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

FILEID**ISSH

```
IIIIII      SSSSSSSS    SSSSSSSS  HH      HH
IIIIII      SSSSSSSS    SSSSSSSS  HH      HH
  II      SS          SS          HH      HH
  II      SS          SS          HH      HH
  II      SS          SS          HH      HH
  II      SS          SS          HH      HH
  II        SSSSSS      SSSSSS    HHHHHHHHHH
  II        SSSSSS      SSSSSS    HHHHHHHHHH
  II              SS          SS  HH      HH
  II              SS          SS  HH      HH
  II              SS          SS  HH      HH      ....
  II              SS          SS  HH      HH      ....
IIIIII    SSSSSSSS    SSSSSSSS    HH      HH      ....
IIIIII    SSSSSSSS    SSSSSSSS    HH      HH      ....
```

```
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
```

```
0000     1          .TITLE  ISSH -- Intercept System Service Handler
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 System Service Monitor
0000    31 ;
0000    32 ; Abstract:
0000    33 ;
0000    34 ;       Intercept system service handler.  This code intercepts the system
0000    35 ;       service, makes the call to user supplied routine under the right
0000    36 ;       privilege first, after returning from user supplied routine,
0000    37 ;       continues system service.
0000    38 ;       Note: this code is copied into P0 space and executed there.
0000    39 ;       If we encounter SYS$RUNDWN, copy the original system vector back,
0000    40 ;       and let it go, this is done by re-entering back to the entry
0000    41 ;       point of DBGSSISHR.EXE.
0000    42 ;
0000    43 ; Environment:  VAX/VMS
0000    44 ;
0000    45 ;--
0000    46 ;
0000    47 ; Author: D.W. Thiel,    Creation Date: 30-Dec-1981
0000    48 ;
0000    49 ; Modified By:
0000    50 ;
0000    51 ;         P. Sager,      20-Sep-1983
0000    52 ;
0000    53 ;**
0000    54 ;
0000    55
0000    56
0000    57          .SBTTL  DECLARATIONS
```

```
                        0000     58
                        0000     59          .ENABLE SUPPRESSION
                        0000     60
                        0000     61
                        0000     62 ;
                        0000     63 ; External routines
                        0000     64 ;
                        0000     65
                        0000     66 ;
                        0000     67 ; INCLUDE FILES:
                        0000     68 ;
                        0000     69
                        0000     70 ;
                        0000     71 ; MACROS:
                        0000     72 ;
                        0000     73
                        0000     74 ;
                        0000     75 ; EQUATED SYMBOLS:
                        0000     76 ;
                        0000     77
                        0000     78          $IPLDEF         ; define IPL symbols
                        0000     79          $PSLDEF         ; define PSL layout symbols
                        0000     80          $SGNDEF         ; define system generation parameters
                        0000     81
                        0000     82 ;
                        0000     83 ; OWN STORAGE: Data area
                        0000     84 ;
                        0000     85
                    00000000     86          .PSECT  $ISSH_CODE$     RD, NOWRT, NOEXE, PIC, CON, GBL, PAGE
                        0000     87
                        0000     88 ISSH_VEC_BASE::
                    00000A00    0000     89 TV_CPY: .BLKB   SGN$C_SYSVECPGS*512     ; copy of transfer vector
                        0A00     90
                        0A00     91 ISSH_DATA_BEG::
                    00000C00    0A00     92 DATA:   .BLKB   512                     ; local/global data area
                        0C00     93 ISSH_DATA_END::
                        0C00     94
                        0C00     95          .ALIGN  PAGE
                        0C00     96
                        0C00     97                                                 ; We have 64 longwords available
                        0C00     98                                                 ;   without making changes to SSI.B32
                        0C00     99 ;****************************************************************************
                        0C00    100 ; This area has potential problems to run over allocated spaces.  Not likely to
                        0C00    101 ; happen. (now the checks are placed in for overflow condition).
                        0C00    102 ;
                    00000BBC    0C00    103 ISSH_STACK==ISSH_DATA_END - 68          ; Local stack (keep track of the
                        0C00    104                                                 ;   current active routine)
                        0C00    105                                                 ; Never intercept again, if system
                        0C00    106                                                 ;   service is originated from the same
                        0C00    107                                                 ;   level)
                    00000BC0    0C00    108 ISSH_STKPTR==ISSH_DATA_END - 64         ; Local stack pointer
                        0C00    109                                                 ; 5 long each for the next two entries,
                        0C00    110                                                 ;   (one byte each per count):
                    00000BD4    0C00    111 ISSH_BIT_15_CNT_4==ISSH_DATA_END - 44   ; Bit 15 in PSW count in prio 4.
                    00000BE8    0C00    112 ISSH_BIT_15_CNT_3==ISSH_DATA_END - 24   ; Bit 15 in PSW count in prio 3.
                        0C00    113                                                 ; It has the following structure:
                        0C00    114                                                 ;    1st byte: total system service
```

```
                     0C00   115                                                ;       count.
                     0C00   116                                                ;     2nd byte, 3rd byte, ...:
                     0C00   117                                                ;       number of priorities it went
                     0C00   118                                                ;       through for each service.
                     0C00   119 ; These two data structures are used by Debug, and Super Debug to
                     0C00   120 ; catch the RET from system service.
                     0C00   121 ; Some services are nested, for example, SYS$PUTMSG calls SYS$GETMSG etc.
                     0C00   122 ; So the 1st byte is used to indicate we are still in SYS$PUTMSG when
                     0C00   123 ; SYS$GETMSG is in.  When this case happens, debug simply lets SYS$GETMSG
                     0C00   124 ; go by.  When SYS$PUTMSG happens, depending on where it is called from
                     0C00   125 ; (User? DBG? or SDBG?), higher priority one always see the lower one, and
                     0C00   126 ; the same level never sees the same level, so we need to know, how many
                     0C00   127 ; levels see SYS$PUTMSG, this info. is recorded in 2nd byte, SYS$GETMSG
                     0C00   128 ; is recorded in 3rd byte, etc.  This really is used by DEBUGGER as a
                     0C00   129 ; means to communicate, in fact, in DEBUGGER, we are really paying attention
                     0C00   130 ; to SYS$PUTMSG only.
                     0C00   131 ;*****************************************************************************
00000BEC             0C00   132 ISSH_BIT_15_CNT_12==ISSH_DATA_END - 20  ; This one has the same flavor as
                     0C00   133                                         ;  the above, except is not used,
                     0C00   134                                         ;  it's only purpose is to fit in
                     0C00   135                                         ;  the code path.
00000BF0             0C00   136 ISSH_CTRL_INDEX==ISSH_DATA_END - 16     ; Total entries been made for prio.
00000BF4             0C00   137 ISSH_CTRL_PRIO_INDEX==ISSH_DATA_END - 12; Entries been made in each prio.
00000BF8             0C00   138 ISSH_RUNNING_FLAG==ISSH_DATA_END - 8    ; Flag to indicate if this system
                     0C00   139                                         ;   service itself is running, if
                     0C00   140                                         ;   so, don't intercept any system
                     0C00   141                                         ;   service called from this routine
00000BFC             0C00   142 ISSH_PRIO_MASK==ISSH_DATA_END - 4       ; Routine Enable/Disable flags for
                     0C00   143                                         ;   all priorities
00000A00             0C00   144 ISSH_USER_ADDR==ISSH_DATA_BEG           ; User routine name Table, the name is
                     0C00   145                                         ;   filled in by the mode and
                     0C00   146                                         ;   priority, declared by the user,
                     0C00   147                                         ;   and is called by the mode
                     0C00   148                                         ;   in current PSL and mask.
                     0C00   149                                         ;
00000B00             0C00   150 ISSH_USER_DATA_BEG==ISSH_DATA_BEG+256   ; Ending of the data area (note:
                     0C00   151                                         ;   data area starting at bigger address)
```

```
                     0C00   153          .SBTTL  INTERCEPT_SS_HANDLER -- System Service Call Intercept Handler
                     0C00   154
                     0C00   155 ;++
                     0C00   156 ;
                     0C00   157 ; FUNCTIONAL DESCRIPTION:
                     0C00   158 ;
                     0C00   159 ;  Receives control whenever a system service is intercepted.
                     0C00   160 ;
                     0C00   161 ; CALLING SEQUENCE:
                     0C00   162 ;
                     0C00   163 ;       JSB with stack frame already established
                     0C00   164 ;
                     0C00   165 ; INPUT PARAMETERS:
                     0C00   166 ;
                     0C00   167 ;       (SP) :  contains the address of the system service entry mask plus 6
                     0C00   168 ;
                     0C00   169 ; IMPLICIT INPUTS:
                     0C00   170 ;
                     0C00   171 ;       none
                     0C00   172 ;
                     0C00   173 ; OUTPUT PARAMETERS:
                     0C00   174 ;
                     0C00   175 ;       none
                     0C00   176 ;
                     0C00   177 ; IMPLICIT OUTPUTS:
                     0C00   178 ;
                     0C00   179 ;       none
                     0C00   180 ;
                     0C00   181 ; COMPLETION CODES:
                     0C00   182 ;
                     0C00   183 ;       none
                     0C00   184 ;
                     0C00   185 ; SIDE EFFECTS:
                     0C00   186 ;
                     0C00   187 ;       none
                     0C00   188 ;
                     0C00   189 ;--
                     0C00   190
```

```
                                0C00    192          .ALIGN  QUAD
                                0C00    193
                                0C00    194 ISSH_ENTRY::                                ; This code is re-entrant many times
                                0C00    195                                             ;   so all the local variables used in
                                0C00    196                                             ;   here should have its own local
                                0C00    197                                             ;   stack, and stack pointers to be
                                0C00    198                                             ;   managed
        50   6E   06   C3      0C00    199          SUBL3   #6, (SP), R0                 ; get system sevice vector address,
                                0C04    200                                             ;   place it in R0
   50   00000000'8F   D1      0C04    201          CMPL    #SYS$RUNDWN, R0                ; run down?
                 11   12      0C0B    202          BNEQ    10$                            ; No
                                0C0D    203                                             ; Yes, time to get out
                                0C0D    204
                                0C0D    205 ; Run Down
                                0C0D    206 ;
   51   000009F4'8F   D0      0C0D    207          MOVL    #SYS$QIOW+<SGN$C_SYSVECPGS*512-12>,R1
                                0C14    208                                             ; get the user defined system service
                                0C14    209                                             ;   address from saved area
                      50   DD  0C14    210          PUSHL   R0                             ; USER_ADDR=SYS$RUNDWN
                   00 B1   01   FB      0C16    211          CALLS   #1, @(R1)              ; Re-enter privileged shareable image
                                0C1A    212                                             ;   DBGSSISHR.EXE to clean up
                      6E   04   C2      0C1A    213          SUBL2   #4, (SP)               ; Pop off return address on
                                0C1D    214                                             ;   the stack
                           05   0C1D    215          RSB                                ; return to retry point, contiune
                                0C1E    216                                             ;   the real run-down
                                0C1E    217
                                0C1E    218
                                0C1E    219 ; Get the address of the system service in P0
                                0C1E    220 ;
        51   F3DE   CF   9E      0C1E    221 10$:     MOVAB   TV_CPY,R1                  ; get the P0 base address
   50   50   00000000'8F   C3      0C23    222          SUBL3   #SYS$QIOW, R0, R0          ; get the offset
        51   51   50   C1      0C2B    223          ADDL3   R0, R1, R1                 ; insert offset into R1, this
                                0C2F    224                                             ;   brings into P0 area
                                0C2F    225
                                0C2F    226 ; Get the current mode
                                0C2F    227 ;
                      7E   DC   0C2F    228          MOVPSL  -(SP)                      ; get PSL
                                0C31    229          ASSUME  <PSL$V_CURMOD+PSL$S_CURMOD>,EQ,PSL$V_IS
                   6E   03   18   EF    0C31    230          EXTZV   #PSL$V_CURMOD, -           ; get current mode + int stk bit
                                0C35    231                  #1+PSL$S_CURMOD, -
                           50           0C35    232                  (SP), -
                                0C36    233                  R0                        ; Mode in R0
                      5E   04   C0      0C36    234          ADDL    #4,SP              ; Pop off PSL
                      03   50   D1      0C39    235          CMPL    R0,#PSL$C_USER     ; Strictly allow interception only in
                                0C3C    236                                             ;  in user mode
                           06   13      0C3C    237          BEQL    15$
                                0C3E    238
                      5E   04   C0      0C3E    239          ADDL    #4,SP              ; Pop off the RET on stack
                        02 A1   17      0C41    240          JMP     2(R1)              ; Continue system service
                                0C44    241
                                0C44    242
                                0C44    243 ; Check to see if this system service is running. If it is, don't intercept
                                0C44    244 ; its own system service calls.
                                0C44    245 ;
                   B1 AF   9F   0C44    246 15$:     PUSHAB  ISSH_RUNNING_FLAG          ; Get the address of the running
              50   00 BE   D0   0C47    247          MOVL    @(SP),R0                   ;   flag (global variable, no
                                0C4B    248                                             ;   need to stack it.)
```

```
                     60   D5  0C4B   249          TSTL    (R0)                    ; Test to see if it is set
                     06   13  0C4D   250          BEQL    20$                     ; No
                  5E 08   C0  0C4F   251          ADDL    #8,SP                   ; Yes, pop off the stack
                  02 A1   17  0C52   252          JMP     2(R1)                   ; Continue with the system service
                          0C55   253
                          0C55   254
                          0C55   255 ; Check to see if there is an index ID, if there is no ID, don't intercept
                          0C55   256 ;
                          0C55   257 20$:
                     98 AF   9F  0C55   258          PUSHAB  ISSH_CTRL_INDEX         ; Get the address of the Index
                  50 00 BE   D0  0C58   259          MOVL    @(SP),R0                ;   (global variable)
                     50   D5  0C5C   260          TSTL    R0                      ; Test to see if it is zero
                     06   12  0C5E   261          BNEQ    25$                     ; There is user routine declared
                  5E 0C   C0  0C60   262          ADDL    #12,SP                  ; Pop Return address and stack locals
                  02 A1   17  0C63   263          JMP     2(R1)                   ; Continue system service
                          0C66   264
                          0C66   265
                          0C66   266 ; We may be able to call the user routine at this point.
                          0C66   267 ;
                          0C66   268 25$:
                  5E   08   C0  0C66   269          ADDL    #8,SP                   ; Pop off RUNNING_FLAG & CTRL_INDEX
                          0C69   270
                          0C69   271
                          0C69   272 34$:
                  0FFC 8F   BB  0C69   273          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>; Save registers
                  57   03   D0  0C6D   274          MOVL    #PSL$C_USER,R7          ; R7 has the user mode only
                  55   51   D0  0C70   275          MOVL    R1,R5                   ; R5 points to P0 saved copy
               5A 28 AE   06   C3  0C73   276          SUBL3   #6,40(SP),R10           ; R10 has the system service index
                  56 FF74 CF   D0  0C78   277          MOVL    ISSH_CTRL_INDEX,R6      ; R6 points to the last entry
                          0C7D   278                                                  ;   in the routine table
                          0C7D   279
                  52 FF3C CF   9E  0C7D   280          MOVAB   ISSH_STACK+1,R2         ; Get local stack address in R2
               53 52 FF3A CF   C3  0C82   281          SUBL3   ISSH_STKPTR,R2,R3       ; Get local stack pointer in R3
                  53   63   9A  0C88   282          MOVZBL  (R3),R3                 ; Get Old ptr value, this tells us
                          0C8B   283                                                  ;   where was last intercepted
                  FF31 CF   D6  0C8B   284          INCL    ISSH_STKPTR             ; Increment local stack pointer
                  52 FF2D CF   C2  0C8F   285          SUBL2   ISSH_STKPTR,R2          ; R2 points to the current priority
                          0C94   286                                                  ;   level
                  50 FE68 CF   9E  0C94   287          MOVAB   ISSH_USER_DATA_BEG,R0   ; Test to see if we overflow the stack
                  50   52   D1  0C99   288          CMPL    R2,R0                   ;   if we are, don't intercept anymore
                          07   14  0C9C   289          BGTR    70$                     ; We are Ok
                  FF1E CF   D7  0C9E   290          DECL    ISSH_STKPTR             ; Take off what we had done
                       00AC   31  0CA2   291          BRW     54$                     ; Continue System service
                          0CA5   292
                          62   94  0CA5   293 70$:    CLRB    (R2)                    ; Clear
                          0CA7   294
                  58 FF51 CF   D0  0CA7   295          MOVL    ISSH_PRIO_MASK,R8       ; R8 has routine enable mask reflecting
                          0CAC   296                                                  ;   all priorities
                          0CAC   297
                          0CAC   298                                                  ; This is used by Prio, 3 and 4 only
                          00   53  D1  0CAC   299          CMPL    R3,#0                   ; R3 could have 0, 1, 2, 3 or 4 value
                          0CAF   300                                                  ;   0 means we are starting a new cycle
                          12   13  0CAF   301          BEQL    36$                     ; Yes, new cycle
                          03   53  D1  0CB1   302          CMPL    R3,#3                   ; No, Did we come from 3 (DBG?)
                          0D   13  0CB4   303          BEQL    36$                     ; Yes, some more checking to do
                  54 FF32 CF   9E  0CB6   304 35$:    MOVAB   ISSH_BIT_15_CNT_12,R4   ; No, so, we must come from 1, 2, or 4
                          0CBB   305                                                  ;   don't need to do anything except
```

```
                        OCBB    306                             ;   to make the code path works.
                64  96  OCBB    307             INCB    (R4)    ; Plunk in address in R4, increment the
          5B    01  54  C1  OCBD    308             ADDL3   R4,#1,R11       ; count, set pointer to next byte in R11
                28  11  OCC1    309             BRB     39$     ; Ready to start
                03  53  D1  OCC3    310 36$:        CMPL    R3,#3   ; We either start a new cycle or came
                        OCC6    311                             ;   from 3
                09  13  OCC6    312             BEQL    37$     ; Yes, we were at 3
          54    FF1C CF  9E  OCC8    313             MOVAB   ISSH_BIT_15_CNT_3,R4    ; No, we are starting a new cycle, so,
                        OCCD    314                             ;   DBG prio. 3 sees this service first,
                        OCCD    315                             ;   passes in the communication variable
            09  58  02  E0  OCCD    316             BBS     #2,R8,38$       ; Is prio. 3 there?  No, more checking!
            E1  58  03  E1  OCD1    317 37$:        BBC     #3,R8,35$       ; We were at 3, or 3 is not here, Is
                        OCD5    318                             ;   prio. 4 there?  No, plung in dummy
          54    FEFB CF  9E  OCD5    319             MOVAB   ISSH_BIT_15_CNT_4,R4    ; Yes, we really need to use prio. 4
                        OCDA    320                             ;   communication path
                64  96  OCDA    321 38$:        INCB    (R4)    ; Mark the fact that we have one system
                        OCDC    322                             ;   service, note, if this count is
                        OCDC    323                             ;   greater than 1 that means we are
                        OCDC    324                             ;   in nested services
                01  64  D1  OCDC    325             CMPL    (R4),#1 ; If we just start a new system service
                06  14  OCDF    326             BGTR    72$     ; No, we are in nested one
          5B    01  54  C1  OCE1    327             ADDL3   R4,#1,R11       ; Set pointer in R11 to point to
                        OCE5    328                             ;   FP RET count for each system
                        OCE5    329                             ;   service in priority 3 or 4.
                        OCE5    330                             ; We need this to set bit 15 to cause
                        OCE5    331                             ;   reserve operand fault in right FP,
                        OCE5    332                             ;   in right priority, and turn on/off
                        OCE5    333                             ;   bit 15 at the right level.
                04  11  OCE5    334             BRB     39$
                        OCE7    335 72$:
          5B    02  54  C1  OCE7    336             ADDL3   R4,#2,R11       ; Mark the fact that we don't cause
                        OCEB    337                             ;   reserve operand fault in FP for
                        OCEB    338                             ;   nested ones
                59  D4  OCEB    339 39$:        CLRL    R9      ; Clear
                51  D4  OCED    340             CLRL    R1      ; Clear
                        OCEF    341
                        OCEF    342 ; Loop till R6 reach to zero.
                        OCEF    343 ;
                        OCEF    344 40$:
                59  56  D1  OCEF    345             CMPL    R6,R9   ; Have we gone through all the
                        OCF2    346                             ;   priorities in this mode?
                40  13  OCF2    347             BEQL    50$     ; Yes, all done
                62  96  OCF4    348             INCB    (R2)    ; No, rememeber which priority we
                        OCF6    349                             ;   are in
                59  53  91  OCF6    350             CMPB    R3,R9   ; We never intercept system service
                        OCF9    351                             ;   that came from the same priroity
                34  14  OCF9    352             BGTR    45$     ; Yes, from same, next
            30  58  59  E1  OCFB    353             BBC     R9,R8,45$       ; Test to see if the given priority
                        OCFF    354                             ;   is enabled in the mask, if it
                        OCFF    355                             ;   is not, loop to next
          50    59  04  C5  OCFF    356             MULL3   #4,R9,R0        ; Index * 4 modes
                50  57  C0  0D03    357             ADDL    R7,R0   ; positioned to current mode
    51    FCF5 CF40  DE  0D06    358             MOVAL   ISSH_DATA_BEG[R0],R1    ; address of the routine
                61  D5  0D0C    359             TSTL    (R1)    ; Is any routine name there?
                1F  13  0D0E    360             BEQL    45$     ; No, go on next
                0FFC 8F  BB  0D10    361             PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                        0D14    362                             ; Get ready to call user routine
```

```
                    58   DD   0D14   363          PUSHL   R8                      ; Current mask
                    54   DD   0D16   364          PUSHL   R4                      ; Total SV count originated from
                         0D18   365                                               ;   a service
                    5B   DD   0D18   366          PUSHL   R11                     ; FP RET count for the same service
                    5D   DD   0D1A   367          PUSHL   FP                      ; FP for this system service
                    5C   DD   0D1C   368          PUSHL   AP                      ; SS argument list
                    5A   DD   0D1E   369          PUSHL   R10                     ; System Service index
               00 B1    06   FB   0D20   370          CALLS   #6,@(R1)                ; Call the user routine
          51   FFFFFFFF  8F   D0   0D24   371          MOVL    #^XFFFFFFFF,R1          ; Indicate the fact that we went
                         0D2B   372                                               ;   through this loop
                  0FFC  8F   BA   0D2B   373          POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                    59   D6   0D2F   374  45$:     INCL    R9                      ; Next higher priority
                      FFBB   31   0D31   375          BRW     40$                     ; Next
                         0D34   376
                         0D34   377  50$:
                  FE88  CF   D7   0D34   378          DECL    ISSH_STKPTR             ; Pop the local stack
               50 FE80  CF   9E   0D38   379          MOVAB   ISSH_BIT_15_CNT_12,R0   ; Get the address
                    54   50   D1   0D3D   380          CMPL    R0,R4                   ; If we have this dummy address set
                         0D40   381                                               ;   in R4, decrement its value,
                         0D40   382                                               ;   for we have increment it, and
                         0D40   383                                               ;   there is no where else to balance
                         0D40   384                                               ;   the count, so do it here.
                    04   12   0D40   385          BNEQ    52$                     ; No, next check
                    64   97   0D42   386          DECB    (R4)                    ; Yes, decrement, and continue
                    0B   11   0D44   387          BRB     54$                     ;
          51   FFFFFFFF  8F   D1   0D46   388  52$:     CMPL    #^XFFFFFFFF,R1          ; If we have gone through the loop,
                    02   13   0D4D   389          BEQL    54$                     ;   (DEBUG interception routine takes
                         0D4F   390                                               ;   care of the count)
                    64   97   0D4F   391          DECB    (R4)                    ; Else, we got to pop off the count
               51   55   D0   0D51   392  54$:     MOVL    R5,R1                   ; Restore the address, continue
               0FFC 8F   BA   0D54   393          POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
               5E   04   C0   0D58   394  100$:    ADDL    #4,SP                   ; Pop of return
                 02 A1   17   0D5B   395          JMP     2(R1)                   ; System service continues
                         0D5E   396
                         0D5E   397
```

```
                0D5E   399          .ALIGN  PAGE
                0E00   400
00000E00        0E00   401 ISSH_CODE_LENGTH==      .-ISSH_VEC_BASE
00000E00        0E00   402 ISSH_VEC_LENGTH==       .-ISSH_VEC_BASE
                0E00   403
                0E00   404          .END
```

```
DATA                              00000A00 R      02
ISSH_BIT_15_CNT_12              = 00000BEC RG     02
ISSH_BIT_15_CNT_3               = 00000BE8 RG     02
ISSH_BIT_15_CNT_4               = 00000BD4 RG     02
ISSH_CODE_LENGTH                = 00000E00   G
ISSH_CTRL_INDEX                 = 00000BF0 RG     02
ISSH_CTRL_PRIO_INDEX            = 00000BF4 RG     02
ISSH_DATA_BEG                     00000A00 RG     02
ISSH_DATA_END                     00000C00 RG     02
ISSH_ENTRY                        00000C00 RG     02
ISSH_PRIO_MASK                  = 00000BFC RG     02
ISSH_RUNNING_FLAG               = 00000BF8 RG     02
ISSH_STACK                      = 00000BBC RG     02
ISSH_STKPTR                     = 00000BC0 RG     02
ISSH_USER_ADDR                  = 00000A00 RG     02
ISSH_USER_DATA_BEG              = 00000B00 RG     02
ISSH_VEC_BASE                     00000000 RG     02
ISSH_VEC_LENGTH                 = 00000E00   G
PSL$C_USER                      = 00000003
PSL$S_CURMOD                    = 00000002
PSL$V_CURMOD                    = 00000018
PSL$V_IS                        = 0000001A
SGN$C_SYSVECPGS                 = 00000005
SYS$QIOW                          ********     X  02
SYS$RUNDWN                        ********     X  02
TV_CPY                            00000000 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
$ISSH_CODE$                     00000E00  ( 3584.)  02 (  2.)    PIC   USR   CON   REL   GBL NOSHR NOEXE   RD  NOWRT NOVEC PAGE
```

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	14	00:00:00.10	00:00:00.87
Command processing	86	00:00:00.85	00:00:03.91
Pass 1	136	00:00:02.13	00:00:07.22
Symbol table sort	0	00:00:00.08	00:00:00.62
Pass 2	89	00:00:00.93	00:00:02.68
Symbol table output	4	00:00:00.03	00:00:00.03
Psect synopsis output	2	00:00:00.02	00:00:00.08
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	333	00:00:04.16	00:00:15.42

```
The working set limit was 1050 pages.
10332 bytes (21 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 90 non-local and 18 local symbols.
404 source lines were read in Pass 1, producing 13 object records in Pass 2.
```

11 pages of virtual memory were used to define 10 macros.

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

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	7

142 GETS were required to define 7 macros.

There were no errors, warnings or information messages.

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

GETMEMORY
LIS

DSTRECORDS
LIS

ISSH
LIS

RESETSSI
LIS