F 13

```
UUU         UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSSS  YYY         YYY
UUU         UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSSS  YYY         YYY
UUU         UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP        SSSSSSSSSSSS  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
UUU         UUU  EEE                    TTT        PPP                          SSS        YYY
UUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP              SSSSSSSSSSSS           YYY
UUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP              SSSSSSSSSSSS           YYY
UUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE        TTT        PPP              SSSSSSSSSSSS           YYY
```

FILEID**SATSSS07

K 10

```
  SSSSSSSS    AAAAAA    TTTTTTTTTT    SSSSSSSS    SSSSSSSS    SSSSSSSS    000000    77777777
  SSSSSSSS    AAAAAA    TTTTTTTTTT    SSSSSSSS    SSSSSSSS    SSSSSSSS    000000    77777777
SS          AA      AA      TT      SS          SS          SS          00      00          77
SS          AA      AA      TT      SS          SS          SS          00      00          77
SS          AA      AA      TT      SS          SS          SS          00    0000          77
SS          AA      AA      TT      SS          SS          SS          00    0000          77
  SSSSSS    AA      AA      TT        SSSSSS      SSSSSS      SSSSSS    00  00  00        77
  SSSSSS    AA      AA      TT        SSSSSS      SSSSSS      SSSSSS    00  00  00        77
        SS  AAAAAAAAAA      TT              SS          SS          SS  0000    00      77
        SS  AAAAAAAAAA      TT              SS          SS          SS  0000    00      77
        SS  AA      AA      TT              SS          SS          SS  00      00    77
        SS  AA      AA      TT              SS          SS          SS  00      00    77          ....
SSSSSSSS    AA      AA      TT      SSSSSSSS    SSSSSSSS    SSSSSSSS      000000    77            ....
SSSSSSSS    AA      AA      TT      SSSSSSSS    SSSSSSSS    SSSSSSSS      000000    77            ....
                                                                                                  ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000            .TITLE  SATSSS07 - SATS SYSTEM SERVICE TESTS  (SUCC S.C.)
0000            .IDENT  'V04-000'
0000            .DEFAULT DISPLACEMENT,WORD
0000            .ENABLE SUPPRESSION
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000    ;
0000    ;++
0000    ; FACILITY:     SATS SYSTEM SERVICE TESTS
0000    ;
0000    ; ABSTRACT:     The SATSSS07 module tests the execution of the following
0000    ;               VMS system services:
0000    ;
0000    ;               $CREMBX
0000    ;               $DELMBX
0000    ;
0000    ; ENVIRONMENT:  User mode image.
0000    ;               Needs TMPMBX and PRMMBX privilege.
0000    ;
0000    ; AUTHOR: Paula Tirak,                  CREATION DATE: SEPTEMBER, 1979
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-005 LDJ0002         Larry D. Jones,         08-May-1984
0000    ;               Fixed bug in V03-004.
0000    ;
0000    ;       V03-004 LDJ0001         Larry D. Jones,         23-Aug-1983
0000    ;               Converted to new logical name length limit of 255.
0000    ;
0000    ;       V03-003 KDM0002         Kathleen D. Morse       28-Jun-1982
0000    ;               Added $IODEF, $PRDEF, $PSLDEF, and $SSDEF.
0000    ;
0000    ;**
0000    ;--
```

```
                    0000           .SBTTL  DECLARATIONS
                    0000   ;
                    0000   ; MACRO LIBRARY CALLS
                    0000   ;
                    0000           .LIBRARY /SYS$LIBRARY:STARLET.MLB/
                    0000           $DIBDEF                          ; device information block definitions
                    0000           $DSCDEF                          ; string descriptor definitions
                    0000           $IODEF                           ; i/o function code definitions
                    0000           $PRDEF                           ; processor register definitions
                    0000           $PRVDEF                          ; privilege name definitions
                    0000           $PSLDEF                          ; program status longword definitions
                    0000           $SFDEF                           ; stack frame definitions
                    0000           $SHR_MESSAGES UETP,116,<<TEXT,INFO>> ; UETP$_TEXT definition
                    0000           $SSDEF                           ; system status code definitions
                    0000           $STSDEF                          ; STS definitions
                    0000           $UETPDEF                         ; UETP message definitions
                    0000   ;
                    0000   ; Equated symbols
                    0000   ;
         00000000   0000   WARNING        = 0                       ; warning severity value for msgs
         00000001   0000   SUCCESS        = 1                       ; success     "      "    "   "
         00000002   0000   ERROR          = 2                       ; error       "      "    "   "
         00000003   0000   INFO           = 3                       ; information "      "    "   "
         00000004   0000   SEVERE         = 4                       ; fatal       "      "    "   "
         00000400   0000   MBBUF          = 1024                    ; MBX max. size for messages
         00002000   0000   HIMSG          = 8192
         00000001   0000   LOMSG          = 1
         00000100   0000   BUF_SIZ_S      = 256                     ; size for _S form
         00000400   0000   BUF_SIZ_G      =1024                     ; size for _G form
         00000001   0000   ENABLE         = 1                       ; for SETPRV calls
         00000000   0000   DISABLE        = 0                       ;  "     "     "
         00000100   0000   TEXT_BUFFER    = 256                     ; misc. message manipulations
                    0000   ;
                    0000   ;
                    0000   ; MACROS
                    0000   ;
```

```
                                00000000          .PSECT  RODATA,RD,NOWRT,NOEXE,LONG
                                    0000    ;
                                    0000    ARGLST:
                          00000001  0000          .LONG   1
                          00000BAC' 0004          .ADDRESS SUPER_MODE
                                    0008    CREMBX:
              58 42 4D 45 52 43 00' 0008          .ASCIC  /CREMBX/
                                 06 0008
                                    000F    CS1:                                ; failure messages
21 20 74 73 65 54 00000017'010E0000' 000F          .ASCID  \Test !AC service name !AC step !UL failed.\
6E 20 65 63 69 76 72 65 73 20 43 41 001D
70 65 74 73 20 43 41 21 20 65 6D 61 0029
2E 64 65 6C 69 61 66 20 4C 55 21 20 0035
                                    0041    CS2:
74 63 65 70 78 45 00000049'010E0000' 0041          .ASCID  \Expected !AS = !XL received !AS = !XL\
4C 58 21 20 3D 20 53 41 21 20 64 65 004F
41 21 20 64 65 76 69 65 63 65 72 20 005B
            4C 58 21 20 3D 20 53    0067
                                    006E    CS3:
74 63 65 70 78 45 00000076'010E0000' 006E          .ASCID  \Expected !AS!UB = !XL received !AS!UB = !XL\
20 3D 20 42 55 21 53 41 21 20 64 65 007C
64 65 76 69 65 63 65 72 20 4C 58 21 0088
58 21 20 3D 20 42 55 21 53 41 21 20 0094
                                 4C 00A0
                                    00A1    CS4:
72 69 75 71 65 52 000000A9'010E0000' 00A1          .ASCID  \Required channel not received.\
6E 20 6C 65 6E 6E 61 68 63 20 64 65 00AF
2E 64 65 76 69 65 63 65 72 20 74 6F 00BB
                                    00C7    CS5:
77 20 65 64 6F 4D 000000CF'010E0000' 00C7          .ASCID  \Mode was !AS.\
            2E 53 41 21 20 73 61    00D5
                                    00DC    NSSSF:
75 73 2D 6E 6F 4E 000000E4'010E0000' 00DC          .ASCID  \Non-subject system service failure of : !/!_      !AS\
6D 65 74 73 79 73 20 74 63 65 6A 62 00EA
69 61 66 20 65 63 69 76 72 65 73 20 00F6
2F 21 20 3A 20 66 6F 20 65 72 75 6C 0102
   53 41 21 20 20 20 20 20 20 5F 21 010E
                                    0119    DCLCMH:
              48 4D 43 4C 43 44 00' 0119          .ASCIC  /DCLCMH/
                                 06 0119
                                    0120    DELMBX:
              58 42 4D 4C 45 44 00' 0120          .ASCIC  /DELMBX/
                                 06 0120
                                    0127    EM:
54 55 43 45 58 45 0000012F'010E0000' 0127          .ASCID  /EXECUTIVE/
                           45 56 49 0135
                                    0138    EXP:
73 75 74 61 74 73 00000140'010E0000' 0138          .ASCID  \status\
                                    0146    KM:
4C 45 4E 52 45 4B 0000014E'010E0000' 0146          .ASCID  /KERNEL/
                                    0154    LOGNAMG:
37 30 53 5F 42 4D 0000015C'010E0000' 0154          .ASCID  /MB_S07_G/
                              47 5F 0162
                                    0164    LOGNAMG1:
37 30 53 5F 42 4D 0000016C'010E0000' 0164          .ASCID  /MB_S07_PG/
                           47 50 5F 0172
                                    0175    LOGNAMS:
```

```
37 30 53 5F 42 4D 0000017D'010E0000' 0175   129          .ASCID  /MB_S07_S/
                              53 5F  0183
                                     0185   130 LOGNAMS1:
37 30 53 5F 42 4D 0000018D'010E0000' 0185   131          .ASCID  /MB_S07_PS/
                           53 50 5F  0193
                                     0196   132 SYS_DEV:                                           ; system disk descriptor
59 53 24 53 59 53 0000019E'010E0000' 0196   133          .ASCID  /SYS$SYSDEVICE/
         45 43 49 56 45 44 53        01A4
                                     01AB   134 TMP_PRV_MASK:                                      ; create temp mailbox priv
                           00008000  01AB   135          .LONG   <1@PRV$V_TMPMBX>
                           00000000  01AF   136          .LONG   0
                                     01B3   137 PRM_PRV_MASK:                                      ; create perm mailbox priv
                           00000800  01B3   138          .LONG   <1@PRV$V_PRMMBX>
                           00000000  01B7   139          .LONG   0
                                     01BB   140 TMP_PRM_PRV_MASK:                                  ; create perm & temp priv
                           00008800  01BB   141          .LONG   <1@PRV$V_PRMMBX>!<1@PRV$V_TMPMBX>
                           00000000  01BF   142          .LONG   0
                                     01C3   143 GRP_PRV_MASK:                                      ; GRPNAM privilege mask
                           00000008  01C3   144          .LONG   <1@PRV$V_GRPNAM>
                           00000000  01C7   145          .LONG   0
                                     01CB   146 CHANNEL_ZERO:                                      ; MBX channel = 0
                               0000  01CB   147          .WORD   0
                                     01CD   148 LEN_00_DESCR:
                               0000  01CD   149          .WORD   0                                  ; zero length descriptor
                                 0E  01CF   150          .BYTE   DSC$K_DTYPE_T
                                 01  01D0   151          .BYTE   DSC$K_CLASS_S
                          00000185'  01D1   152          .ADDRESS LOGNAMS1
                                     01D5   153 LEN_256_DESCR:
                               0100  01D5   154          .WORD   256                                ; 60 char length descriptor
                                 0E  01D7   155          .BYTE   DSC$K_DTYPE_T
                                 01  01D8   156          .BYTE   DSC$K_CLASS_S
                          000001E5'  01D9   157          .ADDRESS LEN_256_NAME
                                     01DD   158 LEN_63_DESCR:
                               003B  01DD   159          .WORD   59                                 ; 59 char length descriptor
                                 0E  01DF   160          .BYTE   DSC$K_DTYPE_T
                                 01  01E0   161          .BYTE   DSC$K_CLASS_S
                          000001E5'  01E1   162          .ADDRESS LEN_256_NAME
                                     01E5   163 LEN_256_NAME:
4F 4C 5F 41 5F 53 49 5F 53 49 48 54  01E5   164          .ASCII  /THIS_IS_A_LONG_MAILBOX_NAME_WITH/
4E 5F 58 4F 42 4C 49 41 4D 5F 47 4E  01F1
            48 54 49 57 5F 45 4D 41  01FD
5F 36 35 32 30 52 45 42 4D 55 4E 5F  0205   165          .ASCII  /_NUMBER0256_CHARACTERS_IN_IT/  ; CREMBX prefixes MBX$
49 5F 53 52 45 54 43 41 52 41 48 43  0211
                        54 49 5F 4E  021D
41'41'41'41'41'41'41'41'41'41'41'41' 0221   166          .BYTE   ^A/A/[196]
41'41'41'41'41'41'41'41'41'41'41'41' 022D
41'41'41'41'41'41'41'41'41'41'41'41' 0239
41'41'41'41'41'41'41'41'41'41'41'41' 0245
41'41'41'41'41'41'41'41'41'41'41'41' 0251
41'41'41'41'41'41'41'41'41'41'41'41' 025D
41'41'41'41'41'41'41'41'41'41'41'41' 0269
41'41'41'41'41'41'41'41'41'41'41'41' 0275
41'41'41'41'41'41'41'41'41'41'41'41' 0281
41'41'41'41'41'41'41'41'41'41'41'41' 028D
41'41'41'41'41'41'41'41'41'41'41'41' 0299
41'41'41'41'41'41'41'41'41'41'41'41' 02A5
41'41'41'41'41'41'41'41'41'41'41'41' 02B1
```

```
41'41'41'41'41'41'41'41'41'41'41'41'  02BD
41'41'41'41'41'41'41'41'41'41'41'41'  02C9
41'41'41'41'41'41'41'41'41'41'41'41'  02D5
                         41'41'41'41'  02E1
                                       02E5  ZERO_ADDR_DESCR:                                  ; address 0 for ACCVIO err
                                0009   02E5          .WORD    9
                                  0E   02E7          .BYTE    DSC$K_DTYPE_T
                                  01   02E8          .BYTE    DSC$K_CLASS_S
                            00000000   02E9          .LONG    0
                                       02ED  D_LOGIC_NAME:
                                0005   02ED          .WORD    5
                                  0E   02EF          .BYTE    DSC$K_DTYPE_T
                                  01   02F0          .BYTE    DSC$K_CLASS_S
                           00000004'   02F1          .ADDRESS LOGIC_NAME
                                       02F5  D_EQUIV_NAME:
                                0005   02F5          .WORD    5
                                  0E   02F7          .BYTE    DSC$K_DTYPE_T
                                  01   02F8          .BYTE    DSC$K_CLASS_S
                           00000009'   02F9          .ADDRESS EQUIV_NAME
                                       02FD  D_MBX_LOGIC_NAME:
                                0009   02FD          .WORD    9                          ; for prefixing 'MBX$'
                                  0E   02FF          .BYTE    DSC$K_DTYPE_T
                                  01   0300          .BYTE    DSC$K_CLASS_S
                           00000000'   0301          .ADDRESS MBX_LOGIC_NAME
                                       0305  MBA:
        41 42 4D 0000030D'010E0000'    0305          .ASCID   \MBA\
                                       0310  MSG1:
65 74 61 65 72 43 00000318'010E0000'   0310          .ASCID   /Created mailbox was not permanent./
61 77 20 78 6F 62 6C 69 61 6D 20 64    031E
6E 61 6D 72 65 70 20 74 6F 6E 20 73    032A
                         2E 74 6E 65   0336
                                       033A  MSGVEC:
                            00000003   033A          .LONG    3                          ; PUTMSG message vector
                            00741133   033E          .LONG    UETP$_TEXT
                            00000001   0342          .LONG    1
                           00002580'   0346          .ADDRESS MESSAGEL
                                       034A  PROT:
4B 53 4D 4F 52 50 00000352'010E0000'   034A          .ASCID   /PROMSK/                   ; protection mask for $CREMBX
                                       0358  SM:
   52 45 50 55 53 00000360'010E0000'   0358          .ASCID   /SUPER/
                                       0365  TEST_DATA:
                            00000001   0365          A=1
                                       0365          .REPT    BUF_SIZ_G/2
                                       0365          .WORD    A
                                       0365          A=A+1
                                0001   0365          .ENDR
                                       0765  TEST_MOD_BEGIN:                            ; start end and fail messages
                6E 75 67 65 62 00'     0765          .ASCIC   /begun/
                                  05   0765
                                       076B  TEST_MOD_FAIL:
             64 65 6C 69 61 66 00'     076B          .ASCIC   /failed/
                                  06   076B
                                       0772  TEST_MOD_NAME:
       37 30 53 53 53 54 41 53 00'     0772          .ASCIC   /SATSSS07/                 ; needed for SATSMS message
                                  08   0772
                                       077B  TEST_MOD_NAME_D:
53 53 53 54 41 53 00000783'010E0000'   077B          .ASCID /SATSSS07/                   ; module name
```

```
                          37 30  0789
                                 078B    214 TEST_MOD_SUCC:
  6C 75 66 73 73 65 63 63 75 73 00' 078B    215          .ASCIC  /successful/
                             0A  078B
                                 0796    216 UM:                                          ; mode messages
    72 65 73 75 0000079E'010E0000' 0796    217          .ASCID  \user\
```

```
                     07A2   ;
                     07A2           .SBTTL  R/W PSECT
                 00000000           .PSECT  RWDATA,RD,WRT,NOEXE,LONG
                     0000   ;
                     0000   ;       ***********************************************************
                     0000   ;       **                                                       **
                     0000   ;       ** The following 2 declaratives must be CONTIGUOUS !!    **
                     0000   ;       **                                                       **
                     0000   ;       ***********************************************************
                     0000   MBX_LOGIC_NAME:
          24 58 42 4D 0000          .ASCII  /MBX$/                          ; for $CREMBX call
                     0004   LOGIC_NAME:                                     ; for $CRELOG sys service
       42 45 4D 41 4E 0004          .ASCII  /NAMEB/
                     0009   EQUIV_NAME:
       41 45 4D 41 4E 0009          .ASCII  /NAMEA/
                     000E   CREATED_FLAG:                                   ; signalling successful mailbox creation
                  00 000E           .BYTE                                   ; for delete service
                     000F   ;
                     000F   ARGLST1:                                        ; argument list for BUF_CHECK
           00000305' 000F           .ADDRESS MBA
            00000023 0013           .BLKL   4
                     0023   BUF:
            00000073 0023           .BLKB   80
                     0073   BUFFER:
            00001FB3 0073           .BLKB   8000
                     1FB3   CHM_CONT:
            00000000 1FB3           .LONG   0                               ; change mode continue address
                     1FB7   CRE:
                     1FB7           $CREMBX 0,MBCHANG,0,0,0,0,0             ; CREMBX paramter list
                     1FD7   CTRSTR:
            00000084 1FD7           .LONG   132                             ; same as above
           00001FDF' 1FDB           .ADDRESS .+4
            00002063 1FDF           .BLKB   132
                     2063   CURRENT_TC:
            00000000 2063           .LONG   0                               ; ptr to current test case
                     2067           .ALIGN LONG                             ; put it on a long word boundry
                     2068   DEL:
                     2068           $DELMBX MBCHANG                         ; DELMBX parameter list
                     2070   GETBUF:
            00000084 2070           .LONG   132                             ; same as above
           00002078' 2074           .ADDRESS .+4
            000020FC 2078           .BLKB   132
                     20FC   MBCHAN:
                0000 20FC           .WORD   0                               ; channel for the $GETCHN
                     20FE   MBCHANG:                                        ; _G mailbox channel
                0000 20FE           .WORD   0
                     2100   MBCHANS:                                        ; _S mailbox channel
                0000 2100           .WORD   0
                     2102   MBCHAND:                                        ; _D mailbox channel for deletions
                0000 2102           .WORD   0
                     2104   MBCHAR:
            00000074 2104           .LONG   DIB$K_LENGTH                    ; length of PROMSK for $GETCHN
           0000210C' 2108           .ADDRESS .+4
                     210C   MBCHR:
            00002180 210C           .BLKB   DIB$K_LENGTH                    ; buffer for PROMSK in $GETCHN
                     2180   MBX_BUF:
            00002580 2180           .BLKB   BUF_SIZ_G                       ; buffer for QIO reads
```

```
                              2580  MESSAGEL:
                   00000000   2580               .LONG    0                        ; message desc.
                   00000023'  2584               .ADDRESS BUF
                   00002078'  2588               .ADDRESS GETBUF+8
                              258C  ML:
                   00000000   258C               .LONG    0                        ; desc. for BUF_CHECK routine
                   00002078'  2590               .ADDRESS GETBUF+8
                              2594  MOD_MSG_CODE:
                   007480D9   2594               .LONG    UETP$_SATSMS             ; test module message code for putmsg
                              2598  MODE:
                   00000000   2598               .LONG    0                        ; current mode string pointer
                              259C  MSGL:
                   00000050   259C               .LONG    80                       ; buffer desc.
                   00000023'  25A0               .ADDRESS BUF
                              25A4  MSGVEC1:                                       ; PUTMSG message vector
                   00000003   25A4               .LONG    3
                   00741133   25A8               .LONG    UETP$_TEXT
                   00000001   25AC               .LONG    1
                   00000000   25B0               .LONG    0
                              25B4  PB:
                   00000074   25B4               .LONG    DIB$K_LENGTH
                   000025BC'  25B8               .ADDRESS .+4
                   00002630   25BC               .BLKB    DIB$K_LENGTH
                              2630  PRIVMASK:
          00000000 00000000   2630               .QUAD    0                        ; priv. mask
                              2638  PRVHND1:
                   00000000   2638               .LONG    0                        ; previous handler address 1
                              263C  PRVPRT:
                         00   263C               .BYTE    0                        ; protection return byte for SETPRT
                              263D  REG:
74 73 69 67 65 72 00002645'010E0000'  263D       .ASCID   \register R\
                52 20 72 65   264B
                              264F  REG_SAVE_AREA:
                   0000268B   264F               .BLKL    15                       ; register save area
                              268B  REGNUM:
                   00000000   268B               .LONG    0                        ; register number
                              268F  RETADR:
                   00002697   268F               .BLKL    2                        ; returned address's from SETPRT
                              2697  SERV_NAME:
                   00000000   2697               .LONG    0                        ; service name pointer
                              269B  STATUS:
                   00000000   269B               .LONG    0
                              269F  TMD_ADDR:
                   00000765'  269F               .ADDRESS TEST_MOD_BEGIN
                              26A3  TMN_ADDR:
                   00000772'  26A3               .ADDRESS TEST_MOD_NAME
                              26A7  TPID:
                   00000000   26A7               .LONG    0                        ; PID for this process
                              26AB  MSG1L:
                   00000100   26AB               .LONG    TEXT_BUFFER              ; Buffer #1 desc.
                   000026B3'  26AF               .ADDRESS BUF1
                              26B3
                              26B3  BUF1:
                   000027B3   26B3               .BLKB    TEXT_BUFFER
                              27B3
                              27B3  MESSAGE1L:
                   00000000   27B3               .LONG    0                        ; Message length
```

```
000026B3' 27B7   332            .ADDRESS BUF1
          27BB   333 BUFFER_PTR:                            ; Fake .ASCID buffer for misc. strings
0000 0100 27BB   334            .WORD   TEXT_BUFFER,0       ; A word for length, a word for desc.
00000073' 27BF   335            .ADDRESS BUFFER
```

```
00000000          .PSECT  SATSSS07,RD,WRT,EXE,LONG
    0000          .SBTTL  SATSSS07
    0000  ;++
    0000  ; FUNCTIONAL DESCRIPTION:
    0000  ;
    0000  ;       After performing some initial housekeeping, such as
    0000  ; printing the module begin message and acquiring needed privileges,
    0000  ; the system services are tested in each of their normal conditions.
    0000  ; Detected failures are identified and  an error message is printed
    0000  ; on the terminal.  Upon completion of the test a success or fail
    0000  ; message is printed on the terminal.
    0000  ;
    0000  ; CALLING SEQUENCE:
    0000  ;
    0000  ;       $ RUN SATSSS07  ...  (DCL COMMAND)
    0000  ;
    0000  ; INPUT PARAMETERS:
    0000  ;
    0000  ;       none
    0000  ;
    0000  ; IMPLICIT INPUTS:
    0000  ;
    0000  ;       none
    0000  ;
    0000  ; OUTPUT PARAMETERS:
    0000  ;
    0000  ;       none
    0000  ;
    0000  ; IMPLICIT OUTPUTS:
    0000  ;
    0000  ;       Messages to SYS$OUTPUT are the only output from SATSSS07.
    0000  ;       They are of the form:
    0000  ;
    0000  ;               %UETP-S-SATSMS, TEST MODULE SATSSS07 BEGUN ... (BEGIN MSG)
    0000  ;               %UETP-S-SATSMS, TEST MODULE SATSSS07 SUCCESSFUL ... (END MSG)
    0000  ;               %UETP-E-SATSMS, TEST MODULE SATSSS07 FAILED ... (END MSG)
    0000  ;               %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
    0000  ;
    0000  ; COMPLETION CODES:
    0000  ;
    0000  ;       The SATSSS07 routine terminates with a $EXIT to the
    0000  ;       operating system with a status code defined by UETP$_SATSMS.
    0000  ;
    0000  ; SIDE EFFECTS:
    0000  ;
    0000  ;       none
    0000  ;
    0000  ;--
    0000
    0000          TEST_START SATSSS07                     ; let the test begin
```

```
                                   0000
                             0000  0000          .ENTRY SATSSS07,0
                  2063'CF     D4   0002          CLRL    W^CURRENT_TC
                       00     DD   0006          PUSHL   #0
                  26A7'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
                  077B'CF     7F   001A          PUSHAQ  W^TEST_MOD_NAME_D
         00000000'GF   01     FB   001E          CALLS   #1,G^SYS$SETPRN
                     15C8     30   0025          BSBW    W^MOD_MSG_PRINT
           269F'CF  078B'CF   DE   0028          MOVAL   W^TEST_MOD_SUCC,W^TMD_ADDR
   2594'CF   03   00   01     F0   002F          INSV    #SUCCESS,#0,#3,W^MOD_MSG_CODE
                       00     DD   0036          PUSHL   #0
            0E16'CF    01     FB   0038          CALLS #1,W^REG_SAVE
                                   003D  STP0:
```

```
                          003D   388          .SBTTL  CREMBX TESTS
                          003D   389  ;
                          003D   390  ;
                          003D   391  ; $CREMBX tests
                          003D   392  ;
                          003D   393  ; Test temporary mailbox.
                          003D   394  ;
                          003D   395  ;-
 2697'CF  0008'CF    DE   003D   396          MOVAL   W^CREMBX,W^SERV_NAME              ; set service name
 2598'CF  0796'CF    DE   0044   397          MOVAL   W^UM,W^MODE                       ; set mode
                          004B   398          $CREMBX_S PRMFLG=#0,-
                          004B   399                  CHAN   =W^MBCHANS,-               ; try _s form
                          004B   400                  MAXMSG=#MBBUF,-                   ; set the max. message size
                          004B   401                  BUFQUO=#BUF_SIZ_S,-               ; set the buffer quota
                          004B   402                  LOGNAM=W^LOGNAMS                  ; try _s form
                          006C   403          FAIL_CHECK SS$_NORMAL                     ; check success
               01    DD   006C                  PUSHL   #SS$_NORMAL
      0E20'CF  01    FB   006E                  CALLS   #1,W^REG_CHECK
         1FBB'CF     D4   0073   404          CLRL    W^CRE+CREMBX$_PRMFLG              ; make it temporary
1FC3'CF  00000400 8F D0   0077   405          MOVL    #MBBUF,W^CRE+CREMBX$_MAXMSG       ; set the max. message size
1FC7'CF  00000400 8F D0   0080   406          MOVL    #BUF_SIZ_G,W^CRE+CREMBX$_BUFQUO   ; set the buffer quota
 1FD3'CF  0154'CF    DE   0089   407          MOVAL   W^LOGNAMG,W^CRE+CREMBX$_LOGNAM    ; set the lognam
                          0090   408          $CREMBX_G W^CRE                           ; try _g form
                          0099   409          FAIL_CHECK SS$_NORMAL                     ; check success
               01    DD   0099                  PUSHL   #SS$_NORMAL
      0E20'CF  01    FB   009B                  CALLS   #1,W^REG_CHECK
      0CFC'CF  00    FB   00A0   410          CALLS   #0,W^VERIFY_MBX                   ; read,write,verify,delete MBX
      0C3D'CF  00    FB   00A5   411          CALLS   #0,W^ERLBUF_DUMP                  ; dump errors
                          00AA   412  ;+
                          00AA   413  ;
                          00AA   414  ; Test permanent mailbox
                          00AA   415  ;
                          00AA   416  ;-
                          00AA   417          NEXT_TEST
                          00AA
                          00AA       STP1:
      2063'CF  01    D0   00AA                  MOVL    #1,W^CURRENT_TC
               00    DD   00AF                  PUSHL   #0
      0E16'CF  01    FB   00B1                  CALLS   #1,W^REG_SAVE
 2697'CF  0008'CF    DE   00B6   418          MOVAL   W^CREMBX,W^SERV_NAME              ; set service name
 2598'CF  0796'CF    DE   00BD   419          MOVAL   W^UM,W^MODE                       ; set mode
                          00C4   420          $CREMBX_S PRMFLG=#1,-
                          00C4   421                  CHAN   =W^MBCHANS,-
                          00C4   422                  MAXMSG=#MBBUF,-
                          00C4   423                  BUFQUO=#BUF_SIZ_S,-
                          00C4   424                  LOGNAM=W^LOGNAMS1                 ; try _s form
                          00E5   425          FAIL_CHECK SS$_NORMAL                     ; check success
               01    DD   00E5                  PUSHL   #SS$_NORMAL
      0E20'CF  01    FB   00E7                  CALLS   #1,W^REG_CHECK
                          00EC   426          $DASSGN_S CHAN=W^MBCHANS                  ; deassign mailbox
                          00F8   427          $ASSIGN_S DEVNAM=W^LOGNAMS1,-
                          00F8   428                  CHAN   =W^MBCHANS                 ; try to reassign MBX
          01   50    D1   0109   429          CMPL    R0,#SS$_NORMAL                    ; is the MBX permanent?
               02    13   010C   430          BEQL    10$                               ; br if yes
               50    11   010E   431          BRB     20$                               ; print perm. MBX error
                          0110   432  10$:
      1FBB'CF  01    D0   0110   433          MOVL    #1,W^CRE+CREMBX$_PRMFLG           ; set prmflg for perm. MBX
```

```
 1FC3'CF   00000400 8F   D0 0115        MOVL    #MBBUF,W^CRE+CREMBX$_MAXMSG      ; set the max. message size
 1FC7'CF   00000400 8F   D0 011E        MOVL    #BUF_SIZ_G,W^CRE+CREMBX$_BUFQUO  ; set the buffer quota_G
    1FD3'CF   0164'CF    DE 0127        MOVAL   W^LOGNAMG1,W^CRE+CREMBX$_LOGNAM  ; set the lognam
                            012E        $CREMBX_G W^CRE                          ; try _G form
                            0137        FAIL_CHECK SS$_NORMAL                    ; check success
                    01   DD 0137                PUSHL   #SS$_NORMAL
         0E20'CF    01   FB 0139                CALLS   #1,W^REG_CHECK
                            013E        $DASSGN_S CHAN=W^MBCHANG                 ; deassign the channel
                            014A        $ASSIGN_S DEVNAM=W^LOGNAMG1,-
                            014A                CHAN =W^MBCHANG                  ; assign the mailbox
              01    50   D1 015B        CMPL    R0,#SS$_NORMAL                   ; is the MBX permanent?
                    09   13 015E        BEQL    30$                              ; br if yes
                            0160 20$:
              0310'CF    DF 0160        PUSHAL  W^MSG1
         1501'CF    01   FB 0164        CALLS   #1,W^PRINT_FAIL                  ; print failure
                            0169 30$:
         0CFC'CF    00   FB 0169        CALLS   #0,W^VERIFY_MBX                  ; read,write,verify,delete MBX
         0C3D'CF    00   FB 016E        CALLS   #0,W^ERLBUF_DUMP                 ; dump errors
                            0173 ;+
                            0173 ;
                            0173 ; $DELMBX test
                            0173 ;
                            0173 ;-
                            0173        $ASSIGN_S DEVNAM=W^LOGNAMS1,-
                            0173                CHAN =W^MBCHANS                  ; reassign the mailbox_S
    2697'CF   0120'CF    DE 0184        MOVAL   W^DELMBX,W^SERV_NAME             ; set service name
                    00   DD 018B        PUSHL   #0
         0E16'CF    01   FB 018D        CALLS   #1,W^REG_SAVE                    ; save the registers
                            0192        $DELMBX_S CHAN=W^MBCHANS                 ; delete the _S mailbox
                            019E        FAIL_CHECK SS$_NORMAL
                    01   DD 019E                PUSHL   #SS$_NORMAL
         0E20'CF    01   FB 01A0                CALLS   #1,W^REG_CHECK
                            01A5        $ASSIGN_S DEVNAM=W^LOGNAMG1,-
                            01A5                CHAN =W^MBCHANG                  ; reassign the mailbox_S
    206C'CF   20FE'CF    3C 01B6        MOVZWL  W^MBCHANG,W^DEL+DELMBX$_CHAN
                            01BD        $DELMBX_G W^DEL                          ; delete the _G mailbox
                            01C6        FAIL_CHECK SS$_NORMAL
                    01   DD 01C6                PUSHL   #SS$_NORMAL
         0E20'CF    01   FB 01C8                CALLS   #1,W^REG_CHECK
                            01CD ;+
                            01CD ;
                            01CD ; Test MAXMSG low limit (1)
                            01CD ;
                            01CD ;-
                            01CD        NEXT_TEST
                            01CD
                            01CD STP2:
         2063'CF    02   D0 01CD                MOVL    #2,W^CURRENT_TC
                    00   DD 01D2                PUSHL   #0
         0E16'CF    01   FB 01D4                CALLS   #1,W^REG_SAVE
    2697'CF   0008'CF    DE 01D9        MOVAL   W^CREMBX,W^SERV_NAME             ; set service name
    2598'CF   0796'CF    DE 01E0        MOVAL   W^UM,W^MODE                      ; set mode
                            01E7        $CREMBX_S PRMFLG=#0,-
                            01E7                CHAN =W^MBCHANS,-
                            01E7                MAXMSG=#LOMSG,-                  ; set the max. message size
                            01E7                BUFQUO=#BUF_SIZ_S,-              ; set the buffer size
                            01E7                LOGNAM=W^LOGNAMS                 ; try _s form
```

```
                                   0204   480        FAIL_CHECK SS$_NORMAL                           ; check success
                     01      DD    0204                        PUSHL    #SS$_NORMAL
          0E20'CF    01      FB    0206                        CALLS    #1,W^REG_CHECK
                                   020B   481
             1FBB'CF         D4    020B   482        CLRL      W^CRE+CREMBX$_PRMFLG                  ; make it temporary
          1FC3'CF    01      D0    020F   483        MOVL      #LOMSG,W^CRE+CREMBX$_MAXMSG           ; set the max. message size
   1FC7'CF   00000400 8F     D0    0214   484        MOVL      #BUF_SIZ_G,W^CRE+CREMBX$_BUFQUO       ; set the buffer quota
        1FD3'CF   0154'CF    DE    021D   485        MOVAL     W^LOGNAMG,W^CRE+CREMBX$_LOGNAM        ; move lognam in
                                   0224   486        $CREMBX_G W^CRE                                 ; try _g form
                                   022D   487        FAIL_CHECK SS$_NORMAL                           ; check success
                     01      DD    022D                        PUSHL    #SS$_NORMAL
          0E20'CF    01      FB    022F                        CALLS    #1,W^REG_CHECK
                                   0234   488        $QIO_S    CHAN=W^MBCHANS,-
                                   0234   489                  FUNC=#IO$_WRITEVBLK,-
                                   0234   490                  P1  =W^TEST_DATA,-
                                   0234   491                  P2  =#LOMSG                           ; write to the mailbox_S
 0100 8F    00    2180'CF    00  2C    0253   492        MOVC5     #0,W^MBX_BUF,#0,#256,W^MBX_BUF        ; zero the MBX buffer
                  2180'CF          025C
                                   025F   493        $QIO_S    CHAN=W^MBCHANS,-
                                   025F   494                  FUNC=#IO$_READVBLK,-
                                   025F   495                  P1  =W^MBX_BUF,-
                                   025F   496                  P2  =#LOMSG                           ; read from the mailbox
               56   2180'CF  DE    027E   497        MOVAL     W^MBX_BUF,R6                          ; set the MBX buffer
               57   0365'CF  DE    0283   498        MOVAL     W^TEST_DATA,R7                        ; find the master data
                     58  01  D0    0288   499        MOVL      #LOMSG,R8
          20FC'CF   2100'CF  B0    028B   500        MOVW      W^MBCHANS,W^MBCHAN                    ; get the channel number
          0C7A'CF    00      FB    0292   501        CALLS     #0,W^BUF_CHECK                        ; check the data
                                   0297   502        $DASSGN_S CHAN=W^MBCHANS                        ; deassign the channel
                                   02A3   503        $QIO_S    CHAN=W^MBCHANG,-
                                   02A3   504                  FUNC=#IO$_WRITEVBLK,-
                                   02A3   505                  P1  =W^TEST_DATA,-
                                   02A3   506                  P2  =#LOMSG                           ; write to the mailbox_G
 0400 8F    00    2180'CF    00  2C    02C2   507        MOVC5     #0,W^MBX_BUF,#0,#1024,W^MBX_BUF       ; zero the MBX buffer
                  2180'CF          02CB
                                   02CE   508        $QIO_S    CHAN=W^MBCHANG,-
                                   02CE   509                  FUNC=#IO$_READVBLK,-
                                   02CE   510                  P1  =W^MBX_BUF,-
                                   02CE   511                  P2  =#LOMSG                           ; read from the mailbox
               56   2180'CF  DE    02ED   512        MOVAL     W^MBX_BUF,R6                          ; set the MBX buffer
               57   0365'CF  DE    02F2   513        MOVAL     W^TEST_DATA,R7                        ; find the master data
                     58  01  D0    02F7   514        MOVL      #LOMSG,R8
          20FC'CF   20FE'CF  B0    02FA   515        MOVW      W^MBCHANG,W^MBCHAN                    ; get the channel number
          0C7A'CF    00      FB    0301   516        CALLS     #0,W^BUF_CHECK                        ; check the data
                                   0306   517        $DASSGN_S CHAN=W^MBCHANG                        ; deassign the channel
                                   0312   518  ;
                                   0312   519  ; Test BUFQUO using BUFQUO = 256 using _S form
                                   0312   520  ;                 BUFQUO = 512 using _G form
                                   0312   521  ;
                                   0312   522  ;-
                                   0312   523        NEXT_TEST
                                   0312
                                   0312        STP3:
          2063'CF    03      D0    0312                        MOVL     #3,W^CURRENT_TC
                     00      DD    0317                        PUSHL    #0
          0E16'CF    01      FB    0319                        CALLS    #1,W^REG_SAVE
        2697'CF   0008'CF    DE    031E   524        MOVAL     W^CREMBX,W^SERV_NAME                  ; set service name
        2598'CF   0796'CF    DE    0325   525        MOVAL     W^UM,W^MODE                           ; set mode
```

```
                                 032C    $CREMBX_S PRMFLG=#0,-
                                 032C              CHAN  =W^MBCHANS,-
                                 032C              MAXMSG=#MBBUF,-
                                 032C              BUFQUO=#BUF_SIZ_S,-
                                 032C              LOGNAM=W^LOGNAMS            ; try _s
                                 034D    FAIL_CHECK SS$_NORMAL                 ; check success
                        01   DD  034D              PUSHL   #SS$_NORMAL
              0E20'CF   01   FB  034F              CALLS   #1,W^REG_CHECK
                   1FBB'CF   D4  0354    CLRL      W^CRE+CREMBX$_PRMFLG
      1FC3'CF  00000400 8F   D0  0358    MOVL      #MBBUF,W^CRE+CREMBX$_MAXMSG         ; set the max. message size
      1FC7'CF  00000200 8F   D0  0361    MOVL      #BUF_SIZ_S*2,W^CRE+CREMBX$_BUFQUO  ; set the buffer quota
         1FD3'CF   0154'CF   DE  036A    MOVAL     W^LOGNAMG,W^CRE+CREMBX$_LOGNAM
                                 0371    $CREMBX_G W^CRE                       ; try _G form
                                 037A    FAIL_CHECK SS$_NORMAL                 ; check success
                        01   DD  037A              PUSHL   #SS$_NORMAL
              0E20'CF   01   FB  037C              CALLS   #1,W^REG_CHECK
                                 0381    $QIO_S    CHAN=W^MBCHANS,-
                                 0381              FUNC=#IO$_WRITEVBLK,-
                                 0381              P1  =W^TEST_DATA,-
                                 0381              P2  =#BUF_SIZ_S             ; write to the mailbox_S
 0100 8F   00   2180'CF   00   2C  03A4    MOVC5     #0,W^MBX_BUF,#0,#256,W^MBX_BUF  ; zero the mailbox buffer
                2180'CF          03AD
                                 03B0    $QIO_S    CHAN=W^MBCHANS,-
                                 03B0              FUNC=#IO$_READVBLK,-
                                 03B0              P1  =W^MBX_BUF,-
                                 03B0              P2  =#BUF_SIZ_S             ; read from the mailbox_S
              56   2180'CF   DE  03D3    MOVAL     W^MBX_BUF,R6                ; set the MBX buffer
              57   0365'CF   DE  03D8    MOVAL     W^TEST_DATA,R7              ; find the master data
           58   00000100 8F  D0  03DD    MOVL      #BUF_SIZ_S,R8
         20FC'CF   2100'CF   B0  03E4    MOVW      W^MBCHANS,W^MBCHAN          ; get the channel number
              0C7A'CF   00   FB  03EB    CALLS     #0,W^BUF_CHECK              ; check the data
                                 03F0    $DASSGN_S CHAN=W^MBCHANS              ; deassign the channel
         20FC'CF   2100'CF   B0  03FC    MOVW      W^MBCHANS,W^MBCHAN          ; get the channel number
                                 0403    $QIO_S    CHAN=W^MBCHANG,-
                                 0403              FUNC=#IO$_WRITEVBLK,-
                                 0403              P1  =W^TEST_DATA,-
                                 0403              P2  =#BUF_SIZ_S*2           ; write to the mailbox_G
 0200 8F   00   2180'CF   00   2C  0426    MOVC5     #0,W^MBX_BUF,#0,#512,W^MBX_BUF  ; zero the mailbox buffer
                2180'CF          042F
                                 0432    $QIO_S    CHAN=W^MBCHANG,-
                                 0432              FUNC=#IO$_READVBLK,-
                                 0432              P1  =W^MBX_BUF,-
                                 0432              P2  =#BUF_SIZ_S*2           ; read from the mailbox_G
              56   2180'CF   DE  0455    MOVAL     W^MBX_BUF,R6                ; set the mailbox buffer
              57   0365'CF   DE  045A    MOVAL     W^TEST_DATA,R7              ; find the master data
           58   00000200 8F  D0  045F    MOVL      #BUF_SIZ_S*2,R8
         20FC'CF   20FE'CF   B0  0466    MOVW      W^MBCHANG,W^MBCHAN          ; get the channel number
              0C7A'CF   00   FB  046D    CALLS     #0,W^BUF_CHECK              ; check the data
                                 0472    $DASSGN_S CHAN=W^MBCHANG              ; deassign the channel
                                 047E ;+
                                 047E ;
                                 047E ; Test PROMSK (protection mask)
                                 047E ;
                                 047E ;-
                                 047E          NEXT_TEST
                                 047E
                                 047E STP4:
```

```
          2063'CF     04    D0  047E               MOVL    #4,W^CURRENT_TC
                      00    DD  0483               PUSHL   #0
          0E16'CF     01    FB  0485               CALLS   #1,W^REG_SAVE
 2697'CF     0008'CF        DE  048A  575        MOVAL   W^CREMBX,W^SERV_NAME          ; set service name
 2598'CF     0796'CF        DE  0491  576        MOVAL   W^UM,W^MODE                   ; set mode
                                0498  577        $CREMBX_S PRMFLG=#0,-
                                0498  578                CHAN  =W^MBCHANS,-
                                0498  579                MAXMSG=#MBBUF,-
                                0498  580                BUFQUO=#BUF_SIZ_S,-
                                0498  581                PROMSK=#^X3303,=
                                0498  582                LOGNAM=LOGNAMS               ; try _s form
                                04BD  583        FAIL_CHECK SS$_NORMAL                 ; check success
                      01    DD  04BD               PUSHL   #SS$_NORMAL
          0E20'CF     01    FB  04BF               CALLS   #1,W^REG_CHECK
                                04C4  584        $GETCHN_S CHAN=W^MBCHANS,-
                                04C4  585                PRIBUF=W^MBCHAR               ; get the MBX_S channel char.
    2124'CF     3303 8F     B1  04DA  586        CMPW    #^X3303,W^MBCHR+DIB$W_VPROT
                      13    13  04E1  587        BEQL    10$                           ; branch if O.K.
                 2124'CF    DD  04E3  588        PUSHL   W^MBCHR+DIB$W_VPROT
            00003303 8F     DD  04E7  589        PUSHL   #^X3303
                 034A'CF    DF  04ED  590        PUSHAL  W^PROT
          1501'CF     03    FB  04F1  591        CALLS   #3,W^PRINT_FAIL               ; print the error
                                04F6  592 10$:
                 1FBB'CF    D4  04F6  593        CLRL    W^CRE+CREMBX$_PRMFLG          ; make it temporary
 1FC3'CF   00000400 8F      D0  04FA  594        MOVL    #MBBUF,W^CRE+CREMBX$_MAXMSG   ; set max. message size
 1FC7'CF   00000400 8F      D0  0503  595        MOVL    #BUF_SIZ_G,W^CRE+CREMBX$_BUFQUO ; set the buffer quota
 1FCB'CF   00003303 8F      D0  050C  596        MOVL    #^X3303,W^CRE+CREMBX$_PROMSK  ; clear protection mask
   1FD3'CF   0154'CF        DE  0515  597        MOVAL   W^LOGNAMG,W^CRE+CREMBX$_LOGNAM ; set the lognam
                                051C  598        $CREMBX_G W^CRE                       ; try _g form
                                0525  599        FAIL_CHECK SS$_NORMAL                 ; check success
                      01    DD  0525               PUSHL   #SS$_NORMAL
          0E20'CF     01    FB  0527               CALLS   #1,W^REG_CHECK
                                052C  600        $GETCHN_S CHAN=W^MBCHANG,-
                                052C  601                PRIBUF=W^MBCHAR               ; get MBX_G channel char.
    2124'CF     3303 8F     B1  0542  602        CMPW    #^X3303,W^MBCHR+DIB$W_VPROT   ; CHECK PROMSK
                      13    13  0549  603        BEQL    20$                           ; BR if O.K.
                 2124'CF    DD  054B  604        PUSHL   W^MBCHR+DIB$W_VPROT
            00003303 8F     DD  054F  605        PUSHL   #^X3303
                 034A'CF    DF  0555  606        PUSHAL  W^PROT
          1501'CF     03    FB  0559  607        CALLS   #3,W^PRINT_FAIL               ; print the error
                                055E  608 20$:
          0CFC'CF     00    FB  055E  609        CALLS   #0,W^VERIFY_MBX               ; read,write,verify,delete MB
          0C3D'CF     00    FB  0563  610        CALLS   #0,W^ERLBUF_DUMP              ; dump errors
                                0568  611 ;+
                                0568  612 ;
                                0568  613 ; Test ACMODE (super mode)
                                0568  614 ;
                                0568  615 ;-
                                0568  616        NEXT_TEST
                                0568
                                0568      STP5:
          2063'CF     05    D0  0568               MOVL    #5,W^CURRENT_TC
                      00    DD  056D               PUSHL   #0
          0E16'CF     01    FB  056F               CALLS   #1,W^REG_SAVE
 2697'CF     0008'CF        DE  0574  617        MOVAL   W^CREMBX,W^SERV_NAME          ; set service name
 2598'CF     0358'CF        DE  057B  618        MOVAL W^SM,W^MODE                     ; declare super mode
                                0582  619        $CMKRNL_S W^SETUP_SUPER,W^ARGLST      ; declare CHMS handler
```

```
          5E      10'  C0 0591          ADDL2   S^#EXE$C_CMSTKSZ+16,SP          ; adjust user stack ptr.
          5D      5E   D0 0594          MOVL    SP,FP                           ; fix the FP
     0C3D'CF      00   FB 0597          CALLS   #0,W^ERLBUF_DUMP                ; dump errors
                  01   BE 059C          CHMS    #1                              ; do the super tests
     0C3D'CF      00   FB 059E          CALLS   #0,W^ERLBUF_DUMP                ; dump any errors
                          05A3    ;+
                          05A3    ;
                          05A3    ; Test EXEC mode
                          05A3    ;
                          05A3    ;-
                          05A3          NEXT_TEST
                          05A3
                          05A3    STP6:
     2063'CF      06   D0 05A3                  MOVL    #6,W^CURRENT_TC
                  00   DD 05A8                  PUSHL   #0
     0E16'CF      01   FB 05AA                  CALLS   #1,W^REG_SAVE
2697'CF      0008'CF   DE 05AF          MOVAL   W^CREMBX,W^SERV_NAME            ; set service name
2598'CF      0127'CF   DE 05B6          MOVAL   W^EM,W^MODE                     ; declare exec mode
                          05BD          $CMEXEC_S B^10$                         ; go to exec mode
               0067    31 05C9          BRW     20$
                          05CC    10$:
                  00   DD 05CC          PUSHL   #0
     0E16'CF      01   FB 05CE          CALLS   #1,W^REG_SAVE                   ; save the registers
                          05D3          $CREMBX_S PRMFLG=#0,-
                          05D3                  CHAN  =W^MBCHANS,-
                          05D3                  MAXMSG=#MBBUF,-                 ; set the max. message size
                          05D3                  BUFQUO=#BUF_SIZ_S,-             ; set the buffer quota
                          05D3                  ACMODE=#PSL$C_EXEC,-            ; set the access mode (super)
                          05D3                  LOGNAM=W^LOGNAMS                ; try _s form
                          05F4          FAIL_CHECKNP SS$_NORMAL                 ; check success
                  01   DD 05F4                  PUSHL   #SS$_NORMAL
     1443'CF      01   FB 05F6                  CALLS   #1,W^REG_CHECKNP
       1FBB'CF         D4 05FB          CLRL    W^CRE+CREMBX$_PRMFLG            ; make it temporary
1FC3'CF  00000400 8F   D0 05FF          MOVL    #MBBUF,W^CRE+CREMBX$_MAXMSG     ; set the max. message size
1FC7'CF  00000400 8F   D0 0608          MOVL    #BUF_SIZ_G,W^CRE+CREMBX$_BUFQUO ; set the buffer quota
     1FCF'CF      01   D0 0611          MOVL    #PSL$C_EXEC,W^CRE+CREMBX$_ACMODE ; move in ACMODE
  1FD3'CF    0154'CF   DE 0616          MOVAL   W^LOGNAMG,W^CRE+CREMBX$_LOGNAM  ; set the lognam
                          061D          $CREMBX_G W^CRE                         ; try _G form
                          0626          FAIL_CHECKNP SS$_NORMAL                 ; check success
                  01   DD 0626                  PUSHL   #SS$_NORMAL
     1443'CF      01   FB 0628                  CALLS   #1,W^REG_CHECKNP
     0CFC'CF      00   FB 062D          CALLS   #0,W^VERIFY_MBX                 ; read,write,verify,delete MBX
                       04 0632          RET
                          0633    20$:
     0C3D'CF      00   FB 0633          CALLS   #0,W^ERLBUF_DUMP                ; dump errors
                          0638    ;+
                          0638    ;
                          0638    ; Test KERNEL mode
                          0638    ;
                          0638    ;-
                          0638          NEXT_TEST
                          0638
                          0638    STP7:
     2063'CF      07   D0 0638                  MOVL    #7,W^CURRENT_TC
                  00   DD 063D                  PUSHL   #0
     0E16'CF      01   FB 063F                  CALLS   #1,W^REG_SAVE
2697'CF      0008'CF   DE 0644          MOVAL   W^CREMBX,W^SERV_NAME            ; set service name
```

```
 2598'CF   0146'CF  DE 064B        MOVAL   W^KM,W^MODE                     ; set mode
                       0652        $CMKRNL_S B^10$
              0067  31 065E        BRW     20$
                       0661  10$:
                00  DD 0661        PUSHL   #0
      0E16'CF   01  FB 0663        CALLS   #1,W^REG_SAVE                   ; save the registers
                       0668        $CREMBX_S PRMFLG=#0,-
                       0668                CHAN   =W^MBCHANS,-
                       0668                MAXMSG=#MBBUF,-
                       0668                BUFQUO=#BUF_SIZ_S,-
                       0668                ACMODE=#PSL$C_KERNEL,-
                       0668                LOGNAM=W^LOGNAMS                ; try _s form
                       0689        FAIL_CHECKNP SS$_NORMAL                 ; check success
                01  DD 0689                PUSHL   #SS$_NORMAL
      1443'CF   01  FB 068B                CALLS   #1,W^REG_CHECKNP
           1FBB'CF  D4 0690        CLRL    W^CRE+CREMBX$_PRMFLG            ; make it temporary
 1FC3'CF  00000400 8F  D0 0694     MOVL    #MBBUF,W^CRE+CREMBX$_MAXMSG     ; get MAXMSG parameter
 1FC7'CF  00000400 8F  D0 069D     MOVL    #BUF_SIZ_G,W^CRE+CREMBX$_BUFQUO ; set the buffer quota
      1FCF'CF   00  D0 06A6        MOVL    #PSL$C_KERNEL,W^CRE+CREMBX$_ACMODE ; move in ACMODE
 1FD3'CF   0154'CF  DE 06AB        MOVAL   W^LOGNAMG,W^CRE+CREMBX$_LOGNAM  ; set the lognam
                       06B2        $CREMBX_G W^CRE                         ; try _g form
                       06BB        FAIL_CHECKNP SS$_NORMAL                 ; check success
                01  DD 06BB                PUSHL   #SS$_NORMAL
      1443'CF   01  FB 06BD                CALLS   #1,W^REG_CHECKNP
      0CFC'CF   00  FB 06C2        CALLS   #0,W^VERIFY_MBX                 ; read,write,verify,delete MBX
                04     06C7        RET
                       06C8  20$:
      0C3D'CF   00  FB 06C8        CALLS   #0,W^ERLBUF_DUMP                ; dump errors
                       06CD  ;+
                       06CD  ;
                       06CD  ; Test USER mode
                       06CD  ;
                       06CD  ;-
                       06CD        NEXT_TEST
                       06CD
                       06CD  STP8:
      2063'CF   08  D0 06CD                MOVL    #8,W^CURRENT_TC
                00  DD 06D2                PUSHL   #0
      0E16'CF   01  FB 06D4                CALLS   #1,W^REG_SAVE
 2598'CF   0796'CF  DE 06D9        MOVAL   W^UM,W^MODE                     ; reset user mode
 2697'CF   0008'CF  DE 06E0        MOVAL   W^CREMBX,W^SERV_NAME            ; set service name
                       06E7        $CREMBX_S PRMFLG=#0,-
                       06E7                CHAN   =W^MBCHANS,-
                       06E7                MAXMSG=#MBBUF,-
                       06E7                BUFQUO=#BUF_SIZ_S,-
                       06E7                ACMODE=#PSL$C_USER,-
                       06E7                LOGNAM=W^LOGNAMS                ; try _S form
                       0708        FAIL_CHECK SS$_NORMAL                   ; check success
                01  DD 0708                PUSHL   #SS$_NORMAL
      0E20'CF   01  FB 070A                CALLS   #1,W^REG_CHECK
           1FBB'CF  D4 070F        CLRL    W^CRE+CREMBX$_PRMFLG            ; make it temporary
 1FC3'CF  00000400 8F  D0 0713     MOVL    #MBBUF,W^CRE+CREMBX$_MAXMSG     ; set MAXMSG parameter
 1FC7'CF  00000400 8F  D0 071C     MOVL    #BUF_SIZ_G,W^CRE+CREMBX$_BUFQUO ; set the buffer quota
      1FCF'CF   01  D0 0725        MOVL    #PSL$C_EXEC,W^CRE+CREMBX$_ACMODE ; set the acmode
 1FD3'CF   0154'CF  DE 072A        MOVAL   W^LOGNAMG,W^CRE+CREMBX$_LOGNAM  ; set the lognam
                       0731        $CREMBX_G W^CRE                         ; try _G form
                       073A        FAIL_CHECK SS$_NORMAL                   ; check success
```

```
                01    DD  073A                   PUSHL   #SS$_NORMAL
     0E20'CF    01    FB  073C                   CALLS   #1,W^REG_CHECK
     0CFC'CF    00    FB  0741  709      CALLS   #0,W^VERIFY_MBX                    ; read,write,verify,delete MBX
     0C3D'CF    00    FB  0746  710      CALLS   #0,W^ERLBUF_DUMP                   ; dump errors
                          074B  711      .SBTTL  FAIL CREMBX TESTS
                          074B  712 ;+++
                          074B  713 ;
                          074B  714 ; Test access violation for logical name
                          074B  715 ;
                          074B  716 ;---
                          074B  717          NEXT_TEST
                          074B
                          074B          STP9:
     2063'CF    09    D0  074B                   MOVL    #9,W^CURRENT_TC
                00    DD  0750                   PUSHL   #0
     0E16'CF    01    FB  0752                   CALLS   #1,W^REG_SAVE
                          0757  718      $CREMBX_S CHAN  =MBCHANS,-
                          0757  719              LOGNAM=ZERO_ADDR_DESCR             ; page zero
                          076E  720      FAIL_CHECK SS$_ACCVIO                      ; check success
                0C    DD  076E                   PUSHL   #SS$_ACCVIO
     0E20'CF    01    FB  0770                   CALLS   #1,W^REG_CHECK
                          0775  721 ;+++
                          0775  722 ;
                          0775  723 ; Test access violation for channel number
                          0775  724 ;
                          0775  725 ;---
                          0775  726          NEXT_TEST
                          0775
                          0775          STP10:
     2063'CF    0A    D0  0775                   MOVL    #10,W^CURRENT_TC
                00    DD  077A                   PUSHL   #0
     0E16'CF    01    FB  077C                   CALLS   #1,W^REG_SAVE
                          0781  727      $CREMBX_S CHAN  =0                         ; channel zero
                          0792  728      FAIL_CHECK SS$_ACCVIO                      ; check success
                0C    DD  0792                   PUSHL   #SS$_ACCVIO
     0E20'CF    01    FB  0794                   CALLS   #1,W^REG_CHECK
                          0799  729 ;+++
                          0799  730 ;
                          0799  731 ; Test invalid logical name descriptor
                          0799  732 ;
                          0799  733 ;---
                          0799  734          NEXT_TEST
                          0799
                          0799          STP11:
     2063'CF    0B    D0  0799                   MOVL    #11,W^CURRENT_TC
                00    DD  079E                   PUSHL   #0
     0E16'CF    01    FB  07A0                   CALLS   #1,W^REG_SAVE
                          07A5  735      $CREMBX_S CHAN  =MBCHANS,-
                          07A5  736              LOGNAM=LEN_00_DESCR                ; zero length name!
                          07BC  737      FAIL_CHECK SS$_IVLOGNAM                    ; check success
   00000154 8F    DD  07BC                   PUSHL   #SS$_IVLOGNAM
     0E20'CF    01    FB  07C2                   CALLS   #1,W^REG_CHECK
                          07C7  738 ;+++
                          07C7  739 ;
                          07C7  740 ; Test valid 255 character logical name descriptor
                          07C7  741 ;                (CREMBX service prefixes 'MBX$' to the string!)
                          07C7  742 ;
```

```
                        07C7   743 ;---
                        07C7   744         NEXT_TEST
                        07C7
                        07C7       STP12:
     2063'CF   0C   D0  07C7                   MOVL    #12,W^CURRENT_TC
               00   DD  07CC                   PUSHL   #0
     0E16'CF   01   FB  07CE                   CALLS   #1,W^REG_SAVE
     000E'CF   01   90  07D3   745         MOVB    #1,CREATED_FLAG                 ; assume it works
                        07D8   746         $CREMBX_S CHAN  =MBCHAND,-
                        07D8   747                 LOGNAM=LEN_63_DESCR             ; 255 chars long
          04   50   E8  07EF   748         BLBS    R0,10$                          ; did it work?
        000E'CF     94  07F2   749         CLRB    CREATED_FLAG                    ; apparently not
                        07F6   750 10$:    FAIL_CHECK SS$_NORMAL                   ; check success
               01   DD  07F6                   PUSHL   #SS$_NORMAL
     0E20'CF   01   FB  07F8                   CALLS   #1,W^REG_CHECK
                        07FD   751 ;+++
                        07FD   752 ;
                        07FD   753 ; Test DELMBX sys service invalid channel ( 0 )
                        07FD   754 ;
                        07FD   755 ;---
                        07FD   756         NEXT_TEST
                        07FD
                        07FD       STP13:
     2063'CF   0D   D0  07FD                   MOVL    #13,W^CURRENT_TC
               00   DD  0802                   PUSHL   #0
     0E16'CF   01   FB  0804                   CALLS   #1,W^REG_SAVE
2697'CF   0120'CF   DE  0809   757         MOVAL   DELMBX,SERV_NAME                ; set service name
                        0810   758         $DELMBX_S CHAN = CHANNEL_ZERO
                        081C   759         FAIL_CHECK SS$_IVCHAN                   ; check int. failure
   0000013C 8F      DD  081C                   PUSHL   #SS$_IVCHAN
     0E20'CF   01   FB  0822                   CALLS   #1,W^REG_CHECK
                        0827   760 ;+++
                        0827   761 ;
                        0827   762 ; Test DELMBX sys service invalid channel ( >#channels )
                        0827   763 ;
                        0827   764 ;---
                        0827   765         NEXT_TEST
                        0827
                        0827       STP14:
     2063'CF   0E   D0  0827                   MOVL    #14,W^CURRENT_TC
               00   DD  082C                   PUSHL   #0
     0E16'CF   01   FB  082E                   CALLS   #1,W^REG_SAVE
     7E   2102'CF   B0  0833   766         MOVW    MBCHAND,-(SP)                   ; save the channel #
2102'CF  00000000'9F  B0  0838   767       MOVW    @#CTL$GW_CHINDX,MBCHAND         ; get max # channel
     2102'CF   10   A0  0841   768         ADDW2   #^X10,MBCHAND                   ; one nibble more
                        0846   769         $DELMBX_S CHAN = MBCHAND
                        0852   770         FAIL_CHECK SS$_IVCHAN                   ; check int. failure
   0000013C 8F      DD  0852                   PUSHL   #SS$_IVCHAN
     0E20'CF   01   FB  0858                   CALLS   #1,W^REG_CHECK
     2102'CF   8E   B0  085D   771         MOVW    (SP)+,MBCHAND                   ; get back the channel
                        0862   772 ;+++
                        0862   773 ;
                        0862   774 ; Test DELMBX sys service - delete one just created
                        0862   775 ;
                        0862   776 ;---
        000E'CF     95  0862   777         TSTB    CREATED_FLAG                    ; not if it wasn't created
               1F   13  0866   778         BEQL    SKIP_DELETE
```

```
                      0868   779            NEXT_TEST
                      0868
                      0868          STP15:
 2063'CF    0F   D0   0868                          MOVL    #15,W^CURRENT_TC
            00   DD   086D                          PUSHL   #0
 0E16'CF    01   FB   086F                          CALLS   #1,W^REG_SAVE
                      0874   780            $DELMBX_S CHAN = MBCHAND
                      0880   781            FAIL_CHECK SS$_NORMAL                       ; should be O.K.
            01   DD   0880                          PUSHL   #SS$_NORMAL
 0E20'CF    01   FB   0882                          CALLS   #1,W^REG_CHECK
                      0887   782 SKIP_DELETE:
                      0887   783 ;+++
                      0887   784 ;
                      0887   785 ; Test DELMBX sys service - channel not assigned
                      0887   786 ;
                      0887   787 ;---
                      0887   788            NEXT_TEST
                      0887
                      0887          STP16:
 2063'CF    10   D0   0887                          MOVL    #16,W^CURRENT_TC
            00   DD   088C                          PUSHL   #0
 0E16'CF    01   FB   088E                          CALLS   #1,W^REG_SAVE
                      0893   789            $DASSGN_S CHAN = MBCHAND                    ; deassign the channel
       07 50     E8   089F   790            BLBS    R0,120$                             ; Skip error report if OK
            50   DD   08A2   791            PUSHL   R0                                  ; Save error code
 14BF'CF    01   FB   08A4   792            CALLS   #1,NONSUB_SSE                       ; Print the failure
                      08A9   793 120$:      $DELMBX_S CHAN = MBCHAND                    ; check int. failure
                      08B5   794            FAIL_CHECK SS$_NOPRIV                       ; 'channel not assigned'
            24   DD   08B5                          PUSHL   #SS$_NOPRIV
 0E20'CF    01   FB   08B7                          CALLS   #1,W^REG_CHECK
                      08BC   795 ;+++
                      08BC   796 ;
                      08BC   797 ; Test DELMBX sys service - device not mailbox
                      08BC   798 ;
                      08BC   799 ;---
                      08BC   800            NEXT_TEST
                      08BC
                      08BC          STP17:
 2063'CF    11   D0   08BC                          MOVL    #17,W^CURRENT_TC
            00   DD   08C1                          PUSHL   #0
 0E16'CF    01   FB   08C3                          CALLS   #1,W^REG_SAVE
                      08C8   801            $ASSIGN_S DEVNAM = SYS_DEV,-                ; system disk assigned
                      08C8   802                    CHAN = MBCHAND
       07 50     E8   08D9   803            BLBS    R0,120$                             ; Skip error report if OK
            50   DD   08DC   804            PUSHL   R0                                  ; Save error code
 14BF'CF    01   FB   08DE   805            CALLS   #1,NONSUB_SSE                       ; Print the failure
                      08E3   806 120$:      $DELMBX_S CHAN = MBCHAND
                      08EF   807            FAIL_CHECK SS$_DEVNOTMBX                    ; check int. failure
 00000074 8F     DD   08EF                          PUSHL   #SS$_DEVNOTMBX
 0E20'CF    01   FB   08F5                          CALLS   #1,W^REG_CHECK
                      08FA   808            $DASSGN_S CHAN = MBCHAND                    ; deassign SYS$SYSDEVICE
                      0906   809 ;+++
                      0906   810 ;
                      0906   811 ; Test no privilege for delete permanent mailbox
                      0906   812 ;
                      0906   813 ;---
                      0906   814            NEXT_TEST
```

```
                          0906
                          0906      STP18:
 2063'CF     12     D0    0906                        MOVL    #18,W^CURRENT_TC
             00     DD    090B                        PUSHL   #0
 0E16'CF     01     FB    090D                        CALLS   #1,W^REG_SAVE
 000E'CF     01     90    0912           MOVB    #1,CREATED_FLAG                        ; assume it works
                          0917           $CREMBX_S PRMFLG=#T,-
                          0917                     CHAN   =MBCHAND
       0B 50        E8    092A           BLBS    R0,120$                                ; Skip error report if OK
    000E'CF         94    092D           CLRB    CREATED_FLAG                           ; it didn't work
             50     DD    0931           PUSHL   R0                                     ; Save error code
 14BF'CF     01     FB    0933           CALLS   #1,NONSUB_SSE                          ; Print the failure
                          0938      120$:
    000E'CF         95    0938           TSTB    CREATED_FLAG                           ; was it created?
             5F     13    093C           BEQL    420$                                   ; branch if not
                          093E           $SETPRV_S ENBFLG=#DISABLE,-                    ; disable perm privilege
                          093E                     PRVADR=PRM_PRV_MASK
       07 50        E8    094F           BLBS    R0,220$                                ; Skip error report if OK
             50     DD    0952           PUSHL   R0                                     ; Save error code
 14BF'CF     01     FB    0954           CALLS   #1,NONSUB_SSE                          ; Print the failure
                          0959      220$:
                          0959           $DELMBX_S CHAN=MBCHAND                         ; attempt deletion
                          0965           FAIL_CHECK SS$_NOPRIV                          ; should fail
             24     DD    0965                        PUSHL   #SS$_NOPRIV
 0E20'CF     01     FB    0967                        CALLS   #1,W^REG_CHECK
                          096C           $SETPRV_S ENBFLG=#ENABLE,-                     ; re-enable perm privilege
                          096C                     PRVADR=PRM_PRV_MASK
       07 50        E8    097D           BLBS    R0,320$                                ; Skip error report if OK
             50     DD    0980           PUSHL   R0                                     ; Save error code
 14BF'CF     01     FB    0982           CALLS   #1,NONSUB_SSE                          ; Print the failure
                          0987      320$:
                          0987           $DELMBX_S CHAN=MBCHAND                         ; delete the mailbox
       07 50        E8    0993           BLBS    R0,420$                                ; Skip error report if OK
             50     DD    0996           PUSHL   R0                                     ; Save error code
 14BF'CF     01     FB    0998           CALLS   #1,NONSUB_SSE                          ; Print the failure
                          099D      420$:
                          099D      ;+++
                          099D      ;
                          099D      ; Test invalid logical name descriptor ( >255 characters )
                          099D      ;                 (CREMBX service prefixes 'MBX$' to the string!);
                          099D      ;
                          099D      ;---
                          099D           NEXT_TEST
                          099D
                          099D      STP19:
 2063'CF     13     D0    099D                        MOVL    #19,W^CURRENT_TC
             00     DD    09A2                        PUSHL   #0
 0E16'CF     01     FB    09A4                        CALLS   #1,W^REG_SAVE
2697'CF   0008'CF   DE    09A9           MOVAL   CREMBX,SERV_NAME                       ; set service name
                          09B0           $CREMBX_S CHAN   =MBCHANS,-
                          09B0                     LOGNAM=LEN_256_DESCR                 ; >255 chars long
                          09C7           FAIL_CHECK SS$_IVLOGNAM                        ; check success
  00000154 8F       DD    09C7                        PUSHL   #SS$_IVLOGNAM
 0E20'CF     01     FB    09CD                        CALLS   #1,W^REG_CHECK
                          09D2      ;+++
                          09D2      ;
                          09D2      ; Test logical name translation limit (=10)
```

```
                          09D2      ;
                          09D2      ;---
                          09D2              NEXT_TEST
                          09D2
                          09D2      STP20:
       2063'CF   14   D0  09D2                      MOVL    #20,W^CURRENT_TC
                 00   DD  09D7                      PUSHL   #0
       0E16'CF   01   FB  09D9                      CALLS   #1,W^REG_SAVE
                 04   BB  09DE              PUSHR   #^M<R2>                       ; Save R2's contents
                 52   D4  09E0              CLRL    R2                            ; Set index variable
   000D'CF    41 8F   90  09E2              MOVB    #^A/A/,EQUIV_NAME+4           ; init equivalence name
   0008'CF    42 8F   90  09E8              MOVB    #^A/B/,LOGIC_NAME+4           ; init logical name
                          09EE      ;
                          09EE      ;       Create 9 levels of logical names ( MBX$NAMEB := NAMEA )
                          09EE      ;                 (10 translations)
                          09EE      ;
                          09EE      10$:
                          09EE              $CRELOG_S LOGNAM = D_MBX_LOGIC_NAME,-
                          09EE                        TBLFLG = #2,-
                          09EE                        EQLNAM = D_EQUIV_NAME       ; Create a level of logical name
              07 50   E8  0A01              BLBS    R0,120$                       ; Skip error report if OK
                 50   DD  0A04              PUSHL   R0                            ; Save error code
       14BF'CF   01   FB  0A06              CALLS   #1,NONSUB_SSE                 ; Print the failure
         0008'CF      96  0A0B      120$:   INCB    LOGIC_NAME+4                  ; Bump the logical name by one
         000D'CF      96  0A0F              INCB    EQUIV_NAME+4                  ; Bump the equiv name by one
         D7 52   09   F2  0A13              AOBLSS  #9,R2,10$                     ; Make 9 levels
                 04   BA  0A17              POPR    #^M<R2>                       ; Restore R2
         0008'CF      97  0A19              DECB    LOGIC_NAME+4                  ; decrement due to loop incr
                          0A1D              $CREMBX_S CHAN  =MBCHANS,-
                          0A1D                        LOGNAM=D_LOGIC_NAME         ; should be O.K.
                          0A34              FAIL_CHECK SS$_NORMAL                 ; check success
                 01   DD  0A34                      PUSHL   #SS$_NORMAL
       0E20'CF   01   FB  0A36                      CALLS   #1,W^REG_CHECK
                          0A3B      ;+++
                          0A3B      ;
                          0A3B      ; Test logical name translation too deep (>10)
                          0A3B      ;
                          0A3B      ;---
                          0A3B              NEXT_TEST
                          0A3B
                          0A3B      STP21:
       2063'CF   15   D0  0A3B                      MOVL    #21,W^CURRENT_TC
                 00   DD  0A40                      PUSHL   #0
       0E16'CF   01   FB  0A42                      CALLS   #1,W^REG_SAVE
         0008'CF      96  0A47              INCB    LOGIC_NAME+4                  ; Bump the logical name by one
                          0A4B              $CRELOG_S LOGNAM = D_MBX_LOGIC_NAME,-
                          0A4B                        TBLFLG = #2,-
                          0A4B                        EQLNAM = D_EQUIV_NAME       ; Create one more level
              07 50   E8  0A5E              BLBS    R0,120$                       ; Skip error report if OK
                 50   DD  0A61              PUSHL   R0                            ; Save error code
       14BF'CF   01   FB  0A63              CALLS   #1,NONSUB_SSE                 ; Print the failure
                          0A68      120$:   $CREMBX_S CHAN  =MBCHANS,-
                          0A68                        LOGNAM=D_LOGIC_NAME         ; should fail
                          0A7F              FAIL_CHECK SS$_TOOMANYLNAM            ; check success
    00000374 8F       DD  0A7F                      PUSHL   #SS$_TOOMANYLNAM
       0E20'CF   01   FB  0A85                      CALLS   #1,W^REG_CHECK
                          0A8A      ;+++
```

```
                        0A8A      ;
                        0A8A      ; Test no privilege for permanent mailbox
                        0A8A      ;
                        0A8A      ;---
                        0A8A                NEXT_TEST
                        0A8A
                        0A8A      STP22:
       2063'CF    16  D0  0A8A                        MOVL    #22,W^CURRENT_TC
                  00  DD  0A8F                        PUSHL   #0
       0E16'CF    01  FB  0A91                        CALLS   #1,W^REG_SAVE
                        0A96                $SETPRV_S ENBFLG=#DISABLE,-
                        0A96                        PRVADR=PRM_PRV_MASK           ; disable perm privilege
             07 50    E8  0AA7                BLBS    R0,120$                       ; Skip error report if OK
                  50  DD  0AAA                PUSHL   R0                            ; Save error code
       14BF'CF    01  FB  0AAC                CALLS   #1,NONSUB_SSE                 ; Print the failure
                        0AB1      120$:     $CREMBX_S PRMFLG=#1,-
                        0AB1                        CHAN   =MBCHANS
                        0AC4                FAIL_CHECK SS$_NOPRIV                 ; check success
                  24  DD  0AC4                        PUSHL   #SS$_NOPRIV
       0E20'CF    01  FB  0AC6                        CALLS   #1,W^REG_CHECK
                        0ACB      ;+++
                        0ACB      ;
                        0ACB      ; Test no privilege for temporary mailbox
                        0ACB      ;
                        0ACB      ;---
                        0ACB                NEXT_TEST
                        0ACB
                        0ACB      STP23:
       2063'CF    17  D0  0ACB                        MOVL    #23,W^CURRENT_TC
                  00  DD  0AD0                        PUSHL   #0
       0E16'CF    01  FB  0AD2                        CALLS   #1,W^REG_SAVE
                        0AD7                $SETPRV_S ENBFLG=#DISABLE,-
                        0AD7                        PRVADR=TMP_PRV_MASK           ; disable temp privilege
             07 50    E8  0AE8                BLBS    R0,120$                       ; Skip error report if OK
                  50  DD  0AEB                PUSHL   R0                            ; Save error code
       14BF'CF    01  FB  0AED                CALLS   #1,NONSUB_SSE                 ; Print the failure
                        0AF2      120$:     $CREMBX_S CHAN  =MBCHANS
                        0B05                FAIL_CHECK SS$_NOPRIV                 ; check success
                  24  DD  0B05                        PUSHL   #SS$_NOPRIV
       0E20'CF    01  FB  0B07                        CALLS   #1,W^REG_CHECK
                        0B0C                $SETPRV_S ENBFLG=#ENABLE,-            ; re-enable privileges
                        0B0C                        PRVADR=TMP_PRM_PRV_MASK
             07 50    E8  0B1D                BLBS    R0,220$                       ; Skip error report if OK
                  50  DD  0B20                PUSHL   R0                            ; Save error code
       14BF'CF    01  FB  0B22                CALLS   #1,NONSUB_SSE                 ; Print the failure
                        0B27      220$:
                        0B27
                  02  BE  0B27                CHMS    #2                            ; reset super mode handler to DCL
                        0B29                TEST_END
            269F'CF   DD  0B29                        PUSHL   W^TMD_ADDR
            26A3'CF   DD  0B2D                        PUSHL   W^TMN_ADDR
                  02  DD  0B31                        PUSHL   #2
            2594'CF   DD  0B33                        PUSHL   W^MOD_MSG_CODE
   00000000'GF    04  FB  0B37                        CALLS   #$$T1,G^LIB$SIGNAL
2594'CF   01      1C  F0  0B3E                        INSV    #1,#STS$V_INHIB_MSG,#1,W^MOD_MSG_CODE
            2594'CF   DD  0B45                        PUSHL   W^MOD_MSG_CODE
   00000000'GF    01  FB  0B49                        CALLS   #1,G^SYS$EXIT
```

```
0B50  .SBTTL  ROUTINES
0B50  .SBTTL SETUP_SUPER ROUTINE
0B50  ;++
0B50  ;
0B50  ;       Routine to declare an initial CHMS handler from user mode.
0B50  ;
0B50  ; FUNCTIONAL DESCRIPTION:
0B50  ;
0B50  ; CALLING SEQUENCE:
0B50  ;
0B50  ;       $CMKRNL_S W^SETUP_SUPER,ARGLST
0B50  ;
0B50  ;               ARGLST = address of a pointer to a one parameter argument list conta
0B50  ;                        the address of the entry mask of the CHMS handler
0B50  ;
0B50  ; INPUT PARAMETERS:
0B50  ;
0B50  ;       ARGLST
0B50  ;
0B50  ; IMPLICIT INPUTS
0B50  ;
0B50  ;       NONE
0B50  ;
0B50  ; OUTPUT PARAMETERS:
0B50  ;
0B50  ;       Declares a change mode handler for super mode which must be
0B50  ;       reset to DCL in the users handler routine when the handler is
0B50  ;       no longer needed.
0B50  ;
0B50  ; IMPLICIT OUTPUTS:
0B50  ;
0B50  ;       NONE
0B50  ;
0B50  ; COMPLETION CODES:
0B50  ;
0B50  ;       NONE
0B50  ;
0B50  ; SIDE EFFECTS:
0B50  ;
0B50  ;       NONE
0B50  ;
0B50  ; ON ENTRY:
0B50  ;                       ---------              ---------
0B50  ;               KSP =>  !    0    !      USP =>  !       !
0B50  ;                       !    0    !              ! USER  !
0B50  ;                       !   AP    !              !       !
0B50  ;                       !   FP    !              ! CALL  !
0B50  ;                       !   PC    !              !       !
0B50  ;                       !    0    !              ! FRAME !
0B50  ;                       !    0    !              !       !
0B50  ;                       !   AP    !              ---------
0B50  ;                       !   FP    !
0B50  ;                       !SRVEXIT  !
0B50  ;                       !   PC    !
0B50  ;                       !  PSL    !
0B50  ;                       ---------
0B50  ;--
```

```
                      0B50  RETURN_PC:
            00000000  0B50          .LONG   0                               ; storage for user return PC
                      0B54  HANDLER_PC:
            00000000  0B54          .LONG   0                               ; storage for handler PC
                      0B58  ;
                      0B58  SETUP_SUPER:
                0000C 0B58          .WORD   ^M<R2,R3>
          53    03 DB 0B5A          MFPR    #PR$_USP,R3                     ; get the user call frame address
    EE AF    10 A3 D0 0B5D          MOVL    SF$L_SAVE_PC(R3),B^RETURN_PC ; get the user return PC
    ED AF    04 AC D0 0B62          MOVL    4(AP),HANDLER_PC                ; save the handler address
       52    0C AD D0 0B67          MOVL    SF$L_SAVE_FP(FP),R2             ; get saved FP
          52    00' C0 0B6B         ADDL    S^#EXE$C_CMSTKSZ,R2             ; back over change mode stack frame
       62    7C' AF 9E 0B6E         MOVAB   B^20$,(R2)                      ; set return address
                0A F0 0B72          INSV    #<<PSL$C_SUPER@PSL$S_CURMOD>+PSL$C_SUPER>,-
                16    0B74                  #PSL$V_PRVMOD,-
       04 A2    04    0B75                  #PSL$S_CURMOD*2,4(R2)           ; set current and previous mode to super
          50    01 D0 0B78          MOVL    S^#SS$_NORMAL,R0                ; set correct return code
                   04 0B7B          RET                                     ; enter super mode
                      0B7C  20$:
                7E D4 0B7C          CLRL    -(SP)                           ; set up dummy PSL
       82'AF    6E FA 0B7E          CALLG   (SP),B^30$                      ; create initial call frame
                      0B82  30$:
                 0000 0B82          .WORD   ^M<>                            ; entry mask
                00 DD 0B84          PUSHL   #0                              ; push a dummy parameter
     0E16'CF    01 FB 0B86          CALLS   #1,W^REG_SAVE                   ; save the registers
                      0B8B          $DCLCMH_S @HANDLER_PC,W^PRVHND1,#0 ; set real handler
                      0B9B          FAIL_CHECKNP SS$_NORMAL                 ; check for success
                01 DD 0B9B                  PUSHL   #SS$_NORMAL
     1443'CF    01 FB 0B9D                  CALLS   #1,W^REG_CHECKNP
      03C00000 8F DD  0BA2          PUSHL   #<<PSL$C_USER@PSL$V_CURMOD>-
                      0BA8                  !<PSL$C_USER@PSL$V_PRVMOD>>; set return to user
          A5 AF    DD 0BA8          PUSHL   RETURN_PC                       ; set the return PC
                   02 0BAB          REI                                     ; return
```

```
                               0BAC  .SBTTL SUPER_MODE
                               0BAC  ;++
                               0BAC  ; FUNCTIONAL DESCRIPTION:
                               0BAC  ;       Routine to handle the CHMS instructions.
                               0BAC  ;
                               0BAC  ; CALLING SEQUENCE:
                               0BAC  ;       CHMS    #N
                               0BAC  ;
                               0BAC  ; INPUT PARAMETERS:
                               0BAC  ;          SP=>  CHMS parameter
                               0BAC  ;                PC
                               0BAC  ;                PSL
                               0BAC  ;
                               0BAC  ;          The CHMS parameter can be one of the following:
                               0BAC  ;
                               0BAC  ;                1 = execute $CREMBX (_S and _G form)
                               0BAC  ;                2 = execute a $DCLCMH_S to reset the CHMS handler to DCL
                               0BAC  ;
                               0BAC  ; OUTPUT PARAMETERS:
                               0BAC  ;       NONE
                               0BAC  ;--
                               0BAC
                               0BAC  SUPER_MODE:
               50    8E    D0  0BAC          MOVL    (SP)+,R0                          ; get CHM parameter off the stack
      02       01    50    8F  0BAF          CASEB   R0,#1,#2                          ; do the right thing
                               0BB3  10$:
                        0004'  0BB3          .WORD   20$-10$
                        006C'  0BB5          .WORD   A30-10$
                               0BB7  20$:
                     00    DD  0BB7          PUSHL   #0
         0E16'CF     01    FB  0BB9          CALLS   #1,W^REG_SAVE                     ; save the registers
                               0BBE          $CREMBX_S PRMFLG=#0,-
                               0BBE                  CHAN  =W^MBCHANS,-
                               0BBE                  MAXMSG=#MBBUF,-
                               0BBE                  BUFQUO=#BUF_SIZ_S,-
                               0BBE                  ACMODE=#PSL$C_SUPER,-
                               0BBE                  LOGNAM=W^LOGNAMS                  ; try _s form
                               0BDF          FAIL_CHECKNP SS$_NORMAL                   ; check success
                     01    DD  0BDF                  PUSHL   #SS$_NORMAL
         1443'CF     01    FB  0BE1                  CALLS   #1,W^REG_CHECKNP
               1FBB'CF     D4  0BE6          CLRL    W^CRE+CREMBX$_PRMFLG              ; make it temporary
  1FC3'CF    00000400  8F  D0  0BEA          MOVL    #MBBUF,W^CRE+CREMBX$_MAXMSG       ; set max. message size
  1FC7'CF    00000400  8F  D0  0BF3          MOVL    #BUF_SIZ_G,W^CRE+CREMBX$_BUFQUO   ; set the buffer quota
         1FCF'CF     02    D0  0BFC          MOVL    #PSL$C_SUPER,W^CRE+CREMBX$_ACMODE ; set the access mode
    1FD3'CF     0154'CF    DE  0C01          MOVAL   W^LOGNAMG,W^CRE+CREMBX$_LOGNAM    ; set the lognam
                               0C08          $CREMBX_G W^CRE                           ; try _g form
                               0C11          FAIL_CHECKNP SS$_NORMAL                   ; check success
                     01    DD  0C11                  PUSHL   #SS$_NORMAL
         1443'CF     01    FB  0C13                  CALLS   #1,W^REG_CHECKNP
         0CFC'CF     00    FB  0C18          CALLS   #0,W^VERIFY_MBX                   ; read,write,verify,delete MBX
                     1D    11  0C1D          BRB     A50                               ; get back to user mode
                               0C1F  A30:
    2697'CF     0119'CF    DE  0C1F          MOVAL   W^DCLCMH,W^SERV_NAME              ; set service name pointer
                               0C26          $DCLCMH_S @PRVHND1,,#0                    ; reset the CHMS handler for DCL
                               0C35          FAIL_CHECK SS$_NORMAL                     ; check for success
                     01    DD  0C35                  PUSHL   #SS$_NORMAL
         0E20'CF     01    FB  0C37                  CALLS   #1,W^REG_CHECK
```

```
       0C3C  1078 A50:
 02    0C3C  1079          REI                                   ; return to user mode
```

```
                           0C3D          .SBTTL  ERLBUF_DUMP
                           0C3D  ;++
                           0C3D  ; FUNCTIONAL DESCRIPTION:
                           0C3D  ;       Routine to check for errors in the error log buffer and
                           0C3D  ;       report any that are there.
                           0C3D  ;
                           0C3D  ; CALLING SEQUENCE:
                           0C3D  ;       CALLS #0,W^ERLBUF_DUMP
                           0C3D  ;
                           0C3D  ; INPUT PARAMETERS:
                           0C3D  ;       FLAG bit 0 = 0 for no errors logged
                           0C3D  ;       FLAG bit 0 = 1 for errors logged
                           0C3D  ;       if errors logged then buffer ERLB must contain legal format errors
                           0C3D  ;
                           0C3D  ; OUTPUT PARAMETERS:
                           0C3D  ;       NONE
                           0C3D  ;
                           0C3D  ;--
                           0C3D
                           0C3D  ERLBUF_DUMP:
                     001C  0C3D          .WORD   ^M<R2,R3,R4>
       2A 0E62'CF     E9   0C3F          BLBC    FLAG,30$               ; br if no errors to report
    52    0E67'CF     DE   0C44          MOVAL   ERLB,R2                ; set up buffer pointer
                           0C49  10$:
              62      D5   0C49          TSTL    (R2)                   ; any more errors?
              21      13   0C4B          BEQL    30$                    ; br if not
       2697'CF  82    D0   0C4D          MOVL    (R2)+,W^SERV_NAME ; reset service name
       2063'CF  82    D0   0C52          MOVL    (R2)+,W^CURRENT_TC ; reset step #
       2598'CF  82    D0   0C57          MOVL    (R2)+,W^MODE           ; reset the mode
           53   82    9A   0C5C          MOVZBL  (R2)+,R3               ; get the longword count
           54   53    D0   0C5F          MOVL    R3,R4                  ; and save it
                           0C62  20$:
                82    DD   0C62          PUSHL   (R2)+                  ; push a parameter
           FB   53    F5   0C64          SOBGTR  R3,20$                 ; and push them all
       1501'CF  54    FB   0C67          CALLS   R4,W^PRINT_FAIL        ; print the failure
                DB    11   0C6C          BRB     10$                    ; do the next one
                           0C6E  30$:
 0E63'CF   0E67'CF    DE   0C6E          MOVAL   W^ERLB,W^ELBP          ; reset the buffer pointer
           0E67'CF    D4   0C75          CLRL    W^ERLB                 ; set fresh terminater
                      04   0C79          RET                            ; bail out
```

```
                          0C7A          .SBTTL BUF_CHECK
                          0C7A  ;++
                          0C7A  ; FUNCTIONAL DESCRIPTION:
                          0C7A  ;       Routine to check the contents of a buffer against known good
                          0C7A  ;       data.
                          0C7A  ;
                          0C7A  ; CALLING SEQUENCE:
                          0C7A  ;       CALLS #0,W^BUF_CHECK            ; check buffer
                          0C7A  ;
                          0C7A  ; INPUT PARAMETERS:
                          0C7A  ;       R6 = buffer address
                          0C7A  ;       R7 = good data address
                          0C7A  ;       R8 = byte count
                          0C7A  ;
                          0C7A  ; OUTPUT PARAMETERS:
                          0C7A  ;       NONE
                          0C7A  ;
                          0C7A  ;--
                          0C7A
                          0C7A  BUF_CHECK:
                    03FC  0C7A          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9>
              59   56  D0  0C7C          MOVL    R6,R9                          ; save a copy of the buffer address
         66   67   58  29  0C7F          CMPC3   R8,(R7),(R6)                   ; check the buffer
              76       13  0C83          BEQL    10$                            ; br if good
0017'CF       53   59  C3  0C85          SUBL3   R9,R3,W^ARGLST1+8              ; get buffer offset
   001B'CF         61  9A  0C8B          MOVZBL  (R1),W^ARGLST1+12              ; get the good data
                          0C90          $GETCHN_S CHAN  =W^MBCHAN,-
                          0C90                    PRIBUF=W^PB                  ; get the unit number
0013'CF     25C8'CF    3C  0CA6          MOVZWL  W^PB+DIB$W_UNIT+8,W^ARGLST1+4  ; get the unit number
   001F'CF         63  9A  0CAD          MOVZBL  (R3),W^ARG[ST1+16              ; get the bad data
                          0CB2          $GETMSG_S MSGID=#UETP$_DATAER,-
                          0CB2                    MSGLEN=W^ML,-
                          0CB2                    BUFADR=W^CTRSTR,-
                          0CB2                    FLAGS =#1                    ; get the ctrstr
                          0CCB          $FAOL_S W^CTRSTR,W^ML,W^GETBUF,W^ARGLST1 ; make it readable
   0DF5'CF         00  FB  0CE2          CALLS   #0,W^STORE_STEP                ; store error info.
   52          0E63'CF D0  0CE7          MOVL    W^ELBP,R2                      ; get error log buffer pointer
              82   01  90  0CEC          MOVB    #1,(R2)+                       ; save longword count
   82          258C'CF DE  0CEF          MOVAL   W^ML,(R2)+                     ; push desc. address
                   62  D4  0CF4          CLRL    (R2)                           ; make a new terminator
   0E63'CF         52  D0  0CF6          MOVL    R2,W^ELBP                      ; reset buffer pointer
                          0CFB  10$:
                       04  0CFB          RET                                    ; return
```

```
                                 0CFC  1166 .SBTTL  VERIFY_MBX
                                 0CFC  1167 ;++
                                 0CFC  1168 ; FUNCTIONAL DESCRIPTION:
                                 0CFC  1169 ;        Routine to write to a mailbox, read the data back, verify the
                                 0CFC  1170 ;        results and delete the mailbox.
                                 0CFC  1171 ;
                                 0CFC  1172 ; CALLING SEQUENCE:
                                 0CFC  1173 ;        CALLS   #1,W^VERIFY_MBX
                                 0CFC  1174 ;
                                 0CFC  1175 ; INPUT PARAMTERS
                                 0CFC  1176 ;
                                 0CFC  1177 ; OUTPUT PARAMETERS:
                                 0CFC  1178 ;        NONE
                                 0CFC  1179 ;
                                 0CFC  1180 ;--
                                 0CFC  1181 VERIFY_MBX:
                           03FC  0CFC  1182         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9>       ; entry mask
                                 0CFE  1183         $QIO_S  CHAN=W^MBCHANS,-
                                 0CFE  1184                 FUNC=#IO$_WRITEVBLK,-
                                 0CFE  1185                 P1  =W^TEST_DATA,-
                                 0CFE  1186                 P2  =#BUF_SIZ_S                    ; write to the mailbox_S
0100 8F   00  2180'CF  00   2C  0D21  1187         MOVC5   #0,W^MBX_BUF,#0,#256,W^MBX_BUF  ; zero the MBX buffer
              2180'CF           0D2A
                                 0D2D  1188         $QIO_S  CHAN=W^MBCHANS,-
                                 0D2D  1189                 FUNC=#IO$_READVBLK,-
                                 0D2D  1190                 P1  =W^MBX_BUF,-
                                 0D2D  1191                 P2  =#BUF_SIZ_S                    ; read from the mailbox
          56  2180'CF       DE  0D50  1192         MOVAL   W^MBX_BUF,R6                       ; set the MBX buffer
          57  0365'CF       DE  0D55  1193         MOVAL   W^TEST_DATA,R7                     ; find the master data
       58  00000100 8F      D0  0D5A  1194         MOVL    #BUF_SIZ_S,R8
     20FC'CF   2100'CF      B0  0D61  1195         MOVW    W^MBCHANS,W^MBCHAN                 ; get the channel number
        FF0D CF      00     FB  0D68  1196         CALLS   #0,W^BUF_CHECK                     ; check the data
                                 0D6D  1197         $DASSGN_S CHAN=W^MBCHANS                   ; deassign the channel
                                 0D79  1198         $QIO_S  CHAN=W^MBCHANG,-
                                 0D79  1199                 FUNC=#IO$_WRITEVBLK,-
                                 0D79  1200                 P1  =W^TEST_DATA,-
                                 0D79  1201                 P2  =#BUF_SIZ_G                    ; write to the mailbox_G
0400 8F   00  2180'CF  00   2C  0D9C  1202         MOVC5   #0,W^MBX_BUF,#0,#1024,W^MBX_BUF ; zero the MBX buffer
              2180'CF           0DA5
                                 0DA8  1203         $QIO_S  CHAN=W^MBCHANG,-
                                 0DA8  1204                 FUNC=#IO$_READVBLK,-
                                 0DA8  1205                 P1  =W^MBX_BUF,-
                                 0DA8  1206                 P2  =#BUF_SIZ_G                    ; read from the mailbox
          56  2180'CF       DE  0DCB  1207         MOVAL   W^MBX_BUF,R6                       ; set the MBX buffer
          57  0365'CF       DE  0DD0  1208         MOVAL   W^TEST_DATA,R7                     ; find the master data
       58  00000400 8F      D0  0DD5  1209         MOVL    #BUF_SIZ_G,R8
     20FC'CF   20FE'CF      B0  0DDC  1210         MOVW    W^MBCHANG,W^MBCHAN                 ; get the channel number
        FE92 CF      00     FB  0DE3  1211         CALLS   #0,W^BUF_CHECK                     ; check the data
                                 0DE8  1212         $DASSGN_S CHAN=W^MBCHANG                   ; deassign the channel
                            04  0DF4  1213         RET
```

```
                        0DF5            .SBTTL STORE_STEP
                        0DF5    ;++
                        0DF5    ; FUNCTIONAL DESCRIPTION:
                        0DF5    ;       Routine to store step information in the error log buffer.
                        0DF5    ;
                        0DF5    ; CALLING SEQUENCE:
                        0DF5    ;       CALLS #0,W^STORE_STEP
                        0DF5    ;
                        0DF5    ; INPUT PARAMETERS:
                        0DF5    ;       ELBP = current errlog buffer pointer
                        0DF5    ;
                        0DF5    ; OUTPUT PARAMETERS:
                        0DF5    ;       FLAG = error logged flag
                        0DF5    ;
                        0DF5    ;--
                        0DF5
                        0DF5    STORE_STEP:
                  0004  0DF5            .WORD   ^M<R2>
   0E62'CF   01     88  0DF7            BISB2   #1,W^FLAG                       ; set the error logged flag
   52   0E63'CF     D0  0DFC            MOVL    W^ELBP,R2                       ; get errlog buf pntr
   82   2697'CF     D0  0E01            MOVL    W^SERV_NAME,(R2)+               ; save the service name
   82   2063'CF     D0  0E06            MOVL    W^CURRENT_TC,(R2)+              ; save the step number
   82   2598'CF     D0  0E0B            MOVL    W^MODE,(R2)+                    ; save the mode
   0E63'CF   52     D0  0E10            MOVL    R2,W^ELBP                       ; reset the errlog buf pntr
                    04  0E15            RET                                     ; return
```

```
                                0E16          .SBTTL REG_SAVE
                                0E16   ;++
                                0E16   ; FUNCTIONAL DESCRIPTION:
                                0E16   ;        Subroutine to save R2-R11 in the register save location.
                                0E16   ;
                                0E16   ; CALLING SEQUENCE:
                                0E16   ;        PUSHL   #0              ; save a dummy parameter
                                0E16   ;        CALLS   #1,W^REG_SAVE   ; save R2-R11
                                0E16   ;
                                0E16   ; INPUT PARAMETERS:
                                0E16   ;        NONE
                                0E16   ;
                                0E16   ; OUTPUT PARAMETERS:
                                0E16   ;        NONE
                                0E16   ;
                                0E16   ;--
                                0E16
                                0E16 REG_SAVE:
                          0FFC  0E16          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
  264F'CF   14 AD   28    28    0E18          MOVC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; save the registers in the program
                            04  0E1F          RET
```

```
                                   0E20          .SBTTL  REG_CHECK
                                   0E20    ;++
                                   0E20    ; FUNCTIONAL DESCRIPTION:
                                   0E20    ;        Subroutine to test R0 & R2-R11 for proper content after a service
                                   0E20    ;        execution. A snapshot is taken by the REG_SAVE routine at the
                                   0E20    ;        beginning of each step and this routine is executed after the
                                   0E20    ;        services have been executed.
                                   0E20    ;
                                   0E20    ; CALLING SEQUENCE:
                                   0E20    ;        PUSHL   #SS$_XXXXXX      ; push expected R0 contents
                                   0E20    ;        CALLS   #1,W^REG_CHECK   ; execute this routine
                                   0E20    ;
                                   0E20    ; INPUT PARAMETERS:
                                   0E20    ;        expected R0 contents on the stack
                                   0E20    ;
                                   0E20    ; OUTPUT PARAMETERS:
                                   0E20    ;        possible error messages printed using $PUTMSG
                                   0E20    ;
                                   0E20    ;--
                                   0E20
                                   0E20    REG_CHECK:
                            0FFC   0E20          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
            50     04 AC     D1    0E22          CMPL    4(AP),R0                            ; is this the right fail code?
                      0E     13    0E26          BEQL    10$                                 ; br if yes
                      50     DD    0E28          PUSHL   R0                                  ; push received data
                   04 AC     DD    0E2A          PUSHL   4(AP)                               ; push expected data
                 0138'CF     DF    0E2D          PUSHAL  W^EXP                               ; push the string variable
          1501'CF     03     FB    0E31          CALLS   #3,W^PRINT_FAIL                     ; print the error message
                                   0E36    10$:
   264F'CF    14 AD    28    29    0E36          CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA      ; check all but R0
                       22    13    0E3D          BEQL    20$                                 ; br if O.K.
   56    53    0000264F'8F   C3    0E3F          SUBL3   #REG_SAVE_AREA,R3,R6                ; calculate the register number
                 56    04    C6    0E47          DIVL2   #4,R6
           7E    56    02    81    0E4A          ADDB3   #^X2,R6,-(SP)                       ; set number past R0-R1 and save
                 51    03    CA    0E4E          BICL2   #3,R1                               ; backup to register boundrys
                 53    03    CA    0E51          BICL2   #3,R3
                       61    DD    0E54          PUSHL   (R1)                                ; push received data
                       63    DD    0E56          PUSHL   (R3)                                ; push expected data
                 263D'CF     DF    0E58          PUSHAL  W^REG                               ; set string pntr param.
          1501'CF     04     FB    0E5C          CALLS   #4,W^PRINT_FAIL                     ; print the error message
                                   0E61    20$:
                             04    0E61          RET
```

```
                            0E62          .SBTTL REG_CHECKNP
                            0E62   ;++
                            0E62   ; FUNCTIONAL DESCRIPTION:
                            0E62   ;        Subroutine to test R0 & R2-R11 for proper content after a service
                            0E62   ;        execution without printing it. A snapshot is taken by the REG_SAVE routine a
                            0E62   ;        beginning of each step and this routine is executed after the
                            0E62   ;        services have been executed. This routine collects the error
                            0E62   ;        information in buffer ERLB instead of printing it.
                            0E62   ;
                            0E62   ; CALLING SEQUENCE:
                            0E62   ;        PUSHL   #SS$_XXXXXX      ; push expected R0 contents
                            0E62   ;        CALLS   #1,W^REG_CHECK   ; execute this routine
                            0E62   ;
                            0E62   ; INPUT PARAMETERS:
                            0E62   ;        expected R0 contents on the stack
                            0E62   ;
                            0E62   ; OUTPUT PARAMETERS:
                            0E62   ;        possible error messages logged in buffer ERLB which are printed
                            0E62   ;        using routine ERLBUF_DUMP.
                            0E62   ;
                            0E62   ;        Error packets are in the following form:
                            0E62   ;
                            0E62   ;                   !-----------------!
                            0E62   ;                   !Service name pntr!
                            0E62   ;                   !-----------------!
                            0E62   ;                   !     Step #      !
                            0E62   ;                   !-----------------!
                            0E62   ;                   !Mode name pointer!
                            0E62   ;                   !-----------------!
                            0E62   ;                   !           !     ! long word count
                            0E62   ;                   !-----------------!
                            0E62   ;                   !\/\/\/\/\/\/\/\/\! 3-4 parameter long words
                            0E62   ;
                            0E62   ;--
                            0E62
                            0E62   FLAG:
                    00      0E62          .BYTE 0                      ; error flags are BIT0 = 0 means no errors in the bu
                            0E63                                       ;                 BIT0 = 1 means errors in the buffe
                            0E63   ELBP:
             00000E67'      0E63          .ADDRESS ERLB                ; error log buffer pointer
                            0E67   ERLB:
              00001443      0E67          .BLKB   1500                 ; error log buffer
                            1443   ;
                            1443   REG_CHECKNP:
                  0FFC      1443          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
          50    04 AC D1    1445          CMPL    4(AP),R0             ; is this the right fail code
                   2E 13    1449          BEQL    10$                  ; br if yes
        F9A5 CF    00 FB    144B          CALLS   #0,W^STORE_STEP      ; store step information
        52   FA0F CF D0     1450          MOVL    ELBP,R2              ; get the current error log pointer
                82 03 90    1455          MOVB    #3,(R2)+             ; save the long word count
                82 50 D0    1458          MOVL    R0,(R2)+             ; save received status
             82 04 AC D0    145B          MOVL    4(AP),(R2)+          ; save expected status
        82   0138'CF DE     145F          MOVAL   W^EXP,(R2)+          ; save the string variable
                   62 D4    1464          CLRL    (R2)                 ; set the terminator
        F9F8 CF    52 D0    1466          MOVL    R2,ELBP              ; reset the buffer pointer
    269F'CF  076B'CF DE     146B          MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR ; set failure message address
 2594'CF   03   00 02 F0    1472          INSV    #ERROR,#0,#3,W^MOD_MSG_CODE ; set severity code
```

```
                                1479  1363 10$:
  264F'CF    14 AD    28    29  1479  1364          CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA ; check all but R0 and R1
                      3C    13  1480  1365          BEQL    20$                  ; br if OK
         F96E CF      00    FB  1482  1366          CALLS #0,W^STORE_STEP        ; store step information
         52   F9D8 CF       D0  1487  1367          MOVL    ELBP,R2              ; get current error log buf pointer
              82      04    90  148C  1368          MOVB    S^#4,(R2)+           ; set longword count
          0000264F'8F       C3  148F  1369          SUBL3   #REG_SAVE_AREA,-
              56      53        1495  1370                  R3,R6                ; calc reg number
              56      04    C6  1497  1371          DIVL2   S^#4,R6              ; make it a longword count
         82   56      02    C1  149A  1372          ADDL3   S^#2,R6,(R2)+        ; correct for R0-R1 and save
              82      61    D0  149E  1373          MOVL    (R1),(R2)+           ; save received results
              82      63    D0  14A1  1374          MOVL    (R3),(R2)+           ; save expected results
         82   263D'CF       DE  14A4  1375          MOVAL   W^REG,(R2)+          ; save string variable
                      62    D4  14A9  1376          CLRL    (R2)                 ; set the terminator
         F9B3 CF      52    D0  14AB  1377          MOVL    R2,ELBP              ; reset the buffer pointer
     269F'CF  076B'CF       DE  14B0  1378          MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR     ; set failure message address
 2594'CF  03   00     02    F0  14B7  1379          INSV    #ERROR,#0,#3,W^MOD_MSG_CODE    ; set severity code
                                14BE  1380 20$:
                            04  14BE  1381          RET                          ; bail out
```

```
                              14BF         .SBTTL NONSUB_SSE
                              14BF   ;++
                              14BF   ; FUNCTIONAL DESCRIPTION:
                              14BF   ;       Subroutine to report the failure of a system service which is not the
                              14BF   ;       subject system service.
                              14BF   ;
                              14BF   ; CALLING SEQUENCE:
                              14BF   ;       PUSHL   R0                ; Save the failure status
                              14BF   ;       CALLS   #1,NONSUB_SSE     ; Print the failure message
                              14BF   ;
                              14BF   ; INPUT PARAMETERS:
                              14BF   ;       4(AP) = Status code of failing system service
                              14BF   ;
                              14BF   ; OUTPUT PARAMETERS:
                              14BF   ;       NONE
                              14BF   ;
                              14BF   ;--
                              14BF   NONSUB_SSE:
                        0FFC  14BF         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                              14C1         $GETMSG_S MSGID  = 4(AP),-
                              14C1                   MSGLEN = BUFFER_PTR,-
                              14C1                   BUFADR = BUFFER_PTR,-
                              14C1                   FLAGS  = #1                    ; Get just the text of the message
                              14D7         $FAO_S  CTRSTR = NSSSF,-
                              14D7                 OUTLEN = MESSAGE1L,-
                              14D7                 OUTBUF = MSG1L,-
                              14D7                 P1     = #BUFFER_PTR             ; Format the failure message
    27BB'CF   0100 8F   B0    14F0         MOVW    #TEXT_BUFFER,BUFFER_PTR          ; Reset the descriptor length
            27B3'CF     DF    14F7         PUSHAL  MESSAGE1L                        ; Push the string address
       1501'CF   01     FB    14FB         CALLS   #1,PRINT_FAIL                    ; Print the failure
                        04    1500         RET
```

```
                                 1501        .SBTTL  PRINT_FAIL
                                 1501  ;++
                                 1501  ; FUNCTIONAL DESCRIPTION:
                                 1501  ;       Subroutine to report failures using $PUTMSG
                                 1501  ;
                                 1501  ; CALLING SEQUENCE:
                                 1501  ; Mode  #1       PUSHL EXPECTED  Mode    #2      PUSHL REG_NUMBER
                                 1501  ;               PUSHL RECEIVED                  PUSHL EXPECTED
                                 1501  ;               PUSHAL STRING_VAR               PUSHL RECEIVED
                                 1501  ;               CALLS #3,W^PRINT_FAIL           PUSHAL STRING_VAR
                                 1501  ;                                               CALLS #4,W^PRINT_FAIL
                                 1501  ; Mode  #3       PUSHAL STRING_VAR
                                 1501  ;               CALLS #1,W^PRINT_FAIL
                                 1501  ;
                                 1501  ; INPUT PARAMETERS:
                                 1501  ;       listed above
                                 1501  ;
                                 1501  ; OUTPUT PARAMETERS:
                                 1501  ;       an error message is printed using $PUTMSG
                                 1501  ;
                                 1501  ;--
                                 1501
                                 1501  PRINT_FAIL:
                          003C   1501          .WORD   ^M<R2,R3,R4,R5>
                                 1503          $FAO_S  W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
                                 1524          $PUTMSG_S W^MSGVEC                      ; print the message
              04     6C     91   1535          CMPB    (AP),#4                         ; is this a register message?
                     26     13   1538          BEQL    10$                             ; br if yes
              01     6C     91   153A          CMPB    (AP),#1                         ; is this just a message?
                     48     13   153D          BEQL    20$                             ; br if yes
                                 153F          $FAO_S  W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
                     40     11   155E          BRB     30$                             ; goto output message
                                 1560  10$:
                                 1560          $FAO_S  W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
                     19     11   1585          BRB     30$                             ; goto output message
                                 1587  20$:
        25B0'CF   04 AC     D0   1587          MOVL    4(AP),W^MSGVEC1+12              ; save string address
                                 158D          $PUTMSG_S W^MSGVEC1                     ; print the message
                     11     11   159E          BRB     40$                             ; skip the other message
                                 15A0  30$:
                                 15A0          $PUTMSG_S W^MSGVEC                      ; print the message
                                 15B1  40$:
           15C5'CF   00     FB   15B1          CALLS   #0,W^MODE_ID                    ; identify the mode
      269F'CF   076B'CF     DE   15B6          MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR      ; set failure message address
  2594'CF   03   00   02    F0   15BD          INSV    #ERROR,#0,#3,W^MOD_MSG_CODE     ; set severity code
                            04   15C4          RET
```

```
              15C5              .SBTTL  MODE_ID
              15C5      ;++
              15C5      ; FUNCTIONAL DESCRIPTION:
              15C5      ;       Subroutine to identify the mode that an exit handler is in.
              15C5      ;
              15C5      ; CALLING SEQUENCE:
              15C5      ;       CALLS   #0,W^MODE_ID
              15C5      ;
              15C5      ; INPUT PARAMETERS:
              15C5      ;       MODE contains an address pointing to an ascii string desc.
              15C5      ;       of the current CPU mode.
              15C5      ;
              15C5      ; OUTPUT PARAMETERS:
              15C5      ;       NONE
              15C5      ;
              15C5      ;--
              15C5
              15C5      MODE_ID:
        003C  15C5              .WORD   ^M<R2,R3,R4,R5>
              15C7              $FAO_S  W^CS5,W^MESSAGEL,W^MSGL,MODE ; format the error message
              15DE              $PUTMSG_S W^MSGVEC                  ; print the mode message
          04  15EF              RET
```

```
                  15F0  MOD_MSG_PRINT:
                  15F0  ;
                  15F0  ;   ***************************************************************
                  15F0  ;   *                                                             *
                  15F0  ;   *  PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES    *
                  15F0  ;   *       (USING THE PUTMSG MACRO).                             *
                  15F0  ;   *                                                             *
                  15F0  ;   ***************************************************************
                  15F0  ;
                  15F0          PUTMSG  <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> ; PRINT MSG
              05  1605          RSB                             ; ... AND RETURN TO CALLER
                  1606  ;
                  1606  CHMRTN:
                  1606  ; ******************************************************************
                  1606  ; *                                                                *
                  1606  ; *     CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER    *
                  1606  ; *     A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED        *
                  1606  ; *     BY THE MODE MACRO ('TO' OPTION).  IT MERELY DOES           *
                  1606  ; *     A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS          *
                  1606  ; *     THE EFFECT OF RETURNING TO THE END OF THE MODE             *
                  1606  ; *     MACRO EXPANSION.                                           *
                  1606  ; *                                                                *
                  1606  ; ******************************************************************
                  1606  ;
            0000  1606          .WORD   0                       ; ENTRY MASK
    1FB3'DF   17  1608          JMP     @CHM_CONT               ; RETURN TO MODE MACRO IN NEW MODE
                  160C  ;
                  160C  ; *     RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
                  160C  ;
                  160C          .END    SATSSS07
```

```
$$ARGS              = 00000001
$$T1                = 00000004
$$T2                = 00000004
A                   = 00000201
A30                   00000C1F R     04
A50                   00000C3C R     04
ARGLST                00000000 R     02
ARGLST1               0000000F R     03
BUF                   00000023 R     03
BUF1                  000026B3 R     03
BUFFER                00000073 R     03
BUFFER_PTR            000027BB R     03
BUF_CHECK             00000C7A R     04
BUF_SIZ_G           = 00000400
BUF_SIZ_S           = 00000100
CHANNEL_ZERO          000001CB R     02
CHMRTN                00001606 R     04
CHM_CONT              00001FB3 R     03
CRE                   00001FB7 R     03
CREATED_FLAG          0000000E R     03
CREMBX                00000008 R     02
CREMBX$_ACMODE      = 00000018
CREMBX$_BUFQUO      = 00000010
CREMBX$_CHAN        = 00000008
CREMBX$_LOGNAM      = 0000001C
CREMBX$_MAXMSG      = 0000000C
CREMBX$_NARGS       = 00000007
CREMBX$_PRMFLG      = 00000004
CREMBX$_PROMSK      = 00000014
CS1                   0000000F R     02
CS2                   00000041 R     02
CS3                   0000006E R     02
CS4                   000000A1 R     02
CS5                   000000C7 R     02
CTL$GW_CHINDX         ********   X   04
CTRSTR                00001FD7 R     03
CURRENT_TC            00002063 R     03
DCLCMH                00000119 R     02
DEL                   00002068 R     03
DELMBX                00000120 R     02
DELMBX$_CHAN        = 00000004
DELMBX$_NARGS       = 00000001
DIB$K_LENGTH        = 00000074
DIB$W_UNIT          = 0000000C
DIB$W_VPROT         = 00000018
DISABLE             = 00000000
DSC$K_CLASS_S       = 00000001
DSC$K_DTYPE_T       = 0000000E
D_EQUIV_NAME          000002F5 R     02
D_LOGIC_NAME          000002ED R     02
D_MBX_LOGIC_NAME      000002FD R     02
ELBP                  00000E63 R     04
EM                    00000127 R     02
ENABLE              = 00000001
EQUIV_NAME            00000009 R     03
ERLB                  00000E67 R     04
ERLBUF_DUMP           00000C3D R     04
ERROR               = 00000002
EXE$C_CMSTKSZ         ********   X   04
EXP                   00000138 R     02
FLAG                  00000E62 R     04
GETBUF                00002070 R     03
GRP_PRV_MASK          000001C3 R     02
HANDLER_PC            00000B54 R     04
HIMSG               = 00002000
INFO                = 00000003
IO$_READVBLK        = 00000031
IO$_WRITEVBLK       = 00000030
KM                    00000146 R     02
LEN_00_DESCR          000001CD R     02
LEN_256_DESCR         000001D5 R     02
LEN_256_NAME          000001E5 R     02
LEN_63_DESCR          000001DD R     02
LIB$SIGNAL            ********   X   04
LOGIC_NAME            00000004 R     03
LOGNAMG               00000154 R     02
LOGNAMG1              00000164 R     02
LOGNAMS               00000175 R     02
LOGNAMS1              00000185 R     02
LOMSG               = 00000001
MBA                   00000305 R     02
MBBUF               = 00000400
MBCHAN                000020FC R     03
MBCHAND               00002102 R     03
MBCHANG               000020FE R     03
MBCHANS               00002100 R     03
MBCHAR                00002104 R     03
MBCHR                 0000210C R     03
MBX_BUF               00002180 R     03
MBX_LOGIC_NAME        00000000 R     03
MESSAGE1L             000027B3 R     03
MESSAGEL              00002580 R     03
ML                    0000258C R     03
MODE                  00002598 R     03
MODE_ID               000015C5 R     04
MOD_MSG_CODE          00002594 R     03
MOD_MSG_PRINT         000015F0 R     04
MSGT                  00000310 R     02
MSG1L                 000026AB R     03
MSGL                  0000259C R     03
MSGVEC                0000033A R     02
MSGVEC1               000025A4 R     03
NONSUB_SSE            000014BF R     04
NSSSF                 000000DC R     02
PB                    000025B4 R     03
PR$_USP             = 00000003
PRINT_FAIL            00001501 R     04
PRIVMASK              00002630 R     03
PRM_PRV_MASK          000001B3 R     02
PROT                  0000034A R     02
PRV$V_GRPNAM        = 00000003
PRV$V_PRMMBX        = 0000000B
PRV$V_TMPMBX        = 0000000F
PRVHND1               00002638 R     03
```

```
PRVPRT                 0000263C R    03
PSL$C_EXEC           = 00000001
PSL$C_KERNEL         = 00000000
PSL$C_SUPER          = 00000002
PSL$C_USER           = 00000003
PSL$S_CURMOD         = 00000002
PSL$V_CURMOD         = 00000018
PSL$V_PRVMOD         = 00000016
REG                    0000263D R    03
REGNUM                 0000268B R    03
REG_CHECK              00000E20 R    04
REG_CHECKNP            00001443 R    04
REG_SAVE               00000E16 R    04
REG_SAVE_AREA          0000264F R    03
RETADR                 0000268F R    03
RETURN_PC              00000B50 R    04
SATSSS07               00000000 RG   04
SERV_NAME              00002697 R    03
SETUP_SUPER            00000B58 R    04
SEVERE               = 00000004
SF$L_SAVE_FP         = 0000000C
SF$L_SAVE_PC         = 00000010
SHR$K_SHRDEF         = 00000001
SHR$_TEXT            = 00001130
SKIP_DELETE            00000887 R    04
SM                     00000358 R    02
SS$_ACCVIO           = 0000000C
SS$_DEVNOTMBX        = 00000074
SS$_IVCHAN           = 0000013C
SS$_IVLOGNAM         = 00000154
SS$_NOPRIV           = 00000024
SS$_NORMAL           = 00000001
SS$_TOOMANYLNAM      = 00000374
STATUS                 0000269B R    03
STEP                 = 00000017
STORE_STEP             00000DF5 R    04
STP0                   0000003D R    04
STP1                   000000AA R    04
STP10                  00000775 R    04
STP11                  00000799 R    04
STP12                  000007C7 R    04
STP13                  000007FD R    04
STP14                  00000827 R    04
STP15                  00000868 R    04
STP16                  00000887 R    04
STP17                  000008BC R    04
STP18                  00000906 R    04
STP19                  0000099D R    04
STP2                   000001CD R    04
STP20                  000009D2 R    04
STP21                  00000A3B R    04
STP22                  00000A8A R    04
STP23                  00000ACB R    04
STP3                   00000312 R    04
STP4                   0000047E R    04
STP5                   00000568 R    04
STP6                   000005A3 R    04
STP7                   00000638 R    04
STP8                   000006CD R    04
STP9                   0000074B R    04
STS$V_INHIB_MSG      = 0000001C
SUCCESS              = 00000001
SUPER_MODE             00000BAC R    04
SYS$ASSIGN             ******** GX   04
SYS$CMEXEC             ******** GX   04
SYS$CMKRNL             ******** GX   04
SYS$CRELOG             ******** GX   04
SYS$CREMBX             ******** GX   04
SYS$DASSGN             ******** GX   04
SYS$DCLCMH             ******** GX   04
SYS$DELMBX             ******** GX   04
SYS$EXIT               ******** GX   04
SYS$FAO                ********  X   04
SYS$FAOL               ******** GX   04
SYS$GETCHN             ******** GX   04
SYS$GETMSG             ******** GX   04
SYS$HIBER              ******** GX   04
SYS$PUTMSG             ******** GX   04
SYS$QIO                ******** GX   04
SYS$SETPRN             ******** GX   04
SYS$SETPRV             ******** GX   04
SYS$WAKE               ******** GX   04
SYS_DEV                00000196 R    02
TEST_DATA              00000365 R    02
TEST_MOD_BEGIN         00000765 R    02
TEST_MOD_FAIL          0000076B R    02
TEST_MOD_NAME          00000772 R    02
TEST_MOD_NAME_D        0000077B R    02
TEST_MOD_SUCC          0000078B R    02
TEXT_BUFFER          = 00000100
TMD_ADDR               0000269F R    03
TMN_ADDR               000026A3 R    03
TMP_PRM_PRV_MASK       000001BB R    02
TMP_PRV_MASK           000001AB R    02
TPID                   000026A7 R    03
UETP$_DATAER         = 00748010
UETP$_SATSMS         = 007480D9
UETP$_TEXT           = 00741133
UM                     00000796 R    02
VERIFY_MBX             00000CFC R    04
WARNING              = 00000000
ZERO_ADDR_DESCR        000002E5 R    02
```

```
                                  +----------------+
                                  ! Psect synopsis !
                                  +----------------+

PSECT name                Allocation          PSECT No.  Attributes
----------                ----------          ---------  ----------
.  ABS  .                 00000000  (     0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
$ABS$                     00000000  (     0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
RODATA                    000007A2  (  1954.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD  NOWRT NOVEC LONG
RWDATA                    000027C3  ( 10179.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD    WRT NOVEC LONG
SATSSS07                  0000160C  (  5644.)  04 (  4.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD    WRT NOVEC LONG

                                  +------------------------+
                                  ! Performance indicators !
                                  +------------------------+

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     31    00:00:00.10    00:00:00.35
Command processing                108    00:00:00.66    00:00:02.28
Pass 1                            609    00:00:25.28    00:00:36.31
Symbol table sort                   0    00:00:02.46    00:00:02.79
Pass 2                            309    00:00:06.60    00:00:08.51
Symbol table output                26    00:00:00.19    00:00:00.19
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             1087    00:00:35.32    00:00:50.46

The working set limit was 1950 pages.
168419 bytes (329 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1495 non-local and 38 local symbols.
1515 source lines were read in Pass 1, producing 41 object records in Pass 2.
61 pages of virtual memory were used to define 57 macros.

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

Macro library name                         Macros defined
------------------                         --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2                      43
_$255$DUA28:[SHRLIB]UETP.MLB;1                         11
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                          0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                       0
TOTALS (all libraries)                                 54

1756 GETS were required to define 54 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SATSSS07/OBJ=OBJ$:SATSSS07 MSRC$:SATSSS07/UPDATE=(ENH$:SATSSS07)+EXECML$/LIB+SHRLIB$:UETP/LIB
```

0421 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SATSSS08
LIS

SATSSS05
LIS

SATSSS22
LIS

SATSSS07
LIS

SATSSS01
LIS