VALUE:
                       108 ;       SS$_WASCLR: success; identifier not previously in list
                       109 ;       SS$_WASSET: success; identifier was previously in list
                       110 ;       SS$_ACCVIO: some argument is unreadable or unwritable
                       111 ;       SS$_NOPRIV: caller lacks privileges over target process, or
                       112 ;                   lacks CMKRNL privilege
                       113 ;       SS$_NOSUCHID: identifier name is not a valid name
                       114 ;       SS$_RIGHTSFULL: the specified rights list is full
                       115 ;
                       116 ; SIDE EFFECTS:
                       117 ;       Identifier entered in or removed from specified rights list
                       118 ;
                       119 ;--
                       120
                  00000000 121          .PSECT  YEXEPAGED
                       122          .ENABLE LSB
                       123
                  000C 0000 124          .ENTRY  EXE$GRANTID,^M<R2,R3>
53 01 D0          0002 125          MOVL    #1,R3                   ; set grant mode
   04 11          0005 126          BRB     10$
                  0007 127
```

```
                     000C  0007   128          .ENTRY  EXE$REVOKID,^M<R2,R3>
                       53   D4   0009   129          CLRL    R3                              ; set revoke mode
                            000B   130
                 6D  0000'CF   9E   000B   131 10$:     MOVAB   W^EXE$SIGTORET,(FP)     ; set local condition handler
                   52    0C AC   D0   0010   132          MOVL    ID(AP),R2               ; get pointer to identifier
                         05   12   0014   133          BNEQ    20$                     ; branch if ID specified
                         7E   7C   0016   134          CLRQ    -(SP)                   ; allocate ID buffer on stack
                       52   5E   D0   0018   135          MOVL    SP,R2                   ; and set pointer
                            001B   136
                         62   D5   001B   137 20$:     TSTL    (R2)                    ; see if a binary ID is supplied
                         1C   12   001D   138          BNEQ    30$                     ; if so, skip conversion
                 50  0114 8F   3C   001F   139          MOVZWL  #SS$_INSFARG,R0
                       10 AC   D5   0024   140          TSTL    NAME(AP)                ; make sure a name is supplied
                         2D   13   0027   141          BEQL    40$
                            0029   142          $ASCTOID_S      ID=(R2),-               ; translate name into identifier
                            0029   143                          ATTRIB=4(R2),-
                            0029   144                          NAME=@NAME(AP)
                       1B 50   E9   0038   145          BLBC    R0,40$                  ; branch if failed
                            003B   146
                            003B   147                                                  ; call kernel mode routine with
                       14 AC   DD   003B   148 30$:     PUSHL   PRVATR(AP)              ; previous attributes
                    7E   52   7D   003E   149          MOVQ    R2,-(SP)                ; identifier and mode
                    7E   04 AC   7D   0041   150          MOVQ    PIDADR(AP),-(SP)        ; PIDADR & PRCNAM
                         05   DD   0045   151          PUSHL   #5                      ; argument count
                         5E   DD   0047   152          PUSHL   SP                      ; arg list address
                 0000002C'EF   9F   0049   153          PUSHAB  GRANT_REVOKE            ; routine address
            00000000'EF   08   FB   004F   154          CALLS   #8,SYS$CMKRNL
                         04   0056   155 40$:     RET
                            0057   156
                            0057   157          .DISABLE LSB
```

```
                     0057   159          .SBTTL  GRANT_REVOKE - Kernel Mode Rights List Handling
                     0057   160
                     0057   161 ;++
                     0057   162 ;
                     0057   163 ;       GRANT_REVOKE - kernel mode rights list handling
                     0057   164 ;
                     0057   165 ; FUNCTIONAL DESCRIPTION:
                     0057   166 ;
                     0057   167 ;       This routine does the kernel mode processing to grant or
                     0057   168 ;       revoke an identifier. It locates the specified process
                     0057   169 ;       and searches and modifies the rights list.
                     0057   170 ;
                     0057   171 ; CALLING SEQUENCE:
                     0057   172 ;       GRANT_REVOKE (PIDADR, PRCNAM, ID, MODE, PRVATR)
                     0057   173 ;
                     0057   174 ; INPUT PARAMETERS:
                     0057   175 ;       PIDADR: address of PID of process
                     0057   176 ;       PRCNAM: address of descriptor of process name
                     0057   177 ;       ID:     address of identifier to grant
                     0057   178 ;       MODE:   0 to revoke identifier, 1 to grant
                     0057   179 ;
                     0057   180 ; IMPLICIT INPUTS:
                     0057   181 ;       NONE
                     0057   182 ;
                     0057   183 ; OUTPUT PARAMETERS:
                     0057   184 ;       PIDADR: address to store resulting PID
                     0057   185 ;       PRVATR: previous attributes of superseded identifier
                     0057   186 ;
                     0057   187 ; IMPLICIT OUTPUTS:
                     0057   188 ;       NONE
                     0057   189 ;
                     0057   190 ; SIDE EFFECTS:
                     0057   191 ;       Identifier entered in specified rights list
                     0057   192 ;
                     0057   193 ;--
                     0057   194
                00000000    195          .PSECT  AEXENONPAGED
                     0000   196          .ENABLE LSB
                     0000   197 ;
                     0000   198 ; To here if EXE$NAMPID returns with an error. Check for the special
                     0000   199 ; case of a PID of -1, indicating that the system rights list is to
                     0000   200 ; be operated on.
                     0000   201 ;
000008E8 8F    50    D1   0000   202 10$:     CMPL    R0,#SS$_NONEXPR         ; check for non-existent process
               1E    12   0007   203          BNEQ    20$                     ; exit if anything else
      51   04 AC    D0   0009   204          MOVL    PIDADR(AP),R1           ; get address of PID
               18    13   000D   205          BEQL    20$                     ; branch if none specified
                     000F   206          IFNORD  #4,(R1),30$             ; ACCVIO if PID not readable
           51  61    D2   0015   207          MCOML   (R1),R1                 ; check for -1
               0D    12   0018   208          BNEQ    20$                     ; branch if not
               57    D6   001A   209          INCL    R7                      ; point to 2nd rights vector entry
    50   2814 8F    3C   001C   210          MOVZWL  #SS$_NOSYSNAM,R0        ; assume no privilege
                     0021   211          IFPRIV  SYSNAM,50$,R6           ; exit if SYSNAM privilege lacking
                04   0027   212 20$:     RET
                     0028   213
           50  0C    D0   0028   214 30$:     MOVL    #SS$_ACCVIO,R0          ; return access violation
                04   002B   215          RET
```

```
                                002C   216 ;
                                002C   217 ; Main subroutine entry point.
                                002C   218 ;
                                002C   219 GRANT_REVOKE:
                          03FC  002C   220         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9>
                       57   7C  002E   221         CLRQ    R7                      ; init rights vector index and free pointer
                    56 54   D0  0030   222         MOVL    R4,R6                   ; save PCB addr in R6
                    FFCA'   30  0033   223         BSBW    W^EXE$NAMPID            ; translate process name to PID
                    C7 50   E9  0036   224         BLBC    R0,10$                  ; branch on failure
                            0039   225
                    56 54   D0  0039   226 40$:     MOVL    R4,R6                   ; save PCB address
                54  00A4 C647   D0  003C   227 50$:     MOVL    PCB$Q_PRIV+ARB$L_RIGHTSLIST(R6)[R7],R4 ; get rights list descriptor
                       3E   13  0042   228         BEQL    100$                    ; branch if none present
             53 84     FD 8F   78  0044   229         ASHL    #-3,(R4)+,R3            ; get rights list length
                    54   64   D0  0049   230         MOVL    (R4),R4                 ; and rights list address
                            004C   231
                    51   0C AC   D0  004C   232 60$:     MOVL    ID(AP),R1               ; get address of identifier
                            0050   233         IFNORD  #8,(R1),30$             ; check readability
                    51   61   7D  0056   234         MOVQ    (R1),R1                 ; get identifier and attributes
                    55   14 AC   D0  0059   235         MOVL    PRVATR(AP),R5           ; get pointer to prev. atr. longword
                         20   13  005D   236         BEQL    90$                     ; branch if none
                            005F   237         IFNOWRT #4,(R5),30$             ; check writability
                         65   D4  0065   238         CLRL    (R5)                    ; initialize to zero
                         16   11  0067   239         BRB     90$                     ; dive into loop
                            0069   240 ;
                            0069   241 ; To here when an empty entry is encountered in a list
                            0069   242 ;
                         58   D5  0069   243 70$:     TSTL    R8                      ; check if we already have one
                         15   12  006B   244         BNEQ    100$                    ; branch if so
                    58   54   D0  006D   245         MOVL    R4,R8                   ; otherwise save the pointer
                         10   11  0070   246         BRB     100$                    ; chain to next list if any
                            0072   247 ;
                            0072   248 ; Search the rights list for the desired identifier
                            0072   249 ;
                 50   64   D0  0072   250 80$:     MOVL    (R4),R0                 ; get next identifier from rights list
                      F2   13  0075   251         BEQL    70$                     ; if zero, end of list
                 51   50   D1  0077   252         CMPL    R0,R1                   ; see if matches desired ID
                      1E   13  007A   253         BEQL    140$                    ; if yes, exit loop
                 54   08   C0  007C   254         ADDL    #8,R4                   ; next list entry
                   F0 53   F4  007F   255 90$:     SOBGEQ  R3,80$                  ; loop throught rights list
                            0082   256 ;
                            0082   257 ; Identifier not found in this list.
                            0082   258 ;
                         57   D5  0082   259 100$:    TSTL    R7                      ; check which list in use
                         05   12  0084   260         BNEQ    110$                    ; branch if not first
                 57      02   C0  0086   261         ADDL    #2,R7                   ; point to extended rights list
                         B1   11  0089   262         BRB     50$                     ; and search it
                            008B   263
              07 10 AC   E9  008B   264 110$:    BLBC    MODE(AP),120$           ; branch if attempted revoke
                         58   D5  008F   265         TSTL    R8                      ; see if empty entry found
                         2B   13  0091   266         BEQL    180$                    ; branch if not
                 68      51   7D  0093   267         MOVQ    R1,(R8)                 ; store identifier in list
                 50      01   D0  0096   268 120$:    MOVL    #SS$_WASCLR,R0          ; if revoke - benign success
                              04  0099   269 130$:    RET
                            009A   270 ;
                            009A   271 ; Specified identifier found in rights list
                            009A   272 ;
```

```
                          55    D5   009A    273 140$:   TSTL    R5                          ; see if prev attributes to be returned
                          04    13   009C    274         BEQL    150$                        ; branch if not
                    65    04    A4   D0   009E    275         MOVL    4(R4),(R5)                  ; store previous attribites
                      05  10    AC   E9   00A2    276 150$:   BLBC    MODE(AP),160$               ; branch to do revoke
                        64      51   7D   00A6    277         MOVQ    R1,(R4)                     ; store identifier in rights list
                                0F   11   00A9    278         BRB     170$
                                     00AB    279
                    53    53    03   78   00AB    280 160$:   ASHL    #3,R3,R3                    ; compute remaining list size
                    50    53    08   C1   00AF    281         ADDL3   #8,R3,R0                    ; compute size plus one entry
64    50    00      08    A4    53   2C   00B3    282         MOVC5   R3,8(R4),#0,R0,(R4)         ; collapse out found list entry
                          50    09   D0   00BA    283 170$:   MOVL    #SS$_WASSET,R0              ; set return status
                                04   00BD    284         RET
                                     00BE    285 ;
                                     00BE    286 ; No empty entries available - extend the rights list
                                     00BE    287 ;
                          51    7C   00BE    288 180$:   CLRQ    R1                          ; assume no block present
              59    00A4    C647   D0   00C0    289         MOVL    PCB$Q_PRIV+ARB$L_RIGHTSLIST(R6)[R7],R9 ; point to rights list again
                          03    13   00C6    290         BEQL    190$                        ; branch if none exists
                    51    69    7D   00C8    291         MOVQ    (R9),R1                     ; get current block size and address
                    53    51    7D   00CB    292 190$:   MOVQ    R1,R3                       ; save size and addr for later
            51    00000050    8F   C0   00CE    293         ADDL    #ARB$S_LOCALRIGHTS+16,R1 ; increase size and add overhead
                        FF28'   30   00D5    294         BSBW    EXE$ALONONPAGED             ; and allocate a new one
                      BE    50   E9   00D8    295         BLBC    R0,130$                     ; branch on failure
                    55    52    D0   00DB    296         MOVL    R2,R5                       ; save block address
              82    51    10    C3   00DE    297         SUBL3   #16,R1,(R2)+                ; set up actual list length
                82    08    A2   9E   00E2    298         MOVAB   8(R2),(R2)+                 ; and descriptor pointer
                    82    51    B0   00E6    299         MOVW    R1,(R2)+                    ; block length
              82    0042    8F   B0   00E9    300         MOVW    #DYN$C_RIGHTSLIST,(R2)+     ; and block type
                    7E    54    7D   00EE    301         MOVQ    R4,-(SP)                    ; save R4 & R5
  62  65  00      64    53    2C   00F1    302         MOVC5   R3,(R4),#0,(R5),(R2)        ; copy the contents and zero rest
                    54    8E    7D   00F7    303         MOVQ    (SP)+,R4                    ; restore regs
                      11  57    E9   00FA    304         BLBC    R7,200$                     ; branch if extended process list
        00000000'EF       65    7D   00FD    305         MOVQ    (R5),EXE$GQ_RIGHTSLIST      ; and store in system descriptor
                    50    54    D0   0104    306         MOVL    R4,R0                       ; get pointer to old block
                          13    13   0107    307         BEQL    220$                        ; branch if none
                    50    0C    C2   0109    308         SUBL    #12,R0                      ; point to start of block
                          0B    11   010C    309         BRB     210$
                                     010E    310
          00A4  C647        55    D0   010E    311 200$:   MOVL    R5,PCB$Q_PRIV+ARB$L_RIGHTSLIST(R6)[R7] ; set up new pointer
                    50    59    D0   0114    312         MOVL    R9,R0
                          03    13   0117    313         BEQL    220$                          branch if no old block
                        FEE4'   30   0119    314 210$:   BSBW    EXE$DEANONPAGED             ; deallocate the old list
                        FF1D    31   011C    315 220$:   BRW     50$                         ; locate free entry and try again
                                     011F    316
                                     011F    317         .DISABLE LSB
                                     011F    318
                                     011F    319
                                     011F    320
                                     011F    321         .END
```

```
ARB$L_RIGHTSLIST                = 00000020
ARB$S_LOCALRIGHTS               = 00000040
DYN$C_RIGHTSLIST                = 00000042
EXE$ALONONPAGED                   ********    X   03
EXE$DEANONPAGED                   ********    X   03
EXE$GQ_RIGHTSLIST                 ********    X   03
EXE$GRANTID                       00000000 RG     02
EXE$NAMPID                        ********    X   03
EXE$REVOKID                       00000007 RG     02
EXE$SIGTORET                      ********    X   02
GRANT_REVOKE                      0000002C R      03
ID                              = 0000000C
MODE                            = 00000010
NAME                            = 00000010
PCB$Q_PRIV                      = 00000084
PIDADR                          = 00000004
PRCNAM                          = 00000008
PRV$V_SYSNAM                    = 00000002
PRVATR                          = 00000014
SS$_ACCVIO                      = 0000000C
SS$_INSFARG                     = 00000114
SS$_NONEXPR                     = 000008E8
SS$_NOSYSNAM                    = 00002814
SS$_WASCLR                      = 00000001
SS$_WASSET                      = 00000009
SYS$ASCTOID                       ********   GX   02
SYS$CMKRNL                        ********    X   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
YEXEPAGED	00000057 (87.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
AEXENONPAGED	0000011F (287.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.07	00:00:00.41
Command processing	129	00:00:00.61	00:00:02.00
Pass 1	267	00:00:07.35	00:00:16.83
Symbol table sort	0	00:00:01.18	00:00:02.48
Pass 2	71	00:00:01.47	00:00:02.56
Symbol table output	5	00:00:00.05	00:00:00.25
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	511	00:00:10.76	00:00:24.56

The working set limit was 1350 pages.
42274 bytes (83 pages) of virtual memory were used to buffer the intermediate code.

There were 50 pages of symbol table space allocated to hold 793 non-local and 26 local symbols.
321 source lines were read in Pass 1, producing 21 object records in Pass 2.
17 pages of virtual memory were used to define 16 macros.

+----------------------------+
! Macro library statistics !
+----------------------------+

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	6
_255DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	13

888 GETS were required to define 13 macros.

There were no errors, warnings or information messages.

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

0387 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SYSPURGWS
LIS

SYSPUTMSG
LIS

SYSPCNTRL
LIS

SYSQIOFDT
LIS

SYSQIOREQ
LIS

SYSRUNDWN
LIS

SYSRDBRES
LIS

SYSRTSLST
LIS