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

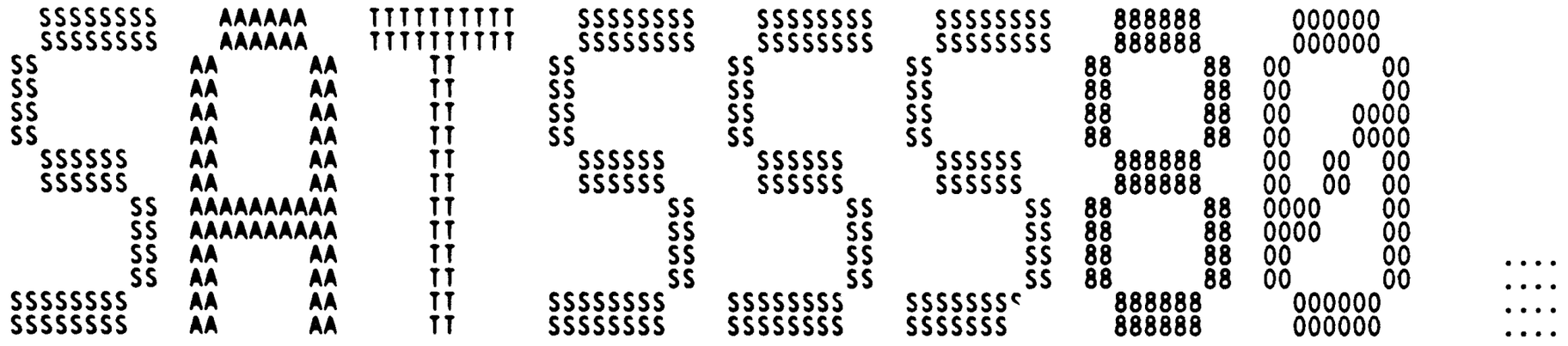
SSSSSSSS AAAAAA TTTTTTTTTT SSSSSSSS SSSSSSSS SSSSSSSS 888888 000000
SSSSSSSS AAAAAA TTTTTTTTTT SSSSSSSS SSSSSSSS SSSSSSSS 888888 000000
SS AA AA TT SS SS SS 88 88 00 00
SS AA AA TT SS SS SS 88 88 00 00
SS AA AA TT SS SS SS 88 88 00 0000
SS AA AA TT SS SS SS 88 88 00 0000
SSSSSS AA AA TT SSSSSS SSSSSS SSSSSS 888888 00 00 00
SSSSSS AA AA TT SSSSSS SSSSSS SSSSSS 888888 00 00 00
SS AAAAAAAAAA TT SS SS SS 88 88 0000 00
SS AAAAAAAAAA TT SS SS SS 88 88 0000 00
SS AA AA TT SS SS SS 88 88 00 00
SS AA AA TT SS SS SS 88 88 00 00
SSSSSSSS AA AA TT SSSSSSSS SSSSSSSS SSSSSSS 888888 000000
SSSSSSSS AA AA TT SSSSSSSS SSSSSSSS SSSSSSS 888888 000000
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
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS

Table of contents

```
0000     1          .TITLE  SATSSS80 - SATS SYSTEM SERVICE TESTS  (SUCC S.C.)
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;**********************************************************************
0000     6  ;*                                                                    *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
0000     9  ;*  ALL RIGHTS RESERVED.                                              *
0000    10  ;*                                                                    *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
0000    16  ;*  TRANSFERRED.                                                       *
0000    17  ;*                                                                    *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0000    20  ;*  CORPORATION.                                                       *
0000    21  ;*                                                                    *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
0000    24  ;*                                                                    *
0000    25  ;*                                                                    *
0000    26  ;**********************************************************************
0000    27  ;
0000    28
0000    29  ;++
0000    30  ; FACILITY:     SATS SYSTEM SERVICE TESTS
0000    31  ;
0000    32  ; ABSTRACT:     The SATSSS80 module tests the execution of the following
0000    33  ;               VMS system services:
0000    34  ;
0000    35  ;               $PURGWS
0000    36  ;
0000    37  ; ENVIRONMENT:  User mode image.
0000    38  ;               Needs CMKRNL privilege and dynamically acquires other
0000    39  ;               privileges, as needed.
0000    40  ;
0000    41  ; AUTHOR: Larry D. Jones,                CREATION DATE: JULY, 1979
0000    42  ;
0000    43  ; MODIFIED BY:
0000    44  ;
0000    45  ;       V03-002 KDM0002         Kathleen D. Morse       28-Jun-1982
0000    46  ;               Added $PRVDEF.
0000    47  ;
0000    48  ;**
0000    49  ;--
```

```
                     0000     51          .SBTTL  DECLARATIONS
                     0000     52 ;
                     0000     53 ; MACRO LIBRARY CALLS
                     0000     54 ;
                     0000     55          .LIBRARY /SYS$LIBRARY:STARLET.MLB/
                     0000     56          $JPIDEF                         ; GETJPI definitions
                     0000     57          $PHDDEF                         ; process header definitions
                     0000     58          $PRVDEF                         ; Privilege bit definitions
                     0000     59          $SHR_MESSAGES UETP,116,<<TEXT,INFO>> ; UETP$_TEXT definition
                     0000     60          $SFDEF                          ; stack frame definitions
                     0000     61          $SSDEF                          ; system status code definitions
                     0000     62          $STSDEF                         ; STS definitions
                     0000     63          $UETPDEF                        ; UETP message definitions
                     0000     64 ;
                     0000     65 ; Equated symbols
                     0000     66 ;
00000000             0000     67 WARNING          = 0                     ; warning severity value for msgs
00000001             0000     68 SUCCESS          = 1                     ; success      "       "   "   "
00000002             0000     69 ERROR            = 2                     ; error        "       "   "   "
00000003             0000     70 INFO             = 3                     ; information  "       "   "   "
00000004             0000     71 SEVERE           = 4                     ; fatal        "       "   "   "
                     0000     72 ;
                     0000     73 ;
                     0000     74 ; MACROS
                     0000     75 ;
```

```
                                    00000000     77          .PSECT  RODATA,RD,NOWRT,NOEXE,PAGE
                                        0000     78 ;
                                        0000     79 TEST_MOD_NAME:
          30 38 53 53 53 54 41 53 00'   0000     80          .ASCIC  /SATSSS80/              ; needed for SATSMS message
                                        08  0000
                                        0009     81 TEST_MOD_NAME_D:
53 53 53 54 41 53 00000011'010E000C'    0009     82          .ASCID /SATSSS80/               ; module name
                                  30 38 0017
                                        0019     83 TEST_MOD_BEGIN:                          ; start end and fail messages
                   6E 75 67 65 62 00'   0019     84          .ASCIC  /begun/
                                        05  0019
                                        001F     85 TEST_MOD_SUCC:
   6C 75 66 73 73 65 63 63 75 73 00'    001F     86          .ASCIC  /successful/
                                        0A  001F
                                        002A     87 TEST_MOD_FAIL:
                64 65 6C 69 61 66 00'   002A     88          .ASCIC  /failed/
                                        06  002A
                                        0031     89 CS1:                                     ; failure messages
21 20 74 73 65 54 00000039'010E0000'    0031     90          .ASCID  \Test !AC service name !AC step !UL failed.\
6E 20 65 63 69 76 72 65 73 20 43 41     003F
70 65 74 73 20 43 41 21 20 65 6D 61     004B
2E 64 65 6C 69 61 66 20 4C 55 21 20     0057
                                        0063     91 CS2:
74 63 65 70 78 45 0000006B'010E0000'    0063     92          .ASCID  \Expected !AS = !XL received !AS = !XL\
4C 58 21 20 3D 20 53 41 21 20 64 65     0071
41 21 20 64 65 76 69 65 63 65 72 20     007D
                4C 58 21 20 3D 20 53    0089
                                        0090     93 CS3:
74 63 65 70 78 45 00000098'010E0000'    0090     94          .ASCID  \Expected !AS!UB = !XL received !AS!UB = !XL\
20 3D 20 42 55 21 53 41 21 20 64 65     009E
64 65 76 69 65 63 65 72 20 4C 58 21     00AA
58 21 20 3D 20 42 55 21 53 41 21 20     00B6
                                  4C    00C2
                                        00C3     95 CS5:
69 20 65 64 6F 4D 000000CB'010E0000'    00C3     96          .ASCID  \Mode is !AS.\
                   2E 53 41 21 20 73    00D1
                                        00D7     97 EXP:
73 75 74 61 74 73 000000DF'010E0000'    00D7     98          .ASCID  \status\
                                        00E5     99 UM:                                      ; mode messages
       72 65 73 75 000000ED'010E0000'   00E5    100          .ASCID  \user\
                                        00F1    101 MSGVEC:
                           00000003     00F1    102          .LONG   3                       ; PUTMSG message vector
                           00741133     00F5    103          .LONG   UETP$_TEXT
                           00000001     00F9    104          .LONG   1
                           0000016F'    00FD    105          .ADDRESS MESSAGEL
                                        0101    106 PURGWS:
                53 57 47 52 55 50 00'   0101    107          .ASCIC  /PURGWS/                ; service name
                                        06  0101
                                        0108    108 WS_STR:
70 20 63 6F 72 70 00000110'010E0000'    0108    109          .ASCID  /proc pg cnt/
                      74 6E 63 20 67    0116
```

```
                                    011B   111 ;
                                    011B   112         .SBTTL  R/W PSECT
                                00000000   113         .PSECT  RWDATA,RD,WRT,NOEXE,PAGE
                                    0000   114 ;
                                    0000   115 PID:
                                00000000   116         .LONG   0                       ; PID for this process
                                    0004   117 CURRENT_TC:
                                00000000   118         .LONG   0                       ; ptr to current test case
                                    0008   119         .ALIGN LONG                     ; put it on a long word boundry
                                    0008   120 REG_SAVE_AREA:
                                00000044   121         .BLKL   15                      ; register save area
                                    0044   122 MOD_MSG_CODE:
                                007480D9   123         .LONG   UETP$_SATSMS            ; test module message code for putmsg
                                    0048   124 TMN_ADDR:
                                00000000'  125         .ADDRESS TEST_MOD_NAME
                                    004C   126 TMD_ADDR:
                                00000019'  127         .ADDRESS TEST_MOD_BEGIN
                                    0050   128 PRVPRT:
                                      00   129         .BYTE   0                       ; protection return byte for SETPRT
                                    0051   130 PRIVMASK:
                    00000000 00000000   131         .QUAD   0                       ; priv. mask
                                    0059   132 CHM_CONT:
                                00000000   133         .LONG   0                       ; change mode continue address
                                    005D   134 RETADR:
                                00000065   135         .BLKL   2                       ; returned address's from SETPRT
                                    0065   136 STATUS:
                                00000000   137         .LONG   0
                                    0069   138 MODE:
                                00000000   139         .LONG   0                       ; current mode string pointer
                                    006D   140 REG:
74 73 69 67 65 72 00000075'010E0000'   141         .ASCID  \register R\
                             52 20 72 65   007B
                                    007F   142 REGNUM:
                                00000000   143         .LONG   0                       ; register number
                                    0083   144 MSGL:
                                00000050   145         .LONG   80                      ; buffer desc.
                                0000008B'  146         .ADDRESS BUF
                                    008B   147 BUF:
                                000000DB   148         .BLKB   80
                                    00DB   149 ML:
                                00000000   150         .LONG   0                       ; desc. for BUF_CHECK routine
                                000000EB'  151         .ADDRESS GETBUF+8
                                    00E3   152 GETBUF:
                                00000084   153         .LONG   132
                                000000EB'  154         .ADDRESS .+4
                                0000016F   155         .BLKB   132
                                    016F   156 MESSAGEL:
                                00000000   157         .LONG   0                       ; message desc.
                                0000008B'  158         .ADDRESS BUF
                                    0177   159 SERV_NAME:
                                00000000   160         .LONG   0                       ; service name pointer
                                    017B   161 MSGVEC1:                             ; PUTMSG message vector
                                00000003   162         .LONG   3
                                00741133   163         .LONG   UETP$_TEXT
                                00000001   164         .LONG   1
                                00000000   165         .LONG   0
                                    018B   166 GET_LIST:
```

```
                0004  018B   167          .WORD   4                       ; GETJPI item list
                030D  018D   168          .WORD   JPI$_PPGCNT
00000198'       018F   169          .LONG   PPG_CNT
00000000        0193   170          .LONG   0
00000000        0197   171          .LONG   0
                019B   172 PPG_CNT:
00000000        019B   173          .LONG   0                       ; before WS peak
                019F   174 PPG_CNT1:
00000000        019F   175          .LONG   0                       ; after WS peak
                01A3   176 PURGE_AREA:
00000000'       01A3   177          .ADDRESS TOUCH_PAGE             ; PURGWS address block
00000000'       01A7   178          .ADDRESS TOUCH_PAGE
                01AB   179 LOCK_AREA:
00000000'       01AB   180          .ADDRESS TEST_MOD_NAME          ; LCKPAG address array
000003C6'       01AF   181          .ADDRESS TEST_END
                01B3   182 PURG:
                01B3   183          $PURGWS PURGE_AREA              ; PURGWS parameter list
            00000000   184          .PSECT TOUCH_PAGE,RD,PAGE
                0000   185          .ALIGN PAGE
                0000   186 TOUCH_PAGE:
00000600        0000   187          .BLKB   1536                    ; 3 pages to touch
```

```
00000000   189          .PSECT  SATSSS80,RD,WRT,EXE,PAGE
     0000  190          .SBTTL  SATSSS80
     0000  191 ;++
     0000  192 ; FUNCTIONAL DESCRIPTION:
     0000  193 ;
     0000  194 ;       After performing some initial housekeeping, such as
     0000  195 ; printing the module begin message and acquiring needed privileges,
     0000  196 ; the system services are tested in each of their normal conditions.
     0000  197 ; Detected failures are identified and  an error message is printed
     0000  198 ; on the terminal.  Upon completion of the test a success or fail
     0000  199 ; message is printed on the terminal.
     0000  200 ;
     0000  201 ; CALLING SEQUENCE:
     0000  202 ;
     0000  203 ;       $ RUN SATSSS80  ...  (DCL COMMAND)
     0000  204 ;
     0000  205 ; INPUT PARAMETERS:
     0000  206 ;
     0000  207 ;       none
     0000  208 ;
     0000  209 ; IMPLICIT INPUTS:
     0000  210 ;
     0000  211 ;       none
     0000  212 ;
     0000  213 ; OUTPUT PARAMETERS:
     0000  214 ;
     0000  215 ;       none
     0000  216 ;
     0000  217 ; IMPLICIT OUTPUTS:
     0000  218 ;
     0000  219 ;       Messages to SYS$OUTPUT are the only output from SATSSS80.
     0000  220 ;       They are of the form:
     0000  221 ;
     0000  222 ;               %UETP-S-SATSMS, TEST MODULE SATSSS80 BEGUN ... (BEGIN MSG)
     0000  223 ;               %UETP-S-SATSMS, TEST MODULE SATSSS80 SUCCESSFUL ... (END MSG)
     0000  224 ;               %UETP-E-SATSMS, TEST MODULE SATSSS80 FAILED ... (END MSG)
     0000  225 ;               %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
     0000  226 ;
     0000  227 ; COMPLETION CODES:
     0000  228 ;
     0000  229 ;       The SATSSS80 routine terminates with a $EXIT to the
     0000  230 ;       operating system with a status code defined by UETP$_SATSMS.
     0000  231 ;
     0000  232 ; SIDE EFFECTS:
     0000  233 ;
     0000  234 ;       none
     0000  235 ;
     0000  236 ;--
     0000  237
     0000  238          TEST_START SATSSS80                     ; let the test begin
```

```
                                0000
                          0000  0000                    .ENTRY SATSSS80,0
                0004'CF    D4   0002                    CLRL    W^CURRENT_TC
                      00   DD   0006                    PUSHL   #0
                0000'CF    DF   0008                    PUSHAL  W^TPID
      00000000'GF    02    FB   000C                    CALLS   #2,G^SYS$WAKE
      00000000'GF    00    FB   0013                    CALLS   #0,G^SYS$HIBER
                0009'CF    7F   001A                    PUSHAQ  W^TEST_MOD_NAME_D
      00000000'GF    01    FB   001E                    CALLS   #1,G^SYS$SETPRN
                  037A     30   0025                    BSBW    W^MOD_MSG_PRINT
    004C'CF   001F'CF      DE   0028                    MOVAL   W^TEST_MOD_SUCC,W^TMD_ADDR
0044'CF  03   00   01      F0   002F                    INSV    #SUCCESS,#0,#3,W^MOD_MSG_CODE
                      00   DD   0036                    PUSHL   #0
            0265'CF    01    FB   0038                    CALLS #1,W^REG_SAVE
                                0003D           STP0:
                                003D  239               .SBTTL PURGWS TESTS
                                003D  240 ;+
                                003D  241 ;
                                003D  242 ; $PURGWS tests
                                003D  243 ;
                                003D  244 ; test _S form with a dry WS and adr array elements =
                                003D  245 ;
                                003D  246 ;-
      017?'CF   0101'CF    DE   003D  247               MOVAL   W^PURGWS,W^SERV_NAME            ; set service name
      0069'CF   00E5'CF    DE   0044  248               MOVAL   W^UM,W^MODE                     ; set the mode
                                004B  249               MODE    TO,10$,KRNL,NOREGS              ; get to kernel mode
       59   00000000'9F    D0   0068  250               MOVL    @#CTL$GL_PHD,R9                 ; get the process header adr
           0051'CF    69    DE   006F  251               MOVAL   PHD$Q_PRIVMSK(R9),W^PRIVMASK    ; get the priv. mask
                                0074  252               MODE    FROM,TO$                        ; return to user mode
                                0075  253               PRIV    ADD,PSWAPM                      ; allow page locking
                      00    DD   0095  254               PUSHL   #0                              ; push a dummy parameter
            0265'CF    01    FB   0097  255               CALLS   #1,W^REG_SAVE                   ; save a reg snapshot
                                009C  256               $LCKPAG_S INADR =W^LOCK_AREA            ; nail down everything but TOUCH_PAG
                                00AB  257               $PURGWS_S INADR =W^PURGE_AREA           ; squeeze the juice out of this proc
                                00B6  258               FAIL_CHECK SS$_NORMAL                   ; check for success
                      01    DD   00B6                             PUSHL   #SS$_NORMAL
            026F'CF    01    FB   00B8                             CALLS   #1,W^REG_CHECK
                                00BD  259               $GETJPI_S ITMLST=W^GET_LIST             ; get the process page count in ques
                                00D2  260 ;+
                                00D2  261 ;
                                00D2  262 ; test _S  form with adr array elements one page apart
                                00D2  263 ;
                                00D2  264 ;-
                                00D2  265               NEXT_TEST
                                00D2
                                00D2            STP1:
            0004'CF    01    DO   00D2                             MOVL    #1,W^CURRENT_TC
                      00    DD   00D7                             PUSHL   #0
            0265'CF    01    FB   00D9                             CALLS   #1,W^REG_SAVE
  01A7'CF    000001FF   8F    C0   00DE  266               ADDL2   #511,W^PURGE_AREA+4             ; set new adr array element
      018F'CF   019F'CF    DE   00E7  267               MOVAL   W^PPG_CNT1,W^GET_LIST+4         ; point to a new storage location
                                00EE  268               $PURGWS_S INADR =W^PURGE_AREA           ; squeeze blood out of a turnip
                                00F9  269               FAIL_CHECK SS$_NORMAL                   ; check for success
                      01    DD   00F9                             PUSHL   #SS$_NORMAL
            026F'CF    01    FB   00FB                             CALLS   #1,W^REG_CHECK
                                0100  270               $GETJPI_S ITMLST=W^GET_LIST             ; get the new process page count
      019F'CF   019B'CF    D1   0115  271               CMPL    W^PPG_CNT,W^PPG_CNT1            ; are they the same?
```

```
                     11   13  011C   272          BEQL    10$                             ; br if they are
                019F'CF   DD  011E   273          PUSHL   W^PPG_CNT1                      ; push received
                019B'CF   DD  0122   274          PUSHL   W^PPG_CNT                       ; push expected
                0108'CF   DF  0126   275          PUSHAL  W^WS_STR                        ; push string variable
           02B1'CF   03   FB  012A   276          CALLS   #3,W^PRINT_FAIL                 ; print the failure
                              012F   277 10$:
                              012F   278 ;+
                              012F   279 ;
                              012F   280 ; test _G form with one page of juice in the process page count
                              012F   281 ;
                              012F   282 ;-
                              012F   283          NEXT_TEST
                              012F
                              012F       STP2:
           0004'CF   02   D0  012F                 MOVL    #2,W^CURRENT_TC
                     00   DD  0134                 PUSHL   #0
           0265'CF   01   FB  0136                 CALLS   #1,W^REG_SAVE
      018F'CF   019B'CF   DE  013B   284          MOVAL   W^PPG_CNT,W^GET_LIST+4          ; reset the process page pointer
                0000'CF   D5  0142   285          TSTL    W^TOUCH_PAGE                    ; suck in a new page
                          0146   286          $GETJPI_S ITMLST=W^GET_LIST             ; get page count after touch
                          015B   287          $PURGWS_G W^PURG                        ; try _G form
                          0164   288          FAIL_CHECK SS$_NORMAL                   ; check success
                     01   DD  0164                 PUSHL   #SS$_NORMAL
           026F'CF   01   FB  0166                 CALLS   #1,W^REG_CHECK
      018F'CF   019F'CF   DE  016B   289          MOVAL   W^PPG_CNT1,W^GET_LIST+4         ; set new page count pointer
                          0172   290          $GETJPI_S ITMLST=W^GET_LIST             ; get the new process page count
                019B'CF   D7  0187   291          DECL    W^PPG_CNT                       ; create expected
      019F'CF   019B'CF   D1  018B   292          CMPL    W^PPG_CNT,W^PPG_CNT1            ; did we squeeze a page out?
                     11   13  0192   293          BEQL    20$                             ; br if yes
                019F'CF   DD  0194   294          PUSHL   W^PPG_CNT1                      ; push recieved
                019B'CF   DD  0198   295          PUSHL   W^PPG_CNT                       ; push expected
                0108'CF   DF  019C   296          PUSHAL  W^WS_STR                        ; push string variable
           02B1'CF   03   FB  01A0   297          CALLS   #3,W^PRINT_FAIL                 ; print the failure
                              01A5   298 20$:
                              01A5   299 ;+
                              01A5   300 ;
                              01A5   301 ; test _S form with more than one page to recover
                              01A5   302 ;
                              01A5   303 ;-
                              01A5   304          NEXT_TEST
                              01A5
                              01A5       STP3:
           0004'CF   03   D0  01A5                 MOVL    #3,W^CURRENT_TC
                     00   DD  01AA                 PUSHL   #0
           0265'CF   01   FB  01AC                 CALLS   #1,W^REG_SAVE
 01A7'CF    00000400   8F   C0  01B1   305          ADDL2   #1024,W^PURGE_AREA+4            ; make a three page purge area
      018F'CF   019F'CF   DE  01BA   306          MOVAL   W^PPG_CNT1,W^GET_LIST+4         ; reset the process page pointer
           56   0000'CF   DE  01C1   307          MOVAL   W^TOUCH_PAGE,R6                 ; set a page pointer
                57   03   D0  01C6   308          MOVL    #3,R7                           ; set a page count
                              01C9   309 30$:
                     66   D5  01C9   310          TSTL    (R6)                            ; touch a page
           56    00000200   8F   C0  01CB   311          ADDL2   #512,R6                 ; point to next page
                      F4   57   F5  01D2   312          SOBGTR  R7,30$                  ; do all pages
                     00   DD  01D5   313          PUSHL   #0                              ; push a dummy paramter
           0265'CF   01   FB  01D7   314          CALLS   #1,W^REG_SAVE                   ; save a reg snapshot
                          01DC   315          $GETJPI_S ITMLST=W^GET_LIST             ; get the process page count
                          01F1   316          $PURGWS_S INADR=W^PURGE_AREA            ; clean it up
```

```
                                 01FC   317          FAIL_CHECK SS$_NORMAL                          ; check for success
                          01     DD  01FC                    PUSHL   #SS$_NORMAL
                026F'CF   01     FB  01FE                    CALLS   #1,W^REG_CHECK
          018F'CF   019B'CF      DE  0203   318          MOVAL   W^PPG_CNT,W^GET_LIST+4                 ; set new PPG pointer
                                 020A   319          $GETJPI_S ITMLST=W^GET_LIST                    ; get new process page count
                019F'CF   03     C2  021F   320          SUBL2   #3,W^PPG_CNT1                          ; set expected PPGCNT
          019F'CF   019B'CF      D1  0224   321          CMPL    W^PPG_CNT,W^PPG_CNT1                   ; did we get at least 3 pages?
                          11     13  022B   322          BEQL    40$                                    ; br if OK
                    019B'CF      DD  022D   323          PUSHL   W^PPG_CNT                              ; push recieved
                    019F'CF      DD  0231   324          PUSHL   W^PPG_CNT1                             ; push expected
                    0108'CF      DF  0235   325          PUSHAL  W^WS_STR                               ; push string variable
                02B1'CF   03     FB  0239   326          CALLS   #3,W^PRINT_FAIL                        ; print the failure
                                 023E   327 40$:
                                 023E   328          TEST_END
                    004C'CF      DD  023E                    PUSHL   W^TMD_ADDR
                    0048'CF      DD  0242                    PUSHL   W^TMN_ADDR
                          02     DD  0246                    PUSHL   #2
                    0044'CF      DD  0248                    PUSHL   W^MOD_MSG_CODE
             00000000'GF   04     FB  024C                    CALLS   #$$T1,G^LIB$SIGNAL
    0044'CF   01     1C   01     F0  0253                    INSV    #1,#STS$V_INHIB_MSG,#1,W^MOD_MSG_CODE
                    0044'CF      DD  025A                    PUSHL   W^MOD_MSG_CODE
             00000000'GF   01     FB  025E                    CALLS   #1,G^SYS$EXIT
```

```
                                0265    331          .SBTTL REG_SAVE
                                0265    332 ;++
                                0265    333 ; FUNCTIONAL DESCRIPTION:
                                0265    334 ;       Subroutine to save R2-R11 in the register save location.
                                0265    335 ;
                                0265    336 ; CALLING SEQUENCE:
                                0265    337 ;       PUSHL   #0                  ; save a dummy parameter
                                0265    338 ;       CALLS   #1,W^REG_SAVE       ; save R2-R11
                                0265    339 ;
                                0265    340 ; INPUT PARAMETERS:
                                0265    341 ;       NONE
                                0265    342 ;
                                0265    343 ; OUTPUT PARAMETERS:
                                0265    344 ;       NONE
                                0265    345 ;
                                0265    346 ;--
                                0265    347
                                0265    348 REG_SAVE:
                          0FFC  0265    349          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
    0008'CF   14 AD   28   28   0267    350          MOVC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; save the registers in the program
                          04    026E    351          RET
                                026F    352          .SBTTL  REG_CHECK
                                026F    353 ;++
                                026F    354 ; FUNCTIONAL DESCRIPTION:
                                026F    355 ;       Subroutine to test R0 & R2-R11 for proper content after a service
                                026F    356 ;       execution. A snapshot is taken by the REG_SAVE routine at the
                                026F    357 ;       beginning of each step and this routine is executed after the
                                026F    358 ;       services have been executed.
                                026F    359 ;
                                026F    360 ; CALLING SEQUENCE:
                                026F    361 ;       PUSHL   #SS$_XXXXXX         ; push expected R0 contents
                                026F    362 ;       CALLS   #1,W^REG_CHECK      ; execute this routine
                                026F    363 ;
                                026F    364 ; INPUT PARAMETERS:
                                026F    365 ;       expected R0 contents on the stack
                                026F    366 ;
                                026F    367 ; OUTPUT PARAMETERS:
                                026F    368 ;       possible error messages printed using $PUTMSG
                                026F    369 ;
                                026F    370 ;--
                                026F    371
                                026F    372 REG_CHECK:
                          0FFC  026F    373          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
          50    04 AC     D1   0271    374          CMPL    4(AP),R0                        ; is this the right fail code?
                     0E     13   0275    375          BEQL    10$                             ; br if yes
                     50     DD   0277    376          PUSHL   R0                              ; push received data
                  04 AC     DD   0279    377          PUSHL   4(AP)                           ; push expected data
              00D7'CF       DF   027C    378          PUSHAL  W^EXP                           ; push the string variable
        02B1'CF             FB   0280    379          CALLS   #3,W^PRINT_FAIL                 ; print the error message
                            0285    380 10$:
 0008'CF    14 AD   28   29   0285    381          CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; check all but R0
                     22     13   028C    382          BEQL    20$                             ; br if O.K.
56   53    00000008'8F     C3   028E    383          SUBL3   #REG_SAVE_AREA,R3,R6            ; calculate the register number
                  56 04     C6   0296    384          DIVL2   #4,R6
           7E     56 02     81   0299    385          ADDB3   #^X2,R6,-(SP)                   ; set number past R0-R1 and save
                  51 03     CA   029D    386          BICL2   #3,R1                           ; backup to register boundrys
                  53 03     CA   02A0    387          BICL2   #3,R3
```

```
                                02B0    392 20$:
                             04 02B0    393         RET
                                02B1    394         .SBTTL  PRINT_FAIL
                                02B1    395 ;++
                                02B1    396 ; FUNCTIONAL DESCRIPTION:
                                02B1    397 ;       Subroutine to report failures using $PUTMSG
                                02B1    398 ;
                                02B1    399 ; CALLING SEQUENCE:
                                02B1    400 ; Mode  #1      PUSHL EXPECTED  Mode    #2      PUSHL REG_NUMBER
                                02B1    401 ;               PUSHL RECEIVED                   PUSHL EXPECTED
                                02B1    402 ;               PUSHAL STRING_VAR                PUSHL RECEIVED
                                02B1    403 ;               CALLS #3,W^PRINT_FAIL            PUSHAL STRING_VAR
                                02B1    404 ;                                                CALLS #4,W^PRINT_FAIL
                                02B1    405 ; Mode  #3      PUSHAL STRING_VAR
                                02B1    406 ;               CALLS #1,W^PRINT_FAIL
                                02B1    407 ;
                                02B1    408 ; INPUT PARAMETERS:
                                02B1    409 ;       listed above
                                02B1    410 ;
                                02B1    411 ; OUTPUT PARAMETERS:
                                02B1    412 ;       an error message is printed using $PUTMSG
                                02B1    413 ;
                                02B1    414 ;--
                                02B1    415
                                02B1    416 PRINT_FAIL:
                           003C 02B1    417         .WORD   ^M<R2,R3,R4,R5>
                                02B3    418         $FAO_S  W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
                                02D4    419         $PUTMSG_S W^MSGVEC                      ; print the message
                          04    6C  91  02E5    420         CMPB    (AP),#4                 ; is this a register message?
                                26  13  02E8    421         BEQL    10$                     ; br if yes
                          01    6C  91  02EA    422         CMPB    (AP),#1                 ; is this just a message?
                                48  13  02ED    423         BEQL    20$                     ; br if yes
                                    02EF    424         $FAO_S  W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
                                40  11  030E    425         BRB     30$                     ; goto output message
                                    0310    426 10$:
                                    0310    427         $FAO_S  W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
                                19  11  0335    428         BRB     30$                     ; goto output message
                                    0337    429 20$:
                  0187'CF   04  AC  D0  0337    430         MOVL    4(AP),W^MSGVEC1+12      ; save string address
                                    033D    431         $PUTMSG_S W^MSGVEC1             ; print the message
                                11  11  034E    432         BRB     40$                     ; skip the other message
                                    0350    433 30$:
                                    0350    434         $PUTMSG_S W^MSGVEC              ; print the message
                                    0361    435 40$:
                0375'CF   00    FB  0361    436         CALLS   #0,W^MODE_ID            ; identify the mode
           004C'CF   002A'CF    DE  0366    437         MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR      ; set failure message address
0044'CF    03   00    02        F0  036D    438         INSV    #ERROR,#0,#3,W^MOD_MSG_CODE     ; set severity code
                                04  0374    439         RET
```

```
                0375   441          .SBTTL  MODE_ID
                0375   442 ;++
                0375   443 ; FUNCTIONAL DESCRIPTION:
                0375   444 ;       Subroutine to identify the mode that an exit handler is in.
                0375   445 ;
                0375   446 ; CALLING SEQUENCE:
                0375   447 ;       CALLS   #0,W^MODE_ID
                0375   448 ;
                0375   449 ; INPUT PARAMETERS:
                0375   450 ;       MODE contains an address pointing to an ascii string desc.
                0375   451 ;       of the current CPU mode.
                0375   452 ;
                0375   453 ; OUTPUT PARAMETERS:
                0375   454 ;       NONE
                0375   455 ;
                0375   456 ;--
                0375   457
                0375   458 MODE_ID:
      003C      0375   459          .WORD   ^M<R2,R3,R4,R5>
                0377   460          $FAO_S  W^CS5,W^MESSAGEL,W^MSGL,MODE ; format the error message
                0390   461          $PUTMSG_S W^MSGVEC                   ; print the mode message
        04      03A1   462          RET
```

```
                     03A2    465 MOD_MSG_PRINT:
                     03A2    466 ;
                     03A2    467 ;  ***************************************************************************
                     03A2    468 ;  *                                                                         *
                     03A2    469 ;  *  PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES                *
                     03A2    470 ;  *       (USING THE PUTMSG MACRO).                                         *
                     03A2    471 ;  *                                                                         *
                     03A2    472 ;  ***************************************************************************
                     03A2    473 ;
                     03A2    474           PUTMSG  <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> ; PRINT MSG
                  05 03BD    475           RSB                                         ; ... AND RETURN TO CALLER
                     03BE    476 ;
                     03BE    477 CHMRTN:
                     03BE    478 ;  ***************************************************************************
                     03BE    479 ;  *                                                                         *
                     03BE    480 ;  *     CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER             *
                     03BE    481 ;  *     A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED                *
                     03BE    482 ;  *     BY THE MODE MACRO ('TO' OPTION).  IT MERELY DOES                    *
                     03BE    483 ;  *     A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS                   *
                     03BE    484 ;  *     THE EFFECT OF RETURNING TO THE END OF THE MODE                      *
                     03BE    485 ;  *     MACRO EXPANSION.                                                    *
                     03BE    486 ;  *                                                                         *
                     03BE    487 ;  ***************************************************************************
                     03BE    488 ;
                0000 03BE    489           .WORD   0                                   ; ENTRY MASK
00000059'FF      17  03C0    490           JMP     @CHM_CONT                           ; RETURN TO MODE MACRO IN NEW MODE
                     03C6    491 ;
                     03C6    492 ;  *     RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
                     03C6    493 ;
                     03C6    494 TEST_END:
                     03C6    495           .END    SATSSS80
```

```
$$ARGS                  =  00000001
$$T1                    =  00000004
$$T2                    =  00000004
BUF                        0000008B  R      03
CHMRTN                     000003BE  R      05
CHM_CONT                   00000059  R      03
CS1                        00000031  R      02
CS2                        00000063  R      02
CS3                        00000090  R      02
CS5                        000000C3  R      02
CTL$GL_PHD                 ********    X    05
CURRENT_TC                 00000004  R      03
ERROR                   =  00000002
EXP                        000000D7  R      02
GETBUF                     000000E3  R      03
GET_LIST                   0000018B  R      03
INFO                    =  00000003
JPI$_PPGCNT             =  0000030D
LIB$SIGNAL                 ********    X    05
LOCK_AREA                  000001AB  R      03
MESSAGEL                   0000016F  R      03
ML                         000000DB  R      03
MODE                       00000069  R      03
MODE_ID                    00000375  R      05
MOD_MSG_CODE               00000044  R      03
MOD_MSG_PRINT              000003A2  R      05
MSGL                       00000083  R      03
MSGVEC                     000000F1  R      02
MSGVEC1                    0000017B  R      03
PHD$Q_PRIVMSK           =  00000000
PPG_CNT                    0000019B  R      03
PPG_CNT1                   0000019F  R      03
PRINT_FAIL                 000002B1  R      05
PRIVMASK                   00000051  R      03
PRIV_ARGS               =  00000002
PRV$V_PSWAPM            =  0000000C
PRVPRT                     00000050  R      03
PURG                       000001B3  R      03
PURGE_AREA                 000001A3  R      03
PURGWS                     00000101  R      02
PURGWS$_INADR           =  00000004
PURGWS$_NARGS           =  00000001
REG                        0000006D  R      03
REGNUM                     0000007F  R      03
REG_CHECK                  0000026F  R      05
REG_SAVE                   00000265  R      05
REG_SAVE_AREA              00000008  R      03
RETADR                     0000005D  R      03
SATSSS80                   00000000  RG     05
SERV_NAME                  00000177  R      03
SEVERE                  =  00000004
SHR$K_SHRDEF            =  00000001
SHR$_TEXT               =  00001130
SS$_NORMAL              =  00000001
STATUS                     00000065  R      03
STEP                    =  00000003
STP0                       0000003D  R      05
STP1                       000000D2  R      05
STP2                       0000012F  R      05
STP3                       000001A5  R      05
STS$V_INHIB_MSG         =  0000001C
SUCCESS                 =  00000001
SYS$CMKRNL                 ********    GX   05
SYS$EXIT                   ********    GX   05
SYS$FAO                    ********     X   05
SYS$GETJPI                 ********    GX   05
SYS$HIBER                  ********    GX   05
SYS$LCKPAG                 ********    GX   05
SYS$PURGWS                 ********    GX   05
SYS$PUTMSG                 ********    GX   05
SYS$SETPRN                 ********    GX   05
SYS$SETPRV                 ********    GX   05
SYS$WAKE                   ********    GX   05
TEST_END                   000003C6  R      05
TEST_MOD_BEGIN             00000019  R      02
TEST_MOD_FAIL              0000002A  R      02
TEST_MOD_NAME              00000000  R      02
TEST_MOD_NAME_D            00000009  R      02
TEST_MOD_SUCC              0000001F  R      02
TMD_ADDR                   0000004C  R      03
TMN_ADDR                   00000048  R      03
TOUCH_PAGE                 00000000  R      04
TPID                       00000000  R      03
UETP$_SATSMS            =  007480D9
UETP$_TEXT              =  00741133
UM                         000000E5  R      02
WARNING                 =  00000000
WS_STR                     00000108  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	0000011B (283.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC PAGE
RWDATA	000001BB (443.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC PAGE
TOUCH_PAGE	00000600 (1536.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC PAGE
SATSSS80	000003C6 (966.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC PAGE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.11	00:00:00.68
Command processing	107	00:00:00.73	00:00:04.45
Pass 1	374	00:00:12.01	00:00:30.06
Symbol table sort	0	00:00:01.53	00:00:02.77
Pass 2	115	00:00:02.52	00:00:05.43
Symbol table output	11	00:00:00.09	00:00:01.03
Psect synopsis output	2	00:00:00.04	00:00:00.06
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	640	00:00:17.03	00:00:44.48

The working set limit was 1350 pages.
67614 bytes (133 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 979 non-local and 12 local symbols.
495 source lines were read in Pass 1, producing 26 object records in Pass 2.
47 pages of virtual memory were used to define 42 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	26
_255DUA28:[SHRLIB]UETP.MLB;1	12
_255DUA28:[SYS.OBJ]LIB.MLB;1	1
_255DUA28:[SYSLIB]STARLET.MLB;2	0
TOTALS (all libraries)	39

1267 GETS were required to define 39 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SATSSS80/OBJ=OBJ$:SATSSS80 MSRC$:SATSSS80/UPDATE=(ENH$:SATSSS80)+EXECML$/LIB+SHRLIB$:UETP/LIB

SATSS574
LIS

SATSS578
LIS

SATSS580
LIS

SATSS582
LIS

SATSS590
LIS

SATSUT01
LIS

SATSUT05
LIS

SATSUT04
LIS

SATSUT06
LIS

SATSS581
LIS

SATSS591
LIS

SATSS579
LIS

SATSS583
LIS