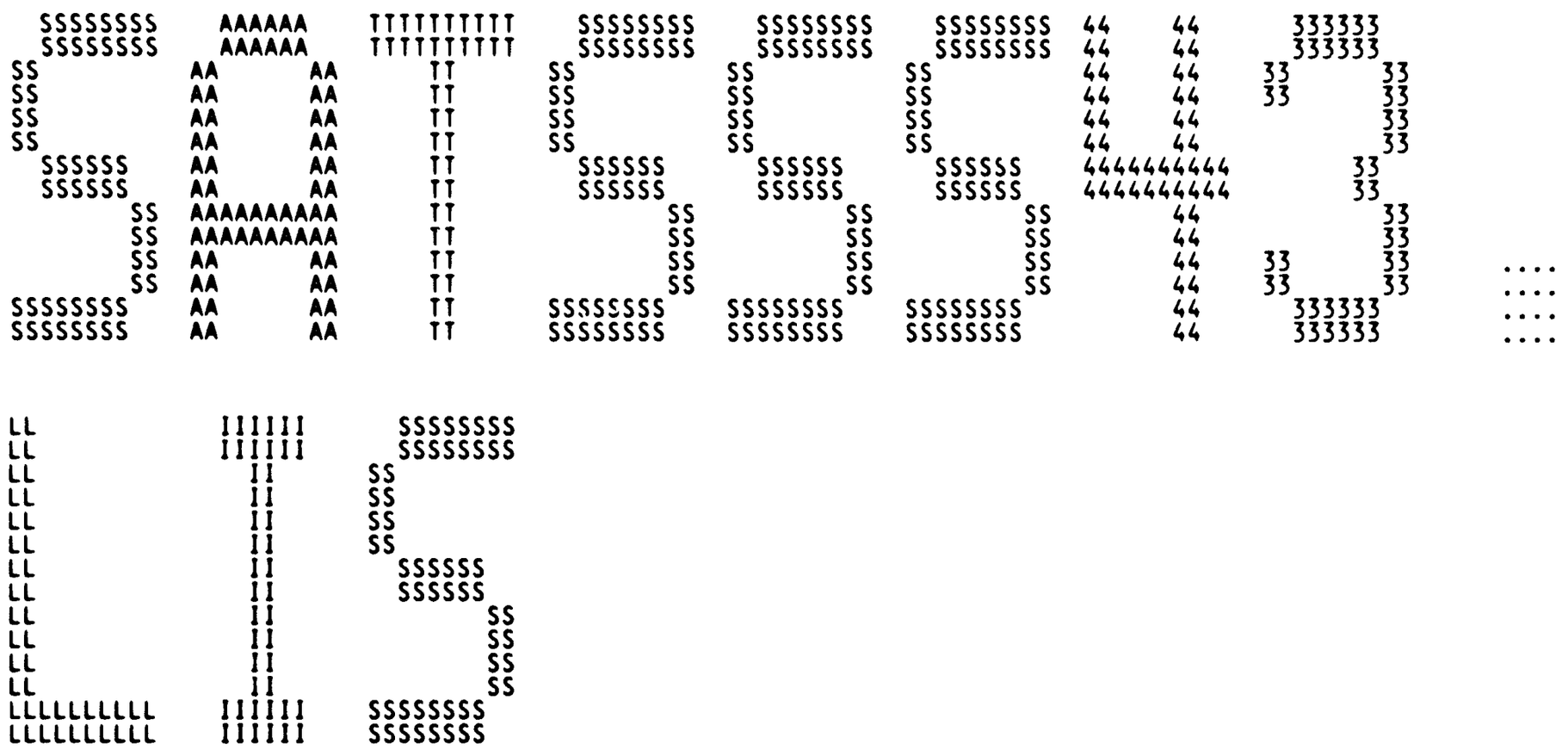
```
UUU          UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSS  YYY          YYY
UUU          UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSS  YYY          YYY
UUU          UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSS  YYY          YYY
UUU          UUU  EEE                    TTT        PPP          PPP  SSS              YYY          YYY
UUU          UUU  EEE                    TTT        PPP          PPP  SSS              YYY          YYY
UUU          UUU  EEE                    TTT        PPP          PPP  SSS              YYY          YYY
UUU          UUU  EEE                    TTT        PPP          PPP  SSS                 YYY    YYY
UUU          UUU  EEE                    TTT        PPP          PPP  SSS                 YYY    YYY
UUU          UUU  EEE                    TTT        PPP          PPP  SSS                 YYY    YYY
UUU          UUU  EEEEEEEEEEEE           TTT        PPPPPPPPPPPP       SSSSSSSSS              YYY
UUU          UUU  EEEEEEEEEEEE           TTT        PPPPPPPPPPPP       SSSSSSSSS              YYY
UUU          UUU  EEEEEEEEEEEE           TTT        PPPPPPPPPPPP       SSSSSSSSS              YYY
UUU          UUU  EEE                    TTT        PPP                        SSS              YYY
UUU          UUU  EEE                    TTT        PPP                        SSS              YYY
UUU          UUU  EEE                    TTT        PPP                        SSS              YYY
UUU          UUU  EEE                    TTT        PPP                        SSS              YYY
UUU          UUU  EEE                    TTT        PPP                        SSS              YYY
UUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP                SSSSSSSSSSS              YYY
UUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP                SSSSSSSSSSS              YYY
UUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP                SSSSSSSSSSS              YYY
```


SSSSSSSS AAAAAA TTTTTTTTTT SSSSSSSS SSSSSSSS SSSSSSSS 44 44 333333
SSSSSSSS AAAAAA TTTTTTTTTT SSSSSSSS SSSSSSSS SSSSSSSS 44 44 333333
SS AA AA TT SS SS SS 44 44 33 33
SS AA AA TT SS SS SS 44 44 33 33
SS AA AA TT SS SS SS 44 44 33
SS AA AA TT SS SS SS 44 44 33
SSSSSS AA AA TT SSSSSS SSSSSS SSSSSS 4444444444 33
SSSSSS AA AA TT SSSSSS SSSSSS SSSSSS 4444444444 33
SS AAAAAAAAAA TT SS SS SS 44 33
SS AAAAAAAAAA TT SS SS SS 44 33
SS AA AA TT SS SS SS 44 33 33
SS AA AA TT SS SS SS 44 33 33
SSSSSSSS AA AA TT SSSSSSSS SSSSSSSS SSSSSSSS 44 333333
SSSSSSSS AA AA TT SSSSSSSS SSSSSSSS SSSSSSSS 44 333333
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  SATSSS43 - SATS SYSTEM SERVICE TESTS  (SUCC S.C.)
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:     SATS SYSTEM SERVICE TESTS
0000    31  ;
0000    32  ; ABSTRACT:     The SATSSS43 module tests the execution of the following
0000    33  ;               VMS system services:
0000    34  ;
0000    35  ;               $DCLCMH
0000    36  ;               $DCLEXH
0000    37  ;               $CANEXH
0000    38  ;
0000    39  ;
0000    40  ; ENVIRONMENT:  User, Supervisor and Executive mode image.
0000    41  ;               Needs CMKRNL privilege and dynamically acquires other
0000    42  ;               privileges, as needed.
0000    43  ;
0000    44  ; AUTHOR: THOMAS L. CAFARELLA,          CREATION DATE: MMM, 1978
0000    45  ;         PAUL D. FAY (DISPSERV & TESTSERV MACROS)
0000    46  ;
0000    47  ; MODIFIED BY:
0000    48  ;
0000    49  ;       V03-001 LDJ0001         Larry D. Jones,         17-Sep-1980
0000    50  ;               Modified to conform to new build command procedures.
0000    51  ;--
0000    52  ;--
```

```
                     0000     54          .SBTTL  DECLARATIONS
                     0000     55 ;
                     0000     56 ; MACRO LIBRARY CALLS
                     0000     57 ;
                     0000     58          $PCBDEF                                 ; PCB definitions
                     0000     59          $PHDDEF                                 ; process header definitions
                     0000     60          $PRDEF                                  ; processor register definitions
                     0000     61          $PRVDEF                                 ; privilege definitions
                     0000     62          $PSLDEF                                 ; PSL definitions
                     0000     63          $SFDEF                                  ; Stack Frame definitions
                     0000     64          $SHR_MESSAGES UETP,116,<<TEXT,INFO>> ; UETP$_TEXT definition
                     0000     65          $SSDEF                                  ; system service definitions
                     0000     66          $STSDEF                                 ; STS definitions
                     0000     67          $UETPDEF                                ; UETP message definitions
                     0000     68 ;
                     0000     69 ; Equated symbols
                     0000     70 ;
00000000             0000     71 WARNING          = 0                             ; warning severity value for msgs
00000001             0000     72 SUCCESS          = 1                             ; success       "       "   "   "
00000002             0000     73 ERROR            = 2                             ; error         "       "   "   "
00000003             0000     74 INFO             = 3                             ; information   "       "   "   "
00000004             0000     75 SEVERE           = 4                             ; fatal         "       "   "   "
                     0000     76 ;
                     0000     77          .SBTTL  MACROS
                     0000     78 ;
                     0000     79          .MACRO  EHDB    MODE,NUM
                     0000     80          .LIST   MEB
                     0000     81 MODE'NUM:
                     0000     82                  .LONG   0
                     0000     83                  .ADDRESS MODE'H'NUM
                     0000     84                  .LONG   2
                     0000     85                  .ADDRESS STATUS
                     0000     86                  .LONG   NUM
                     0000     87          .NLIST MEB
                     0000     88          .ENDM   EHDB
                     0000     89 ;
```

```
                                00000000     91          .PSECT  RODATA,RD,NOWRT,NOEXE,LONG
                                    0000     92 ;
                                    0000     93 TEST_MOD_NAME:
        33 34 53 53 53 54 41 53 00' 0000     94          .ASCIC  /SATSSS43/              ; needed for SATSMS message
                                    08   0000
                                    0009     95 TEST_MOD_NAME_D:
53 53 53 54 41 53 00000011'010E0000' 0009     96          .ASCID  /SATSSS43/              ; module name
                              33 34  0017
                                    0019     97 TEST_MOD_BEGIN:
              6E 75 67 65 62 00'    0019     98          .ASCIC  /begun/
                                    05   0019
                                    001F     99 TEST_MOD_SUCC:
   6C 75 66 73 73 65 63 63 75 73 00' 001F    100          .ASCIC  /successful/
                                    0A   001F
                                    002A    101 TEST_MOD_FAIL:
           64 65 6C 69 61 66 00'    002A    102          .ASCIC  /failed/
                                    06   002A
                                    0031    103 DCLCMH:
           48 4D 43 4C 43 44 00'    0031    104          .ASCIC  /DCLCMH/
                                    06   0031
                                    0038    105 DCLEXH:
           48 58 45 4C 43 44 00'    0038    106          .ASCIC  /DCLEXH/
                                    06   0038
                                    003F    107 CANEXH:
           48 58 45 4E 41 43 00'    003F    108          .ASCIC  /CANEXH/
                                    06   003F
                                    0046    109 CS1:
21 20 74 73 65 54 0000004E'010E0000' 0046    110          .ASCID  \Test !AC service name !AC step !UL failed.\
6E 20 65 63 69 76 72 65 73 20 43 41  0054
70 65 74 73 20 43 41 21 20 65 6D 61  0060
2E 64 65 6C 69 61 66 20 4C 55 21 20  006C
                                    0078    111 CS2:
74 63 65 70 78 45 00000080'010E0000' 0078    112          .ASCID  \Expected !AS = !XL received !AS = !XL\
4C 58 21 20 3D 20 53 41 21 20 64 65  0086
41 21 20 64 65 76 69 65 63 65 72 20  0092
              4C 58 21 20 3D 20 53   009E
                                    00A5    113 CS3:
74 63 65 70 78 45 000000AD'010E0000' 00A5    114          .ASCID  \Expected !AS!UB = !XL received !AS!UB = !XL\
20 3D 20 42 55 21 53 41 21 20 64 65  00B3
64 65 76 69 65 63 65 72 20 4C 58 21  00BF
58 21 20 3D 20 42 55 21 53 41 21 20  00CB
                              4C     00D7
                                    00D8    115 CS4:
65 70 78 65 6E 55 000000E0'010E0000' 00D8    116          .ASCID  \Unexpected !AS mode exit handler found in !AS.\
64 6F 6D 20 53 41 21 20 64 65 74 63  00E6
6C 64 6E 61 68 20 74 69 78 65 20 65  00F2
20 6E 69 20 64 6E 75 6F 66 20 72 65  00FE
                        2E 53 41 21  010A
                                    010E    117 CS5:
77 20 65 64 6F 4D 00000116'010E0000' 010E    118          .ASCID  \Mode was !AS.\
              2E 53 41 21 20 73 61   011C
                                    0123    119 UM:
       72 65 73 75 0000012B'010E0000' 0123    120          .ASCID  \user\
                                    012F    121 SM:
   72 65 70 75 73 00000137'010E0000' 012F    122          .ASCID  \super\
                                    013C    123 EM:
74 75 63 65 78 65 00000144'010E0000' 013C    124          .ASCID  \executive\
```

```
                                                B 6
                          65 76 69  014A
                                    014D    125 EXP:
73 75 74 61 74 73 00000155'010E0000' 014D    126         .ASCID  \status\
```

```
                     015B   128 ;
                     015B   129         .SBTTL  R/W PSECT
                 00000000   130         .PSECT  RWDATA,RD,WRT,NOEXE,LONG
                     0000   131 ;
                     0000   132 TPID:
                 00000000   0000   133         .LONG   0                       ; PID for this process
                     0004   134 CURRENT_TC:
                 00000000   0004   135         .LONG   0                       ; ptr to current test case
                     0008   136         .ALIGN LONG
                     0008   137 REG_SAVE_AREA:
                 00000044   0008   138         .BLKL   15                      ; register save area
                     0044   139 MOD_MSG_CODE:
                 00748009   0044   140         .LONG   UETP$_SATSMS            ; test module message code for putmsg
                     0048   141 TMN_ADDR:
                 00000000'  0048   142         .ADDRESS TEST_MOD_NAME
                     004C   143 TMD_ADDR:
                 00000019'  004C   144         .ADDRESS TEST_MOD_BEGIN
                     0050   145 PRVPRT:
                       00   0050   146         .BYTE   0                       ; protection return byte for SETPRT
                     0051   147 PRIVMASK:
        00000000 00000000   0051   148         .QUAD   0                       ; priv. mask
                     0059   149 CHM_CONT:
                 00000000   0059   150         .LONG   0                       ; change mode continue address
                     005D   151 RETADR:
                 00000065   005D   152         .BLKL   2                       ; returned address's from SETPRT
                     0065   153 STATUS:
                 00000000   0065   154         .LONG   0
                     0069   155 MODE:
                 00000000   0069   156         .LONG   0
                     006D   157 DCL:
                            006D   158         $DCLCMH DUMMY,OHC,0             ; DCLCMH parameter list
                     007D   159 DCL1:
                            007D   160         $DCLEXH EXEC3                   ; DCLEXH parameter list
                     0C85   161 CAN:
                            0085   162         $CANEXH EXEC1                   ; CANEXH parameter list
                     008D   163 REG:
74 73 69 67 65 72 00000095'010E0000'  008D   164         .ASCID  \register R\
                   52 20 72 65  009B
                     009F   165 REGNUM:
                 00000000   009F   166         .LONG   0                       ; register number
                     00A3   167 MSGL:
                 00000050   00A3   168         .LONG   80                      ; buffer desc.
                 000000AB'  00A7   169         .ADDRESS BUF
                     00AB   170 BUF:
                 000000FB   00AB   171         .BLKB   80
                     00FB   172 MESSAGEL:
                 00000000   00FB   173         .LONG   0                       ; message desc.
                 000000AB'  00FF   174         .ADDRESS BUF
                     0103   175 SERV_NAME:
                 00000000   0103   176         .LONG   0                       ; service name pointer
                     0107   177 PRVHND1:
                 00000000   0107   178         .LONG   0                       ; previous handler address 1
                     010B   179 PRVHND2:
                 00000000   010B   180         .LONG   0                       ; previous handler address 2
                     010F   181 PRVHND3:
                 00000000   010F   182         .LONG   0                       ; previous handler address 3
                     0113   183 OHC:
```

```
                                                D 6
00000000  0113   184          .LONG   0                       ; old handler check location
          0117   185 ARGLST:
00000001  0117   186          .LONG   1                       ; super mode setup arg list
000003E6' 011B   187          .ADDRESS SUPER_MODE
          011F   188 MSGVEC:                                  ; PUTMSG message vector
00000003  011F   189          .LONG   3
00741133  0123   190          .LONG   UETP$_TEXT
00000001  0127   191          .LONG   1
000000FB' 012B   192          .ADDRESS MESSAGEL
          012F   193 MSGVEC1:
00000004  012F   194          .LONG   4                       ; PUTMSG message vector for exit
00000000  0133   195          .LONG   0
00000002  0137   196          .LONG   2
00000143  013B   197          .BLKL   2
```

```
                     0143   199 ; exit handler desc. blocks
                     0143   200         EHDB    USER,1                  ; user #1 will be deleted
                     0143       USER1:
00000000             0143                .LONG   0
00000735'            0147                .ADDRESS USERH1
00000002             014B                .LONG   2
00000065'            014F                .ADDRESS STATUS
00000001             0153                .LONG   1
                     0157   201         EHDB    USER,2                  ; user #2 will be used
                     0157       USER2:
00000000             0157                .LONG   0
00000353'            015B                .ADDRESS USERH2
00000002             015F                .LONG   2
00000065'            0163                .ADDRESS STATUS
00000002             0167                .LONG   2
                     016B   202         EHDB    USER,3                  ; user #3 will be deleted
                     016B       USER3:
00000000             016B                .LONG   0
00000735'            016F                .ADDRESS USERH3
00000002             0173                .LONG   2
00000065'            0177                .ADDRESS STATUS
00000003             017B                .LONG   3
                     017F   203         EHDB    USER,4                  ; user #4 will be used
                     017F       USER4:
00000000             017F                .LONG   0
0000034C'            0183                .ADDRESS USERH4
00000002             0187                .LONG   2
00000065'            018B                .ADDRESS STATUS
00000004             018F                .LONG   4
                     0193   204         EHDB    SUPER,1                 ; super #1 will be deleted
                     0193       SUPER1:
00000000             0193                .LONG   0
00000740'            0197                .ADDRESS SUPERH1
00000002             019B                .LONG   2
00000065'            019F                .ADDRESS STATUS
00000001             01A3                .LONG   1
                     01A7   205         EHDB    SUPER,3                 ; super #3 will be deleted
                     01A7       SUPER3:
00000000             01A7                .LONG   0
00000740'            01AB                .ADDRESS SUPERH3
00000002             01AF                .LONG   2
00000065'            01B3                .ADDRESS STATUS
00000003             01B7                .LONG   3
                     01BB   206         EHDB    EXEC,1                  ; exec #1 will be deleted
                     01BB       EXEC1:
00000000             01BB                .LONG   0
0000074B'            01BF                .ADDRESS EXECH1
00000002             01C3                .LONG   2
00000065'            01C7                .ADDRESS STATUS
00000001             01CB                .LONG   1
                     01CF   207         EHDB    EXEC,3                  ; exec #3 will be deleted
                     01CF       EXEC3:
00000000             01CF                .LONG   0
0000074B'            01D3                .ADDRESS EXECH3
00000002             01D7                .LONG   2
00000065'            01DB                .ADDRESS STATUS
00000003             01DF                .LONG   3
```

```
00000000   209          .PSECT  SATSSS43,RD,WRT,EXE,LONG
     0000  210          .SBTTL  SATSSS43
     0000  211 ;++
     0000  212 ; FUNCTIONAL DESCRIPTION:
     0000  213 ;
     0000  214 ;       After performing some initial housekeeping, such as
     0000  215 ; printing the module begin message and acquiring needed privileges,
     0000  216 ; the system services are tested in each of their normal conditions.
     0000  217 ; Detected failures are identified and  an error message is printed
     0000  218 ; on the terminal.  Upon completion of the test a success or fail
     0000  219 ; message is printed on the terminal.
     0000  220 ;
     0000  221 ; CALLING SEQUENCE:
     0000  222 ;
     0000  223 ;       $ RUN SATSSS43  ...  (DCL COMMAND)
     0000  224 ;
     0000  225 ; INPUT PARAMETERS:
     0000  226 ;
     0000  227 ;       none
     0000  228 ;
     0000  229 ; IMPLICIT INPUTS:
     0000  230 ;
     0000  231 ;       none
     0000  232 ;
     0000  233 ; OUTPUT PARAMETERS:
     0000  234 ;
     0000  235 ;       none
     0000  236 ;
     0000  237 ; IMPLICIT OUTPUTS:
     0000  238 ;
     0000  239 ;       Messages to SYS$OUTPUT are the only output from SATSSS43.
     0000  240 ;       They are of the form:
     0000  241 ;
     0000  242 ;               %UETP-S-SATSMS, TEST MODULE SATSSS43 BEGUN ... (BEGIN MSG)
     0000  243 ;               %UETP-S-SATSMS, TEST MODULE SATSSS43 SUCCESSFUL ... (END MSG)
     0000  244 ;               %UETP-E-SATSMS, TEST MODULE SATSSS43 FAILED ... (END MSG)
     0000  245 ;               %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
     0000  246 ;
     0000  247 ; COMPLETION CODES:
     0000  248 ;
     0000  249 ;       The SATSSS43 routine terminates with a $EXIT to the
     0000  250 ;       operating system with a status code defined by UETP$_SATSMS.
     0000  251 ;
     0000  252 ; SIDE EFFECTS:
     0000  253 ;
     0000  254 ;       none
     0000  255 ;
     0000  256 ;--
     0000  257
     0000  258
     0000  259
     0000  260          TEST_START SATSSS43             ; let the test begin
```

```
                                     0000
                              0000   0000                        .ENTRY SATSSS43,0
                 0004'CF       D4    0002                        CLRL    W^CURRENT_TC
                      00       DD    0006                        PUSHL   #0
                 0000'CF       DF    0008                        PUSHAL  W^TPID
          00000000'GF  02      FB    000C                        CALLS   #2,G^SYS$WAKE
          00000000'GF  00      FB    0013                        CALLS   #0,G^SYS$HIBER
                 0009'CF       7F    001A                        PUSHAQ  W^TEST_MOD_NAME_D
          00000000'GF  01      FB    001E                        CALLS   #1,G^SYS$SETPRN
                    0798       30    0025                        BSBW    W^MOD_MSG_PRINT
         004C'CF   001F'CF     DE    0028                        MOVAL   W^TEST_MOD_SUCC,W^TMD_ADDR
  0044'CF   03     00    01    F0    002F                        INSV    #SUCCESS,#0,#3,W^MOD_MSG_CODE
                      00       DD    0036                        PUSHL   #0
              0549'CF 01       FB    0038                        CALLS #1,W^REG_SAVE
                                     003D              STP0:
                                     003D                    .SBTTL  DCLCMH TESTS
                                     003D              ;+
                                     003D              ;
                                     003D              ; $DCLCMH tests
                                     003D              ; test super mode handler declaration
                                     003D              ;
                                     003D              ;-
      0103'CF    0031'CF       DE    003D                MOVAL   W^DCLCMH,W^SERV_NAME      ; set service name
                                     0044                $CMKRNL_S W^SETUP_SUPER,W^ARGLST ; test super mode declaration
           0663'CF   00        FB    0053                CALLS   #0,W^ERLBUF_DUMP          ; report any errors
                     01        BE    0058                CHMS    #1                        ; declare dummy handler
    0113'CF   000003E6'8F      D1    005A                CMPL    #SUPER_MODE,W^OHC         ; make sure it happened
                     11        13    0063                BEQL    10$                       ; br if yes
                 0113'CF       DD    0065                PUSHL   W^OHC                     ; else setup to report the error
                 03E6'CF       DF    0069                PUSHAL  W^SUPER_MODE              ; save the expected results
                 014D'CF       DF    006D                PUSHAL  W^EXP                     ; push the message address
           0691'CF   03        FB    0071                CALLS   #3,W^PRINT_FAIL           ; report the failure
                                     0076          10$:
                     02        BE    0076                CHMS    #2                        ; remove the dummy handler
                                     0078              ;+
                                     0078              ;
                                     0078              ; test user mode handler declaration
                                     0078              ;
                                     0078              ;-
                                     0078                NEXT_TEST
                                     0078
                                     0078              STP1:
           0004'CF   01        D0    0078                        MOVL    #1,W^CURRENT_TC
                     00        DD    007D                        PUSHL   #0
           0549'CF   01        FB    007F                        CALLS   #1,W^REG_SAVE
      0069'CF    0123'CF       DE    0084                MOVAL   W^UM,W^MODE               ; set the mode
      0071'CF    049A'CF       DE    008B                MOVAL   W^DUMMY,W^DCL+DCLCMH$_ADDRES ; reset the handler address
      0075'CF    010B'CF       DE    0092                MOVAL   W^PRVHND2,W^DCL+DCLCMH$_PRVHND ; set new handler save address
                                     0099                $DCLCMH_G W^DCL               ; check _G form
                                     00A2                FAIL_CHECK SS$_NORMAL         ; check for success
                     01        DD    00A2                        PUSHL   #SS$_NORMAL
           0553'CF   01        FB    00A4                        CALLS   #1,W^REG_CHECK
                                     00A9                $DCLCMH_S W^USER_MODE,W^OHC   ; set real handler
                                     00BA                FAIL_CHECK SS$_NORMAL         ; check for success
                     01        DD    00BA                        PUSHL   #SS$_NORMAL
           0553'CF   01        FB    00BC                        CALLS   #1,W^REG_CHECK
    0113'CF   0000049A'8F      D1    00C1                CMPL    #DUMMY,W^OHC              ; is handler address correct?
```

```
              11   13  00CA            BEQL    10$                     ; br if yes
         0113'CF   DD  00CC            PUSHL   W^OHC                   ; push received address
         049A'CF   DF  00D0            PUSHAL  W^DUMMY                 ; push expected address
         014D'CF   DF  00D4            PUSHAL  W^EXP                   ; push string variable
    0691'CF   03   FB  00D8            CALLS   #3,W^PRINT_FAIL         ; print the error
                       00DD      10$:
                       00DD      ;+
                       00DD      ;
                       00DD      ; test for compatiblity mode handler declaration
                       00DD      ;
                       00DD      ;-
                       00DD            NEXT_TEST
                       00DD
                       00DD      STP2:
    0004'CF   02   D0  00DD                    MOVL    #2,W^CURRENT_TC
              00   DD  00E2                    PUSHL   #0
    0549'CF   01   FB  00E4                    CALLS   #1,W^REG_SAVE
 0075'CF   010F'CF DE  00E9            MOVAL   W^PRVHND3,W^DCL+DCLCMH$_PRVHND ; set new handler save location
         0079'CF   D6  00F0            INCL    W^DCL+DCLCMH$_TYPE      ; set to compatiblity mode type
                       00F4            $DCLCMH_G W^DCL                 ; check _G form
                       00FD            FAIL_CHECK SS$_NORMAL           ; check for success
              01   DD  00FD                    PUSHL   #SS$_NORMAL
    0553'CF   01   FB  00FF                    CALLS   #1,W^REG_CHECK
                       0104            $DCLCMH_S W^COMP_MODE,W^OHC,#1  ; set real handler
                       0115            FAIL_CHECK SS$_NORMAL           ; check for success
              01   DD  0115                    PUSHL   #SS$_NORMAL
    0553'CF   01   FB  0117                    CALLS   #1,W^REG_CHECK
 0113'CF  0000049A'8F  D1  011C        CMPL    #DUMMY,W^OHC            ; is handler address correct?
              11   13  0125            BEQL    10$                     ; br if yes
         0113'CF   DD  0127            PUSHL   W^OHC                   ; push received address
         049A'CF   DF  012B            PUSHAL  W^DUMMY                 ; push expected address
         014D'CF   DF  012F            PUSHAL  W^EXP                   ; push string variable
    0691'CF   03   FB  0133            CALLS   #3,W^PRINT_FAIL         ; print the error
                       0138      10$:
                       0138      ;+
                       0138      ;
                       0138      ; check the compatibility mode handler
                       0138      ;
                       0138      ;-
                       0138            NEXT_TEST
                       0138
                       0138      STP3:
    0004'CF   03   D0  0138                    MOVL    #3,W^CURRENT_TC
              00   DD  013D                    PUSHL   #0
    0549'CF   01   FB  013F                    CALLS   #1,W^REG_SAVE
     83C00000 8F   DD  0144            PUSHL   #<<PSL$M_CM>!<PSL$C_USER@PSL$V_PRVMOD>-
                       014A            !<PSL$C_USER@PSL$V_CURMOD>>     ; set compatibility mode
            4E'AF  DF  014A            PUSHAL  B^10$                   ; set new address
              02   014D                REI                             ; enter compatibility mode
```

```
                         014E        .ALIGN  WORD                  ; adjust addressing for PDP-11's
                         014E  10$:
                   15F7  014E        .WORD   ^0012767              ; MOV   #-1,TEST ;prove we were here
                   FFFF  0150        .WORD   ^0177777
                   0002  0152        .WORD   ^0000002
                   0000  0154        .WORD   ^0000000              ; HALT          ;cause an exception
                         0156  TEST:
                   0000  0156        .WORD   ^0000000              ; compatibility mode flag location
                         0158  RETURN:                             ; return to the good life
                         0158  ;+
                         0158  ;
                         0158  ; test the user mode handler
                         0158  ;
                         0158  ;-
                         0158        NEXT_TEST
                         0158
                         0158  STP4:
       0004'CF    04    D0  0158             MOVL    #4,W^CURRENT_TC
                  00    DD  015D             PUSHL   #0
       0549'CF    01    FB  015F             CALLS   #1,W^REG_SAVE
                  05    BF  0164     CHMU    #5                    ; use a param of 5
                         0166  ;+
                         0166  ;
                         0166  ; reset handlers to the original address
                         0166  ;
                         0166  ;-
                         0166        NEXT_TEST
                         0166
                         0166  STP5:
       0004'CF    05    D0  0166             MOVL    #5,W^CURRENT_TC
                  00    DD  016B             PUSHL   #0
       0549'CF    01    FB  016D             CALLS   #1,W^REG_SAVE
 0103'CF   0031'CF      DE  0172     MOVAL   W^DCLCMH,W^SERV_NAME  ; set service name
                         0179        $DCLCMH_S 0,W^PRVHND2         ; reset CHMU handler
                         0188        FAIL_CHECK SS$_NORMAL         ; check for success
                  01    DD  0188             PUSHL   #SS$_NORMAL
       0553'CF    01    FB  018A             CALLS   #1,W^REG_CHECK
                         018F        $DCLCMH_S 0,W^PRVHND3,#1      ; reset CM handler
                         019E        FAIL_CHECK SS$_NORMAL         ; check for success
                  01    DD  019E             PUSHL   #SS$_NORMAL
       0553'CF    01    FB  01A0             CALLS   #1,W^REG_CHECK
```

```
                         01A5   358          .SBTTL  DCLEXH TESTS #1
                         01A5   359 ;+
                         01A5   360 ;
                         01A5   361 ; $DCLEXH tests
                         01A5   362 ;
                         01A5   363 ; These tests are divided into two parts.  This part is the declaration
                         01A5   364 ; tests.  The second part is the servicinn part.
                         01A5   365 ;
                         01A5   366 ; test for exec mode exit handler declaration
                         01A5   367 ;
                         01A5   368 ;-
                         01A5   369          NEXT_TEST
                         01A5
                         01A5        STP6:
    0004'CF    06   DO   01A5                  MOVL    #6,W^CURRENT_TC
               00   DD   01AA                  PUSHL   #0
    0549'CF    01   FB   01AC                  CALLS   #1,W^REG_SAVE
0069'CF  013C'CF   DE   01B1   370          MOVAL   W^EM,W^MODE             ; set the mode
0103'CF  0038'CF   DE   01B8   371          MOVAL   W^DCLEXH,W^SERV_NAME    ; set service name
                        01BF   372          $CMEXEC_S B^10$                 ; get to exec mode
               2C   11  01CB   373          BRB     20$                     ; skip over exec routine
                        01CD   374 10$:
                0000    01CD   375          .WORD   0
               00   DD  01CF   376          PUSHL   #0                      ; push a dummy parameter
    0549'CF    01   FB  01D1   377          CALLS   #1,W^REG_SAVE           ; save a reg snapshot
                        01D6   378          $DCLEXH_S W^EXECT               ; declare #1 exec exit handler
                        01E1   379          FAIL_CHECKNP SS$_NORMAL         ; check for success
               01   DD  01E1                  PUSHL   #SS$_NORMAL
    05EA'CF    01   FB  01E3                  CALLS   #1,W^REG_CHECKNP
                        01E8   380          $DCLEXH_G W^DCL1                ; declare #3 exec exit handler
                        01F1   381          FAIL_CHECKNP SS$_NORMAL         ; check for success
               01   DD  01F1                  PUSHL   #SS$_NORMAL
    05EA'CF    01   FB  01F3                  CALLS   #1,W^REG_CHECKNP
               04   BE  01F8   382          RET                             ; go back to user mode
                        01F9   383 20$:
    0663'CF    00   FB  01F9   384          CALLS   #0,W^ERLBUF_DUMP        ; dump any errors that occured
                        01FE   385 ;+
                        01FE   386 ;
                        01FE   387 ; test super mode exit handler declaration
                        01FE   388 ;
                        01FE   389 ;-
                        01FE   390          NEXT_TEST
                        01FE
                        01FE        STP7:
    0004'CF    07   DO   01FE                  MOVL    #7,W^CURRENT_TC
               00   DD   0203                  PUSHL   #0
    0549'CF    01   FB   0205                  CALLS   #1,W^REG_SAVE
0069'CF  012F'CF   DE   020A   391          MOVAL   W^SM,W^MODE             ; set the mode
               04   BE   0211   392          CHMS    #4                      ; declare 2 super mode exit handlers
```

```
                                0213   394 ;+
                                0213   395 ;
                                0213   396 ; test user mode exit handler declaration
                                0213   397 ;
                                0213   398 ;-
                                0213   399         NEXT_TEST
                                0213
                                0213       STP8:
     0004'CF    08    DO      0213                 MOVL    #8,W^CURRENT_TC
                  00    DD      0218                 PUSHL   #0
     0549'CF    01    FB      021A                 CALLS   #1,W^REG_SAVE
0069'CF   0123'CF    DE      021F   400           MOVAL   W^UM,W^MODE                ; set the mode
                                0226   401           $DCLEXH_S W^USER1                  ; declare #1 user mode exit handler
                                0231   402           FAIL_CHECK SS$_NORMAL              ; check for success
                  01    DD      0231                 PUSHL   #SS$_NORMAL
     0553'CF    01    FB      0233                 CALLS   #1,W^REG_CHECK
0081'CF   0157'CF    DE      0238   403           MOVAL   W^USER2,W^DCL1+DCLEXH$_DESBLK ; set exit handler address
                                023F   404           $DCLEXH_G W^DCL1                   ; declare #2 user mode exit handler
                                0248   405           FAIL_CHECK SS$_NORMAL              ; check for success
                  01    DD      0248                 PUSHL   #SS$_NORMAL
     0553'CF    01    FB      024A                 CALLS   #1,W^REG_CHECK
                                024F   406           $DCLEXH_S W^USER3                  ; declare #3 user mode exit handler
                                025A   407           FAIL_CHECK SS$_NORMAL              ; check for success
                  01    DD      025A                 PUSHL   #SS$_NORMAL
     0553'CF    01    FB      025C                 CALLS   #1,W^REG_CHECK
0081'CF   017F'CF    DE      0261   408           MOVAL   W^USER4,W^DCL1+DCLEXH$_DESBLK ; set exit handler address
                                0268   409           $DCLEXH_G W^DCL1                   ; declare #4 user mode exit handler
                                0271   410           FAIL_CHECK SS$_NORMAL              ; check for success
                  01    DD      0271                 PUSHL   #SS$_NORMAL
     0553'CF    01    FB      0273                 CALLS   #1,W^REG_CHECK
```

```
                          0278             .SBTTL CANEXH TESTS
                          0278       ;+
                          0278       ;
                          0278       ; $CANEXH tests
                          0278       ; test for exec mode exit handler deletion
                          0278       ;
                          0278       ;-
                          0278             NEXT_TEST
                          0278
                          0278       STP9:
      0004'CF    09   D0  0278                       MOVL    #9,W^CURRENT_TC
                 00   DD  027D                       PUSHL   #0
      0549'CF    01   FB  027F                       CALLS   #1,W^REG_SAVE
0069'CF    013C'CF    DE  0284             MOVAL   W^EM,W^MODE                 ; set the mode
0103'CF    003F'CF    DE  028B             MOVAL   W^CANEXH,W^SERV_NAME        ; set service name
                          0292             $CMEXEC_S B^10$                     ; get to exec mode
                 33   11  029E             BRB     20$                         ; skip over the routine
                          02A0       10$:
               0000       02A0             .WORD   0                           ; entry mask
                 00   DD  02A2             PUSHL   #0                          ; push a dummy parameter
      0549'CF    01   FB  02A4             CALLS   #1,W^REG_SAVE               ; save a reg snapshot
                          02A9             $CANEXH_S W^FXECT                   ; cancel exec exit handler #1
                          02B4             FAIL_CHECKNP SS$_NORMAL             ; check for success
                 01   DD  02B4                       PUSHL   #SS$_NORMAL
      05EA'CF    01   FB  02B6                       CALLS   #1,W^REG_CHECKNP
0089'CF    01CF'CF    DE  02BB             MOVAL   W^EXEC3,W^CAN+CANEXH$_DESBLK ; set handler adr
                          02C2             $CANEXH_G W^CAN                     ; cancel exec exit handler #3
                          02CB             FAIL_CHECKNP SS$_NORMAL             ; check for success
                 01   DD  02CB                       PUSHL   #SS$_NORMAL
      05EA'CF    01   FB  02CD                       CALLS   #1,W^REG_CHECKNP
                      04  02D2             RET                                 ; return
                          02D3       20$:
      0663'CF    00   FB  02D3             CALLS #0,W^ERLBUF_DUMP              ; dump any errors that occured
                          02D8       ;+
                          02D8       ;
                          02D8       ; test super mode exit handler cancellation
                          02D8       ;
                          02D8       ;-
                          02D8             NEXT_TEST
                          02D8
                          02D8       STP10:
      0004'CF    0A   D0  02D8                       MOVL    #10,W^CURRENT_TC
                 00   DD  02DD                       PUSHL   #0
      0549'CF    01   FB  02DF                       CALLS   #1,W^REG_SAVE
0069'CF    012F'CF    DE  02E4             MOVAL   W^SM,W^MODE                 ; set the mode
                 05   BE  02EB             CHMS    #5                          ; cancel super exit handlers #1 and #3
```

```
                                02ED    445 ;+
                                02ED    446 ;
                                02ED    447 ; test user mode exit handler cancellation
                                02ED    448 ;
                                02ED    449 ;-
                                02ED    450         NEXT_TEST
                                02ED
                                02ED        STP11:
      0004'CF    0B   DO   02ED                       MOVL    #11,W^CURRENT_TC
                 00   DD   02F2                       PUSHL   #0
      0549'CF    01   FB   02F4                       CALLS   #1,W^REG_SAVE
0069'CF   0123'CF    DE   02F9    451         MOVAL   W^UM,W^MODE                 ; set the mode
                                0300    452         $CANEXH_S W^USER1               ; cancel user exit handler #1
                                030B    453         FAIL_CHECK SS$_NORMAL           ; check for success
                 01   DD   030B                       PUSHL   #SS$_NORMAL
      0553'CF    01   FB   030D                       CALLS   #1,W^REG_CHECK
0089'CF   016B'CF    DE   0312    454         MOVAL   W^USER3,W^CAN+CANEXH$_DESBLK ; set handler adr
                                0319    455         $CANEXH_G W^CAN                 ; cancel user exit handler #3
                                0322    456         FAIL_CHECK SS$_NORMAL           ; check for success
                 01   DD   0322                       PUSHL   #SS$_NORMAL
      0553'CF    01   FB   0324                       CALLS   #1,W^REG_CHECK
```

```
                                0329   458          .SBTTL  DCLEXH TESTS #2
                                0329   459 ;+
                                0329   460 ;
                                0329   461 ; $DCLEXH tests
                                0329   462 ;
                                0329   463 ; This is the se.ond of two parts of the DCLEXH tests.
                                0329   464 ; This part tests the servicing of the exit handlers.
                                0329   465 ; At this time there should be 2 user mode exit handlers declared.
                                0329   466 ;
                                0329   467 ; test user mode exit handler #4
                                0329   468 ;
                                0329   469 ;-
                                0329   470          NEXT_TEST
                                0329
                                0329         STP12:
     0004'CF    0C    DO        0329                 MOVL    #12,W^CURRENT_TC
                00    DD        032E                 PUSHL   #0
     0549'CF    01    FB        0330                 CALLS   #1,W^REG_SAVE
0103'CF  0038'CF    DE        0335   471          MOVAL   W^DCLEXH,W^SERV_NAME    ; set service name
     0065'CF    01    DO        033C   472          MOVL    S^#SS$_NORMAL,W^STATUS  ; set the expected status return
                              0341   473          $EXIT_S W^MOD_MSG_CODE          ; kick off ALL exit handlers
                              034C   474 USERH4:
                     0000     034C   475          .WORD   0
          52    04    9A     034E   476          MOVZBL  S^#4,R2                 ; set expected handler code
                11    11     0351   477          BRB     HNDLR_COM
                              0353   478 ;+
                              0353   479 ;
                              0353   480 ; test user exit handler #2
                              0353   481 ;
                              0353   482 ;-
                              0353   483 USERH2:
                     0000     0353   484          .WORD   0
                              0355   485          NEXT_TEST
                              0355
                              0355         STP13:
     0004'CF    0D    DO        0355                 MOVL    #13,W^CURRENT_TC
                00    DD        035A                 PUSHL   #0
     0549'CF    01    FB        035C                 CALLS   #1,W^REG_SAVE
          52    02    9A     0361   486          MOVZBL  S^#2,R2                 ; set expected handler code
                              0364   487 HNDLR_COM:
0065'CF   04 BC    D1        0364   488          CMPL    @B^4(AP),W^STATUS       ; is the status adr field OK?
                15    13     036A   489          BEQL    10$                     ; br if yes
             04 AC    DD     036C   490          PUSHL   4(AP)                   ; push received code
          0065'CF    DF     036F   491          PUSHAL  W^STATUS                ; push expected code
          014D'CF    DF     0373   492          PUSHAL  W^EXP                   ; push string variable
     0691'CF    03    FB     0377   493          CALLS   #3,W^PRINT_FAIL         ; print the error
     0793'CF    00    FB     037C   494          CALLS   #0,W^MODE_ID            ; identify the handler mode
                              0381   495 10$:
   08 AC    52    D1        0381   496          CMPL    R2,8(AP)                ; is the argument field OK?
                13    13     0385   497          BEQL    20$                     ; br if yes
             08 AC    DD     0387   498          PUSHL   8(AP)                   ; push received code
                52    DD     038A   499          PUSHL   R2                      ; push expected code
          014D'CF    DF     038C   500          PUSHAL  W^EXP                   ; push string variable
     0691'CF    03    FB     0390   501          CALLS   #3,W^PRINT_FAIL         ; print the error
     0793'CF    00    FB     0395   502          CALLS   #0,W^MODE_ID            ; identify the exit handler mode
                              039A   503 20$:
   08 AC    02    91        039A   504          CMPB    S^#2,8(AP)              ; is this the last handler?
```

```
                     01   13  039E    505          BEQL    30$                     ; br if yes
                          04  03A0    506          RET                             ; do the next handler
                              03A1    507 30$:
                     00   DD  03A1    508          PUSHL   #0                      ; push dummy parameter
          0549'CF    01   FB  03A3    509          CALLS   #1,W^REG_SAVE           ; save the registers
0069'CF   012F'CF    DE  03A8    510          MOVAL   W^SM,W^MODE             ; set the mode
                     03   BE  03AF    511          CHMS    #3                      ; reset the CHMS handler
0133'CF   0044'CF    D0  03B1    512          MOVL    W^MOD_MSG_CODE,W^MSGVEC1+4 ; set message code
013B'CF   0048'CF    D0  03B8    513          MOVL    W^TMN_ADDR,W^MSGVEC1+12 ; set up parameters
013F'CF   004C'CF    D0  03BF    514          MOVL    W^TMD_ADDR,W^MSGVEC1+16
                              03C6    515          $PUTMSG_S W^MSGVEC1             ; print the message
                1C   01   F0  03D7    516          INSV    #1,#STS$V_INHIB_MSG,-
 00000044'EF    01        03DA    517                  #1,MOD_MSG_CODE         ; set inhibit printing on the exit status
          50   0044'CF    D0  03E0    518          MOVL    W^MOD_MSG_CODE,R0       ; save the new code in R0
                          04  03E5    519          RET                             ; leave for good!
```

```
                        03E6    522 .SBTTL SUPER_MODE
                        03E6    523 ;++
                        03E6    524 ; FUNCTIONAL DESCRIPTION:
                        03E6    525 ;       Routine to handle the CHMS instructions.
                        03E6    526 ;
                        03E6    527 ; CALLING SEQUENCE:
                        03E6    528 ;       CHMS    #N
                        03E6    529 ;
                        03E6    530 ; INPUT PARAMETERS:
                        03E6    531 ;         SP=>  CHMS parameter
                        03E6    532 ;               PC
                        03E6    533 ;               PSL
                        03E6    534 ;
                        03E6    535 ;         The CHMS parameter can be one of the following:
                        03E6    536 ;
                        03E6    537 ;               1 = execute a $DCLCHM_G to declate a dummy handler
                        03E6    538 ;               2 = execute a $DCLCMH_G to clear the dummy CHMS handler
                        03E6    539 ;               3 = execute a $DCLCMH_S to reset the CHMS handler
                        03E6    540 ;               4 = declare 2 exit handlers in super mode
                        03E6    541 ;               5 = delete 2 exit handlers in super mode
                        03E6    542 ;
                        03E6    543 ; OUTPUT PARAMETERS:
                        03E6    544 ;       NONE
                        03E6    545 ;--
                        03E6    546
                        03E6    547 SUPER_MODE:
      50    8E    D0    03E6    548         MOVL    (SP)+,R0                ; get CHM parameter off the stack
05    01    50    8F    03E9    549         CASEB   R0,#1,#5                ; do the right thing
                        03ED    550 10$:
                000A'   03ED    551         .WORD   20$-10$
                001D'   03EF    552         .WORD   30$-10$
                0037'   03F1    553         .WORD   40$-10$
                0058'   03F3    554         .WORD   50$-10$
                0083'   03F5    555         .WORD   60$-10$
                        03F7    556 20$:
                        03F7    557         $DCLCMH_G W^DCL                 ; declare a dummy CHMS handler
                        0400    558         FAIL_CHECK SS$_NORMAL           ; check for success
          01    DD    0400                        PUSHL   #SS$_NORMAL
0553'CF   01    FB    0402                        CALLS   #1,W^REG_CHECK
              008F    31  0407    559         BRW     70$                     ; carry on
                        040A    560 30$:
0071'CF FFD8 CF   DE    040A    561         MOVAL   W^SUPER_MODE,W^DCL+DCLCMH$_ADDRES ; set up to delete dummy handler
                        0411    562         $DCLCMH_G W^DCL                 ; clear the dummy handler
                        041A    563         FAIL_CHECK SS$_NORMAL           ; check for success
          01    DD    041A                        PUSHL   #SS$_NORMAL
0553'CF   01    FB    041C                        CALLS   #1,W^REG_CHECK
              0075    31  0421    564         BRW     70$                     ; carry on
                        0424    565 40$:
0103'CF   0031'CF   DE  0424    566         MOVAL   W^DCLCMH,W^SERV_NAME    ; set service name pointer
                        042B    567         $DCLCMH_S @PRVHND1,,#0          ; reset the CHMS handler for DCL
                        043C    568         FAIL_CHECK SS$_NORMAL           ; check for success
          01    DD    043C                        PUSHL   #SS$_NORMAL
0553'CF   01    FB    043E                        CALLS   #1,W^REG_CHECK
              54    11  0443    569         BRB     70$                     ; carry on
                        0445    570 50$:
                        0445    571         $DCLEXH_S W^SUPER1              ; declare #1 super mode exit handler
                        0450    572         FAIL_CHECK SS$_NORMAL           ; check for success
```

```
                    01   DD  0450                        PUSHL   #SS$_NORMAL
          0553'CF   01   FB  0452                        CALLS   #1,W^REG_CHECK
0081'CF   01A7'CF   DE  0457   573          MOVAL   W^SUPER3,W^DCL1+DCLEXH$_DESBLK ; set handler adr for #3
                    045E   574          $DCLEXH_G W^DCL1                       ; declare #3 super mode exit handler
                    0467   575          FAIL_CHECK SS$_NORMAL                  ; check for success
                    01   DD  0467                        PUSHL   #SS$_NORMAL
          0553'CF   01   FB  0469                        CALLS   #1,W^REG_CHECK
                    29   11  046E   576          BRB     70$                    ; carry on
                    0470   577 60$:
                    0470   578          $CANEXH_S W^SUPER1                     ; delete #1 super mode exit handler
                    047B   579          FAIL_CHECK SS$_NORMAL                  ; check for success
                    01   DD  047B                        PUSHL   #SS$_NORMAL
          0553'CF   01   FB  047D                        CALLS   #1,W^REG_CHECK
0089'CF   01A7'CF   DE  0482   580          MOVAL W^SUPER3,W^CAN+CANEXH$_DESBLK ; set handler adr for #3
                    0489   581          $CANEXH_G W^CAN                        ; delete #3 super mode exit handler
                    0492   582          FAIL_CHECK SS$_NORMAL                  ; check for success
                    01   DD  0492                        PUSHL   #SS$_NORMAL
          0553'CF   01   FB  0494                        CALLS   #1,W^REG_CHECK
                    0499   583 70$:
                    02   0499   584          REI                                    ; go back to user mode
                    049A   585 DUMMY:
                FF49   31   049A   586          BRW     SUPER_MODE             ; dummy handler address
                    049D   587          .SBTTL USER_MODE
                    049D   588 ;++
                    049D   589 ; FUNCTIONAL DESCRIPTION:
                    049D   590 ;       Routine to handle the CHMU instruction
                    049D   591 ;
                    049D   592 ; CALLING SEQUENCE:
                    049D   593 ;       CHMU    #5
                    049D   594 ;
                    049D   595 ; INPUT PARAMETERS:
                    049D   596 ;       SP=>    #5
                    049D   597 ;               PC
                    049D   598 ;               PSL
                    049D   599 ;
                    049D   600 ; OUTPUT PARAMETERS:
                    049D   601 ;       NONE
                    049D   602 ;
                    049D   603 ;--
                    049D   604
                    049D   605 USER_MODE:
          50   8E   D0   049D   606          MOVL    (SP)+,R0               ; get CHM parameter off the stack
          05   50   D1   04A0   607          CMPL    R0,S^#5                ; is it correct?
               0D   13   04A3   608          BEQL    10$                    ; br if yes
               50   DD   04A5   609          PUSHL   R0                     ; save received
               05   DD   04A7   610          PUSHL   S^#5                   ; save expected
          C14D'CF   DF   04A9   611          PUSHAL  W^EXP                  ; save the string variable
0691'CF   03   FB   04AD   612          CALLS   #3,W^PRINT_FAIL        ; print the error message
                    04B2   613 10$:
                    02   04B2   614          REI                            ; return
```

```
                        04B3   616          .SBTTL  COMP_MODE
                        04B3   617 ;++
                        04B3   618 ; FUNCTIONAL DESCRIPTION:
                        04B3   619 ;       Compatibility mode exception handler
                        04B3   620 ;
                        04B3   621 ; CALLING SEQUENCE:
                        04B3   622 ;       execute a compatibility mode exception
                        04B3   623 ;
                        04B3   624 ; INPUT PARAMETERS:
                        04B3   625 ;       NONE
                        04B3   626 ;
                        04B3   627 ; OUTPUT PARAMETERS:
                        04B3   628 ;       NONE
                        04B3   629 ;
                        04B3   630 ;--
                        04B3   631 COMP_MODE:
                FC A0   95     04B3   632          TSTB    -4(R0)                  ; see if we got the correct exception
                   0E   13     04B6   633          BEQL    10$                     ; br if correct
                FC A0   DD     04B8   634          PUSHL   -4(R0)                  ; push received code
                   00   DD     04BB   635          PUSHL   #0                      ; push expected code
             014D'CF    DF     04BD   636          PUSHAL  W^EXP                   ; push string variable
        0691'CF    03   FB     04C1   637          CALLS   #3,W^PRINT_FAIL         ; print the error
                        04C6   638 10$:
    FFFF 8F    FC8C CF   B1     04C6   639          CMPW    W^TEST,#-1              ; were we really in compatibility mode?
                   14   13     04CD   640          BEQL    20$                     ; br if yes
        7E    FC83 CF   3C     04CF   641          MOVZWL  W^TEST,-(SP)            ; push received code
         0000FFFF 8F    DD     04D4   642          PUSHL   #^X0000FFFF             ; push expected code
             014D'CF    DF     04DA   643          PUSHAL  W^EXP                   ; push string variable
        0691'CF    03   FB     04DE   644          CALLS   #3,W^PRINT_FAIL         ; print the error
                        04E3   645 20$:
                FC72    31     04E3   646          BRW     RETURN                  ; carry on
```

```
                        .SBTTL SETUP_SUPER ROUTINE
                ;++
                ; FUNCTIONAL DESCRIPTION:
                ;       Routine to declare an initial CHMS handler from user mode.
                ;
                ; CALLING SEQUENCE:
                ;       $CMKRNL_S W^SETUP_SUPER,ARGLST
                ;
                ;               ARGLST = address of a pointer to a one parameter argument list conta
                ;                        the address of the entry mask of the CHMS handler
                ;
                ; INPUT PARAMETERS:
                ;       ARGLST
                ;
                ; IMPLICIT INPUTS
                ;       NONE
                ;
                ; OUTPUT PARAMETERS:
                ;       Declares a change mode handler for super mode which must be
                ;       reset to DCL in the users handler routine when the handler is
                ;       no longer needed.
                ;
                ; IMPLICIT OUTPUTS:
                ;       NONE
                ;
                ; COMPLETION CODES:
                ;       NONE
                ;
                ; SIDE EFFECTS:
                ;       NONE
                ;
                ; ON ENTRY:
                ;
                ;                                KSP =>  ---------          USP =>  ---------
                ;                                        |   0   |                  |       |
                ;                                        |   0   |                  |  USER |
                ;                                        |  AP   |                  |       |
                ;                                        |  FP   |                  |  CALL |
                ;                                        |  PC   |                  |       |
                ;                                        |   0   |                  | FRAME |
                ;                                        |   0   |                  |       |
                ;                                        |  AP   |                  ---------
                ;                                        |  FP   |
                ;                                        | SRVEXIT|
                ;                                        |  PC   |
                ;                                        |  PSL  |
                ;                                        ---------
                ;--
```

```
                          04E6  RETURN_PC:
                00000000  04E6          .LONG   0                           ; storage for user return PC
                          04EA  HANDLER_PC:
                00000000  04EA          .LONG   0                           ; storage for handler PC
                          04EE  ;
                          04EE  SETUP_SUPER:
                    000C  04EE          .WORD   ^M<R2,R3>
             53   03  DB  04F0          MFPR    #PR$_USP,R3                 ; get the user call frame address
       EE AF   10 A3  D0  04F3          MOVL    SF$L_SAVE_PC(R3),B^RETURN_PC ; get the user return PC
       ED AF   04 AC  D0  04F8          MOVL    4(AP),HANDLER_PC            ; save the handler address
          52   0C AD  D0  04FD          MOVL    SF$L_SAVE_FP(FP),R2         ; get saved FP
             52  00'  C0  0501          ADDL    S^#EXE$C_CMSTKSZ,R2         ; back over change mode stack frame
          62   12'AF  9E  0504          MOVAB   B^20$,(R2)                  ; set return address
                  0A  F0  0508          INSV    #<<PSL$C_SUPER@PSL$S_CURMOD>+PSL$C_SUPER>,-
                  16      050A                  #PSL$V_PRVMOD,-
          04 A2   04      050B                  #PSL$S_CURMOD*2,4(R2)       ; set current and previous mode to super
             50   01  D0  050E          MOVL    S^#SS$_NORMAL,R0            ; set correct return code
                      04  0511          RET                                 ; enter super mode
                          0512  20$:
                  7E  D4  0512          CLRL    -(SP)                       ; set up dummy PSL
          18'AF   6E  FA  0514          CALLG   (SP),B^30$                  ; create initial call frame
                          0518  30$:
                    0000  0518          .WORD   ^M<>                        ; entry mask
                  00  DD  051A          PUSHL   #0                          ; push a dummy parameter
        0549'CF   01  FB  051C          CALLS   #1,W^REG_SAVE               ; save the registers
   0069'CF   012F'CF  DE  0521          MOVAL   W^SM,W^MODE                 ; set the mode
                          0528          $DCLCMH_S @HANDLER_PC,W^PRVHND1,#0 ; set real handler
                          0538          FAIL_CHECKNP SS$_NORMAL             ; check for success
                  01  DD  0538                  PUSHL   #SS$_NORMAL
        05EA'CF   01  FB  053A                  CALLS   #1,W^REG_CHECKNP
      03C00000 8F     DD  053F          PUSHL   #<<PSL$C_USER@PSL$V_CURMOD>-
                          0545                  !<PSL$C_USER@PSL$V_PRVMOD>>; set return to user
             9E AF    DD  0545          PUSHL   RETURN_PC                   ; set the return PC
                      02  0548          REI                                 ; return to user mode
                          0549          .SBTTL REG_SAVE
                          0549  ;++
                          0549  ; FUNCTIONAL DESCRIPTION:
                          0549  ;       Subroutine to save R2-R11 in the register save location.
                          0549  ;
                          0549  ; CALLING SEQUENCE:
                          0549  ;       PUSHL   #0              ; save a dummy parameter
                          0549  ;       CALLS   #1,W^REG_SAVE   ; save R2-R11
                          0549  ;
                          0549  ; INPUT PARAMETERS:
                          0549  ;       NONE
                          0549  ;
                          0549  ; OUTPUT PARAMETERS:
                          0549  ;       NONE
                          0549  ;
                          0549  ;--
                          0549
                          0549  REG_SAVE:
                    0FFC  0549          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
  0008'CF   14 AD   28  28  054B        MOVC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; save the registers in the program
                      04  0552          RET
```

```
                                                                .SBTTL  REG_CHECK
                                                        ;++
                                                        ; FUNCTIONAL DESCRIPTION:
                                                        ;       Subroutine to test R0 & R2-R11 for proper content after a service
                                                        ;       execution. A snapshot is taken by the REG_SAVE routine at the
                                                        ;       beginning of each step and this routine is executed after the
                                                        ;       services have been executed.
                                                        ;
                                                        ; CALLING SEQUENCE:
                                                        ;       PUSHL   #SS$_XXXXXX     ; push expected R0 contents
                                                        ;       CALLS   #1,W^REG_CHECK  ; execute this routine
                                                        ;
                                                        ; INPUT PARAMETERS:
                                                        ;       expected R0 contents on the stack
                                                        ;
                                                        ; OUTPUT PARAMETERS:
                                                        ;       possible error messages printed using $PUTMSG
                                                        ;
                                                        ;--
                                                        REG_CHECK:
                                        0FFC    0553    .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                  50    04 AC   D1      CMPL    4(AP),R0                ; is this the right fail code?
                                        0E      13      BEQL    10$                     ; br if yes
                                        50      DD      PUSHL   R0                      ; push received data
                                     04 AC      DD      PUSHL   4(AP)                   ; push expected data
                                014D'CF         DF      PUSHAL  W^EXP                   ; push the string variable
                            0691'CF     03      FB      CALLS   #3,W^PRINT_FAIL         ; print the error message
                                                10$:
                0008'CF     14 AD       28      29      CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; check all but R0
                                        22      13      BEQL    20$                     ; br if O.K.
                53  00000008'CF        8F      C3      SUBL3   #REG_SAVE_AREA,R3,R6    ; calculate the register number
                                56      04      C6      DIVL2   #4,R6
                            7E  56      02      81      ADDB3   #^X2,R6,-(SP)           ; set number past R0-R1 and save
                                51      03      CA      BICL2   #3,R1                   ; backup to register boundrys
                                53      03      CA      BICL2   #3,R3
                                        61      DD      PUSHL   (R1)                    ; push received data
                                        63      DD      PUSHL   (R3)                    ; push expected data
                                008D'CF         DF      PUSHAL  W^REG                   ; set string pntr param.
                            0691'CF     04      FB      CALLS   #4,W^PRINT_FAIL         ; print the error message
                                                20$:
                                            04  0594    RET
```

```
                                 0595            .SBTTL REG_CHECKNP
                                 0595   ;++
                                 0595   ; FUNCTIONAL DESCRIPTION:
                                 0595   ;        Subroutine to test R0 & R2-R11 for proper content after a service
                                 0595   ;        execution without printing it. A snapshot is taken by the REG_SAVE routine a
                                 0595   ;        beginning of each step and this routine is executed after the
                                 0595   ;        services have been executed. This routine collects the error
                                 0595   ;        information in buffer ERLB instead of printing it.
                                 0595   ;
                                 0595   ; CALLING SEQUENCE:
                                 0595   ;        PUSHL   #SS$_XXXXXX      ; push expected R0 contents
                                 0595   ;        CALLS   #1,W^REG_CHECK   ; execute this routine
                                 0595   ;
                                 0595   ; INPUT PARAMETERS:
                                 0595   ;        expected R0 contents on the stack
                                 0595   ;
                                 0595   ; OUTPUT PARAMETERS:
                                 0595   ;        possible error messages logged in buffer ERLB which are printed
                                 0595   ;        using routine ERLBUF_DUMP.
                                 0595   ;
                                 0595   ;--
                                 0595
                                 0595  FLAG:
                           00    0595           .BYTE 0                    ; error flags are BIT0 = 0 means no errors in the bu
                                 0596                                      ;                 BIT0 = 1 means errors in the buffe
                                 0596  ELBP:
                    0000059A'    0596           .ADDRESS ERLB              ; error log buffer pointer
                                 059A  ERLB:
                    000005EA     059A           .BLKB   80                 ; error log buffer
                                 05EA  ;
                                 05EA  REG_CHECKNP:
                         0FFC    05EA           .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
         50    04 AC      D1     05EC           CMPL    4(AP),R0           ; is this the right fail code
                  2B      13     05F0           BEQL    10$                ; br if yes
         9F AF    01      88     05F2           BISB2   #1,FLAG            ; set the error logged flag bit
         52    9D AF      D0     05F6           MOVL    ELBP,R2            ; get the current error log pointer
            82    03      90     05FA           MOVB    #3,(R2)+           ; save the long word count
            82    50      D0     05FD           MOVL    R0,(R2)+           ; save received status
         82    04 AC      D0     0600           MOVL    4(AP),(R2)+        ; save expected status
       82    014D'CF      DE     0604           MOVAL   W^EXP,(R2)+        ; save the string variable
                  62      D4     0609           CLRL    (R2)               ; set the terminator
         87 AF    52      D0     060B           MOVL    R2,ELBP            ; reset the buffer pointer
   004C'CF    002A'CF     DE     060F           MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR ; set failure message address
0044'CF   03     00   02  F0     0616           INSV    #ERROR,#0,#3,W^MOD_MSG_CODE ; set severity code
                                 061D  10$:
 0008'CF    14 AD   28    29     061D           CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA ; check all but R0 and R1
                  3C      13     0624           BEQL    20$                ; br if OK
      FF6A CF     01      88     0626           BISB2   #1,FLAG            ; set error logged flag bit
      52   FF67 CF        D0     062B           MOVL    ELBP,R2            ; get current error log buf pointer
            82    04      90     0630           MOVB    S^#4,(R2)+         ; set longword count
       00000008'8F        C3     0633           SUBL3   #REG_SAVE_AREA,-
            56    53             0639           R3,R6                      ; calc reg number
            56    04      C6     063B           DIVL2   S^#4,R6            ; make it a longword count
      82    56    02      C1     063E           ADDL3   S^#2,R6,(R2)+      ; correct for R0-R1 and save
            82    61      D0     0642           MOVL    (R1),(R2)+         ; save received results
            82    63      D0     0645           MOVL    (R3),(R2)+         ; save expected results
       82    008D'CF      DE     0648           MOVAL   W^REG,(R2)+        ; save string variable
```

```
                          62   D4  064D   850          CLRL    (R2)                    ; set the terminator
                    FF42  CF   52   D0  064F   851          MOVL    R2,ELBP                 ; reset the buffer pointer
              004C'CF     002A'CF   DE  0654   852          MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR      ; set failure message address
        0044'CF   03   00   02      F0  0658   853          INSV    #ERROR,#0,#3,W^MOD_MSG_CODE     ; set severity code
                                        0662   854 20$:
                                    04  0662   855          RET                             ; bail out
```

```
                                0663   857          .SBTTL  ERLBUF_DUMP
                                0663   858 ;++
                                0663   859 ; FUNCTIONAL DESCRIPTION:
                                0663   860 ;       Routine to check for errors in the error log buffer and
                                0663   861 ;       report any that are there.
                                0663   862 ;
                                0663   863 ; CALLING SEQUENCE:
                                0663   864 ;       CALLS #0,W^ERLBUF_DUMP
                                0663   865 ;
                                0663   866 ; INPUT PARAMETERS:
                                0663   867 ;       FLAG bit 0 = 0 for no errors logged
                                0663   868 ;       FLAG bit 0 = 1 for errors logged
                                0663   869 ;       if errors logged then buffer ERLB must contain legal format errors
                                0663   870 ;
                                0663   871 ; OUTPUT PARAMETERS:
                                0663   872 ;       NONE
                                0663   873 ;
                                0663   874 ;--
                                0663   875
                                0663   876 ERLBUF_DUMP:
                         001C   0663   877          .WORD   ^M<R2,R3,R4>
      1B FF2C CF          E9    0665   878          BLBC    FLAG,30$        ; br if no errors to report
52    FF2C CF               DE 066A   879          MOVAL   ERLB,R2         ; set up buffer pointer
                                066F   880 10$:
                      62        D5 066F   881          TSTL    (R2)            ; any more errors?
                      12        13 0671   882          BEQL    30$             ; br if not
                   53 82        9A 0673   883          MOVZBL  (R2)+,R3        ; get the longword count
                   54 53        D0 0676   884          MOVL    R3,R4           ; and save it
                                0679   885 20$:
                      82        DD 0679   886          PUSHL   (R2)+           ; push a parameter
                         FB     53 F5 067B   887          SOBGTR  R3,20$          ; and push them all
                0691'CF          54 FB 067E   888          CALLS   R4,W^PRINT_FAIL ; print the failure
                                EA 11 0683   889          BRB     10$             ; do the next one
                                0685   890 30$:
FF0A CF    FF11 CF              DE 0685   891          MOVAL   ERLB,ELBP       ; reset the buffer pointer
           FF0A CF              D4 068C   892          CLRL    W^ERLB          ; set fresh terminater
                                04 0690   893          RET                     ; bail out
```

```
                                0691   895          .SBTTL  PRINT_FAIL
                                0691   896 ;++
                                0691   897 ; FUNCTIONAL DESCRIPTION:
                                0691   898 ;       Subroutine to report failures using $PUTMSG
                                0691   899 ;
                                0691   900 ; CALLING SEQUENCE:
                                0691   901 ; Mode  #1       PUSHL EXPECTED  Mode    #2      PUSHL REG_NUMBER
                                0691   902 ;                PUSHL RECEIVED                  PUSHL EXPECTED
                                0691   903 ;                PUSHAL STRING_VAR               PUSHL RECEIVED
                                0691   904 ;                CALLS #3,W^PRINT_FAIL           PUSHAL STRING_VAR
                                0691   905 ;                                                CALLS #4,W^PRINT_FAIL
                                0691   906 ; INPUT PARAMETERS:
                                0691   907 ;       listed above
                                0691   908 ;
                                0691   909 ; OUTPUT PARAMETERS:
                                0691   910 ;       an error message is printed using $PUTMSG
                                0691   911 ;
                                0691   912 ;--
                                0691   913
                                0691   914 PRINT_FAIL:
                           003C 0691   915          .WORD   ^M<R2,R3,R4,R5>
                                0693   916          $FAO_S  W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
                                06B4   917          $PUTMSG_S W^MSGVEC                      ; print the message
                     04  6C   91 06C5   918          CMPB    (AP),#4                         ; is this a register message?
                         21   13 06C8   919          BEQL    10$                             ; br if yes
                                06CA   920          $FAO_S  W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
                         25   11 06E9   921          BRB     20$                             ; goto output message
                                06EB   922 10$:
                                06EB   923          $FAO_S  W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
                                0710   924 20$:
                                0710   925
                                0710   926          $PUTMSG_S W^MSGVEC                      ; print the message
           0793'CF   00   FB    0721   927          CALLS   #0,W^MODE_ID                    ; identify the mode
      004C'CF   002A'CF   DE    0726   928          MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR      ; set failure message address
0044'CF   03   00   02   F0    072D   929          INSV    #ERROR,#0,#3,W^MOD_MSG_CODE     ; set severity code
                              04 0734   930          RET
                                0735   931 USERH1:
                                0735   932 USERH3:
                           0000 0735   933          .WORD   0
      0069'CF   0123'CF   DE    0737   934          MOVAL   W^UM,W^MODE             ; set the mode string
                        14   11 073E   935          BRB     CEP
                                0740   936 SUPERH1:
                                0740   937 SUPERH3:
                           0000 0740   938          .WORD   0
      0069'CF   012F'CF   DE    0742   939          MOVAL   W^SM,W^MODE             ; set the mode string
                        09   11 0749   940          BRB     CEP
                                074B   941 EXECH1:
                                074B   942 EXECH3:
                           0000 074B   943          .WORD   0
      0069'CF   013C'CF   DE    074D   944          MOVAL   W^EM,W^MODE             ; set the mode string
                                0754   945 CEP:
                                0754   946          $FAO_S  W^CS4,W^MESSAGEL,W^MSGL,MODE,#TEST_MOD_NAME ; format the error strin
                                0773   947          $PUTMSG_S W^MSGVEC                      ; print the message
      004C'CF   002A'CF   DE    0784   948          MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR ; set failure message address
0044'CF   03   00   02   F0    078B   949          INSV    #ERROR,#0,#3,W^MOD_MSG_CODE ; set severity code
                              04 0792   950          RET
```

```
                0793   952          .SBTTL  MODE_ID
                0793   953 ;+
                0793   954 ; FUNCTIONAL DESCRIPTION:
                0793   955 ;       Subroutine to identify the mode that an exit handler is in.
                0793   956 ;
                0793   957 ; CALLING SEQUENCE:
                0793   958 ;       CALLS   #0,W^MODE_ID
                0793   959 ;
                0793   960 ; INPUT PARAMETERS:
                0793   961 ;       MODE contains an address pointing to an ascii string desc.
                0793   962 ;       of the current CPU mode.
                0793   963 ;
                0793   964 ; OUTPUT PARAMETERS:
                0793   965 ;       NONE
                0793   966 ;
                0793   967 ;--
                0793   968
                0793   969 MODE_ID:
        003C    0793   970          .WORD   ^M<R2,R3,R4,R5>
                0795   971          $FAO_S  W^CS5,W^MESSAGEL,W^MSGL,MODE ; format the error message
                07AE   972          $PUTMSG_S W^MSGVEC                   ; print the mode message
          04    07BF   973          RET
```

```
                        07C0   975 MOD_MSG_PRINT:
                        07C0   976 ;
                        07C0   977 ;  ************************************************************************
                        07C0   978 ;  *                                                                      *
                        07C0   979 ;  *  PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES              *
                        07C0   980 ;  *       (USING THE PUTMSG MACRO).                                      *
                        07C0   981 ;  *                                                                      *
                        07C0   982 ;  ************************************************************************
                        07C0   983 ;
                        07C0   984          PUTMSG  <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> ; PRINT MSG
                     05 07DB   985          RSB                                         ; ... AND RETURN TO CALLER
                        07DC   986 ;
                        07DC   987 CHMRTN:
                        07DC   988 ;  ************************************************************************
                        07DC   989 ;  *                                                                      *
                        07DC   990 ;  *     CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER           *
                        07DC   991 ;  *     A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED              *
                        07DC   992 ;  *     BY THE MODE MACRO ('TO' OPTION).  IT MERELY DOES                 *
                        07DC   993 ;  *     A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS                *
                        07DC   994 ;  *     THE EFFECT OF RETURNING TO THE END OF THE MODE                   *
                        07DC   995 ;  *     MACRO EXPANSION.                                                 *
                        07DC   996 ;  *                                                                      *
                        07DC   997 ;  ************************************************************************
                        07DC   998 ;
                   0000 07DC   999          .WORD   0                                   ; ENTRY MASK
00000059'FF          17 07DE  1000          JMP     @CHM_CONT                           ; RETURN TO MODE MACRO IN NEW MODE
                        07E4  1001 ;
                        07E4  1002 ;  *     RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
                        07E4  1003 ;
                        07E4  1004          .END    SATSSS43
```

$$ARGS	=	00000001			PRIVMASK		00000051	R	03		
$$T1	=	00000004			PRVHND1		00000107	R	03		
$$T2	=	00000004			PRVHND2		0000010B	R	03		
ARGLST		00000117	R	03	PRVHND3		0000010F	R	03		
BUF		000000AB	R	03	PRVPRT		00000050	R	03		
CAN		00000085	R	03	PSL$C_SUPER	=	00000002				
CANEXH		0000003F	R	02	PSL$C_USER	=	00000003				
CANEXH$_DESBLK	=	00000004			PSL$M_CM	=	80000000				
CANEXH$_NARGS	=	00000001			PSL$S_CURMOD	=	00000002				
CEP		00000754	R	04	PSL$V_CURMOD	=	00000018				
CHMRTN		000007DC	R	04	PSL$V_PRVMOD	=	00000016				
CHM_CONT		00000059	R	03	REG		0000008D	R	03		
COMP_MODE		000004B3	R	04	REGNUM		0000009F	R	03		
CS1		00000046	R	02	REG_CHECK		00000553	R	04		
CS2		00000078	R	02	REG_CHECKNP		000005EA	R	04		
CS3		000000A5	R	02	REG_SAVE		00000549	R	04		
CS4		000000D8	R	02	REG_SAVE_AREA		00000008	R	03		
CS5		0000010E	R	02	RETADR		0000005D	R	03		
CURRENT_TC		00000004	R	03	RETURN		00000158	R	04		
DCL		0000006D	R	03	RETURN_PC		000004E6	R	04		
DCL1		0000007D	R	03	SATSSS43		00000000	RG	04		
DCLCMH		00000031	R	02	SERV_NAME		00000103	R	03		
DCLCMH$_ADDRES	=	00000004			SETUP_SUPER		000004EE	R	04		
DCLCMH$_NARGS	=	00000003			SEVERE	=	00000004				
DCLCMH$_PRVHND	=	00000008			SF$L_SAVE_FP	=	0000000C				
DCLCMH$_TYPE	=	0000000C			SF$L_SAVE_PC	=	00000010				
DCLEXH		00000038	R	02	SHR$R_SHRDEF	=	00000001				
DCLEXH$_DESBLK	=	00000004			SHR$_TEXT	=	00001130				
DCLEXH$_NARGS	=	00000001			SM		0000012F	R	02		
DUMMY		0000049A	R	04	SS$_NORMAL	=	00000001				
ELBP		00000596	R	04	STATUS		00000065	R	03		
EM		0000013C	R	02	STEP	=	0000000D				
ERLB		0000059A	R	04	STP0		0000003D	R	04		
ERLBUF_DUMP		00000663	R	04	STP1		00000078	R	04		
ERROR	=	00000002			STP10		000002D8	R	04		
EXE$C_CMSTKSZ		********		X 04	STP11		000002ED	R	04		
EXEC1		000001BB	R	03	STP12		00000329	R	04		
EXEC3		000001CF	R	03	STP13		00000355	R	04		
EXECH1		0000074B	R	04	STP2		000000DD	R	04		
EXECH3		0000074B	R	04	STP3		00000138	R	04		
EXP		0000014D	R	02	STP4		00000158	R	04		
FLAG		00000595	R	04	STP5		00000166	R	04		
HANDLER_PC		000004EA	R	04	STP6		000001A5	R	04		
HNDLR_COM		00000364	R	04	STP7		000001FE	R	04		
INFO	=	00000003			STP8		00000213	R	04		
LIB$SIGNAL		********		X 04	STP9		00000278	R	04		
MESSAGEL		000000FB	R	03	STS$V_INHIB_MSG	=	0000001C				
MODE		00000069	R	03	SUCCESS	=	00000001				
MODE_ID		00000793	R	04	SUPER1		00000193	R	03		
MOD_MSG_CODE		00000044	R	03	SUPER3		000001A7	R	03		
MOD_MSG_PRINT		000007C0	R	04	SUPERH1		00000740	R	04		
MSGL		000000A3	R	03	SUPERH3		00000740	R	04		
MSGVEC		0000011F	R	03	SUPER_MODE		000003E6	R	04		
MSGVEC1		0000012F	R	03	SYS$CANEXH		********		GX 04		
OHC		00000113	R	03	SYS$CMEXEC		********		GX 04		
PR$_USP	=	00000003			SYS$CMKRNL		********		GX 04		
PRINT_FAIL		00000691	R	04	SYS$DCLCMH		********		GX 04		

```
SYS$DCLEXH                      ********  GX    04
SYS$EXIT                        ********  GX    04
SYS$FAO                         ********   X    04
SYS$HIBER                       ********  GX    04
SYS$PUTMSG                      ********  GX    04
SYS$SETPRN                      ********  GX    04
SYS$WAKE                        ********  GX    04
TEST                            00000156 R      04
TEST_MOD_BEGIN                  00000019 R      02
TEST_MOD_FAIL                   0000002A R      02
TEST_MOD_NAME                   00000000 R      02
TEST_MOD_NAME_D                 00000009 R      02
TEST_MOD_SUCC                   0000001F R      02
TMD_ADDR                        0000004C R      03
TMN_ADDR                        00000048 R      03
TPID                            00000000 R      03
UETP$_SATSMS                  = 007480D9
UETP$_TEXT                    = 00741133
UM                              00000123 R      02
USER1                           00000143 R      03
USER2                           00000157 R      03
USER3                           0000016B R      03
USER4                           0000017F R      03
USERH1                          00000735 R      04
USERH2                          00000353 R      04
USERH3                          00000735 R      04
USERH4                          0000034C R      04
USER_MODE                       0000049D R      04
WARNING                       = 00000000
```

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
RODATA	0000015B (347.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
RWDATA	000001E3 (483.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
SATSSS43	000007E4 (2020.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	37	00:00:00.07	00:00:00.30
Command processing	133	00:00:00.66	00:00:02.62
Pass 1	411	00:00:14.67	00:00:25.19
Symbol table sort	0	00:00:01.91	00:00:02.62
Pass 2	206	00:00:03.72	00:00:08.03
Symbol table output	18	00:00:00.14	00:00:00.34
Psect synopsis output	2	00:00:00.02	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	809	00:00:21.19	00:00:39.13

The working set limit was 1800 pages.
88714 bytes (174 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1170 non-local and 32 local symbols.
1004 source lines were read in Pass 1, producing 28 object records in Pass 2.
52 pages of virtual memory were used to define 48 macros.

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

Macro library name	Macros defined
_255DUA28:[SHRLIB]UETP.MLB;1	10
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	32
TOTALS (all libraries)	44

1367 GETS were required to define 44 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:SATSSS43/OBJ=OBJ$:SATSSS43 MSRC$:SATSSS43/UPDATE=(ENH$:SATSSS43)+EXECML$/LIB+SHRLIB$:UETP/LIB
```

SATSS552
LIS

SATSS542
LIS

SATSS540
LIS

SATSS550
LIS

SATSS544
LIS

SATSS546
LIS

SATSS543
LIS

SATSS545
LIS

SATSS541
LIS

SATSS547
LIS