```
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    333333    5555555555
  SSSSSSSS    AAAAAA    TTTTTTTTTT    SSSSSSSS    SSSSSSSS    SSSSSSSS    333333    5555555555
SS           AA      AA      TT      SS          SS          SS          33      33  55
SS           AA      AA      TT      SS          SS          SS          33      33  55
SS           AA      AA      TT      SS          SS          SS                  33  555555
SS           AA      AA      TT      SS          SS          SS                  33  555555
  SSSSSS     AA      AA      TT        SSSSSS      SSSSSS      SSSSSS          33          55
  SSSSSS     AA      AA      TT        SSSSSS      SSSSSS      SSSSSS          33          55
        SS   AAAAAAAAAA      TT              SS          SS          SS          33          55
        SS   AAAAAAAAAA      TT              SS          SS          SS          33          55
        SS   AA      AA      TT              SS          SS          SS  33      33  55      55      ....
        SS   AA      AA      TT              SS          SS          SS  33      33  55      55      ....
SSSSSSSS     AA      AA      TT      SSSSSSSS    SSSSSSSS    SSSSSSSS      333333      555555      ....
SSSSSSSS     AA      AA      TT      SSSSSSSS    SSSSSSSS    SSSSSSSS      333333      555555      ....

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     1          .TITLE  SATSSS35 - SATS SYSTEM SERVICE TESTS  (SUCC S.C.)
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
000C     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 SATSSS35 module tests the execution of the following
0000    33  ;               VMS system services:
0000    34  ;
0000    35  ;               $CREPRC
0000    36  ;               $GETJPI
0000    37  ;
0000    38  ; ENVIRONMENT:  User mode image,
0000    39  ;               Needs CMKRNL privilege and dynamically acquires other
0000    40  ;               privileges, as needed.
0000    41  ;
0000    42  ; AUTHOR: Larry D. Jones,                CREATION DATE: JULY, 1979
0000    43  ;
0000    44  ; MODIFIED BY:
0000    45  ;
0000    46  ;       V03-002 LDJ0006         Larry D. Jones,         23-Mar-1982
0000    47  ;               Made the quota list be absolute minimum to test the
0000    48  ;               SYSBOOT minimum values.
0000    49  ;
0000    50  ;       V03-001 RNP0005         Robert N. Perron,       23-Mar-1982
0000    51  ;               Removed EXCVEC and FINALEXC from the JPI_GOOD list.
0000    52  ;
0000    53  ;       V02-006 RNP0004         Robert N. Perron,       09-Dec-1981
0000    54  ;               Removed ASTEN from the JPI_GOOD list.
0000    55  ;
0000    56  ;       V02-005 RNP0003         Robert N. Perron,       02-Oct-1981
0000    57  ;               Removed ASTACT from the JPI_GOOD list.
```

```
0000     58 ;
0000     59 ;       V02-004 LDJ0002         Larry D. Jones,         06-Sep-1981
0000     60 ;               Fixed GETJPI P1 reference to CTL$AQ_EXCVEC and CTL$AL_FINALEXC.
0000     61 ;
0000     62 ;       V02-003 RNP0002         Robert N. Perron,       01-Jun-1981
0000     63 ;               To eliminate dependence on the SYSTEST account privileges
0000     64 ;               being a specific list, privileges are now set to a fixed list
0000     65 ;               before the GETJPI tests are started.
0000     66 ;
0000     67 ;       V02-002 RNP0001         Robert N. Perron,       09-Apr-1981
0000     68 ;               Fixed problem of STS field changing due to Swapper activity.
0000     69 ;               Prevent failure when privileges are added to the SYSTEST
0000     70 ;               account. Cleaned up some format problems.
0000     71 ;
0000     72 ;       V02-001 LDJ0001         Larry D. Jones,         17-Sep-1980
0000     73 ;               Modified to conform to new build command procedures.
0000     74 ;**
0000     75 ;--
```

```
                     0000     77          .SBTTL  DECLARATIONS
                     0000     78 ;
                     0000     79 ; MACRO LIBRARY CALLS
                     0000     80 ;
                     0000     81          $ACCDEF                         ; account record offset definitions
                     0000     82          $DIBDEF                         ; device info block definitions
                     0000     83          $JPIDEF                         ; JPI offset definitions
                     0000     84          $PCBDEF                         ; process control block definitions
                     0000     85          $PHDDEF                         ; Process header definitions
                     0000     86          $PQLDEF
                     0000     87          $PRVDEF                         ; privilege definitions
                     0000     88          $SHRDEF                         ; shared message definitions
                     0000     89          $SFDEF                          ; stack frame definitions
                     0000     90          $STSDEF                         ; STS definitions
                     0000     91          $UETPDEF                        ; UETP message definitions
                     0000     92 ;
                     0000     93 ;
00000001             0000     94          SUCCESS         = 1             ; success
00000002             0000     95          ERROR           = 2             ; error
                     0000     96 ;
                     0000     97 ; SHR message definitions
                     0000     98 ;
00740000             0000     99          UETP = 116@STS$V_FAC_NO         ;define the UETP facility code
                     0000    100 ;
00741038             0000    101          UETP$_BEGIND = UETP!SHR$_BEGIND ;define the UETP messages
00741130             0000    102          UETP$_TEXT   = UETP!SHR$_TEXT
007410E0             0000    103          UETP$_ABENDD = UETP!SHR$_ABENDD
00741080             0000    104          UETP$_ENDEDD = UETP!SHR$_ENDEDD
                     0000    105 ;
                     0000    106 ; Mask of bits for the STS field in a $CREPRC system service as they are
                     0000    107 ; returned from a $GETJPI system service.
                     0000    108 ;
                     0000    109          JPI_STS_MASK = <<1@PCB$V_NETWRK>!<1@PCB$V_SSFEXCU>!<1@PCB$V_SSRWAIT>!-
                     0000    110                          <1@PCB$V_BATCH> !<1@PCB$V_NOACNT> !<1@PCB$V_HIBER>  !-
0038C600             0000    111                          <1@PCB$V_LOGIN>>
                     0000    112 ;
                     0000    113 ; The opposite of JPI_STS_MASK
                     0000    114 ;
FFC739FF             0000    115          JPI_STS_NMASK = ^CJPI_STS_MASK
                     0000    116
                     0000    117 ;
                     0000    118 ; Mask of bits for the Privilege field as they are returned from a $GETJPI
                     0000    119 ;
                     0000    120          JPI_PRV_MASK = <<1@PRV$V_CMEXEC>!<1@PRV$V_CMKRNL>!<1@PRV$V_DETACH>!-
                     0000    121                          <1@PRV$V_DIAGNOSE>!<1@PRV$V_GROUP>!<1@PRV$V_GRPNAM>!-
                     0000    122                          <1@PRV$V_LOG_IO>!<1@PRV$V_NETMBX>!<1@PRV$V_NOACNT>!-
                     0000    123                          <1@PRV$V_PHY_IO>!<1@PRV$V_PRMCEB>!<1@PRV$V_PRMMBX>!-
                     0000    124                          <1@PRV$V_PSWAPM>!<1@PRV$V_SETPRI>!<1@PRV$V_SYSNAM>!-
1070BFEF             0000    125                          <1@PRV$V_SYSPRV>!<1@PRV$V_TMPMBX>!<1@PRV$V_VOLPRO>>
                     0000    126 ;
                     0000    127 ; The compliment of JPI_PRV_MASK
                     0000    128 ;
EF8F4010             0000    129          JPI_PRV_NMASK = ^CJPI_PRV_MASK
                     0000    130 ;
                     0000    131 ; MACROS
                     0000    132 ;
                     0000    133          .MACRO JPI,NAME,SIZE
```

```
                                G 8
0000    134          .WORD   SIZE
0000    135          .WORD   JPI$_'NAME'
0000    136          .ADDRESS NAME
0000    137          .ADDRESS NAME'L
0000    138          .SAVE_PSECT
0000    139          .PSECT ITEM_LIST
0000    140 NAME:
0000    141          .BLKB   SIZE
0000    142 NAME'L:
0000    143          .WORD   0
0000    144          .RESTORE_PSECT
0000    145          .ENDM JPI
```

```
                              00000000   147            .PSECT  ITEM_LIST,RD,WRT,NOEXE,LONG ; psect to store JPI results in
                              00000000   148            .PSECT  RODATA,RD,NOWRT,NOEXE,LONG
                                  0000   149 ;
                                  0000   150 TEST_MOD_NAME:
         35 33 53 53 53 54 41 53 00'  0000   151            .ASCIC  /SATSSS35/              ; needed for SATSMS message
                                  08  0000
                                  0009   152 TEST_MOD_NAME_D:
53 53 53 54 41 53 00000011'010E0000'  0009   153            .ASCID /SATSSS35/              ; module name
                               35 33  0017
                                  0019   154 TEST_MOD_BEGIN:                            ; start end and fail messages
                  6E 75 67 65 62 00'  0019   155            .ASCIC  /begun/
                                  05  0019
                                  001F   156 TEST_MOD_SUCC:
   6C 75 66 73 73 65 63 63 75 73 00'  001F   157            .ASCIC  /successful/
                                  0A  001F
                                  002A   158 TEST_MOD_FAIL:
               64 65 6C 69 61 66 00'  002A   159            .ASCIC  /failed/
                                  06  002A
                                  0031   160 CS1:                                       ; failure messages
21 20 74 73 65 54 00000039'010E0000'  0031   161            .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   162 CS2:
74 63 65 70 78 45 0000006B'010E0000'  0063   163            .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   164 CS3:
74 63 65 70 78 45 00000098'010E0000'  0090   165            .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   166 CS5:
77 20 65 64 6F 4D 000000CB'010E0000'  00C3   167            .ASCID  \Mode was !AS.\
            2E 53 41 21 20 73 61      00D1
                                  00D8   168 EXP:
73 75 74 61 74 73 000000E0'010E0000'  00D8   169            .ASCID  \status\
                                  00E6   170 AST_PARAM:
61 70 20 54 53 41 000000EE'010E0000'  00E6   171            .ASCID  \AST param.\
                      2E 6D 61 72     00F4
                                  00F8   172 BP:
70 20 65 73 61 62 00000100'010E0000'  00F8   173            .ASCID  \base pri.\
                         2E 69 72     0106
                                  0109   174 PNS:
73 65 63 6F 72 50 00000111'010E0000'  0109   175            .ASCID  \Process name was not set correctly.\
6E 20 73 61 77 20 65 6D 61 6E 20 73   0117
65 72 72 6F 63 20 74 65 73 20 74 6F   0123
                   2E 79 6C 74 63     012F
                                  0134   176 STSFLGS:
47 4C 46 53 54 53 0000013C'010E0000'  0134   177            .ASCID  \STSFLG's\
                               73 27  0142
                                  0144   178 UIC_MSG:
         43 49 55 0000014C'010E0000'  0144   179            .ASCID  \UIC\
                                  014F   180 EFC_NAME:
46 53 53 54 41 53 00000157'010E0000'  014F   181            .ASCID  \SATSSF06_DET\
```

```
                    54 45 44 5F 36 30  015D
                                       0163   182 PID_STR:
            44 49 50 0000016B'010E0000' 0163   183         .ASCID  \PID\
                                       016E   184 CREPRC:
                  43 52 50 45 52 43 00' 016E   185         .ASCIC  \CREPRC\
                                    06 016E
                                       0175   186 GETJPI:
                  49 50 4A 54 45 47 00' 0175   187         .ASCIC  \GETJPI\
                                    06 0175
                                       017C   188 UM:                              ; mode messages
          72 65 73 75 00000184'01CE0000' 017C   189         .ASCID  \user\
                                       0188   190 MBNAM:
58 42 4D 35 33 53 00000190'010E0000' 0188   191         .ASCID  \S35MBX\
                                       0196   192 PRVMASK:
                    00000000 1070BFEF  0196   193         .QUAD   JPI_PRV_MASK        ; used for setting privileges to
                                       019E   194                                     ; known value
                                       019E   195 NPRVMASK:
                    00000000 EF8F4010  019E   196         .QUAD   JPI_PRV_NMASK       ; used for clearing any extra
                                       01A6   197                                     ; privileges
                                       01A6   198 MSGVEC:
                             00000003  01A6   199         .LONG   3                   ; PUTMSG message vector
                             00741130  01AA   200         .LONG   UETP$_TEXT
                             00000001  01AE   201         .LONG   1
                             00000217' 01B2   202         .ADDRESS MESSAGEL
                                       01B6   203 QUOTA_LIST:
                                   01  01B6   204         .BYTE   PQL$_ASTLM          ; minimum quota list
                             00000001  01B7   205         .LONG   1
                                   02  01BB   206         .BYTE   PQL$_BIOLM
                             00000001  01BC   207         .LONG   1
                                   03  01C0   208         .BYTE   PQL$_BYTLM
                             00000001  01C1   209         .LONG   1
                                   04  01C5   210         .BYTE   PQL$_CPULM
                             00000000  01C6   211         .LONG   0
                                   05  01CA   212         .BYTE   PQL$_DIOLM
                             00000001  01CB   213         .LONG   1
                                   06  01CF   214         .BYTE   PQL$_FILLM
                             00000001  01D0   215         .LONG   1
                                   07  01D4   216         .BYTE   PQL$_PGFLQUOTA
                             00000001  01D5   217         .LONG   1
                                   08  01D9   218         .BYTE   PQL$_PRCLM
                             00000000  01DA   219         .LONG   0
                                   09  01DE   220         .BYTE   PQL$_TQELM
                             00000000  01DF   221         .LONG   0
                                   0B  01E3   222         .BYTE   PQL$_WSDEFAULT
                             00000001  01E4   223         .LONG   1
                                   0A  01E8   224         .BYTE   PQL$_WSQUOTA
                             00000001  01E9   225         .LONG   1
                                   00  01ED   226         .BYTE   PQL$_LISTEND
```

```
                                    J 8
                     01EE    228 GET_LIST:                           ; GETJPI list of items and results
                     01EE    229         JPI ACCOUNT,8
                     01FA    230 SHORT_LIST:
                     01FA    231         JPI CPULIM,4
                     0206    232         JPI CURPRIV,8
                     0212    233         JPI GRP,4
                     021E    234         JPI IMAGPRIV,8
                     022A    235         JPI MEM,4
                     0236    236         JPI PRCLM,4
                     0242    237         JPI TQLM,4
                     024E    238         JPI UIC,4
                     025A    239         JPI USERNAME,12
00000044             0266    240         JPI_LIST_SIZE=<USERNAME+2>-ACCOUNT
0000003A             0266    241         JPI_LIST_SIZE1=<USERNAME+2>-CPULIM
                     0266    242 DIRTY:                              ; GETJPI entrys which will vary
                     0266    243         JPI APTCNT,4
                     0272    244         JPI ASTACT,4
                     027E    245         JPI ASTEN,4
                     028A    246         JPI ASTCNT,4
                     0296    247         JPI ASTLM,4
                     02A2    248         JPI AUTHPRIV,8
                     02AE    249         JPI BIOCNT,4
                     02BA    250         JPI BIOLM,4
                     02C6    251         JPI BUFIO,4
                     02D2    252         JPI BYTCNT,4
                     02DE    253         JPI BYTLM,4
                     02EA    254         JPI CPUTIM,4
                     02F6    255         JPI DFPFC,4
                     0302    256         JPI DFWSCNT,4
                     030E    257         JPI DIOCNT,4
                     031A    258         JPI DIOLM,4
                     0326    259         JPI DIRIO,4
                     0332    260         JPI EFCS,4
                     033E    261         JPI EFCU,4
                     034A    262         JPI EFWM,4
                     0356    263         JPI EXCVEC,4
                     0362    264         JPI FINALEXC,4
                     036E    265         JPI FILCNT,4
                     037A    266         JPI FILLM,4
                     0386    267         JPI FREPOVA,4
                     0392    268         JPI FREP1VA,4
                     039E    269         JPI GPGCNT,4
                     03AA    270         JPI IMAGNAME,128
                     03B6    271         JPI LOGINTIM,4
                     03C2    272         JPI OWNER,4
                     03CE    273         JPI PAGEFLTS,4
                     03DA    274         JPI PGFLQUOTA,4
                     03E6    275         JPI PID,4
                     03F2    276         JPI PPGCNT,4
                     03FE    277         JPI PRCCNT,4
                     040A    278         JPI PRCNAM,15
                     0416    279         JPI PROCPRIV,8
                     0422    280         JPI PRI,4
                     042E    281         JPI PRIB,4
                     043A    282         JPI STATE,4
                     0446    283         JPI STS,4
                     0452    284         JPI TMBU,4
```

```
                     045E   285          JPI TQCNT,4
                     046A   286          JPI VOLUMES,4
                     0476   287          JPI VIRTPEAK,4
                     0482   288          JPI WSAUTH,4
                     048E   289          JPI WSQUOTA,4
                     049A   290          JPI WSPEAK,4
                     04A6   291          JPI WSSIZE,4
00000000             04B2   292          .LONG   0                       ; list terminator
```

```
                                04B6   294 JPI_GOOD:                                   ; expected GETJPI results
                                04B6   295                                             ; Item name          buffer offset
            20 54 53 45 54 53 59 53 04B6   296          .ASCII  /SYSTEST /             ; ACCOUNT                 00
                               0008 04BE   297          .WORD   8                      ; ACCOUNTL                08
                                04C0   298 JPI_GOOD_SHRT:
                           00000000 04C0   299          .LONG   0                      ; CPULIM                  0A
                               0004 04C4   300          .WORD   4                      ; CPULIML                 0E
                   00000000 1070BFEF 04C6   301          .QUAD   JPI_PRV_MASK           ; CURPRIV                 10
                               0008 04CE   302          .WORD   8                      ; CURPRIVL                18
                           00000001 04D0   303          .LONG   1                      ; GRP                     1A
                               0002 04D4   304          .WORD   2                      ; GRPL                    1E
                   00000000 00000000 04D6   305          .QUAD   0                      ; IMAGEPRIV               20
                               0008 04DE   306          .WORD   8                      ; IMAGEPRIVL              28
                           00000007 04E0   307          .LONG   7                      ; MEM                     2A
                               0002 04E4   308          .WORD   2                      ; MEML                    2E
                           00000008 04E6   309          .LONG   8                      ; PRCLM                   30
                               0002 04EA   310          .WORD   2                      ; PRCLML                  38
                           00000014 04EC   311          .LONG   ^X14                   ; TQLM                    3A
                               0002 04F0   312          .WORD   2                      ; TQLML                   3E
                           00010007 04F2   313          .LONG   ^X10007                ; UIC                     40
                               0004 04F6   314          .WORD   4                      ; UICL                    44
20 20 20 20 20 54 53 45 54 53 59 53 04F8   315          .ASCII  /SYSTEST     /         ; USERNAME                46
                               000C 0504   316          .WORD   ^XC                    ; USERNAMEL               52
                                0506   317 IN:
4E 49 24 53 59 53 0000050E'010E0000' 0506   318          .ASCID  /SYS$INPUT/
                                54 55 50  0514
                                0517   319 OUT:
55 4F 24 53 59 53 0000051F'010E0000' 0517   320          .ASCID  /SYS$OUTPUT/
                             54 55 50 54  0525
                                0529   321 ERR:
52 45 24 53 59 53 00000531'010E0000' 0529   322          .ASCID  /SYS$ERROR/
                                52 4F 52  0537
                                053A   323 IMAGE_NAME:
54 55 53 54 41 53 00000542'010E0000' 053A   324          .ASCID  /SATSUT01.EXE/
                          45 58 45 2E 31 30  0548
                                054E   325 PROC_NAME:
54 55 53 54 41 53 00000556'010E0000' 054E   326          .ASCID  /SATSUT35/
                                35 33  055C
                           00000565  055E   327          .BLKB   7
                                0565   328 PROC_UIC:
                           00010007  0565   329          .LONG   ^X10007                ; process UIC
```

```
                                0569   331 ;
                                0569   332         .SBTTL  R/W PSECT
                           00000000    333         .PSECT  RWDATA,RD,WRT,NOEXE,LONG
                                0000   334 ;
                                0000   335 PID:
                           00000000     0000   336         .LONG   0                       ; PID for this process
                                0004   337 CURRENT_TC:
                           00000000     0004   338         .LONG   0                       ; ptr to current test case
                                0008   339         .ALIGN LONG                     ; put it on a long word boundry
                                0008   340 REG_SAVE_AREA:
                           00000044     0008   341         .BLKL   15                      ; register save area
                                0044   342 MOD_MSG_CODE:
                           007480D9     0044   343         .LONG   UETP$_SATSMS            ; test module message code for putmsg
                                0048   344 TMN_ADDR:
                           00000000'    0048   345         .ADDRESS TEST_MOD_NAME
                                004C   346 TMD_ADDR:
                           00000019'    004C   347         .ADDRESS TEST_MOD_BEGIN
                                0050   348 PRVPRT:
                                  00    0050   349         .BYTE   0                       ; protection return byte for SETPRT
                                0051   350 PRIVMASK:
                   00000000 00000000    0051   351         .QUAD   0                       ; priv. mask
                                0059   352 CHM_CONT:
                           00000000     0059   353         .LONG   0                       ; change mode continue address
                                005D   354 RETADR:
                           00000065     005D   355         .BLKL   2                       ; returned address's from SETPRT
                                0065   356 STATUS:
                           00000000     0065   357         .LONG   0
                                0069   358 MODE:
                           00000000     0069   359         .LONG   0                       ; current mode string pointer
                                006D   360 REG:
74 73 69 67 65 72 00000075'010E0000'    006D   361         .ASCID  \register R\
                        52 20 72 65     007B
                                007F   362 REGNUM:
                           00000000     007F   363         .LONG   0                       ; register number
                                0083   364 MSGL:
                           00000050     0083   365         .LONG   80                      ; buffer desc.
                           00000133'    0087   366         .ADDRESS BUF
                                008B   367 CRE:
                                008B   368         $CREPRC PID1,0,0,0,0,0,QUOTA_LIST,-
                                008B   369                 0,0,0,0,0                   ; CREPRC parameter list
                                00C3   370 GET:                                         ; GETJPI parameter list's
                                00C3   371         $GETJPI EFN=1,PIDADR=PID1,PRCNAM=TEST_MOD_NAME_D,ITMLST=GET_LIST
                                00E3   372 GET1:
                                00E3   373         $GETJPI ITMLST=GET_LIST
                                0103   374 ITEM_LIST:
                           00000133     0103   375         .BLKL   12
                                0133   376 BUF:
                           00000183     0133   377         .BLKB   80
                                0183   378 ML:
                           00000000     0183   379         .LONG   0                       ; desc. for BUF_CHECK routine
                           00000193'    0187   380         .ADDRESS GETBUF+8
                                018B   381 GETBUF:
                           00000084     018B   382         .LONG   132
                           00000193'    018F   383         .ADDRESS .+4
                           00000217     0193   384         .BLKB   132
                                0217   385 MESSAGEL:
                           00000000     0217   386         .LONG   0                       ; message desc.
```

```
                00000133' 021B   387          .ADDRESS BUF
                          021F   388 SERV_NAME:
                00000000  021F   389          .LONG   0                       ; service name pointer
                          0223   390 MSGVEC1:                                 ; PUTMSG message vector
                00000003  0223   391          .LONG   3
                00741130  0227   392          .LONG   UETP$_TEXT
                00000001  022B   393          .LONG   1
                00000000  022F   394          .LONG   0
                          0233   395 IOSTAT:
            00000000 00000000  0233   396          .QUAD   0                       ; IO status block
                          023B   397 PID1:
                00000000  023B   398          .LONG   0                       ; PID storage location
                          023F   399 MBCHAN:
                    0000  023F   400          .WORD   0                       ; MBX channel location
                          0241   401 MBXUN:
                    0000  0241   402          .WORD   0                       ; MBX unit number
                          0243   403 IOSTATUS:
                0000024B  0243   404          .BLKL   2                       ; MBX read IO status block
                          024B   405 MBUF:
                000002AF  024B   406          .BLKB   100                     ; MBX read buffer
                          02AF   407 TEST_PID:
                00000000  02AF   408          .LONG   0                       ; GETJPI parameter
                          02B3   409 PRIVS:
            00000000 00000000  02B3   410          .QUAD   0                       ; privilege mask
```

```
00000000   412          .PSECT  SATSSS35,RD,WRT,EXE,LONG
     0000  413          .SBTTL  SATSSS35
     0000  414 ;++
     0000  415 ; FUNCTIONAL DESCRIPTION:
     0000  416 ;
     0000  417 ;       After performing some initial housekeeping, such as
     0000  418 ; printing the module begin message and acquiring needed privileges,
     0000  419 ; the system services are tested in each of their normal conditions.
     0000  420 ; Detected failures are identified and  an error message is printed
     0000  421 ; on the terminal.  Upon completion of the test a success or fail
     0000  422 ; message is printed on the terminal.
     0000  423 ;
     0000  424 ; CALLING SEQUENCE:
     0000  425 ;
     0000  426 ;       $ RUN SATSSS35  ...  (DCL COMMAND)
     0000  427 ;
     0000  428 ; INPUT PARAMETERS:
     0000  429 ;
     0000  430 ;       none
     0000  431 ;
     0000  432 ; IMPLICIT INPUTS:
     0000  433 ;
     0000  434 ;       none
     0000  435 ;
     0000  436 ; OUTPUT PARAMETERS:
     0000  437 ;
     0000  438 ;       none
     0000  439 ;
     0000  440 ; IMPLICIT OUTPUTS:
     0000  441 ;
     0000  442 ;       Messages to SYS$OUTPUT are the only output from SATSSS35.
     0000  443 ;       They are of the form:
     0000  444 ;
     0000  445 ;               %UETP-S-SATSMS, TEST MODULE SATSSS35 BEGUN ... (BEGIN MSG)
     0000  446 ;               %UETP-S-SATSMS, TEST MODULE SATSSS35 SUCCESSFUL ... (END MSG)
     0000  447 ;               %UETP-E-SATSMS, TEST MODULE SATSSS35 FAILED ... (END MSG)
     0000  448 ;               %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
     0000  449 ;
     0000  450 ; COMPLETION CODES:
     0000  451 ;
     0000  452 ;       The SATSSS35 routine terminates with a $EXIT to the
     0000  453 ;       operating system with a status code defined by UETP$_SATSMS.
     0000  454 ;
     0000  455 ; SIDE EFFECTS:
     0000  456 ;
     0000  457 ;       none
     0000  458 ;
     0000  459 ;--
     0000  460
     0000  461          TEST_START SATSSS35                     ; let the test begin
```

```
                                0000
                          0000  0000                    .ENTRY SATSSS35,0
                0004'CF    D4   0002                    CLRL    W^CURRENT_TC
                           00   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
                   0DEF    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   0036                    PUSHL   #0
          0BF3'CF    01    FB   0038                    CALLS #1,W^REG_SAVE
                                003D            STP0:
                                003D  462               .SBTTL  CREPRC TESTS
                                003D  463 ;+
                                003D  464 ;
                                003D  465 ; $CREPRC tests
                                003D  466 ;
                                003D  467 ; test the minimum quota all defaults subprocess with _S
                                003D  468 ;
                                003D  469 ;-
      0069'CF    017C'CF    DE   003D  470               MOVAL   W^UM,W^MODE                     ; set the mode
      021F'CF    016E'CF    DE   0044  471               MOVAL   W^CREPRC,W^SERV_NAME            ; set the service name
                                004B  472               $CREMBX_S CHAN=W^MBCHAN,-
                                004B  473                       LOGNAM=W^MBNAM                  ; make something to listen with
                                0062  474               $GETCHN_S CHAN=W^MBCHAN,-
                                0062  475                       PRIBUF=W^GETBUF                 ; get the unit number
      0241'CF    019F'CF    B0   0078  476               MOVW    W^GETBUF+8+DIB$W_UNIT,W^MBXUN   ; and save it
                                007F  477               $CREPRC_S QUOTA=W^QUOTA_LIST,-
                                007F  478                        MBXUNT=W^MBXUN                 ; create a subprocess with _S
                                00A5  479               FAIL_CHECK SS$_NORMAL                   ; check for success
       00000000'8F    DD   00A5                            PUSHL   #SS$_NORMAL
    0BFD'CF    01    FB   00AB                            CALLS   #1,W^REG_CHECK
56   00000000'8F    D0   00B0  480               MOVL    #RMS$_FNF,R6                    ; set exit status code
                57    D4   00B7  481               CLRL    R7                              ; disable PID checking this time
    0D30'CF    00    FB   00B9  482               CALLS   #0,W^CRE_CHECK                  ; check the process exit code
                                00BE  483 ;+
                                00BE  484 ;
                                00BE  485 ; test the PIDADR parameter with _G
                                00BE  486 ;
                                00BE  487 ;-
                                00BE  488               NEXT_TEST
                                00BE
                                00BE            STP1:
    0004'CF    01    D0   00BE                    MOVL    #1,W^CURRENT_TC
                           00   00C3                    PUSHL   #0
    0BF3'CF    01    FB   00C5                    CALLS   #1,W^REG_SAVE
00B7'CF   0241'CF    B0   00CA  489               MOVW    W^MBXUN,W^CRE+CREPRC$_MBXUNT    ; set the MBX unit number
                                00D1  490               $CREPRC_G W^CRE                 ; try _G and PIDADR param.
                                00DA  491               FAIL_CHECK SS$_NORMAL           ; check for success
       00000000'8F    DD   00DA                            PUSHL   #SS$_NORMAL
    0BFD'CF    01    FB   00E0                            CALLS   #1,W^REG_CHECK
                57    D6   00E5  492               INCL    R7                              ; enable PID checking
    0D30'CF    00    FB   00E7  493               CALLS   #0,W^CRE_CHECK                  ; check the process exit code
56   00000000'8F    D0   00EC  494               MOVL    #SS$_NORMAL,R6                  ; set expected status return
```

```
                        00F3   495 ;+
                        00F3   496 ;
                        00F3   497 ; test the IMAGE param. with _S
                        00F3   498 ;
                        00F3   499 ;-
                        00F3   500          NEXT_TEST
                        00F3
                        00F3        STP2:
0004'CF    02    D0     00F3                    MOVL    #2,W^CURRENT_TC
           00    DD     00F8                    PUSHL   #0
0BF3'CF    01    FB     00FA                    CALLS   #1,W^REG_SAVE
                        00FF   501          $CREPRC_S QUOTA=W^QUOTA_LIST,-
                        00FF   502                    IMAGE=W^IMAGE_NAME,-
                        00FF   503                   MBXUNT=W^MBXUN,-
                        00FF   504                   PIDADR=W^PID1                  ; try _S with IMAGE param.
                        0129   505          FAIL_CHECK SS$_NORMAL                   ; check success
 00000000'8F    DD     0129                    PUSHL   #SS$_NORMAL
0BFD'CF    01    FB     012F                    CALLS   #1,W^REG_CHECK
                        0134   506          $WAKE_S PIDADR = W^PID1                 ; cause process termination
0D30'CF    00    FB     0141   507          CALLS   #0,W^CRE_CHECK                  ; check the process exit code
                        0146   508 ;+
                        0146   509 ;
                        0146   510 ; test the INPUT param. with _G
                        0146   511 ;
                        0146   512 ;-
                        0146   513          NEXT_TEST
                        0146
                        0146        STP3:
0004'CF    03    D0     0146                    MOVL    #3,W^CURRENT_TC
           00    DD     014B                    PUSHL   #0
0BF3'CF    01    FB     014D                    CALLS   #1,W^REG_SAVE
0093'CF    053A'CF  DE   0152   514          MOVAL   W^IMAGE_NAME,W^CRE+CREPRC$_IMAGE ; set image name
008F'CF    023B'CF  DE   0159   515          MOVAL   W^PID1,W^CRE+CREPRC$_PIDADR     ; set the PID save address
0097'CF    0506'CF  DE   0160   516          MOVAL   W^IN,W^CRE+CREPRC$_INPUT        ; set the INPUT param.
                        0167   517          $CREPRC_G W^CRE                         ; try it
                        0170   518          FAIL_CHECK SS$_NORMAL                   ; check success
 00000000'8F    DD     0170                    PUSHL   #SS$_NORMAL
0BFD'CF    01    FB     0176                    CALLS   #1,W^REG_CHECK
                        017B   519          $WAKE_S PIDADR = W^PID1                 ; cause process termination
0D30'CF    00    FB     0188   520          CALLS   #0,W^CRE_CHECK                  ; check the process exit code
                        018D   521 ;+
                        018D   522 ;
                        018D   523 ; test the OUTPUT param. with _S
                        018D   524 ;
                        018D   525 ;-
                        018D   526          NEXT_TEST
                        018D
                        018D        STP4:
0004'CF    04    D0     018D                    MOVL    #4,W^CURRENT_TC
           00    DD     0192                    PUSHL   #0
0BF3'CF    01    FB     0194                    CALLS   #1,W^REG_SAVE
                        0199   527          $CREPRC_S PIDADR=W^PID1,-
                        0199   528                    IMAGE =W^IMAGE_NAME,-
                        0199   529                    INPUT =W^IN,-
                        0199   530                    OUTPUT=W^OUT,-
                        0199   531                    MBXUNT=W^MBXUN,-
                        0199   532                    QUOTA =W^QUOTA_LIST          ; try _S with OUT param.
```

```
                     01C7   533          FAIL_CHECK SS$_NORMAL                    ; chec success
 00000000'8F   DD    01C7                       PUSHL   #SS$_NORMAL
0BFD'CF    01    FB  01CD                       CALLS   #1,W^REG_CHECK
                     01D2   534          $WAKE_S PIDADR = W^PID1                  ; cause process termination
0D30'CF    00    FB  01DF   535          CALLS   #0,W^CRE_CHECK                   ; check process exit code
                     01E4   536 ;+
                     01E4   537 ;
                     01E4   538 ; test ERROR param. with _G
                     01E4   539 ;
                     01E4   540 ;-
                     01E4   541          NEXT_TEST
                     01E4
                     01E4        STP5:
0004'CF    05    D0  01E4                       MOVL    #5,W^CURRENT_TC
           00    DD  01E9                       PUSHL   #0
0BF3'CF    01    FB  01EB                       CALLS   #1,W^REG_SAVE
009B'CF  0517'CF   DE  01F0   542          MOVAL   W^OUT,W^CRE+CREPRC$_OUTPUT       ; set the output param.
009F'CF  0529'CF   DE  01F7   543          MOVAL   W^ERR,W^CRE+CREPRC$_ERROR        ; set the error output param
                     01FE   544          $CREPRC_G W^CRE                          ; try _G with ERROR param
                     0207   545          FAIL_CHECK SS$_NORMAL                    ; check for success
 00000000'8F   DD    0207                       PUSHL   #SS$_NORMAL
0BFD'CF    01    FB  020D                       CALLS   #1,W^REG_CHECK
                     0212   546          $WAKE_S PIDADR = W^PID1                  ; cause process termination
0D30'CF    00    FB  021F   547          CALLS   #0,W^CRE_CHECK                   ; check process exit code
                     0224   548 ;+
                     0224   549 ;
                     0224   550 ; test PRVADR param with _S
                     0224   551 ;
                     0224   552 ;-
                     0224   553          NEXT_TEST
                     0224
                     0224        STP6:
0004'CF    06    D0  0224                       MOVL    #6,W^CURRENT_TC
           00    DD  0229                       PUSHL   #0
0BF3'CF    01    FB  022B                       CALLS   #1,W^REG_SAVE
                     0230   554          $CREPRC_S PIDADR=W^PID1,-
                     0230   555                    IMAGE =W^IMAGE_NAME,-
                     0230   556                    INPUT =W^IN,-
                     0230   557                    OUTPUT=W^OUT,-
                     0230   558                    ERROR =W^ERR,-
                     0230   559                    PRVADR=W^PRIVS,-
                     0230   560                    MBXUNT=W^MBXUN,-
                     0230   561                    QUOTA =W^QUOTA_LIST              ; try _S with PRVADR param
                     0262   562          FAIL_CHECK SS$_NORMAL                    ; check success
 00000000'8F   DD    0262                       PUSHL   #SS$_NORMAL
0BFD'CF    01    FB  0268                       CALLS   #1,W^REG_CHECK
                     026D   563          $WAKE_S PIDADR = W^PID1                  ; cause process termination
0D30'CF    00    FB  027A   564          CALLS   #0,W^CRE_CHECK                   ; check image exit status
                     027F   565 ;+
                     027F   566 ;
                     027F   567 ; test PRCNAM param with _G
                     027F   568 ;
                     027F   569 ;-
                     027F   570          NEXT_TEST
                     027F
                     027F        STP7:
0004'CF    07    D0  027F                       MOVL    #7,W^CURRENT_TC
```

```
                    00    DD  0284                        PUSHL   #0
          0BF3'CF   01    FB  0286                        CALLS   #1,W^REG_SAVE
009F'CF   0529'CF       DE  028B   571              MOVAL   W^ERR,W^CRE+CREPRC$_ERROR        ; set the ERROR param.
00A3'CF   02B3'CF       DE  0292   572              MOVAL   W^PRIVS,W^CRE+CREPRC$_PRVADR     ; set the PRVADR param.
00AB'CF   054E'CF       DE  0299   573              MOVAL   W^PROC_NAME,W^CRE+CREPRC$_PRCNAM ; set the process name
                            02A0   574              $CREPRC_G W^CRE                          ; try _G with a PRCNAM
                            02A9   575              FAIL_CHECK SS$_NORMAL                    ; check success
     00000000'8F    DD  02A9                        PUSHL   #SS$_NORMAL
          0BFD'CF   01    FB  02AF                        CALLS   #1,W^REG_CHECK
021F'CF   0175'CF       DE  02B4   576              MOVAL   W^GETJPI,W^SERV_NAME             ; set service name
                            02BB   577              $GETJPI_S PIDADR = W^PID1,-
                            02BB   578                        ITMLST = W^GET_LIST            ; get the process name
                            02D2   579              FAIL_CHECK SS$_NORMAL                    ; check success
     00000000'8F    DD  02D2                        PUSHL   #SS$_NORMAL
          0BFD'CF   01    FB  02D8                        CALLS   #1,W^REG_CHECK
021F'CF   016E'CF       DE  02DD   580              MOVAL   W^CREPRC,W^SERV_NAME             ; set service name
0556'CF   01A2'CF       0F  29  02E4   581              CMPC3   #15,W^PRCNAM,W^PROC_NAME+8        ; correct process name?
                    09  13  02EC   582              BEQL    10$                              ; br if OK
          0109'CF   DF  02EE   583              PUSHAL  W^PNS                            ; push string variable
          0C3F'CF   01  FB  02F2   584              CALLS   #1,W^PRINT_FAIL                  ; print the failure
                            02F7   585 10$:
                            02F7   586              $WAKE_S PIDADR = W^PID1                  ; cause process termination
          0D30'CF   00  FB  0304   587              CALLS   #0,W^CRE_CHECK                   ; check image exit status
                            0309   588 ;+
                            0309   589 ;
                            0309   590 ; test BASPRI with _S and a lower priority
                            0309   591 ;
                            0309   592 ;-
                            0309   593              NEXT_TEST
                            0309
                            0309       STP8:
0004'CF   08  D0  0309                          MOVL    #8,W^CURRENT_TC
                    0C  DD  030E                        PUSHL   #0
          0BF3'CF   01  FB  0310                        CALLS   #1,W^REG_SAVE
                            0315   594              $CREPRC_S PIDADR = W^PIDT,-
                            0315   595                        IMAGE = W^IMAGE_NAME,-
                            0315   596                        INPUT = W^IN,-
                            0315   597                       OUTPUT = W^OUT,-
                            0315   598                        ERROR = W^ERR,-
                            0315   599                       BASPRI = #1,-
                            0315   600                       PRVADR = W^PRIVS,-
                            0315   601                       MBXUNT = W^MBXUN,-
                            0315   602                        QUOTA = W^QUOTA_LIST        ; try all that
                            0347   603              FAIL_CHECK SS$_NORMAL                    ; check success
     00000000'8F    DD  0347                        PUSHL   #SS$_NORMAL
          0BFD'CF   01    FB  034D                        CALLS   #1,W^REG_CHECK
021F'CF   0175'CF       DE  0352   604              MOVAL   W^GETJPI,W^SERV_NAME             ; set service name
                            0359   605              $GETJPI_S PIDADR = W^PID1,-
                            0359   606                        ITMLST = W^GET_LIST            ; get the base priority
                            0370   607              FAIL_CHECK SS$_NORMAL                    ; check success
     00000000'8F    DD  0370                        PUSHL   #SS$_NORMAL
          0BFD'CF   01    FB  0376                        CALLS   #1,W^REG_CHECK
021F'CF   016E'CF       DE  037B   608              MOVAL   W^CREPRC,W^SERV_NAME             ; set service name
          01C3'CF   01  D1  0382   609              CMPL    #1,W^PRIB                        ; is it correct?
                    0F  13  0387   610              BEQL    20$                              ; br if OK
               01C3'CF  DD  0389   611              PUSHL   W^PRIB                           ; push received
                    01  DD  038D   612              PUSHL   #1                               ; push expected
```

```
                                                G 9
                00F8'CF    DF  038F    613          PUSHAL  W^BP                                ; push str variable
           0C3F'CF   03    FB  0393    614          CALLS   #3,W^PRINT_FAIL                     ; print the failure
                           0398    615 20$:
                           0398    616          $WAKE_S PIDADR = W^PID1                        ; cause process termination
           0D30'CF   00    FB  03A5    617          CALLS   #0,W^CRE_CHECK                     ; check image exit status
                           03AA    618 ;+
                           03AA    619 ;
                           03AA    620 ; test BASPRI with _S and a higher priority
                           03AA    621 ;
                           03AA    622 ;-
                           03AA    623          NEXT_TEST
                           03AA
                           03AA        STP9:
           0004'CF   09    D0  03AA                 MOVL    #9,W^CURRENT_TC
                     00    DD  03AF                 PUSHL   #0
           0BF3'CF   01    FB  03B1                 CALLS   #1,W^REG_SAVE
                           03B6    624          MODE    TO,25$,KRNL,NOREGS                     ; kernal mode to access PHD
     59  00000000'9F   D0  03D3    625          MOVL    @#CTL$GL_PHD,R9                        ; get process header address
           0051'CF   69    DE  03DA    626          MOVAL   PHD$Q_PRIVMSK(R9),W^PRIVMASK           ; get priv mask address
                           03DF    627          MODE    FROM,25$                               ; get back to user mode
                           03E0    628          PRIV    ADD,SETPRI                             ; add SETPRI priv
                     00    DD  0400    629          PUSHL   #0                                     ; push a dummy parameter
           0BF3'CF   01    FB  0402    630          CALLS   #1,W^REG_SAVE                          ; save the registers
                           0407    631          $CREPRC_S PIDADR = W^PID1,-
                           0407    632                    IMAGE = W^IMAGE_NAME,-
                           0407    633                    INPUT = W^IN,-
                           0407    634                   OUTPUT = W^OUT,-
                           0407    635                    ERROR = W^ERR,-
                           0407    636                   BASPRI = #4,-
                           0407    637                   PRVADR = W^PRIVS,-
                           0407    638                   MBXUNT = W^MBXUN,-
                           0407    639                    QUOTA = W^QUOTA_LIST                  ; try _S higher priority
                           0439    640          FAIL_CHECK SS$_NORMAL                          ; check success
      00000000'8F   DD  0439                        PUSHL   #SS$_NORMAL
           0BFD'CF   01    FB  043F                 CALLS   #1,W^REG_CHECK
021F'CF    0175'CF   DE  0444    641          MOVAL   W^GETJPI,W^SERV_NAME                   ; set the service name
                           044B    642          $GETJPI_S PIDADR = W^PID1,-
                           044B    643                    ITMLST = W^GET_LIST                   ; get the base priority
                           0462    644          FAIL_CHECK SS$_NORMAL                          ; check success
      00000000'8F   DD  0462                        PUSHL   #SS$_NORMAL
           0BFD'CF   01    FB  0468                 CALLS   #1,W^REG_CHECK
021F'CF    016E'CF   DE  046D    645          MOVAL   W^CREPRC,W^SERV_NAME                   ; reset the service name
           01C3'CF   04    D1  0474    646          CMPL    #4,W^PRIB                              ; is the priority OK?
                     0F    13  0479    647          BEQL    30$                                    ; br if OK
                01C3'CF    DD  047B    648          PUSHL   W^PRIB                                 ; push received
                     04    DD  047F    649          PUSHL   #4                                     ; push expected
                00F8'CF    DF  0481    650          PUSHAL  W^BP                                   ; push the str variable
           0C3F'CF   03    FB  0485    651          CALLS   #3,W^PRINT_FAIL                        ; print the failure
                           048A    652 30$:
                           048A    653          $WAKE_S PIDADR = W^PID1                        ; cause process termination
           0D30'CF   00    FB  0497    654          CALLS   #0,W^CRE_CHECK                     ; check image exit status
                           049C    655 ;+
                           049C    656 ;
                           049C    657 ; test detached process
                           049C    658 ;
                           049C    659 ;-
                           049C    660          NEXT_TEST
```

```
                                049C
                                049C        STP10:
        0004'CF    0A    D0     049C                MOVL    #10,W^CURRENT_TC
                   00    DD     04A1                PUSHL   #0
        0BF3'CF    01    FB     04A3                CALLS   #1,W^REG_SAVE
                                04A8    661         $CREPRC_S PIDADR = W^PID1,-
                                04A8    662                   IMAGE = W^IMAGE_NAME,-
                                04A8    663                   INPUT = W^IN,-
                                04A8    664                  OUTPUT = W^OUT,-
                                04A8    665                   ERROR = W^ERR,-
                                04A8    666                  BASPRI = #2,-
                                04A8    667                  PRVADR = W^PRIVS,-
                                04A8    668                  MBXUNT = W^MBXUN,-
                                04A8    669                   QUOTA = W^QUOTA_LIST,-
                                04A8    670                     UIC = W^PROC_UIC       ; try _S and all this
                                04DC    671         FAIL_CHECK SS$_NORMAL              ; check success
         00000000'8F    DD     04DC                PUSHL   #SS$_NORMAL
        0BFD'CF    01    FB     04E2                CALLS   #1,W^REG_CHECK
021F'CF    0175'CF    DE     04E7    672         MOVAL   W^GETJPI,W^SERV_NAME       ; set service name
                                04EE    673         $GETJPI_S PIDADR = W^PID1,-
                                04EE    674                   ITMLST = W^GET_LIST  ; get the process UIC
                                0505    675         FAIL_CHECK SS$_NORMAL              ; check success
         00000000'8F    DD     0505                PUSHL   #SS$_NORMAL
        0BFD'CF    01    FB     050B                CALLS   #1,W^REG_CHECK
021F'CF    016E'CF    DE     0510    676         MOVAL   W^CREPRC,W^SERV_NAME       ; reset the service name
0565'CF    003C'CF    D1     0517    677         CMPL    W^UIC,W^PROC_UIC           ; is the UIC correct?
                   11    13     051E    678         BEQL    40$                        ; br if OK
           003C'CF    DD     0520    679         PUSHL   W^UIC                      ; push received
           0565'CF    DD     0524    680         PUSHL   W^PROC_UIC                 ; push expected
           0144'CF    DF     0528    681         PUSHAL  W^UIC_MSG                  ; push the string variable
        0C3F'CF    03    FB     052C    682         CALLS   #3,W^PRINT_FAIL            ; print the failure
                                0531    683 40$:
                                0531    684         $WAKE_S PIDADR = W^PID1            ; cause process termination
        0D30'CF    00    FB     053E    685         CALLS   #0,W^CRE_CHECK             ; check the process exit status
                                0543    686 ;+
                                0543    687 ;
                                0543    688 ; test the STSFLG's _S with all set
                                0543    689 ;
                                0543    690 ;-
                                0543    691         NEXT_TEST
                                0543
                                0543        STP11:
        0004'CF    0B    D0     0543                MOVL    #11,W^CURRENT_TC
                   00    DD     0548                PUSHL   #0
        0BF3'CF    01    FB     054A                CALLS   #1,W^REG_SAVE
                                054F    692         MODE    TO,45$,KRNL,NOREGS         ; kernal mode to access PHD
 59      00000000'9F    D0     056C    693         MOVL    @#CTL$GL_PHD,R9            ; get process header address
        0051'CF    69    DE     0573    694         MOVAL   PHD$Q_PRIVMSK(R9),W^PRIVMASK ; get priv mask address
                                0578    695         MODE    FROM,45$                   ; get back to user mode
                                0579    696         PRIV    ADD,PSWAPM                 ; add PSWAPM priv
                                0599    697         PRIV    ADD,NOACNT                 ; add NOACNT priv
                                05B9    698         PRIV    ADD,NETMBX                 ; add NETMBX priv
                   00    DD     05D9    699         PUSHL   #0                         ; push a dummy param
        0BF3'CF    01    FB     05DB    700         CALLS   #1,W^REG_SAVE              ; save a reg snap shot
021F'CF    0175'CF    DE     05E0    701         MOVAL   W^GETJPI,W^SERV_NAME       ; set service name
                                05E7    702         $GETJPI_S ITMLST = W^GET_LIST      ; get the current process privs
                                05FC    703         FAIL_CHECK SS$_NORMAL              ; check success
```

```
                00000000'8F    DD  05FC                        PUSHL   #SS$_NORMAL
              0BFD'CF    01    FB  0602                        CALLS   #1,W^REG_CHECK
      021F'CF    016E'CF       DE  0607   704          MOVAL   W^CREPRC,W^SERV_NAME            ; set the service name
      02B3'CF    0010'CF       7D  060E   705          MOVQ    W^CURPRIV,W^PRIVS               ; set the detached process privs
                               0615   706          $CREPRC_S PIDADR = W^PID1,-
                               0615   707                    IMAGE = W^IMAGE_NAME,-
                               0615   708                    INPUT = W^IN,-
                               0615   709                   OUTPUT = W^OUT,-
                               0615   710                    ERROR = W^ERR,-
                               0615   711                   BASPRI = #2,-
                               0615   712                   PRVADR = W^PRIVS,-
                               0615   713                   MBXUNT = W^MBXUN,-
                               0615   714                    QUOTA = W^QUOTA_LIST,-
                               0615   715                      UIC = W^PROC_UIC,-
                               0615   716                   STSFLG = #^XFF            ; try every thing _S
                               064D   717          FAIL_CHECK SS$_NORMAL             ; check success
                00000000'8F    DD  064D                        PUSHL   #SS$_NORMAL
              0BFD'CF    01    FB  0653                        CALLS   #1,W^REG_CHECK
      021F'CF    0175'CF       DE  0658   718          MOVAL   W^GETJPI,W^SERV_NAME            ; set service name
                               065F   719          $GETJPI_S PIDADR = W^PID1,-
                               065F   720                    ITMLST = W^GET_LIST           ; get process status flags
                               0676   721          FAIL_CHECK SS$_NORMAL             ; check success
                00000000'8F    DD  0676                        PUSHL   #SS$_NORMAL
              0BFD'CF    01    FB  067C                        CALLS   #1,W^REG_CHECK
      021F'CF    016E'CF       DE  0681   722          MOVAL   W^CREPRC,W^SERV_NAME            ; reset service name
 000001CF'EF     FFC739FF 8F   CA  0688   723          BICL    #JPI_STS_NMASK,STS              ; clear out any extraneous
                               0693   724                                                    ; bits set by the Swapper
     0038C600 8F    01CF'CF    D1  0693   725          CMPL    W^STS,#JPI_STS_MASK             ; flags OK?
                       13      13  069C   726          BEQL    50$                             ; br if OK
                01CF'CF        DD  069E   727          PUSHL   W^STS                           ; push received
             0038C600 8F       DD  06A2   728          PUSHL   #JPI_STS_MASK                   ; push expected
                 0134'CF       DF  06A8   729          PUSHAL  W^STSFLGS                       ; push str variable
              0C3F'CF    03    FB  06AC   730          CALLS   #3,W^PRINT_FAIL                 ; print the failure
                               06B1   731 50$:
                               06B1   732          $DELPRC_S PIDADR = W^PID1                 ; needed for bit 5 in STS
                       56      D4  06BE   733          CLRL    R6                              ; set expected status return
              0D30'CF    00    FB  06C0   734          CALLS   #0,W^CRE_CHECK                  ; check image exit status
          56   00000000'8F    D0  06C5   735          MOVL    #SS$_NORMAL,R6                  ; reset the expected status return
                               06CC   736 ;+
                               06CC   737 ;
                               06CC   738 ; test the STSFLG's _G all clear
                               06CC   739 ;
                               06CC   740 ;-
                               06CC   741          NEXT_TEST
                               06CC
                               06CC       STP12:
                0004'CF    0C  D0  06CC                        MOVL    #12,W^CURRENT_TC
                           00  DD  06D1                        PUSHL   #0
                0BF3'CF    01  FB  06D3                        CALLS   #1,W^REG_SAVE
      00A3'CF    02B3'CF       DE  06D8   742          MOVAL   W^PRIVS,W^CRE+CREPRC$_PRVADR    ; setup PRVADR parameter
      00AB'CF    054E'CF       DE  06DF   743          MOVAL   W^PROC_NAME,W^CRE+CREPRC$_PRCNAM ; setup PRCNAM parameter
          00AF'CF    02       D0  06E6   744          MOVL    #2,W^CRE+CREPRC$_BASPRI         ; setup BASPRI parameter
      00B3'CF    0565'CF       D0  06EB   745          MOVL    W^PROC_UIC,W^CRE+CREPRC$_UIC    ; setup UIC parameter
                               06F2   746          $CREPRC_G W^CRE                         ; try it all _G
      021F'CF    0175'CF       DE  06FB   747          MOVAL   W^GETJPI,W^SERV_NAME            ; set service name
                               0702   748          $GETJPI_S PIDADR = W^PID1,-
                               0702   749                    ITMLST = W^GET_LIST           ; get the process status flags
```

```
                                    0719    750          FAIL_CHECK SS$_NORMAL                          ; check success
                 00000000'8F    DD  0719                             PUSHL   #SS$_NORMAL
               0BFD'CF    01    FB  071F                             CALLS   #1,W^REG_CHECK
          021F'CF    016E'CF    DE  0724    751          MOVAL   W^CREPRC,W^SERV_NAME                   ; set the service name
     000001CF'EF    FFC739FF 8F    CA  072B    752          BICL    #JPI_STS_NMASK,STS                     ; clear out any extraneous
                                    0736    753                                                         ; bits set by the Swapper
                 00    01CF'CF    D1  0736    754          CMPL    W^STS,#^X0                             ; are they all OK?
                             0F    13  073B    755          BEQL    60$                                    ; br if OK
                       01CF'CF    DD  073D    756          PUSHL   W^STS                                  ; push received
                             00    DD  0741    757          PUSHL   #^X0                                   ; push expected
                       0134'CF    DF  0743    758          PUSHAL  W^STSFLGS                              ; push str variable
                 0C3F'CF    03    FB  0747    759          CALLS   #3,W^PRINT_FAIL                        ; print the failure
                                    074C    760 60$:
                                    074C    761          $WAKE_S PIDADR = W^PID1                        ; cause process termination
                 0D30'CF    00    FB  0759    762          CALLS   #0,W^CRE_CHECK                         ; check image exit status
```

```
                                075E    764 .SBTTL GETJPI TESTS
                                075E    765 ;+
                                075E    766 ;
                                075E    767 ; $GETJPI tests
                                075E    768 ;
                                075E    769 ; test the default case with all items _S
                                075E    770 ;
                                075E    771 ;-
                                075E    772         NEXT_TEST
                                075E
                                075E        STP13:
        0004'CF    0D   DO      075E                MOVL    #13,W^CURRENT_TC
                   00   DD      0763                PUSHL   #0
        0BF3'CF    01   FB      0765                CALLS   #1,W^REG_SAVE
021F'CF   0175'CF   DE      076A    773         MOVAL   W^GETJPI,W^SERV_NAME            ; set service name
0069'CF   017C'CF   DE      0771    774         MOVAL   W^UM,W^MODE                     ; set the mode
                            0778    775         PRIV    ADD,SETPRV                      ; get ready to set privileges
                            0798    776                                                 ; to known set for testing
                            0798    777         $SETPRV_S ENBFLG = #1,-                 ; enable expected privileges
                            0798    778                   PRVADR = PRVMASK
                            07AB    779         $SETPRV_S PRVADR = NPRVMASK             ; disable any extra privileges
                            07BE    780                                                 ; set in SYSTEST account
                            07BE    781         FAIL_CHECK SS$_NORMAL                   ; check success
 00000000'8F    DD      07BE                PUSHL   #SS$_NORMAL
    0BFD'CF    01   FB      07C4                CALLS   #1,W^REG_CHECK
                            07C9    782         $GETJPI_S ITMLST = W^GET_LIST           ; try _S
                            07DE    783         FAIL_CHECK SS$_NORMAL                   ; check success
 00000000'8F    DD      07DE                PUSHL   #SS$_NORMAL
    0BFD'CF    01   FB      07E4                CALLS   #1,W^REG_CHECK
    56   0000'CF   DE      07E9    784         MOVAL   W^ACCOUNT,R6                    ; set questionable data adr
    57   04B6'CF   DE      07EE    785         MOVAL   W^JPI_GOOD,R7                   ; set good data adr
58   00000044 8F   DO      07F3    786         MOVL    #JPI_LIST_SIZE,R8               ; set the byte count
    0DDA'CF    00   FB      07FA    787         CALLS   #0,W^JPI_CHECK                  ; check the results
                            07FF    788 ;+
                            07FF    789 ;
                            07FF    790 ; test _G default case with all items
                            07FF    791 ;
                            07FF    792 ;-
                            07FF    793         NEXT_TEST
                            07FF
                            07FF        STP14:
    0004'CF    0E   DO      07FF                MOVL    #14,W^CURRENT_TC
                   00   DD      0804                PUSHL   #0
    0BF3'CF    01   FB      0806                CALLS   #1,W^REG_SAVE
                            080B    794         $GETJPI_G W^GET1                        ; try _G
                            0814    795         FAIL_CHECK SS$_NORMAL                   ; check success
 00000000'8F    DD      0814                PUSHL   #SS$_NORMAL
    0BFD'CF    01   FB      081A                CALLS   #1,W^REG_CHECK
    0DDA'CF    00   FB      081F    796         CALLS   #0,W^JPI_CHECK                  ; check the results
                            0824    797 ;+
                            0824    798 ;
                            0824    799 ; test local EFN
                            0824    800 ;
                            0824    801 ;-
                            0824    802         NEXT_TEST
                            0824
                            0824        STP15:
```

```
          0004'CF   0F   D0  0824                 MOVL    #15,W^CURRENT_TC
                    00   DD  0829                 PUSHL   #0
          0BF3'CF   01   FB  082B                 CALLS   #1,W^REG_SAVE
  021F'CF   016E'CF      DE  0830  803       MOVAL   W^CREPRC,W^SERV_NAME          ; set service name
                             0837  804       $CREPRC_S QUOTA  = W^QUOTA_LIST,-
                             0837  805                 IMAGE  = W^IMAGE_NAME,-
                             0837  806                 PIDADR = W^PID1,-
                             0837  807                 PRCNAM = W^PROC_NAME        ; create the target process
                             0861  808       FAIL_CHECK SS$_NORMAL                 ; check for success
       00000000'8F       DD  0861                 PUSHL   #SS$_NORMAL
          0BFD'CF   01   FB  0867                 CALLS   #1,W^REG_CHECK
  021F'CF   0175'CF      DE  086C  809       MOVAL   W^GETJPI,W^SERV_NAME          ; reset the service name
                             0873  810       $GETJPI_S EFN    = #1,-
                             0873  811                 ITMLST = W^GET_LIST         ; try _S with EFN
                             0888  812       FAIL_CHECK SS$_NORMAL                 ; check success
       00000000'8F       DD  0888                 PUSHL   #SS$_NORMAL
          0BFD'CF   01   FB  088E                 CALLS   #1,W^REG_CHECK
                             0893  813       $WAITFR_S EFN    = #1                 ; wait for completion
          0DDA'CF   00   FB  089C  814       CALLS   #0,W^JPI_CHECK                ; check the results
          00C7'CF   01   D0  08A1  815       MOVL    #1,W^GET+GETJPI$_EFN          ; set the EFN
  00CB'CF   023B'CF      DE  08A6  816       MOVAL   W^PID1,W^GET+GETJPI$_PIDADR   ; set the target process PID
                             08AD  817       $GETJPI_G W^GET                       ; try _G with target process
                             08B6  818       FAIL_CHECK SS$_NORMAL                 ; check success
       00000000'8F       DD  08B6                 PUSHL   #SS$_NORMAL
          0BFD'CF   01   FB  08BC                 CALLS   #1,W^REG_CHECK
                             08C1  819       $WAITFR_S EFN    = #1                 ; wait for completion
          0DDA'CF   00   FB  08CA  820       CALLS   #0,W^JPI_CHECK                ; check the results
                             08CF  821  ;+
                             08CF  822  ;
                             08CF  823  ; test common EFN with _S
                             08CF  824  ;
                             08CF  825  ;-
                             08CF  826       NEXT_TEST
                             08CF
                             08CF       STP16:
          0004'CF   10   D0  08CF                 MOVL    #16,W^CURRENT_TC
                    00   DD  08D4                 PUSHL   #0
          0BF3'CF   01   FB  08D6                 CALLS   #1,W^REG_SAVE
                             08DB  827       $ASCEFC_S EFN  = #65,-
                             08DB  828                 NAME = W^EFC_NAME           ; get a common EF
                             08F0  829       $GETJPI_S EFN  = #65,-
                             08F0  830               ITMLST = W^GET_LIST           ; try _S with CEFN
                             0909  831       FAIL_CHECK SS$_NORMAL                 ; check success
       00000000'8F       DD  0909                 PUSHL   #SS$_NORMAL
          0BFD'CF   01   FB  090F                 CALLS   #1,W^REG_CHECK
                             0914  832       $WAITFR_S EFN  = #65                  ; wait for completion
          0DDA'CF   00   FB  0921  833       CALLS   #0,W^JPI_CHECK                ; check results
 00C7'CF   00000041 8F   D0  0926  834       MOVL    #65,W^GET+GETJPI$_EFN         ; set the common EFC
                             092F  835       $GETJPI_G  W^GET                      ; try _G, CEFC, and target process
                             0938  836       FAIL_CHECK SS$_NORMAL                 ; check for success
       00000000'8F       DD  0938                 PUSHL   #SS$_NORMAL
          0BFD'CF   01   FB  093E                 CALLS   #1,W^REG_CHECK
                             0943  837       $WAITFR_S EFN  = #65                  ; wait for completion
          0DDA'CF   00   FB  0950  838       CALLS   #0,W^JPI_CHECK                ; check the results
                             0955  839       $DACEFC_S EFN  = #65                  ; release the CEFC
                             0962  840
                             0962  841  ;+
```

```
                                                M 9
                     0962   842 ;
                     0962   843 ; test PIDADR
                     0962   844 ;
                     0962   845 ;-
                     0962   846         NEXT_TEST
                     0962
                     0962       STP17:
0004'CF    11   D0  0962                 MOVL    #17,W^CURRENT_TC
           00   DD  0967                 PUSHL   #0
0BF3'CF    01   FB  0969                 CALLS   #1,W^REG_SAVE
                     096E   847         $GETJPI_S EFN    = #2,-
                     096E   848                   PIDADR = W^PID1,-
                     096E   849                   ITMLST = W^GET_LIST          ; try _S with PID
                     0985   850         FAIL_CHECK SS$_NORMAL                  ; check success
 00000000'8F   DD  0985                 PUSHL   #SS$_NORMAL
0BFD'CF    01   FB  098B                 CALLS   #1,W^REG_CHECK
                     0990   851         $WAITFR_S EFN = #2                      ; wait for completion
0DDA'CF    00   FB  0999   852         CALLS   #0,W^JPI_CHECK                 ; check the results
                     099E   853 ;+
                     099E   854 ;
                     099E   855 ; test PRCNAM
                     099E   856 ;
                     099E   857 ;-
                     099E   858         NEXT_TEST
                     099E
                     099E       STP18:
0004'CF    12   D0  099E                 MOVL    #18,W^CURRENT_TC
           00   DD  09A3                 PUSHL   #0
0BF3'CF    01   FB  09A5                 CALLS   #1,W^REG_SAVE
                     09AA   859         $GETJPI_S EFN    = #3,-
                     09AA   860                   PRCNAM = W^PROC_NAME,-
                     09AA   861                   ITMLST = W^GET_LIST          ; try _S with PRCNAM
                     09C1   862         FAIL_CHECK SS$_NORMAL                  ; check success
 00000000'8F   DD  09C1                 PUSHL   #SS$_NORMAL
0BFD'CF    01   FB  09C7                 CALLS   #1,W^REG_CHECK
                     09CC   863         $WAITFR_S EFN    = #3                  ; wait for completion
0DDA'CF    00   FB  09D5   864         CALLS   #0,W^JPI_CHECK                 ; check the results
                     09DA   865         $GETJPI_S EFN    = #16,-
                     09DA   866                   PRCNAM = W^TEST_MOD_NAME_D,-
                     09DA   867                   ITMLST = W^GET_LIST          ; try _S with PRCNAM on self
                     09F1   868         FAIL_CHECK SS$_NORMAL                  ; check success
 00000000'8F   DD  09F1                 PUSHL   #SS$_NORMAL
0BFD'CF    01   FB  09F7                 CALLS   #1,W^REG_CHECK
                     09FC   869         $WAITFR_S EFN    = #16                 ; wait for completion
0DDA'CF    00   FB  0A05   870         CALLS   #0,W^JPI_CHECK                 ; check the results
                     0A0A   871 ;+
                     0A0A   872 ;
                     0A0A   873 ; test IOSB
                     0A0A   874 ;
                     0A0A   875 ;-
                     0A0A   876         NEXT_TEST
                     0A0A
                     0A0A       STP19:
0004'CF    13   D0  0A0A                 MOVL    #19,W^CURRENT_TC
           00   DD  0A0F                 PUSHL   #0
0BF3'CF    01   FB  0A11                 CALLS   #1,W^REG_SAVE
                     0A16   877         $GETJPI_S EFN    = #4,-
```

```
                        0A16   878                     IOSB    = W^IOSTAT,-
                        0A16   879                     PRCNAM  = W^IMAGE_NAME,-
                        0A16   880                     PIDADR  = W^PID1,-
                        0A16   881                     ITMLST  = W^GET_LIST            ; try all this stuff
                        0A31   882             FAIL_CHECK SS$_NORMAL                    ; check success
 00000000'8F    DD      0A31                           PUSHL   #SS$_NORMAL
0BFD'CF    01    FB      0A37                           CALLS   #1,W^REG_CHECK
                        0A3C   883             $WAITFR_S EFN = #4                       ; wait for completion
0DDA'CF    00    FB      0A45   884             CALLS   #0,W^JPI_CHECK                  ; check the results
                        0A4A   885 ;+
                        0A4A   886 ;
                        0A4A   887 ; test ASTADR and ASTPRM _S
                        0A4A   888 ;
                        0A4A   889 ;-
                        0A4A   890             NEXT_TEST
                        0A4A
                        0A4A           STP20:
0004'CF    14    D0      0A4A                           MOVL    #20,W^CURRENT_TC
           00    DD      0A4F                           PUSHL   #0
0BF3'CF    01    FB      0A51                           CALLS   #1,W^REG_SAVE
                        0A56   891             $SETAST_S ENBFLG = #0                    ; disable AST's
                        0A5F   892             $GETJPI_S ASTADR = B^20$,-
                        0A5F   893                       ASTPRM = #5,-
                        0A5F   894                       IOSB   = W^IOSTAT,-
                        0A5F   895                       PRCNAM = W^IMAGE_NAME,-
                        0A5F   896                       PIDADR = W^PID1,-
                        0A5F   897                       ITMLST = W^GET_LIST           ; try an AST
                        0A7D   898             FAIL_CHECK SS$_NORMAL                    ; check success
 00000000'8F    DD      0A7D                           PUSHL   #SS$_NORMAL
0BFD'CF    01    FB      0A83                           CALLS   #1,W^REG_CHECK
                        0A88   899             $SETAST_S ENBFLG = #1                    ; let er rip
                        0A91   900             $HIBER_S                                 ; wait here for completion
           27    11      0A98   901             BRB     40$                              ; jump over the AST routine
                        0A9A   902 20$:
                   001C 0A9A   903             .WORD ^M<R2,R3,R4>
  04 AC    05    D1      0A9C   904             CMPL    #5,4(AP)                         ; is this the right param
           0E    13      0AA0   905             BEQL    30$                              ; br if OK
        04 AC    DD      0AA2   906             PUSHL   4(AP)                            ; push the received
           05    DD      0AA5   907             PUSHL   #5                               ; push expected
     00E6'CF    DF      0AA7   908             PUSHAL  W^AST_PARAM                      ; push the string variable
0C3F'CF    03    FB      0AAB   909             CALLS   #3,W^PRINT_FAIL                  ; print the failure
                        0AB0   910 30$:
0DDA'CF    00    FB      0AB0   911             CALLS   #0,W^JPI_CHECK                   ; check the results
                        0AB5   912             $WAKE_S                                  ; time to wake up
                  04    0AC0   913             RET                                      ; return
                        0AC1   914 40$:
                        0AC1   915 ;+
                        0AC1   916 ;
                        0AC1   917 ; test ASTADR and ASTPRM _G to test all offset definitions
                        0AC1   918 ;
                        0AC1   919 ;-
                        0AC1   920             NEXT_TEST
                        0AC1
                        0AC1           STP21:
0004'CF    15    D0      0AC1                           MOVL    #21,W^CURRENT_TC
           00    DD      0AC6                           PUSHL   #0
0BF3'CF    01    FB      0AC8                           CALLS   #1,W^REG_SAVE
```

```
                                                B 10
                                 0ACD   921          $SETAST_S ENBFLG = #0                          ; disable AST's
          00C7'CF    0C   DO    0AD6   922          MOVL    #12,W^GET+GETJPI$_EFN                  ; setup EFN param
    00CB'CF    023B'CF   DE    0ADB   923          MOVAL   W^PID1,W^GET+GETJPI$_PIDADR            ; setup PIDADR param
    00CF'CF    053A'CF   DE    0AE2   924          MOVAL   W^IMAGE_NAME,W^GET+GETJPI$_PRCNAM ; setup PRCNAM param
    00D7'CF    0233'CF   DE    0AE9   925          MOVAL   W^IOSTAT,W^GET+GETJPI$_IOSB            ; setup IOSB param
      00DB'CF    25'AF   DE    0AF0   926          MOVAL   B^20$,W^GET+GETJPI$_ASTADR             ; setup ASTADR param
00DF'CF    FFFFFFFF 8F   DO    0AF6   927          MOVL    #-1,W^GET+GETJPI$_ASTPRM               ; setup ASTPRM param
                                 0AFF   928          $GETJPI_G W^GET                                ; try it all _G
                                 0B08   929          FAIL_CHECK SS$_NORMAL                          ; check success
          00000000'8F   DD    0B08                          PUSHL   #SS$_NORMAL
        0BFD'CF    01   FB    0B0E                          CALLS   #1,W^REG_CHECK
                                 0B13   930          $SETAST_S ENBFLG = #1                          ; let er rip
                                 0B1C   931          $HIBER_S                                       ; wait here for completion
                     23   11    0B23   932          BRB     40$                                    ; jump over the AST routine
                                 0B25   933 20$:
                   001C   0B25   934          .WORD ^M<R2,R3,R4>
  04 AC    FFFFFFFF 8F   D1    0B27   935          CMPL    #-1,4(AP)                              ; is this the right param
                     12   13    0B2F   936          BEQL    30$                                    ; br if OK
                  04 AC   DD    0B31   937          PUSHL   4(AP)                                  ; push the received
          FFFFFFFF 8F   DD    0B34   938          PUSHL   #-1                                    ; push expected
              00E6'CF   DF    0B3A   939          PUSHAL  W^AST_PARAM                            ; push the string variable
        0C3F'CF    03   FB    0B3E   940          CALLS   #3,W^PRINT_FAIL                        ; print the failure
                                 0B43   941 30$:
        0DDA'CF    00   FB    0B43   942          CALLS   #0,W^JPI_CHECK                         ; check the results
                                 0B48   943 40$:
                                 0B48   944 ;+
                                 0B48   945 ;
                                 0B48   946 ; test a shorter list starting with a PCB item followed by a JIB item
                                 0B48   947 ;
                                 0B48   948 ;-
                                 0B48   949          NEXT_TEST
                                 0B48
                                 0B48       STP22:
0004'CF    16   DO    0B48                          MOVL    #22,W^CURRENT_TC
           00   DD    0B4D                          PUSHL   #0
0BF3'CF    01   FB    0B4F                          CALLS   #1,W^REG_SAVE
                                 0B54   950          $GETJPI_S EFN    = #17,-
                                 0B54   951                   PIDADR = W^PID1,-
                                 0B54   952                   ITMLST = W^SHORT_LIST             ; try _S, target process, short list
                                 0B6B   953          FAIL_CHECK SS$_NORMAL                      ; check success
 00000000'8F   DD    0B6B                          PUSHL   #SS$_NORMAL
0BFD'CF    01   FB    0B71                          CALLS   #1,W^REG_CHECK
56   000A'CF   DE    0B76   954          MOVAL   W^CPULIM,R6                                ; set questionable data adr
57   04C0'CF   DE    0B7B   955          MOVAL   W^JPI_GOOD_SHRT,R7                         ; set good data adr
     58   3A   DO    0B80   956          MOVL    #JPI_LIST_SIZE1,R8                         ; set the byte count
0DDA'CF    00   FB    0B83   957          CALLS   #0,W^JPI_CHECK                             ; check the results
           00   DD    0B88   958          PUSHL   #0                                         ; push a dummy parameter
0BF3'CF    01   FB    0B8A   959          CALLS   #1,W^REG_SAVE                              ; save a reg snapshot
                                 0B8F   960          $GETJPI_S EFN    = #18,-
                                 0B8F   961                   PRCNAM = W^TEST_MOD_NAME_D,-
                                 0B8F   962                   ITMLST = W^SHORT_LIST             ; try _S, self, short list
                                 0BA6   963          FAIL_CHECK SS$_NORMAL                      ; check for success
 00000000'8F   DD    0BA6                          PUSHL   #SS$_NORMAL
0BFD'CF    01   FB    0BAC                          CALLS   #1,W^REG_CHECK
                                 0BB1   964          $WAITFR_S EFN    = #18                     ; wait for completion
0DDA'CF    00   FB    0BBA   965          CALLS   #0,W^JPI_CHECK                             ; check the results
                                 0BBF   966          $WAKE_S PIDADR = W^PID1                    ; get rid of the target process
```

```
                                 0BCC   967          TEST_END
                    004C'CF    DD  0BCC                     PUSHL   W^TMD_ADDR
                    0048'CF    DD  0BD0                     PUSHL   W^TMN_ADDR
                         02    DD  0BD4                     PUSHL   #2
                    0044'CF    DD  0BD6                     PUSHL   W^MOD_MSG_CODE
          00000000'GF    04    FB  0BDA                     CALLS   #$$T1,G^LIB$SIGNAL
0044'CF    01    1C    01    F(  0BE1                     INSV    #1,#STS$V_INHIB_MSG,#1,W^MOD_MSG_CODE
                    0044'CF    DD  0BE8                     PUSHL   W^MOD_MSG_CODE
          00000000'GF    01    FB  0BEC                     CALLS   #1,G^SYS$EXIT
```

```
                        0BF3    969 .SBTTL  ROUTINES
                        0BF3    970         .SBTTL REG_SAVE
                        0BF3    971 ;++
                        0BF3    972 ; FUNCTIONAL DESCRIPTION:
                        0BF3    973 ;       Subroutine to save R2-R11 in the register save location.
                        0BF3    974 ;
                        0BF3    975 ; CALLING SEQUENCE:
                        0BF3    976 ;       PUSHL   #0                  ; save a dummy parameter
                        0BF3    977 ;       CALLS   #1,W^REG_SAVE       ; save R2-R11
                        0BF3    978 ;
                        0BF3    979 ; INPUT PARAMETERS:
                        0BF3    980 ;       NONE
                        0BF3    981 ;
                        0BF3    982 ; OUTPUT PARAMETERS:
                        0BF3    983 ;       NONE
                        0BF3    984 ;
                        0BF3    985 ;--
                        0BF3    986
                        0BF3    987 REG_SAVE:
                    0FFC 0BF3    988         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
    0008'CF    14 AD 28  28 0BF5    989         MOVC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; save the registers in the program
                  04    0BFC    990         RET
                        0BFD    991         .SBTTL  REG_CHECK
                        0BFD    992 ;++
                        0BFD    993 ; FUNCTIONAL DESCRIPTION:
                        0BFD    994 ;       Subroutine to test R0 & R2-R11 for proper content after a service
                        0BFD    995 ;       execution. A snapshot is taken by the REG_SAVE routine at the
                        0BFD    996 ;       beginning of each step and this routine is executed after the
                        0BFD    997 ;       services have been executed.
                        0BFD    998 ;
                        0BFD    999 ; CALLING SEQUENCE:
                        0BFD   1000 ;       PUSHL   #SS$_XXXXXX     ; push expected R0 contents
                        0BFD   1001 ;       CALLS   #1,W^REG_CHECK  ; execute this routine
                        0BFD   1002 ;
                        0BFD   1003 ; INPUT PARAMETERS:
                        0BFD   1004 ;       expected R0 contents on the stack
                        0BFD   1005 ;
                        0BFD   1006 ; OUTPUT PARAMETERS:
                        0BFD   1007 ;       possible error messages printed using $PUTMSG
                        0BFD   1008 ;
                        0BFD   1009 ;--
                        0BFD   1010
                        0BFD   1011 REG_CHECK:
                    0FFC 0BFD   1012         .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
          50    04 AC   D1  0BFF   1013         CMPL    4(AP),R0                ; is this the right fail code?
                   0E   13  0C03   1014         BEQL    10$                     ; br if yes
                   50   DD  0C05   1015         PUSHL   R0                      ; push received data
                04 AC   DD  0C07   1016         PUSHL   4(AP)                   ; push expected data
             00D8'CF    DF  0C0A   1017         PUSHAL  W^EXP                   ; push the string variable
          0C3F'CF       03  FB  0C0E   1018         CALLS   #3,W^PRINT_FAIL         ; print the error message
                        0C13   1019 10$:
 0008'CF    14 AD   28  29  0C13   1020         CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; check all but R0
                   22   13  0C1A   1021         BEQL    20$                     ; br if O.K.
56   53    00000008'8F   C3  0C1C   1022         SUBL3   #REG_SAVE_AREA,R3,R6    ; calculate the register number
                   56   04  C6  0C24   1023         DIVL2   #4,R6
             7E    56   02  81  0C27   1024         ADDB3   #^X2,R6,-(SP)           ; set number past R0-R1 and save
                   51   03  CA  0C2B   1025         BICL2   #3,R1                   ; backup to register t undrys
```

```
                                                E 10
                53   03   CA  0C2E  1026          BICL2   #3,R3
                     61   DD  0C31  1027          PUSHL   (R1)                    ; push received data
                     63   DD  0C33  1028          PUSHL   (R3)                    ; push expected data
              006D'CF     DF  0C35  1029          PUSHAL  W^REG                   ; set string pntr param.
        0C3F'CF    04    FB  0C39  1030          CALLS   #4,W^PRINT_FAIL         ; print the error message
                               0C3E  1031 20$:
                         04    0C3E  1032          RET
                               0C3F  1033          .SBTTL  PRINT_FAIL
                               0C3F  1034 ;++
                               0C3F  1035 ; FUNCTIONAL DESCRIPTION:
                               0C3F  1036 ;       Subroutine to report failures using $PUTMSG
                               0C3F  1037 ;
                               0C3F  1038 ; CALLING SEQUENCE:
                               0C3F  1039 ; Mode  #1      PUSHL EXPECTED  Mode    #2      PUSHL REG_NUMBER
                               0C3F  1040 ;               PUSHL RECEIVED                   PUSHL EXPECTED
                               0C3F  1041 ;               PUSHAL STRING_VAR                PUSHL RECEIVED
                               0C3F  1042 ;               CALLS #3,W^PRINT_FAIL            PUSHAL STRING_VAR
                               0C3F  1043 ;                                                CALLS #4,W^PRINT_FAIL
                               0C3F  1044 ; Mode  #3      PUSHAL STRING_VAR
                               0C3F  1045 ;               CALLS #1,W^PRINT_FAIL
                               0C3F  1046 ;
                               0C3F  1047 ; INPUT PARAMETERS:
                               0C3F  1048 ;       listed above
                               0C3F  1049 ;
                               0C3F  1050 ; OUTPUT PARAMETERS:
                               0C3F  1051 ;       an error message is printed using $PUTMSG
                               0C3F  1052 ;
                               0C3F  1053 ;--
                               0C3F  1054
                               0C3F  1055 PRINT_FAIL:
                          003C 0C3F  1056          .WORD   ^M<R2,R3,R4,R5>
                               0C41  1057          $FAO_S  W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
                               0C62  1058          $PUTMSG_S W^MSGVEC              ; print the message
                    04   6C    91  0C73  1059          CMPB    (AP),#4             ; is this a register message?
                         26    13  0C76  1060          BEQL    10$                 ; br if yes
                    01   6C    91  0C78  1061          CMPB    (AP),#1             ; is this just a message?
                         48    13  0C7B  1062          BEQL    20$                 ; br if yes
                               0C7D  1063          $FAO_S  W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
                         40    11  0C9C  1064          BRB     30$                 ; goto output message
                               0C9E  1065 10$:
                               0C9E  1066          $FAO_S  W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
                         19    11  0CC3  1067          BRB     30$                 ; goto output message
                               0CC5  1068 20$:
            022F'CF    04 AC    D0  0CC5  1069          MOVL    4(AP),W^MSGVEC1+12  ; save string address
                               0CCB  1070          $PUTMSG_S W^MSGVEC1             ; print the message
                         11    11  0CDC  1071          BRB     40$                 ; skip the other message
                               0CDE  1072 30$:
                               0CDE  1073          $PUTMSG_S W^MSGVEC              ; print the message
                               0CEF  1074 40$:
              0D03'CF    00    FB  0CEF  1075          CALLS   #0,W^MODE_ID        ; identify the mode
         004C'CF    002A'CF    DE  0CF4  1076          MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR  ; set failure message address
0044'CF    03    00    02    F0  0CFB  1077          INSV    #ERROR,#0,#3,W^MOD_MSG_CODE  ; set severity code
                         04    0D02  1078          RET
```

```
                     0D03  1080          .SBTTL  MODE_ID
                     0D03  1081 ;++
                     0D03  1082 ; FUNCTIONAL DESCRIPTION:
                     0D03  1083 ;       Subroutine to identify the mode that an exit handler is in.
                     0D03  1084 ;
                     0D03  1085 ; CALLING SEQUENCE:
                     0D03  1086 ;       CALLS   #0,W^MODE_ID
                     0D03  1087 ;
                     0D03  1088 ; INPUT PARAMETERS:
                     0D03  1089 ;       MODE contains an address pointing to an ascii string desc.
                     0D03  1090 ;       of the current CPU mode.
                     0D03  1091 ;
                     0D03  1092 ; OUTPUT PARAMETERS:
                     0D03  1093 ;       NONE
                     0D03  1094 ;
                     0D03  1095 ;--
                     0D03  1096
                     0D03  1097 MODE_ID:
                003C 0D03  1098          .WORD   ^M<R2,R3,R4,R5>
                     0D05  1099          $FAO_S  W^CS5,W^MESSAGEL,W^MSGL,MODE ; format the error message
                     0D1E  1100          $PUTMSG_S W^MSGVEC                   ; print the mode message
                  04 0D2F  1101          RET
                     0D30  1102          .SBTTL CRE_CHECK
                     0D30  1103 ;++
                     0D30  1104 ; FUNCTIONAL DESCRIPTION:
                     0D30  1105 ;       Routine to check the process exit status of a created process.
                     0D30  1106 ;
                     0D30  1107 ; CALLING SEQUENCE:
                     0D30  1108 ;       CALLS   #0,W^CRE_CHECK  ; save R2-R11
                     0D30  1109 ;
                     0D30  1110 ; INPUT PARAMETERS:
                     0D30  1111 ;       R6 = Expected process exit status
                     0D30  1112 ;       R7 = PID check flag BIT0 = 1 means check the PID
                     0D30  1113 ;
                     0D30  1114 ; OUTPUT PARAMETERS:
                     0D30  1115 ;       NONE
                     0D30  1116 ;
                     0D30  1117 ;--
                     0D30  1118
                     0D30  1119 CRE_CHECK:
                0FFC 0D30  1120          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                     0D32  1121          $QIOW_S EFN  = #1,-
                     0D32  1122                  FUNC = #IO$_READVBLK,-
                     0D32  1123                  CHAN = W^MBCHAN,-
                     0D32  1124                  IOSB = W^IOSTATUS,-
                     0D32  1125                    P1 = W^MBUF,-
                     0D32  1126                    P2 = #100                          ; read the mail
             56 024F'CF   D1 0D59  1127          CMPL    W^MBUF+ACC$L_FINALSTS,R6   ; is the status as expected?
                    0F    13 0D5E  1128          BEQL    10$                        ; br if OK
                024F'CF   DD 0D60  1129          PUSHL   W^MBUF+ACC$L_FINALSTS      ; push received
                      56  DD 0D64  1130          PUSHL   R6                         ; push expected
                00D8'CF   DF 0D66  1131          PUSHAL  W^EXP                      ; push string variable
             FED0 CF    03  FB 0D6A  1132          CALLS   #3,W^PRINT_FAIL            ; print the failure
                           0D6F  1133 10$:
                     1A 57   E9 0D6F  1134          BLBC    R7,20$                     ; should we check the PID?
          0247'CF    023B'CF  D1 0D72  1135          CMPL    W^PID1,W^IOSTATUS+4        ; check the PID
                         11   13 0D79  1136          BEQL    20$                        ; br if its good
```

```
                     0247'CF    DD  0D7B  1137          PUSHL   W^IOSTATUS+4                    ; push received
                     023B'CF    DD  0D7F  1138          PUSHL   W^PID1                          ; push expected
                     0163'CF    DF  0D83  1139          PUSHAL  W^PID_STR                       ; push the string variable
                FEB3 CF    03   FB  0D87  1140          CALLS   #3,W^PRINT_FAIL                 ; print the failure
                                    0D8C  1141 20$:
                               04   0D8C  1142          RET
                                    0D8D  1143          .SBTTL JPI_CHECK
                                    0D8D  1144 ;++
                                    0D8D  1145 ; FUNCTIONAL DESCRIPTION:
                                    0D8D  1146 ;       Subroutine to check the results of a JPI service
                                    0D8D  1147 ;
                                    0D8D  1148 ; CALLING SEQUENCE:
                                    0D8D  1149 ;       CALLS   #0,W^JPI_CHECK  ; check the results
                                    0D8D  1150 ;
                                    0D8D  1151 ; INPUT PARAMETERS:
                                    0D8D  1152 ;       R6 = questionable data address
                                    0D8D  1153 ;       R7 = good data address
                                    0D8D  1154 ;       R8 = byte count
                                    0D8D  1155 ;
                                    0D8D  1156 ; OUTPUT PARAMETERS:
                                    0D8D  1157 ;       NONE
                                    0D8D  1158 ;
                                    0D8D  1159 ;--
                                    0D8D  1160
                                    0D8D  1161 ARGLST1:
                          00000D99  0D8D  1162          .BLKL   3
                                    0D99  1163 CTRSTR:
65 20 61 74 61 64 00000DA1'010E0000' 0D99  1164          .ASCID  /data error at offset !XW, good data = !XB bad data = !XB./
73 66 66 6F 20 74 61 20 72 6F 72 72  0DA7
64 6F 6F 67 20 2C 57 58 21 20 74 65  0DB3
20 42 58 21 20 3D 20 61 74 61 64 20  0DBF
21 20 3D 20 61 74 61 64 20 64 61 62  0DCB
                        2E 42 58     0DD7
                                    0DDA  1165 ;
                                    0DDA  1166 JPI_CHECK:
                               0FFC 0DDA  1167          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                          67   66   58   29  0DDC  1168          CMPC3   R8,(R6),(R7)            ; check the buffer
                                    34   13  0DE0  1169          BEQL    10$                     ; br if good
                     FFA1 CF    51   00000000'8F   C3  0DE2  1170          SUBL3   #ACCOUNT,R1,W^ARGLST1   ; get buffer offset
                          FFA0 CF    63   9A  0DEC  1171          MOVZBL  (R3),W^ARGLST1+4        ; get the good data
                          FF9F CF    61   9A  0DF1  1172          MOVZBL  (R1),W^ARGLST1+8        ; get the bad data
                                    0DF6  1173          $FAOL_S W^CTRSTR,W^ML,W^GETBUF,W^ARGLST1 ; make it readable
                               0183'CF    DF  0E0D  1174          PUSHAL  W^ML                    ; push the desc. address
                          FE29 CF    01   FB  0E11  1175          CALLS #1,W^PRINT_FAIL           ; print the failure
                                    0E16  1176 10$:
                                    04  0E16  1177          RET
```

```
                                H 10
                     0E17  1180 MOD_MSG_PRINT:
                     0E17  1181 ;
                     0E17  1182 ;  ***************************************************************
                     0E17  1183 ;  *                                                             *
                     0E17  1184 ;  *  PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES     *
                     0E17  1185 ;  *       (USING THE PUTMSG MACRO).                             *
                     0E17  1186 ;  *                                                             *
                     0E17  1187 ;  ***************************************************************
                     0E17  1188 ;
                     0E17  1189          PUTMSG  <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> ; PRINT MSG
                  05 0E32  1190          RSB                                         ; ... AND RETURN TO CALLER
                     0E33  1191 ;
                     0E33  1192 CHMRTN:
                     0E33  1193 ;  ***************************************************************
                     0E33  1194 ;  *                                                             *
                     0E33  1195 ;  *     CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER  *
                     0E33  1196 ;  *     A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED     *
                     0E33  1197 ;  *     BY THE MODE MACRO ('TO' OPTION).  IT MERELY DOES        *
                     0E33  1198 ;  *     A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS       *
                     0E33  1199 ;  *     THE EFFECT OF RETURNING TO THE END OF THE MODE          *
                     0E33  1200 ;  *     MACRO EXPANSION.                                        *
                     0E33  1201 ;  *                                                             *
                     0E33  1202 ;  ***************************************************************
                     0E33  1203 ;
                0000 0E33  1204          .WORD   0                                   ; ENTRY MASK
00000059'FF       17 0E35  1205          JMP     @CHM_CONT                           ; RETURN TO MODE MACRO IN NEW MODE
                     0E3B  1206 ;
                     0E3B  1207 ;  *     RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
                     0E3B  1208 ;
                     0E3B  1209          .END    SATSSS35
```

```
$$ARGS                  =  00000007
$$T1                    =  00000004
$$T2                    =  00000004
ACC$L_FINALSTS          =  00000004
ACCOUNT                    00000000 R      02
ACCOUNTL                   00000008 R      02
APTCNT                     00000050 R      02
APTCNTL                    00000054 R      02
ARGLST1                    00000D8D R      05
ASTACT                     00000056 R      02
ASTACTL                    0000005A R      02
ASTCNT                     00000062 R      02
ASTCNTL                    00000066 R      02
ASTEN                      0000005C R      02
ASTENL                     00000060 R      02
ASTLM                      00000068 R      02
ASTLML                     0000006C R      02
AST_PARAM                  000000E6 R      03
AUTHPRIV                   0000006E R      02
AUTHPRIVL                  00000076 R      02
BIOCNT                     00000078 R      02
BIOCNTL                    0000007C R      02
BIOLM                      0000007E R      02
BIOLML                     00000082 R      02
BP                         00000CF8 R      03
BUF                        00000133 R      04
BUFIO                      00000084 R      02
BUFIOL                     00000088 R      02
BYTCNT                     0000008A R      02
BYTCNTL                    0000008E R      02
BYTLM                      00000090 R      02
BYTLML                     00000094 R      02
CHMRTN                     00000E33 R      05
CHM_CONT                   00000059 R      04
CPUTIM                     0000000A R      02
CPULIML                    0000000E R      02
CPUTIM                     00000096 R      02
CPUTIML                    0000009A R      02
CRE                        0000008B R      04
CREPRC                     0000016E R      03
CREPRC$_BASPRI          =  00000024
CREPRC$_ERROR           =  00000014
CREPRC$_IMAGE           =  00000008
CREPRC$_INPUT           =  0000000C
CREPRC$_ITMLST          =  00000034
CREPRC$_MBXUNT          =  0000002C
CREPRC$_NARGS           =  0000000D
CREPRC$_OUTPUT          =  00000010
CREPRC$_PIDADR          =  00000004
CREPRC$_PRCNAM          =  00000020
CREPRC$_PRVADR          =  00000018
CREPRC$_QUOTA           =  0000001C
CREPRC$_STSFLG          =  00000030
CREPRC$_UIC             =  00000028
CRE_CHECK                  00000D30 R      05
CS1                        00000031 R      03
CS2                        00000063 R      03
CS3                        00000090 R      03
CS5                        000000C3 R      03
CTL$GL_PHD                 ********   X    05
CTRSTR                     00000D99 R      05
CURPRIV                    00000010 R      02
CURPRIVL                   00000018 R      02
CURRENT_TC                 00000004 R      04
DFPFC                      0000009C R      02
DFPFCL                     000000A0 R      02
DFWSCNT                    000000A2 R      02
DFWSCNTL                   000000A6 R      02
DIB$W_UNIT              =  0000000C
DIOCNT                     000000A8 R      02
DIOCNTL                    000000AC R      02
DIOLM                      000000AE R      02
DIOLML                     000000B2 R      02
DIRIO                      000000B4 R      02
DIRIOL                     000000B8 R      02
DIRTY                      00000266 R      03
EFCS                       000000BA R      02
EFCSL                      000000BE R      02
EFCU                       000000C0 R      02
EFCUL                      000000C4 R      02
EFC_NAME                   0000014F R      03
EFWM                       000000C6 R      02
EFWML                      000000CA R      02
ERR                        00000529 R      03
ERROR                   =  00000002
EXCVEC                     000000CC R      02
EXCVECL                    000000D0 R      02
EXP                        000000D8 R      03
FILCNT                     000000D8 R      02
FILCNTL                    000000DC R      02
FILLM                      000000DE R      02
FILLML                     000000E2 R      02
FINALEXC                   000000D2 R      02
FINALEXCL                  000000D6 R      02
FREPOVA                    000000E4 R      02
FREPOVAL                   000000E8 R      02
FREP1VA                    000000EA R      02
FREP1VAL                   000000EE R      02
GET                        000000C3 R      04
GET1                       000000E3 R      04
GETBUF                     00000188 R      04
GETJPI                     00000175 R      03
GETJPI$_ASTADR          =  00000018
GETJPI$_ASTPRM          =  0000001C
GETJPI$_EFN             =  00000004
GETJPI$_IOSB            =  00000014
GETJPI$_ITMLST          =  00000010
GETJPI$_NARGS           =  00000007
GETJPI$_PIDADR          =  00000008
GETJPI$_PRCNAM          =  0000000C
GET_LIST                   000001EE R      03
GPGCNT                     000000F0 R      02
GPGCNTL                    000000F4 R      02
GRP                        0000001A R      02
```

```
GRPL                    0000001E R      02
IMAGE_NAME              0000053A R      03
IMAGNAME                000000F6 R      02
IMAGNAMEL               00000176 R      02
IMAGPRIV                00000020 R      02
IMAGPRIVL               00000028 R      02
IN                      00000506 R      03
IO$_READVBLK            ******** X      05
IOSTAT                  00000233 R      04
IOSTATUS                00000243 R      04
ITEM_LIST               00000103 R      04
JPI$_ACCOUNT            =  00000203
JPI$_APTCNT             =  0000030A
JPI$_ASTACT             =  00000300
JPI$_ASTCNT             =  0000030E
JPI$_ASTEN              =  00000301
JPI$_ASTLM              =  00000409
JPI$_AUTHPRIV           =  00000412
JPI$_BIOCNT             =  0000030F
JPI$_BIOLM              =  00000310
JPI$_BUFIO              =  0000040C
JPI$_BYTCNT             =  00000311
JPI$_BYTLM              =  0000031A
JPI$_CPULIM             =  0000040D
JPI$_CPUTIM             =  00000407
JPI$_CURPRIV            =  00000400
JPI$_DFPFC              =  00000406
JPI$_DFWSCNT            =  00000403
JPI$_DIOCNT             =  00000312
JPI$_DIOLM              =  00000313
JPI$_DIRIO              =  0000040B
JPI$_EFCS               =  00000317
JPI$_EFCU               =  00000318
JPI$_EFWM               =  00000316
JPI$_EXCVEC             =  00000100
JPI$_FILCNT             =  00000314
JPI$_FILLM              =  0000040F
JPI$_FINALEXC           =  00000101
JPI$_FREPOVA            =  00000404
JPI$_FREP1VA            =  00000405
JPI$_GPGCNT             =  0000030C
JPI$_GRP                =  00000308
JPI$_IMAGNAME           =  00000207
JPI$_IMAGPRIV           =  00000413
JPI$_LOGINTIM           =  00000206
JPI$_MEM                =  00000307
JPI$_OWNER              =  00000303
JPI$_PAGEFLTS           =  0000040A
JPI$_PGFLQUOTA          =  0000040E
JPI$_PID                =  00000319
JPI$_PPGCNT             =  0000030D
JPI$_PRCCNT             =  0000031B
JPI$_PRCLM              =  00000408
JPI$_PRCNAM             =  0000031C
JPI$_PRI                =  00000302
JPI$_PRIB               =  00000309
JPI$_PROCPRIV           =  00000204
JPI$_STATE              =  00000306
JPI$_STS                =  00000305
JPI$_TMBU               =  0000030B
JPI$_TQCNT              =  00000315
JPI$_TQLM               =  00000410
JPI$_UIC                =  00000304
JPI$_USERNAME           =  00000202
JPI$_VIRTPEAK           =  00000200
JPI$_VOLUMES            =  00000205
JPI$_WSAUTH             =  00000401
JPI$_WSPEAK             =  00000201
JPI$_WSQUOTA            =  00000402
JPI$_WSSIZE             =  00000411
JPI_CHECK                  00000DDA R      05
JPI_GOOD                   000004B6 R      03
JPI_GOOD_SHRT              000004C0 R      03
JPI_LIST_SIZE           =  00000044
JPI_LIST_SIZE1          =  0000003A
JPI_PRV_MASK            =  1070BFEF
JPI_PRV_NMASK           =  EF8F4010
JPI_STS_MASK            =  0038C600
JPI_STS_NMASK           =  FFC739FF
LIB$SIGNAL                 ******** X      05
LOGINTIM                   00000178 R      02
LOGINTIML                  0000017C R      02
MBCHAN                     0000023F R      04
MBNAM                      00000188 R      03
MBUF                       0000024B R      04
MBXUN                      00000241 R      04
MEM                        0000002A R      02
MEML                       0000002E R      02
MESSAGEL                   00000217 R      04
ML                         00000183 R      04
MODE                       00000069 R      04
MODE_ID                    00000D03 R      05
MOD_MSG_CODE               00000044 R      04
MOD_MSG_PRINT              00000E17 R      05
MSGL                       00000083 R      04
MSGVEC                     000001A6 R      03
MSGVEC1                    00000223 R      04
NPRVMASK                   0000019E R      03
OUT                        00000517 R      03
OWNER                      0000017E R      02
OWNERL                     00000182 R      02
PAGEFLTS                   00000184 R      02
PAGEFLTSL                  00000188 R      02
PCB$V_BATCH             =  0000000E
PCB$V_HIBER             =  00000013
PCB$V_LOGIN             =  00000014
PCB$V_NETWRK            =  00000015
PCB$V_NOACNT            =  0000000F
PCB$V_SSFEXCU           =  00000009
PCB$V_SSRWAIT           =  0000000A
PGFLQUOTA                  0000018A R      02
PGFLQUOTAL                 0000018E R      02
PHD$Q_PRIVMSK           =  00000000
PID                        00000190 R      02
```

```
PID1                            0000023B R      04
PIDL                            00000194 R      02
PID_STR                         00000163 R      03
PNS                             00000109 R      03
PPGCNT                          00000196 R      02
PPGCNTL                         0000019A R      02
PQL$_ASTLM                    = 00000001
PQL$_BIOLM                    = 00000002
PQL$_BYTLM                    = 00000003
PQL$_CPULM                    = 00000004
PQL$_DIOLM                    = 00000005
PQL$_FILLM                    = 00000006
PQL$_LISTEND                  = 00000000
PQL$_PGFLQUOTA                = 00000007
PQL$_PRCLM                    = 00000008
PQL$_TQELM                    = 00000009
PQL$_WSDEFAULT                = 0000000B
PQL$_WSQUOTA                  = 0000000A
PRCCNT                          0000019C R      02
PRCCNTL                         000001A0 R      02
PRCLM                           00000030 R      02
PRCLML                          00000034 R      02
PRCNAM                          000001A2 R      02
PRCNAML                         000001B1 R      02
PRI                             000001BD R      02
PRIB                            000001C3 R      02
PRIBL                           000001C7 R      02
PRIL                            000001C1 R      02
PRINT_FAIL                      00000C3F R      05
PRIVMASK                        00000051 R      04
PRIVS                           000002B3 R      04
PRIV_ARGS                     = 00000002
PROCPRIV                        000001B3 R      02
PROCPRIVL                       000001BB R      02
PROC_NAME                       0000054E R      03
PROC_UIC                        00000565 R      03
PRV$V_CMEXEC                  = 00000001
PRV$V_CMKRNL                  = 00000000
PRV$V_DETACH                  = 00000005
PRV$V_DIAGNOSE                = 00000006
PRV$V_GROUP                   = 00000008
PRV$V_GRPNAM                  = 00000003
PRV$V_LOG_IO                  = 00000007
PRV$V_NETMBX                  = 00000014
PRV$V_NOACNT                  = 00000009
PRV$V_PHY_IO                  = 00000016
PRV$V_PRMCEB                  = 0000000A
PRV$V_PRMMBX                  = 0000000B
PRV$V_PSWAPM                  = 0000000C
PRV$V_SETPRI                  = 0000000D
PRV$V_SETPRV                  = 0000000E
PRV$V_SYSNAM                  = 00000002
PRV$V_SYSPRV                  = 0000001C
PRV$V_TMPMBX                  = 0000000F
PRV$V_VOLPRO                  = 00000015
PRVMASK                         00000196 R      03
PRVPRT                          00000050 R      04
QUOTA_LIST                      000001B6 R      03
REG                             0000006D R      04
REGNUM                          0000007F R      04
REG_CHECK                       00000BFD R      05
REG_SAVE                        00000BF3 R      05
REG_SAVE_AREA                   00000008 R      04
RETADR                          0000005D R      04
RMS$_FNF                        ********   X    05
SATSSS35                        00000000 RG     05
SERV_NAME                       0000021F R      04
SHORT_LIST                      000001FA R      03
SHR$_ABENDD                   = 000010E0
SHR$_BEGIND                   = 00001038
SHR$_ENDEDD                   = 00001080
SHR$_TEXT                     = 00001130
SS$_NORMAL                      ********   X    05
STATE                           000001C9 R      02
STATEL                          000001CD R      02
STATUS                          00000065 R      04
STEP                          = 00000016
STP0                            0000003D R      05
STP1                            000000BE R      05
STP10                           0000049C R      05
STP11                           00000543 R      05
STP12                           000006CC R      05
STP13                           0000075E R      05
STP14                           000007FF R      05
STP15                           00000824 R      05
STP16                           000008CF R      05
STP17                           00000962 R      05
STP18                           0000099E R      05
STP19                           00000A0A R      05
STP2                            000000F3 R      05
STP20                           00000A4A R      05
STP21                           00000AC1 R      05
STP22                           00000B48 R      05
STP3                            00000146 R      05
STP4                            0000018D R      05
STP5                            000001E4 R      05
STP6                            00000224 R      05
STP7                            0000027F R      05
STP8                            00000309 R      05
STP9                            000003AA R      05
STS                             000001CF R      02
STS$V_FAC_NO                  = 00000010
STS$V_INHIB_MSG               = 0000001C
STSFLGS                         00000134 R      03
STSL                            000001D3 R      02
SUCCESS                       = 00000001
SYS$ASCEFC                      ********  GX    05
SYS$CMKRNL                      ********  GX    05
SYS$CREMBX                      ********  GX    05
SYS$CREPRC                      ********  GX    05
SYS$DACEFC                      ********  GX    05
SYS$DELPRC                      ********  GX    05
SYS$EXIT                        ********  GX    05
SYS$FAO                         ********   X    05
```

```
SYS$FAOL                        ********  GX    05
SYS$GETCHN                      ********  GX    05
SYS$GETJPI                      ********  GX    05
SYS$HIBER                       ********  GX    05
SYS$PUTMSG                      ********  GX    05
SYS$QIOW                        ********  GX    05
SYS$SETAST                      ********  GX    05
SYS$SETPRN                      ********  GX    05
SYS$SETPRV                      ********  GX    05
SYS$WAITFR                      ********  GX    05
SYS$WAKE                        ********  GX    05
TEST_MOD_BEGIN                  00000019  R     03
TEST_MOD_FAIL                   0000002A  R     03
TEST_MOD_NAME                   00000000  R     03
TEST_MOD_NAME_D                 00000009  R     03
TEST_MOD_SUCC                   0000001F  R     03
TEST_PID                        000002AF  R     04
TMBU                            000001D5  R     02
TMBUL                           000001D9  R     02
TMD_ADDR                        0000004C  R     04
TMN_ADDR                        00000048  R     04
TPID                            00000000  R     04
TQCNT                           000001DB  R     02
TQCNTL                          000001DF  R     02
TQLM                            00000036  R     02
TQLML                           0000003A  R     02
UETP                          = 00740000
UETP$_ABENDD                  = 007410E0
UETP$_BEGIND                  = 00741038
UETP$_ENDEDD                  = 00741080
UETP$_SATSMS                  = 007480D9
UETP$_TEXT                    = 00741130
UIC                             0000003C  R     02
UICL                            00000040  R     02
UIC_MSG                         00000144  R     03
UM                              0000017C  R     03
USERNAME                        00000042  R     02
USERNAMEL                       0000004E  R     02
VIRTPEAK                        000001E7  R     02
VIRTPEAKL                       000001EB  R     02
VOLUMES                         000001E1  R     02
VOLUMESL                        000001E5  R     02
WSAUTH                          000001ED  R     02
WSAUTHL                         000001F1  R     02
WSPEAK                          000001F9  R     02
WSPEAKL                         000001FD  R     02
WSQUOTA                         000001F3  R     02
WSQUOTAL                        000001F7  R     02
WSSIZE                          000001FF  R     02
WSSIZEL                         00000203  R     02
```

M 10

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
ITEM_LIST	00000205 (517.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
RODATA	00000569 (1385.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
RWDATA	000002BB (699.)	04 (4.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
SATSSS35	00000E3B (3643.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.08	00:00:00.31
Command processing	134	00:00:00.63	00:00:02.15
Pass 1	472	00:00:19.08	00:00:32.37
Symbol table sort	0	00:00:01.51	00:00:01.52
Pass 2	257	00:00:04.93	00:00:08.65
Symbol table output	46	00:00:00.38	00:00:00.54
Psect synopsis output	3	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	949	00:00:26.66	00:00:45.59

The working set limit was 2000 pages.
116238 bytes (228 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1010 non-local and 25 local symbols.
1209 source lines were read in Pass 1, producing 42 object records in Pass 2.
63 pages of virtual memory were used to define 57 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SHRLIB]UETP.MLB;1	10
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	41
TOTALS (all libraries)	53

1207 GETS were required to define 53 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:SATSSS35/OBJ=OBJ$:SATSSS35 MSRC$:SATSSS35/UPDATE=(ENH$:SATSSS35)+EXECML$/LIB+SHRLIB$:UETP/LIB
```

SATSSS35
LIS

SATSSS26
LIS

SATSSS38
LIS

SATSS530
LIS

SATSSS37
LIS

SATSSS39
LIS

SATSSS36
LIS