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

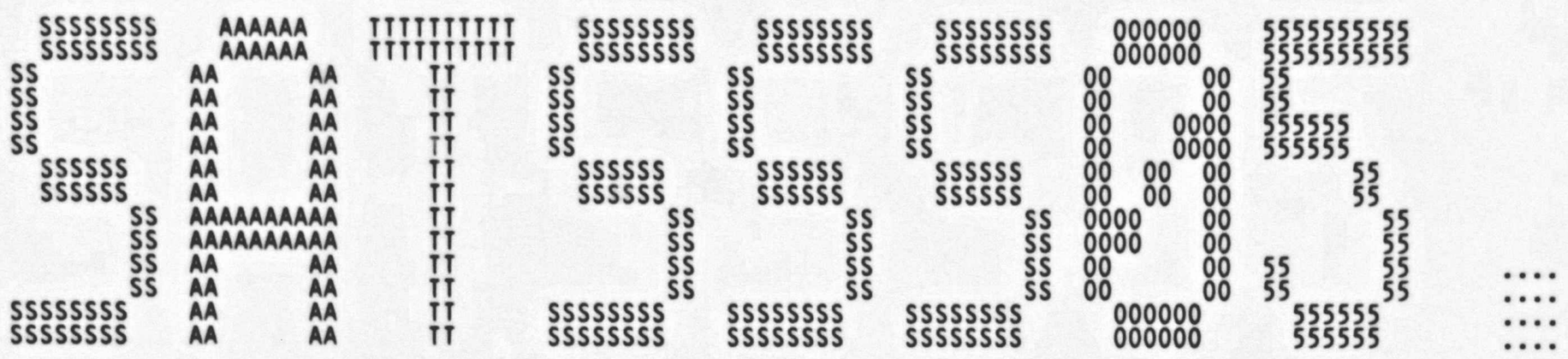
SSSSSSSS AAAAAA TTTTTTTTTT SSSSSSSS SSSSSSSS SSSSSSSS 000000 5555555555
SSSSSSSS AAAAAA TTTTTTTTTT SSSSSSSS SSSSSSSS SSSSSSSS 000000 5555555555
SS AA AA TT SS SS SS 00 00 55
SS AA AA TT SS SS SS 00 00 55
SS AA AA TT SS SS SS 00 0000 555555
SS AA AA TT SS SS SS 00 0000 555555
SSSSSS AA AA TT SSSSSS SSSSSS SSSSSS 00 00 00 55
SSSSSS AA AA TT SSSSSS SSSSSS SSSSSS 00 00 00 55
SS AAAAAAAAAA TT SS SS SS 0000 00 55
SS AAAAAAAAAA TT SS SS SS 0000 00 55
SS AA AA TT SS SS SS 00 00 55 55
SS AA AA TT SS SS SS 00 00 55 55
SSSSSSSS AA AA TT SSSSSSSS SSSSSSSS SSSSSSSS 000000 555555
SSSSSSSS AA AA TT SSSSSSSS SSSSSSSS SSSSSSSS 000000 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

```
                                J 7
0000     1          .TITLE  SATSSS05 - SATS SYSTEM SERVICE TESTS  (SUCC S.C.)
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16  ;*  TRANSFERRED.                                                              *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20  ;*  CORPORATION.                                                              *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28
0000    29  ;++
0000    30  ; FACILITY:     SATS SYSTEM SERVICE TESTS
0000    31  ;
0000    32  ; ABSTRACT:     The SATSSS05 module tests the execution of the following
0000    33  ;               VMS system services:
0000    34  ;
0000    35  ;               $SNDACC
0000    36  ;               $SNDERR
0000    37  ;               $SNDOPR
0000    38  ;               $SNDSMB
0000    39  ;
0000    40  ;
0000    41  ; ENVIRONMENT:  User mode image,
0000    42  ;               Needs CMKRNL privilege and dynamically acquires other
0000    43  ;               privileges, as needed.
0000    44  ;
0000    45  ; AUTHOR: Larry D. Jones,                CREATION DATE: JULY, 1978
0000    46  ;
0000    47  ; MODIFIED BY:
0000    48  ;
0000    49  ;       V03-002 PCG0001         Peter C. George         16-Feb-1981
0000    50  ;               Add OPCMSG macro expansion
0000    51  ;
0000    52  ;       V03-001 LDJ0001         Larry D. Jones,         17-Sep-1980
0000    53  ;               Modified to conform to new build command procedures.
0000    54  ;--
0000    55  ;--
```

```
                        57          .SBTTL  DECLARATIONS
                        58 ;
                        59 ; MACRO LIBRARY CALLS
                        60 ;
                        61          $ACCDEF                                 ; accounting definitions
                        62          $DIBDEF                                 ; device info block offsets
                        63          $EMBDEF                                 ; error log buffer definitions
                        64          $JBCMSGDEF                              ; job controller definitions
                        65          $OPCDEF                                 ; operator communications def.
                        66          $OPCMSG                                 ; operator communications messages
                        67          $OPRDEF                                 ; operator message definitions
                        68          $PHDDEF                                 ; process header definitions
                        69          $PRVDEF                                 ; privilege definitions
                        70          $SHR_MESSAGES UETP,116,<<TEXT,INFO>> ; UETP$_TEXT definition
                        71          $SMRDEF                                 ; symbiot manager definitions
                        72          $STSDEF                                 ; STS definitions
                        73          $UETPDEF                                ; UETP message definitions
                        74 ;
                        75 ; Equated symbols
                        76 ;
00000000  0000          77 WARNING          = 0                             ; warning severity value for msgs
00000001  0000          78 SUCCESS          = 1                             ; success       "       "   "   "
00000002  0000          79 ERROR            = 2                             ; error         "       "   "   "
00000003  0000          80 INFO             = 3                             ; information   "       "   "   "
00000004  0000          81 SEVERE           = 4                             ; fatal         "       "   "   "
                        82 ;
0000000D  0000          83          CR = 13                                 ; terminal definitions
0000000A  0000          84          LF = 10
                        85 ;
00000006  0000          86          FIDSIZ = 6                              ; ID sizes
00000006  0000          87          DIDSIZ = 6
00000014  0000          88          FILNAMSIZ = 20
00000007  0000          89          COM_FIL_SIZ = 7
                        90 ;
00000064  0000          91          BUF_SIZE=100                            ; buffer size
                        92 ;
                        93          ALL_OPR = OPC$M_NM_CENTRL!OPC$M_NM_PRINT!-
                        94                    OPC$M_NM_TAPES!OPC$M_NM_DISKS!-
                        95                    OPC$M_NM_DEVICE!OPC$M_NM_OPER1!-
                        96                    OPC$M_NM_OPER2!OPC$M_NM_OPER3!-
                        97                    OPC$M_NM_OPER4!OPC$M_NM_OPER5!-
                        98                    OPC$M_NM_OPER6!OPC$M_NM_OPER7!-
                        99                    OPC$M_NM_OPER8!OPC$M_NM_OPER9!-
                       100                    OPC$M_NM_OPER10!OPC$M_NM_OPER11!-
00FFF01F  0000         101                    OPC$M_NM_OPER12
                       102 ;
                       103 ; ***** NOTE *****
                       104 ;
                       105 ; THE FOLLOWING DEFINITION IS TO BE REMOVED WHEN VMS RELEASE 2 IS FIXED.
                       106 ;
00000008  0000         107          SNDACC$_CHAN = 8
                       108 ; MACROS
                       109 ;
```

```
                                 00000000        .PSECT  RODATA,RD,NOWRT,NOEXE,LONG
                                     0000  ;
                                     0000  TEST_MOD_NAME:
       35 30 53 53 53 54 41 53 00'   0000          .ASCIC  /SATSSS05/              ; needed for SATSMS message
                                 08  0000
                                     0009  TEST_MOD_NAME_D:
53 53 53 54 41 53 00000011'010E0000' 0009          .ASCID /SATSSS05/               ; module name
                              35 30  0017
                                     0019  TEST_MOD_BEGIN:
             6E 75 67 65 62 00'      0019          .ASCIC  /begun/
                                 05  0019
                                     001F  TEST_MOD_SUCC:
   6C 75 66 73 73 65 63 63 75 73 00' 001F          .ASCIC  /successful/
                                 0A  001F
                                     002A  TEST_MOD_FAIL:
          64 65 6C 69 61 66 00'      002A          .ASCIC  /failed/
                                 06  002A
                                     0031  SNDACC:
          43 43 41 44 4E 53 00'      0031          .ASCIC  /SNDACC/
                                 06  0031
                                     0038  SNDERR:
          52 52 45 44 4E 53 00'      0038          .ASCIC  /SNDERR/
                                 06  0038
                                     003F  SNDOPR:
          52 50 4F 44 4E 53 00'      003F          .ASCIC  /SNDOPR/
                                 06  003F
                                     0046  SNDSMB:
          42 4D 53 44 4E 53 00'      0046          .ASCIC  /SNDSMB/
                                 06  0046
                                     004D  CS1:
21 20 74 73 65 54 00000055'010E0000' 004D          .ASCID  \Test !AC service name !AC step !UL failed.\
6E 20 65 63 69 76 72 65 73 20 43 41  005B
70 65 74 73 20 43 41 21 20 65 6D 61  0067
2E 64 65 6C 69 61 66 20 4C 55 21 20  0073
                                     007F  CS2:
74 63 65 70 78 45 00000087'010E0000' 007F          .ASCID  \Expected !AS = !XL received !AS = !XL\
4C 58 21 20 3D 20 53 41 21 20 64 65  008D
41 21 20 64 65 76 69 65 63 65 72 20  0099
            4C 58 21 20 3D 20 53     00A5
                                     00AC  CS3:
74 63 65 70 78 45 000000B4'010E0000' 00AC          .ASCID  \Expected !AS!UB = !XL received !AS!UB = !XL\
20 3D 20 42 55 21 53 41 21 20 64 65  00BA
64 65 76 69 65 63 65 72 20 4C 58 21  00C6
58 21 20 3D 20 42 55 21 53 41 21 20  00D2
                                 4C  00DE
                                     00DF  CS5:
77 20 65 64 6F 4D 000000E7'010E0000' 00DF          .ASCID  \Mode was !AS.\
            2E 53 41 21 20 73 61     00ED
                                     00F4  CS6:
74 63 65 70 78 45 000000FC'010E0000' 00F4          .ASCID  \Expected byte offset !UB(10) = !XB(16) received !XB(16).\
73 66 66 6F 20 65 74 79 62 20 64 65  0102
3D 20 29 30 31 28 42 55 21 20 74 65  010E
63 65 72 20 29 36 31 28 42 58 21 20  011A
36 31 28 42 58 21 20 64 65 76 69 65  0126
                              2E 29  0132
                                     0134  UM:
      72 65 73 75 0000013C'010E0000' 0134          .ASCID  \user\
```

```
                                                M 7
                                0140   143 MBNAM:
42 4D 24 54 53 53 00000148'010E0000' 0140   144         .ASCID  \SST$MBX\
                                  58  014E
                                0140   145 TTNAM:
                     41 54 54 5F 00'  014F   146         .ASCIC  \_TTA\                  ; terminal name to send opr messages to
                                  04  014F
                                0154   147 TTUNIT:
                               0001  0154   148         .WORD   1                       ; unit number for above
                                0156   149 EXP:
73 75 74 61 74 73 0000015E'010E0000' 0156   150         .ASCID  \status\
                                0164   151 BAT_IMP_EXC:
20 68 63 74 61 42 0000016C'010E0000' 0164   152         .ASCID  \Batch job improperly executed.\
72 65 70 6F 72 70 6D 69 20 62 6F 6A  0172
2E 64 65 74 75 63 65 78 65 20 79 6C  017E
                                018A   153 YES_DESC:
                           00000003  018A   154         .LONG   3
                           00000192' 018E   155         .ADDRESS SYM_NAME
                                0192   156 SYM_NAME:                                  ; batch job symbol name
                           4D 59 53  0192   157         .ASCII  \SYM\
                                0195   158 SYM_DESC:
                           00000014  0195   159         .LONG   20
                           0000038A' 0199   160         .ADDRESS SYM
                                019D   161 YES:
                     53 45 59 00'  019D   162         .ASCIC  \YES\                   ; parameter for SNDSMB
                                  03  019D
                                01A1   163 QUENAM1:
55 51 5F 54 41 42 5F 50 54 45 55 00' 01A1   164         .ASCIC  /UETP_BAT_QUE1/
                               31 45  01AD
                                  0D  01A1
                           0000000E  01AF   165         QUENAM1L=.-QUENAM1
                                01AF   166 QUENAM2:
55 51 5F 54 41 42 5F 50 54 45 55 00' 01AF   167         .ASCIC  /UETP_BAT_QUE2/
                               32 45  01BB
                                  0D  01AF
                           0000000E  01BD   168         QUENAM2L=.-QUENAM2
                                01BD   169 MSGVEC:
                           00000003  01BD   170         .LONG   3                       ; PUTMSG message vector
                           00741133  01C1   171         .LONG   UETP$_TEXT
                           00000001  01C5   172         .LONG   1
                           00000169' 01C9   173         .ADDRESS MESSAGEL
                                01CD   174 TEST_ERROR:                                 ; SNDERR test data
                           00000064  01CD   175         .LONG BUF_SIZE
                           000001D5' 01D1   176         .ADDRESS .+4
                           00000000  01D5   177         A=0
                                01D5   178         .REPT BUF_SIZE
                                01D5   179         .BYTE A
                                01D5   180         A=A+1
                                  00  01D5   181         .ENDR
                                0239   182 OPNAME:
                     41 50 4F 5F 00'  0239   183         .ASCIC  /_OPA/                  ; operator console name
                                  04  0239
                                023E   184 OP_MSG1:
                           00000036' 023E   185         .LONG   MSG1L                   ; GENREQ routine OPRMSG buffer
                           00000246' 0242   186         .ADDRESS .+4
                                03  0246   187         .BYTE   OPC$_RQ_RQST
                           00000001  0247   188         .LONG   OPC$M_NM_CENTRL         ; request operator type
                           0000024A  024B   189         .=.-1                           ; is only 3 bytes big
```

```
                       00000000  024A              .LONG   0                         ; global request ID of 0
                                 024E  OP_MESG:
52 50 4F 44 4E 53 24 20 50 54 45 55  024E          .ASCII  /UETP $SNDOPR system service test user message./
76 72 65 73 20 6D 65 74 73 79 73 20  025A
65 73 75 20 74 73 65 74 20 65 63 69  0266
         2E 65 67 61 73 73 65 6D 20 72  0272
                       0000002E  027C              OP_MESG_LEN=.-OP_MESG
                       00000036  027C              MSG1L=.-OP_MSG1-8         ; message buffer size
                                 027C  FILE_NAME:
         4D 4F 43 2E 35 30 53 00'  027C            .ASCIC  /S05.COM/
                             07  027C
                       00000008  0284              NAME_SIZE=.-FILE_NAME
                       00000290  0284              .BLKB   <FILNAMSIZ-NAME_SIZE>     ; filler for SNDSMB
                                 0290  FILE_NAME1:
            47 4F 4C 2E 35 30 53  0290             .ASCII  /S05.LOG/                 ; log file name
                                 0297  COM_FILE:
20 35 30 53 53 53 54 41 53 20 21 24  0297          .ASCII  /$! SATSSS05 SNDSMB test batch job/<CR><LF>
20 74 73 65 74 20 42 4D 53 44 4E 53  02A3
   0A 0D 62 6F 6A 20 68 63 74 61 62  02AF
                          21 24  02BA              .ASCII  /$!/
                       00000025  02BC              REC0_SIZE=.-COM_FILE      ; record 0 size
                                 02BC  REC1:
   27 31 50 27 3D 3A 4D 59 53 20 24  02BC          .ASCII  /$ SYM:='P1'/
                       0000000B  02C7              REC1_SIZE=.-REC1          ; record 1 size
                                 02C7  REC2:
53 51 45 2E 4D 59 53 20 46 49 20 24  02C7          .ASCII  \$ IF SYM.EQS."YES" THEN DEF/GR SYM 'P1'\<CR><LF>
20 4E 45 48 54 20 22 53 45 59 22 2E  02D3
27 20 4D 59 53 20 52 47 2F 46 45 44  02DF
                  0A 0D 27 31 50  02EB
                       00000029  02F0              REC2_SIZE=.-REC2
                                 02F0  OL1:
                            07'  02F0              .BYTE   OL1S
                             21  02F1              .BYTE   SMO$K_HOLD
                             26  02F2              .BYTE   SMO$K_PARAMS
                   53 45 59 00'  02F3              .ASCIC  /YES/
                             03  02F3
                             00  02F7              .BYTE   0
                       00000007  02F8              OL1S=.-OL1-1
                                 02F8  OL2:
                            04'  02F8              .BYTE   OL2S
                             22  02F9              .BYTE   SMO$K_JOBPRI
                             03  02FA              .BYTE   3
                             21  02FB              .BYTE   SMO$K_HOLD
                             00  02FC              .BYTE   0
                       00000004  02FD              OL2S=.-OL2-1
                                 02FD  JN1:
      31 4D 55 4E 5F 42 4F 4A 00'  02FD            .ASCIC  /JOB_NUM1/
                             08  02FD
                             00  0306              .BYTE   0
                                 0307  JN2:
      32 4D 55 4E 5F 42 4F 4A 00'  0307            .ASCIC  /JOB_NUM2/
                             08  0307
                             00  0310              .BYTE   0
                                 0311  JN3:
      33 4D 55 4E 5F 42 4F 4A 00'  0311            .ASCIC  /JOB_NUM3/
                             08  0311
                             00  031A              .BYTE   0
```

```
                     031B   235 ;
                     031B   236         .SBTTL  R/W PSECT
                 00000000   237         .PSECT  RWDATA,RD,WRT,NOEXE,LONG
                     0000   238 ;
                     0000   239 PID:
                 00000000   0000   240         .LONG   0                       ; PID for this process
                     0004   241 CURRENT_TC:
                 00000000   0004   242         .LONG   0                       ; ptr to current test case
                     0008   243         .ALIGN LONG
                     0008   244 REG_SAVE_AREA:
                 00000044   0008   245         .BLKL   15                      ; register save area
                     0044   246 MOD_MSG_CODE:
                 007480D9   0044   247         .LONG   UETP$_SATSMS            ; test module message code for putmsg
                     0048   248 TMN_ADDR:
                 00000000'  0048   249         .ADDRESS TEST_MOD_NAME
                     004C   250 TMD_ADDR:
                 00000019'  004C   251         .ADDRESS TEST_MOD_BEGIN
                     0050   252 PRVPRT:
                       00   0050   253         .BYTE   0                       ; protection return byte for SETPRT
                     0051   254 PRIVMASK:
             00000000 00000000  0051   255         .QUAD   0                       ; priv. mask
                     0059   256 CHM_CONT:
                 00000000   0059   257         .LONG   0                       ; change mode continue address
                     005D   258 RETADR:
                 00000065   005D   259         .BLKL   2                       ; returned address's from SETPRT
                     0065   260 STATUSM:
                 00000000   0065   261         .LONG   0
                     0069   262 QIO:
                     0069   263         $QIO 2,MBCHAN,IO$_READVBLK,,,,BUF,BUF_SIZE+30 ; QIO parameter list
                     009D   264 SNDA:
                     009D   265         $SNDACC ACC_DESC,MBCHAN         ; SNDACC parameter list
                     00A9   266 SNDE:
                     00A9   267         $SNDERR TEST_ERROR              ; SNDERR paramter list
                     00B1   268 SNDO:
                     00B1   269         $SNDOPR OPMSG_DESC,0            ; SNDOPR parameter list
                     00BD   270 SNDS:
                     00BD   271         $SNDSMB SMSG_DESC,0             ; SNDSMB paramter list
                     00C9   272 REG:
74 73 69 67 65 72 000000D1'010E0000'  00C9   273         .ASCID  \register R\
                   52 20 72 65   00D7
                     00DB   274 REGNUM:
                 00000000   00DB   275         .LONG   0                       ; register number
                     00DF   276 MSGL:
                 00000082   00DF   277         .LONG   130                     ; buffer desc.
                 000000E7'  00E3   278         .ADDRESS BUF
                     00E7   279 BUF:
                 00000169   00E7   280         .BLKB   130
                     0169   281 MESSAGEL:
                 00000000   0169   282         .LONG   0                       ; message desc.
                 000000E7'  016D   283         .ADDRESS BUF
                     0171   284 SERV_NAME:
                 00000000   0171   285         .LONG   0                       ; service name pointer
                     0175   286 MBCHAN:
                     0000   0175   287         .WORD   0                       ; mailbox channel number
                     0177   288 MODE:
                 00000000   0177   289         .LONG   0                       ; current mode string pointer
                     017B   290 MBUF:
```

```
                                                     C 8
000001CB  017B   291          .BLKB   80                      ; mailbox buffer
          01CB   292 MSGVEC1:                                 ; PUTMSG message vector
00000003  01CB   293          .LONG   3
00741133  01CF   294          .LONG   UETP$_TEXT
00000001  01D3   295          .LONG   1
00000000  01D7   296          .LONG   0
          01DB   297 STATUS:
000001E3  01DB   298          .BLKL   2                       ; mailbox status block
          01E3   299 ACC_MSG:
    0001  01E3   300          .WORD   ACC$K_INSMESG           ; starting message code
    0052' 01E5   301          .WORD   MSG_SIZE                ; message size
00000001  01E7   302          .LONG   1                       ; final exit status
00000000  01EB   303          .LONG   0                       ; PID
00000002  01EF   304          .LONG   2                       ; job ID
00000000 00000000 01F3   305  .QUAD   0                       ; system job termination time
54 53 45 54 53 59 53 00' 01FB   306  .ASCIC  /SYSTEST/        ; account name
                   07    01FB
43 41 44 4E 53 24 20 50 54 45 55 00' 0203   307  .ASCIC  /UETP $SNDACC system service test user data record/     ; user data
72 65 73 20 6D 65 74 73 79 73 20 43  020F
73 75 20 74 73 65 74 20 65 63 69 76  021B
6F 63 65 72 20 61 74 61 64 20 72 65  0227
                64 72    0233
                31       0203
00000052  0235   308          MSG_SIZE=.-ACC_MSG
          0235   309 ACC_MSG1:
    0006  0235   310          .WORD   ACC$K_DISASEL           ; function code
      02  0237   311          .BYTE   ACC$K_BATTRM            ; batch job type
      11  0238   312          .BYTE   ACC$K_INSMSG            ; arbitrary message type
      03  0239   313          .BYTE   ACC$K_INTTRM            ; interactive job type
      04  023A   314          .BYTE   ACC$K_LOGTRM            ; login failure termination type
      01  023B   315          .BYTE   ACC$K_PRCTRM            ; non-interactive process type
      10  023C   316          .BYTE   ACC$K_PRTJOB            ; print job type
      00  023D   317          .BYTE   0                       ; terminator byte
00000009  023E   318          MSG1_SIZE=.-ACC_MSG1
          023E   319 ACC_DESC:
00000052  023E   320          .LONG   MSG_SIZE                ; descriptor for accounting message
000001E3' 0242   321          .ADDRESS ACC_MSG
```

```
          0246  OPTYPE:
01000001  0246          .LONG   OPC$M_NM_CENTRL!<1@24>  ; opr type & ID table
02000002  024A          .LONG   OPC$M_NM_PRINT!<2@24>
03000004  024E          .LONG   OPC$M_NM_TAPES!<3@24>
04000008  0252          .LONG   OPC$M_NM_DISKS!<4@24>
05000010  0256          .LONG   OPC$M_NM_DEVICE!<5@24>
06001000  025A          .LONG   OPC$M_NM_OPER1!<6@24>
07002000  025E          .LONG   OPC$M_NM_OPER2!<7@24>
08004000  0262          .LONG   OPC$M_NM_OPER3!<8@24>
09008000  0266          .LONG   OPC$M_NM_OPER4!<9@24>
0A010000  026A          .LONG   OPC$M_NM_OPER5!<10@24>
0B020000  026E          .LONG   OPC$M_NM_OPER6!<11@24>
0C040000  0272          .LONG   OPC$M_NM_OPER7!<12@24>
0D080000  0276          .LONG   OPC$M_NM_OPER8!<13@24>
0E100000  027A          .LONG   OPC$M_NM_OPER9!<14@24>
0F200000  027E          .LONG   OPC$M_NM_OPER10!<15@24>
10400000  0282          .LONG   OPC$M_NM_OPER11!<16@24>
11800000  0286          .LONG   OPC$M_NM_OPER12!<17@24>
12000001  028A          .LONG   OPC$M_NM_CENTRL!<18@24> ; just to make an even number
          028E  OPMSG_DESC:
00000080' 028E          .LONG   MSG_LEN                 ; SNDOPR msg buffer desc
00000296' 0292          .ADDRESS OPMSG
          0296  OPMSG:
      03  0296          .BYTE   OPC$_RQ_RQST            ; function code
0000029A  0297          .BLKB   3                       ; operator type
00000000  029A          .LONG   0                       ; ID
00000316  029E          .BLKB   120                     ; message or terminal info
00000080  0316          MSG_LEN=.-OPMSG
          0316  SMSG_DESC:
0000006C' 0316          .LONG   SMSG_LEN                ; SNDSMB msg buffer desc
0000031E' 031A          .ADDRESS SMSG
          031E  SMSG:
    0000  031E          .WORD   SMR$K_INITIAL           ; SNDSMB msg buffer
00000330  0320          .BLKB   16                      ; queue name
          0330  SMSG1:
00000340  0330          .BLKB   16                      ; device name
00000346  0340          .BLKB   6                       ; file ID
0000034C  0346          .BLKB   6                       ; directory ID
00000360  034C          .BLKB   20                      ; filename
00000362  0360          .BLKB   2                       ; Job ID
0000036A  0362          .BLKB   8                       ; job name
0000038A  036A          .BLKB   32                      ; room for options and option data
0000006C  038A          SMSG_LEN=.-SMSG
          038A  SYM:
0000039E  038A          .BLKB   20
          039E  ;
          039E          .ALIGN LONG
          03A0  NAMBLK:
          03A0          $NAM
          0400  FAB:
          0400          $FAB    FAC=PUT,-
          0400                  FNA=FILE_NAME+1,-
          0400                  FNS=COM_FIL_SIZ,-
          0400                  NAM=NAMBLK,-
          0400                  RAT=CR,-
          0400                  RFM=VAR
          0450  RAB:
```

```
                                E 8
0450    380             $RAB    FAB=FAB,-
0450    381                     MBF=1,-
0450    382                     RBF=COM_FILE,-
0450    383                     RSZ=RECO_SIZE
0494    384 FAB1:
0494    385             $FAB    FAC=PUT,-
0494    386                     FNA=FILE_NAME1,-
0494    387                     FNS=COM_FIL_SIZ
```

```
00000000   389          .PSECT  SATSSS05,RD,WRT,EXE,LONG
     0000  390          .SBTTL  SATSSS05
     0000  391 ;++
     0000  392 ; FUNCTIONAL DESCRIPTION:
     0000  393 ;
     0000  394 ;       After performing some initial housekeeping, such as
     0000  395 ; printing the module begin message and acquiring needed privileges,
     0000  396 ; the system services are tested in each of their normal conditions.
     0000  397 ; Detected failures are identified and  an error message is printed
     0000  398 ; on the terminal.  Upon completion of the test a success or fail
     0000  399 ; message is printed on the terminal.
     0000  400 ;
     0000  401 ; CALLING SEQUENCE:
     0000  402 ;
     0000  403 ;       $ RUN SATSSS05  ...  (DCL COMMAND)
     0000  404 ;
     0000  405 ; INPUT PARAMETERS:
     0000  406 ;
     0000  407 ;       none
     0000  408 ;
     0000  409 ; IMPLICIT INPUTS:
     0000  410 ;
     0000  411 ;       none
     0000  412 ;
     0000  413 ; OUTPUT PARAMETERS:
     0000  414 ;
     0000  415 ;       none
     0000  416 ;
     0000  417 ; IMPLICIT OUTPUTS:
     0000  418 ;
     0000  419 ;       Messages to SYS$OUTPUT are the only output from SATSSS05.
     0000  420 ;       They are of the form:
     0000  421 ;
     0000  422 ;               %UETP-S-SATSMS, TEST MODULE SATSSS05 BEGUN ... (BEGIN MSG)
     0000  423 ;               %UETP-S-SATSMS, TEST MODULE SATSSS05 SUCCESSFUL ... (END MSG)
     0000  424 ;               %UETP-E-SATSMS, TEST MODULE SATSSS05 FAILED ... (END MSG)
     0000  425 ;               %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
     0000  426 ;
     0000  427 ; COMPLETION CODES:
     0000  428 ;
     0000  429 ;       The SATSSS05 routine terminates with a $EXIT to the
     0000  430 ;       operating system with a status code defined by UETP$_SATSMS.
     0000  431 ;
     0000  432 ; SIDE EFFECTS:
     0000  433 ;
     0000  434 ;       none
     0000  435 ;
     0000  436 ;--
     0000  437
     0000  438          TEST_START SATSSS05             ; let the test begin
```

```
                        0000
                   0000 0000                    .ENTRY SATSSS05,0
          0004'CF   D4   0002                   CLRL    W^CURRENT_TC
                00  DD   0006                   PUSHL   #0
          0000'CF   DF   0008                   PUSHAL  W^TPID
 00000000'GF    02   FB   000C                   CALLS   #2,G^SYS$WAKE
 00000000'GF    00   FB   0013                   CALLS   #0,G^SYS$HIBER
          0009'CF   7F   001A                   PUSHAQ  W^TEST_MOD_NAME_D
 00000000'GF    01   FB   001E                   CALLS   #1,G^SYS$SETPRN
            0ED4   30   0025                   BSBW    W^MOD_MSG_PRINT
     004C'CF 001F'CF   DE   0028                MOVAL   W^TEST_MOD_SUCC,W^TMD_ADDR
0044'CF 03  00   01   F0   002F                INSV    #SUCCESS,#0,#3,W^MOD_MSG_CODE
                00   DD   0036                PUSHL   #0
          0AFE'CF   01   FB   0038                CALLS #1,W^REG_SAVE
                        003D            STP0:
                        003D    439             .SBTTL  SNDACC TESTS
                        003D    440 ;+
                        003D    441 ;
                        003D    442 ; $SNDACC tests
                        003D    443 ;
                        003D    444 ; test ACC$K_NEWFILE
                        003D    445 ;
                        003D    446 ; This function will not be tested because of the possible interference
                        003D    447 ; that it might cause with the ACCOUNTNG.DAT file on a customer's system.
                        003D    448 ;
                        003D    449 ; test ACC$K_INSMESG
                        003D    450 ;
                        003D    451 ;-
                        003D    452             MODE    TO,10$,KRNL,NOREGS      ; kernal mode to access PHD
  59  00000000'9F   D0  005A    453             MOVL    a#CTL$GL_PHD,R9         ; get process header address
      0051'CF   69  DE  0061    454             MOVAL   PHD$Q_PRIVMSK(R9),W^PRIVMASK ; get priv mask address
                        0066    455             MODE    FROM,TO$                ; get back to user mode
                        0067    456             PRIV    ADD,OPER                ; add the OPER priv.
0171'CF   0031'CF   DE  0087    457             MOVAL   W^SNDACC,W^SERV_NAME    ; set service name
0177'CF   0134'CF   DE  008E    458             MOVAL   W^UM,W^MODE             ; set the mode
                00  DD  0095    459             PUSHL   #0                      ; push a dummy param
      0AFE'CF   01  FB  0097    460             CALLS   #1,W^REG_SAVE           ; save a reg snapshot
                        009C    461             $CREMBX_S CHAN=W^MBCHAN         ; create a mailbox
              09 50   E8  00AF  462             BLBS    R0,20$                  ; br if OK
                        00B2    463             $EXIT_S R0                      ; exit and show why
                        00BB    464 20$:
                        00BB    465             $SNDACC_S MSGBUF=W^ACC_DESC,-
                        00BB    466                       CHAN  =W^MBCHAN       ; try a ACC$K_NEWFILE
                        00CB    467             FAIL_CHECK SS$_NORMAL           ; check for success
 00000000'8F    DD   00CB                               PUSHL   #SS$_NORMAL
      0B08'CF   01  FB  00D1                            CALLS   #1,W^REG_CHECK
      0C0E'CF   00  FB  00D6    468             CALLS #0,W^READ_CHECK           ; check the mailbox
                        00DB    469 ;+
                        00DB    470 ;
                        00DB    471 ; test ACC$K_DISACC
                        00DB    472 ;
                        00DB    473 ;-
                        00DB    474             NEXT_TEST
                        00DB
                        00DB            STP1:
      0004'CF   01  D0  00DB                    MOVL    #1,W^CURRENT_TC
                00  DD  00E0                    PUSHL   #0
```

```
      0AFE'CF   01   FB  00E2                     CALLS   #1,W^REG_SAVE
      01E3'CF   04   B0  00E7            MOVW    #ACC$K_DISAACC,W^ACC_MSG ; set the new function
 00A5'CF   0175'CF   3C  00EC            MOVZWL  W^MBCHAN,W^SNDA+SNDACC$_CHAN ; set up the channel number
                         00F3            $SNDACC_G W^SNDA                ; try ACC$K_DISAACC with a little _G
                         00FC            FAIL_CHECK SS$_NORMAL           ; check for success
    00000000'8F          DD  00FC                     PUSHL   #SS$_NORMAL
      0B08'CF   01   FB  0102                     CALLS   #1,W^REG_CHECK
      0C0E'CF   00   FB  0107            CALLS #0,W^READ_CHECK           ; check the mailbox
                         010C       ;+
                         010C       ;
                         010C       ; test ACC$K_ENABACC
                         010C       ;
                         010C       ;-
                         010C            NEXT_TEST
                         010C
                         010C       STP2:
      0004'CF   02   D0  010C                     MOVL    #2,W^CURRENT_TC
                00   DD  0111                     PUSHL   #0
      0AFE'CF   01   FB  0113                     CALLS   #1,W^REG_SAVE
      01E3'CF   03   B0  0118            MOVW    #ACC$K_ENABACC,W^ACC_MSG ; set function code
                         011D            $SNDACC_S MSGBUF=W^ACC_DESC,-
                         011D                 CHAN  =W^MBCHAN            ; try ACC$K_ENABACC with a little _S
                         012D            FAIL_CHECK SS$_NORMAL           ; check for success
    00000000'8F          DD  012D                     PUSHL   #SS$_NORMAL
      0B08'CF   01   FB  0133                     CALLS   #1,W^REG_CHECK
      0C0E'CF   00   FB  0138            CALLS #0,W^READ_CHECK           ; check the mailbox
                         013D       ;+
                         013D       ;
                         013D       ; test ACC$K_DISASEL with all types selected
                         013D       ;
                         013D       ;-
                         013D            NEXT_TEST
                         013D
                         013D       STP3:
      0004'CF   03   D0  013D                     MOVL    #3,W^CURRENT_TC
                00   DD  0142                     PUSHL   #0
      0AFE'CF   01   FB  0144                     CALLS   #1,W^REG_SAVE
 0242'CF   0235'CF   DE  0149            MOVAL   W^ACC_MSG1,W^ACC_DESC+4 ; set new message address
      023E'CF   09   D0  0150            MOVL    #MSG1_SIZE,W^ACC_DESC   ; set new message size
                         0155            $SNDACC_G W^SNDA                        ; try ACC$K_DISASEL
                         015E            FAIL_CHECK SS$_NORMAL                   ; check for success
    00000000'8F          DD  015E                     PUSHL   #SS$_NORMAL
      0B08'CF   01   FB  0164                     CALLS   #1,W^REG_CHECK
      0C0E'CF   00   FB  0169            CALLS #0,W^READ_CHECK                   ; check the mailbox
                         016E       ;+
                         016E       ;
                         016E       ; test ACC$K_ENABSEL
                         016E       ;
                         016E       ;-
                         016E            NEXT_TEST
                         016E
                         016E       STP4:
      0004'CF   04   D0  016E                     MOVL    #4,W^CURRENT_TC
                00   DD  0173                     PUSHL   #0
      0AFE'CF   01   FB  0175                     CALLS   #1,W^REG_SAVE
      0235'CF   05   B0  017A            MOVW    #ACC$K_ENABSEL,W^ACC_MSG1       ; set new function
                         017F            $SNDACC_S MSGBUF=W^ACC_DESC,-
```

```
                          017F   510          CHAN   =W^MBCHAN              ; try ACC$K_ENABSEL
                          018F   511   FAIL_CHECK SS$_NORMAL                ; check for success
  00000000'8F       DD    018F                 PUSHL   #SS$_NORMAL
 0B08'CF    01      FB    0195                 CALLS   #1,W^REG_CHECK
 0C0E'CF    00      FB    019A   512   CALLS #0,W^READ_CHECK               ; check the mailbox
```

```
                                  019F          .SBTTL  SNDERR_S TESTS
                                  019F   ;+
                                  019F   ;
                                  019F   ; $SNDERR_S tests
                                  019F   ;
                                  019F   ;-
                                  019F          NEXT_TEST
                                  019F
                                  019F   STP5:
           0004'CF     05    D0   019F                  MOVL    #5,W^CURRENT_TC
                       00    DD   01A4                  PUSHL   #0
           0AFE'CF     01    FB   01A6                  CALLS   #1,W^REG_SAVE
                                  01AB          PRIV    ADD,BUGCHK                      ; add the BUGCHK priv.
    0171'CF    0038'CF       DE   01CB          MOVAL   W^SNDERR,W^SERV_NAME            ; set service name
                                  01D2          $CREMBX_S CHAN=W^MBCHAN,-
                                  01D2                  LOGNAM=W^MBNAM,-
                                  01D2                  PRMFLG=#0                       ; make a mailbox
                                  01E9          $GETCHN_S CHAN=W^MBCHAN,-
                                  01E9                  PRIBUF=W^MSGL                   ; get the unit number
          7E    00F3'CF      3C   01FF          MOVZWL  W^BUF+DIB$W_UNIT,-(SP)          ; push the MBX unit #
    00000000'GF        01    FB   0204          CALLS   #1,G^SYS$DERLMB                 ; declare errorlog MBX
                                  020B          $SNDERR_S MSGBUF=W^TEST_ERROR           ; try _S form
                                  0216          FAIL_CHECK SS$_NORMAL                   ; check for success
        00000000'8F          DD   0216                  PUSHL   #SS$_NORMAL
           0B08'CF     01    FB   021C                  CALLS   #1,W^REG_CHECK
                                  0221   GET1:
    0071'CF    0175'CF       B0   0221          MOVW    W^MBCHAN,W^QIO+QIO$_CHAN        ; get the channel number
                                  0228          $QIO_G  W^QIO                           ; do a read
                                  0231          $WAITFR_S EFN=#2                        ; wait for it to complete
      0075'CF    00'8F       88   023A          BISB2   #IO$M_NOW,W^QIO+QIO$_FUNC       ; set the NOW modifier
         00EB'CF       27    B1   0240          CMPW    #EMB$C_SS,W^BUF+EMB$W_HD_ENTRY  ; is this the right entry?
                       DA    12   0245          BNEQ    GET1                            ; br if not
          56    00F9'CF      DE   0247          MOVAL   W^BUF+18,R6                     ; set buffer address
          57    01D5'CF      DE   024C          MOVAL   W^TEST_ERROR+8,R7               ; set good data address
       58    00000064 8F     D0   0251          MOVL    #BUF_SIZE,R8                    ; set byte count
         0E1A'CF       00    FB   0258          CALLS   #0,W^BUF_CHECK                  ; check results
                                  025D   ;+
                                  025D   ;
                                  025D   ; $SNDERR_G tests
                                  025D   ;
                                  025D   ;-
                                  025D          NEXT_TEST
                                  025D
                                  025D   STP6:
           0004'CF     06    D0   025D                  MOVL    #6,W^CURRENT_TC
                       00    DD   0262                  PUSHL   #0
           0AFE'CF     01    FB   0264                  CALLS   #1,W^REG_SAVE
 0064 8F    00   00E7'CF   00  2C 0269          MOVC5   #0,W^BUF,#0,#BUF_SIZE,W^BUF     ; zero the buffer
                00E7'CF           0272
                       00    DD   0275          PUSHL   #0                              ; push a dummy parameter
           0AFE'CF     01    FB   0277          CALLS   #1,W^REG_SAVE                   ; save a reg snapshot
                                  027C          $SNDERR_G W^SNDE                        ; try _G
                                  0285          FAIL_CHECK SS$_NORMAL                   ; check for success
        00000000'8F          DD   0285                  PUSHL   #SS$_NORMAL
           0B08'CF     01    FB   028B                  CALLS   #1,W^REG_CHECK
   0075'CF    00000000'8F    CA   0290          BICL2   #IO$M_NOW,W^QIO+QIO$_FUNC       ; set to wait for mailbox
                                  0299   GET2:
```

```
                                0299    556         $QIO_G  W^QIO                                   ; read the mailbox
                                02A2    557         $WAITFR_S EFN=#2                                ; wait for completion
          0075'CF   00'8F   88  02AB    558         BISB2   #IO$M_NOW,W^QIO+QIO$_FUNC               ; set to read it until found
             00EB'CF    27   B1  02B1    559         CMPW    #EMB$C_SS,W^BUF+EMB$W_HD_ENTRY          ; the right entry?
                        E1   12  02B6    560         BNEQ    GET2                                    ; br if not
             0E1A'CF    00   FB  02B8    561         CALLS   #0,W^BUF_CHECK                          ; check results
                        7E   D4  02BD    562         CLRL    -(SP)                                   ; set channel to 0
       00000000'GF      01   FB  02BF    563         CALLS   #1,G^SYS$DERLMB                         ; reset the error logger
```

```
                                02C6            .SBTTL SNDOPR TESTS
                                02C6    ;+
                                02C6    ;
                                02C6    ;
                                02C6    ;$SNDOPR tests
                                02C6    ;
                                02C6    ; DISABLE tests with _S
                                02C6    ;
                                02C6    ;-
                                02C6            NEXT_TEST
                                02C6
                                02C6    STP7:
       0004'CF     07    D0     02C6                    MOVL    #7,W^CURRENT_TC
                   00    DD     02CB                    PUSHL   #0
       0AFE'CF     01    FB     02CD                    CALLS   #1,W^REG_SAVE
  0171'CF   003F'CF      DE     02D2            MOVAL   W^SNDOPR,W^SERV_NAME                    ; set service name
       0296'CF     01    90     02D9            MOVB    #OPC$_RQ_TERME,W^OPMSG                  ; set the function code
             0297'CF     D4     02DE            CLRL    W^OPMSG+OPC$B_MS_ENAB                   ; set disable ID mask
 029A'CF    FFFFFFFF 8F  D0     02E2            MOVL    #-1,W^OPMSG+OPC$L_MS_MASK               ; set operators to be disabled
             029E'CF     B4     02EB            CLRW    W^OPMSG+OPC$W_MS_OUNIT                  ; set unit to zero
  02A0'CF   0239'CF      90     02EF            MOVB    W^OPNAME,W^OPMSG+OPC$T_MS_ONAME         ; set operator name size
  02A1'CF   023A'CF      D0     02F6            MOVL    W^OPNAME+1,W^OPMSG+OPC$T_MS_ONAME+1     ; set operator device name
                                02FD            $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
                                02FD                    CHAN=W^MBCHAN                           ; try _S
                                030D            FAIL_CHECK SS$_NORMAL                           ; check success
     00000000'8F         DD     030D                    PUSHL   #SS$_NORMAL
       0B08'CF     01    FB     0313                    CALLS   #1,W^REG_CHECK
                                0318    ;+
                                0318    ;
                                0318    ; ENABLE tests with _S
                                0318    ;
                                0318    ;-
                                0318            NEXT_TEST
                                0318
                                0318    STP8:
       0004'CF     08    D0     0318                    MOVL    #8,W^CURRENT_TC
                   00    DD     031D                    PUSHL   #0
       0AFE'CF     01    FB     031F                    CALLS   #1,W^REG_SAVE
 0297'CF    00FFF01F 8F  D0     0324            MOVL    #ALL_OPR,W^OPMSG+OPC$B_MS_ENAB          ; set oprators to enable
 029A'CF    FFFFFFFF 8F  D0     032D            MOVL    #-1,W^OPMSG+OPC$L_MS_MASK               ; set enableable bits
  029E'CF   0154'CF      B0     0336            MOVW    W^TTUNIT,W^OPMSG+OPC$W_MS_OUNIT         ; set the terminal unit number
  02A0'CF   014F'CF      90     033D            MOVB    W^TTNAM,W^OPMSG+OPC$T_MS_ONAME          ; set the terminal name size
  02A1'CF   0150'CF      D0     0344            MOVL    W^TTNAM+1,W^OPMSG+OPC$T_MS_ONAME+1 ; set the terminal name
                                034B            $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
                                034B                    CHAN=W^MBCHAN                           ; enable the alternate terminal
                                035B            FAIL_CHECK SS$_NORMAL                           ; check for success
     00000000'8F         DD     035B                    PUSHL   #SS$_NORMAL
       0B08'CF     01    FB     0361                    CALLS   #1,W^REG_CHECK
                                0366    ;+
                                0366    ;
                                0366    ; RQST tests to make a request with ID = 1-18
                                0366    ;
                                0366    ;-
                                0366            NEXT_TEST
                                0366
                                0366    STP9:
       0004'CF     09    D0     0366                    MOVL    #9,W^CURRENT_TC
```

```
             00   DD 036B                       PUSHL   #0
      0AFE'CF 01  FB 036D                       CALLS   #1,W^REG_SAVE
00B9'CF 0175'CF 3C 0372          MOVZWL  W^MBCHAN,W^SNDO+SNDOPR$_CHAN         ; set the channel number
   0296'CF 03  90 0379           MOVB    #OPC$_RQ_RQST,W^OPMSG                ; set function code
   024E'CF 2E  28 037E           MOVC3   #OP_MESG_LEN,W^OP_MESG,-
       029E'CF       0383                W^OPMSG+OPC$L_MS_TEXT                 ; put the text in the message
            57    D4 0386        CLRL    R7                                    ; init loop variable
       029A'CF    D4 0388        CLRL    W^OPMSG+OPC$L_MS_RQSTID               ; init the ID field
  52   0246'CF    DE 038C        MOVAL   W^OPTYPE,R2                           ; set oper type list pointer
                     0391   10$:
   0297'CF 82  D0 0391           MOVL    (R2)+,W^OPMSG+1                       ; set opr type & ID
            00    DD 0396        PUSHL   #0                                    ; push a dummy parameter
   0AFE'CF 01  FB 0398           CALLS   #1,W^REG_SAVE                         ; save the registers
                     039D        $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
                     039D                CHAN=W^MBCHAN                         ; try _S form
                     03AD        FAIL_CHECK SS$_NORMAL                         ; check for success
  00000000'8F     DD 03AD                       PUSHL   #SS$_NORMAL
   0B08'CF 01  FB 03B3                       CALLS   #1,W^REG_CHECK
   0297'CF 82  D0 03B8           MOVL    (R2)+,W^OPMSG+1                       ; set opr type & ID
            00    DD 03BD        PUSHL   #0                                    ; push a dummy param
   0AFE'CF 01  FB 03BF           CALLS   #1,W^REG_SAVE                         ; save a reg snapshot
                     03C4        $SNDOPR_G W^SNDO                              ; try _G
                     03CD        FAIL_CHECK SS$_NORMAL                         ; check for success
  00000000'8F     DD 03CD                       PUSHL   #SS$_NORMAL
   0B08'CF 01  FB 03D3                       CALLS   #1,W^REG_CHECK
     B5 57 09  F2 03D8           AOBLSS  #9,R7,10$                             ; do all opr types
                     03DC   ;+
                     03DC   ;
                     03DC   ; CANCEL tests to cancel requests 1-18
                     03DC   ;
                     03DC   ;-
                     03DC        NEXT_TEST
                     03DC
                     03DC   STP10:
   0004'CF 0A  D0 03DC                       MOVL    #10,W^CURRENT_TC
            00    DD 03E1                       PUSHL   #0
   0AFE'CF 01  FB 03E3                       CALLS   #1,W^REG_SAVE
   0296'CF 05  90 03E8           MOVB    #OPC$_RQ_CANCEL,W^OPMSG               ; set function code
  52   0246'CF    DE 03ED        MOVAL   W^OPTYPE,R2                           ; set table pointer
56  00008084 8F   D0 03F2        MOVL    #<OPC$_RQSTCAN&^XFFFF>,R6             ; set completion code
            57    D4 03F9        CLRL    R7                                    ; set loop variable
                     03FB   10$:
   0297'CF 82  D0 03FB           MOVL    (R2)+,W^OPMSG+1                       ; set opr type & ID
            00    DD 0400        PUSHL   #0                                    ; push a dummy parameter
   0AFE'CF 01  FB 0402           CALLS   #1,W^REG_SAVE                         ; save a reg snapshot
                     0407        $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
                     0407                CHAN=W^MBCHAN                         ; try _S form
                     0417        FAIL_CHECK SS$_NORMAL                         ; check success
  00000000'8F     DD 0417                       PUSHL   #SS$_NORMAL
   0B08'CF 01  FB 041D                       CALLS   #1,W^REG_CHECK
   0E59'CF 00  FB 0422           CALLS   #0,W^SND_CHECK                        ; check the results
   0297'CF 82  D0 0427           MOVL    (R2)+,W^OPMSG+1                       ; set opr type & ID
            00    DD 042C        PUSHL   #0                                    ; push a dummy parameter
   0AFE'CF 01  FB 042E           CALLS   #1,W^REG_SAVE                         ; save a reg snapshot
                     0433        $SNDOPR_G W^SNDO                              ; try _G form
                     043C        FAIL_CHECK SS$_NORMAL                         ; check success
  00000000'8F     DD 043C                       PUSHL   #SS$_NORMAL
```

```
        0B08'CF  01  FB 0442                      CALLS   #1,W^REG_CHECK
        0E59'CF  00  FB 0447        CALLS   #0,W^SND_CHECK                  ; check the results
          AB 57  09  F2 044C        AOBLSS  #9,R7,10$                       ; do all opr types
                        0450  ;+
                        0450  ;
                        0450  ; REPLY tests to respond to requests
                        0450  ;
                        0450  ;-
                        0450        NEXT_TEST
                        0450
                        0450  STP11:
        0004'CF  0B  D0 0450                      MOVL    #11,W^CURRENT_TC
                 00  DD 0455                      PUSHL   #0
        0AFE'CF  01  FB 0457                      CALLS   #1,W^REG_SAVE
   56   00008029 8F  D0 045C        MOVL    #<OPC$_RQSTCMPLTE&^XFFFF>,R6    ; set expected status return
        0296'CF  04  90 0463        MOVB    #OPC$_RQ_REPLY,W^OPMSG          ; set the function
        0298'CF  56  B0 0468        MOVW    R6,W^OPMSG+OPC$W_MS_STATUS      ; set status reply return
            029A'CF  D4 046D        CLRL    W^OPMSG+OPC$L_MS_RPLYID         ; set the message ID
    029E'CF  0154'CF B0 0471        MOVW    W^TTUNIT,W^OPMSG+OPC$W_MS_OUNIT ; set the unit number
02A0'CF  014F'CF  05  28 0478       MOVC3   #5,W^TTNAM,W^OPMSG+OPC$T_MS_ONAME ; set the device name
        024E'CF  2E  28 0480        MOVC3   #OP_MESG_LEN,W^OP_MESG,-
            02B0'CF     0485                W^OPMSG+OPC$L_MS_OTEXT          ; set the message text
                 00  DD 0488        PUSHL   #0                              ; push a dummy parameter
        0AFE'CF  01  FB 048A        CALLS   #1,W^REG_SAVE                   ; save a reg snapshot
        0EB1'CF  00  FB 048F        CALLS   #0,W^GENREQ                     ; generate a pending request
                        0494        $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
                        0494                CHAN=W^MBCHAN                   ; try _S
                        04A4        FAIL_CHECK SS$_NORMAL                   ; check success
    00000000'8F      DD 04A4                      PUSHL   #SS$_NORMAL
        0B08'CF  01  FB 04AA                      CALLS   #1,W^REG_CHECK
        0E59'CF  00  FB 04AF        CALLS   #0,W^SND_CHECK                  ; check results
   56    801C 8F     B0 04B4        MOVW    #<OPC$_RQSTABORT&^XFFFF>,R6     ; set expected status return
        0298'CF  56  B0 04B9        MOVW    R6,W^OPMSG+OPC$W_MS_STATUS      ; set reply status code
                 00  DD 04BE        PUSHL   #0                              ; push a dummy parameter
        0AFE'CF  01  FB 04C0        CALLS   #1,W^REG_SAVE                   ; save a reg snapshot
        0EB1'CF  00  FB 04C5        CALLS   #0,W^GENREQ                     ; generate a pending request
                        04CA        $SNDOPR_G W^SNDO                        ; try _G
                        04D3        FAIL_CHECK SS$_NORMAL                   ; check success
    00000000'8F      DD 04D3                      PUSHL   #SS$_NORMAL
        0B08'CF  01  FB 04D9                      CALLS   #1,W^REG_CHECK
        0E59'CF  00  FB 04DE        CALLS   #0,W^SND_CHECK                  ; check results
   56    8021 8F     B0 04E3        MOVW    #<OPC$_RQSTPEND&^XFFFF>,R6      ; set expected status return
        0298'CF  56  B0 04E8        MOVW    R6,W^OPMSG+OPC$W_MS_STATUS      ; set the reply status code
                 00  DD 04ED        PUSHL   #0                              ; push a dummy parameter
        0AFE'CF  01  FB 04EF        CALLS   #1,W^REG_SAVE                   ; save a reg snapshot
        0EB1'CF  00  FB 04F4        CALLS   #0,W^GENREQ                     ; generate a pending request
                        04F9        $SNDOPR_S MSGBUF=W^OPMSG_DESC,-
                        04F9                CHAN=W^MBCHAN                   ; try _S and leave the request pendi
                        0509        FAIL_CHECK SS$_NORMAL                   ; check success
    00000000'8F      DD 0509                      PUSHL   #SS$_NORMAL
        0B08'CF  01  FB 050F                      CALLS   #1,W^REG_CHECK
        0E59'CF  00  FB 0514        CALLS   #0,W^SND_CHECK                  ; check results
   56    8084 8F     B0 0519        MOVW    #<OPC$_RQSTCAN&^XFFFF>,R6       ; set expected status return
        0298'CF  56  B0 051E        MOVW    R6,W^OPMSG+OPC$W_MS_STATUS      ; set reply status
                 00  DD 0523        PUSHL   #0                              ; push a dummy parameter
        0AFE'CF  01  FB 0525        CALLS   #1,W^REG_SAVE                   ; save a reg snapshot
                        052A        $SNDOPR_G W^SNDO                        ; try _G
```

```
                                  0533          FAIL_CHECK SS$_NORMAL                   ; check success
          00000000'8F      DD     0533                    PUSHL   #SS$_NORMAL
          0B08'CF    01    FB     0539                    CALLS   #1,W^REG_CHECK
          0E59'CF    00    FB     053E          CALLS   #0,W^SND_CHECK                  ; check results
                                  0543     ;+
                                  0543     ;
                                  0543     ; DISABLE tests with _G
                                  0543     ;
                                  0543     ;-
                                  0543          NEXT_TEST
                                  0543
                                  0543     STP12:
          0004'CF    0C    D0     0543                    MOVL    #12,W^CURRENT_TC
                     00    DD     0548                    PUSHL   #0
          0AFE'CF    01    FB     054A                    CALLS   #1,W^REG_SAVE
          0296'CF    01    90     054F          MOVB    #OPC$_RQ_TERME,W^OPMSG          ; set the function code
                0297'CF    D4     0554          CLRL    W^OPMSG+OPC$B_MS_ENAB           ; set disable ID mask
029A'CF   FFFFFFFF 8F      D0     0558          MOVL    #-1,W^OPMSG+OPC$L_MS_MASK       ; set operators to disable
                                  0561          $SNDOPR_G W^SNDO                        ; disable the alternate TTY
                                  056A          FAIL_CHECK SS$_NORMAL                   ; check for success
          00000000'8F      DD     056A                    PUSHL   #SS$_NORMAL
          0B08'CF    01    FB     0570                    CALLS   #1,W^REG_CHECK
                                  0575     ;+
                                  0575     ;
                                  0575     ; ENABLE tests with _G
                                  0575     ;
                                  0575     ;-
                                  0575          NEXT_TEST
                                  0575
                                  0575     STP13:
          0004'CF    0D    D0     0575                    MOVL    #13,W^CURRENT_TC
                     00    DD     057A                    PUSHL   #0
          0AFE'CF    01    FB     057C                    CALLS   #1,W^REG_SAVE
0297'CF   00FFF01F 8F      D0     0581          MOVL    #ALL_OPR,W^OPMSG+OPC$B_MS_ENAB  ; set enable ID mask
029A'CF   FFFFFFFF 8F      D0     058A          MOVL    #-1,W^OPMSG+OPC$L_MS_MASK       ; set all enables
                029E'CF    B4     0593          CLRW    W^OPMSG+OPC$W_MS_OUNIT          ; set unit number
   02A0'CF   0239'CF       90     0597          MOVB    W^OPNAME,W^OPMSG+OPC$T_MS_ONAME ; set name size
   02A1'CF   023A'CF       D0     059E          MOVL    W^OPNAME+1,W^OPMSG+OPC$T_MS_ONAME+1 ;  set name
                                  05A5          $SNDOPR_G W^SNDO                        ; enable the console again
                                  05AE          FAIL_CHECK SS$_NORMAL                   ; check for failure
          00000000'8F      DD     05AE                    PUSHL   #SS$_NORMAL
          0B08'CF    01    FB     05B4                    CALLS   #1,W^REG_CHECK
```

```
                          05B9  .SBTTL  SNDSMB TESTS
                          05B9  ;+
                          05B9  ;
                          05B9  ; $SNDSMB tests
                          05B9  ;
                          05B9  ; The following request types cannot be tested because of the lack of a
                          05B9  ; queueable device in the minimum configuration.
                          05B9  ;
                          05B9  ;       SMR$K_ABORT, SMR$K_ASSIGN, SMR$K_JUSTIFY, SMR$K_ENTER
                          05B9  ;
                          05B9  ; test SMR$K_INITIAL by creating que1 and que2
                          05B9  ;
                          05B9  ;-
                          05B9          NEXT_TEST
                          05B9
                          05B9  STP14:
        0004'CF   0E   D0 05B9                  MOVL    #14,W^CURRENT_TC
                  00   DD 05BE                  PUSHL   #0
        0AFE'CF   01   FB 05C0                  CALLS   #1,W^REG_SAVE
   0171'CF   0046'CF   DE 05C5          MOVAL   W^SNDSMB,W^SERV_NAME                    ; set service name
   00C5'CF   0175'CF   B0 05CC          MOVW    W^MBCHAN,W^SNDS+SNDSMB$_CHAN            ; set the mailbox channel #
        53   0320'CF   DE 05D3          MOVAL   W^SMSG+2,R3                             ; set argument pointer
   63   01A1'CF   0E   28 05D8          MOVC3   #QUENAM1L,W^QUENAM1,(R3)                ; set the queue name
        53   0330'CF   DE 05DE          MOVAL   W^SMSG1,R3                              ; set to proper end of que name
        83     43 8F   90 05E3          MOVB    #SMO$K_DETJOB,(R3)+                     ; set to BATCH
                  63   94 05E7          CLRB    (R3)                                    ; set option terminator
                  00   DD 05E9          PUSHL   #0                                      ; push a dummy parameter
        0AFE'CF   01   FB 05EB          CALLS   #1,W^REG_SAVE                           ; save a reg snapshot
                          05F0          $SNDSMB_S MSGBUF = W^SMSG_DESC,-
                          05F0                  CHAN = W^MBCHAN                         ; try _S INITIAL
                          0600          FAIL_CHECK SS$_NORMAL                           ; check failure
      00000000'8F      DD 0600                  PUSHL   #SS$_NORMAL
        0B08'CF   01   FB 0606                  CALLS   #1,W^REG_CHECK
   56   00040001 8F    D0 060B          MOVL    #JBC$_NORMAL,R6                         ; set expected return status
        0E59'CF   00   FB 0612          CALLS   #0,W^SND_CHECK                          ; check results
        83     4E 8F   90 0617          MOVB    #SMO$K_DISWAP,(R3)+                     ; set to disable swapping
        83     4D 8F   90 061B          MOVB    #SMO$K_INIPRI,(R3)+                     ; set a new job priority
             83   02   90 061F          MOVB    #2,(R3)+                                ; by default of 2
        83     4C 8F   90 0622          MOVB    #SMO$K_JOBLIM,(R3)+                     ; set a job limit of
             83   02   90 0626          MOVB    #2,(R3)+                                ; 2
                  63   94 0629          CLRB    (R3)                                    ; set the terminator
0320'CF   01AF'CF   0E 28 062B          MOVC3   #QUENAM2L,W^QUENAM2,W^SMSG+2            ; set new que name
                  00   DD 0633          PUSHL   #0                                      ; push a dummy parameter
        0AFE'CF   01   FB 0635          CALLS   #1,W^REG_SAVE                           ; save a register snapshot
                          063A          $SNDSMB_G W^SNDS                                ; init the next que
                          0643          FAIL_CHECK SS$_NORMAL                           ; check for failure
      00000000'8F      DD 0643                  PUSHL   #SS$_NORMAL
        0B08'CF   01   FB 0649                  CALLS   #1,W^REG_CHECK
        0E59'CF   00   FB 064E          CALLS   #0,W^SND_CHECK                          ; check the results
                          0653  ;+
                          0653  ;
                          0653  ; test SMR$K_START by starting que1 and que2
                          0653  ;
                          0653  ;-
                          0653          NEXT_TEST
                          0653
                          0653  STP15:
```

```
          0004'CF   0F   D0  0653                     MOVL    #15,W^CURRENT_TC
                    00   DD  0658                     PUSHL   #0
          0AFE'CF   01   FB  065A                     CALLS   #1,W^REG_SAVE
          031E'CF   02   B0  065F          MOVW    #SMR$K_START,W^SMSG           ; set request code
             0330'CF     94  0664          CLRB    W^SMSGT                       ; set for no start options
                             0668          $SNDSMB_G W^SNDS                      ; try _G START
                             0671          FAIL_CHECK SS$_NORMAL                 ; check failure
        00000000'8F      DD  0671                     PUSHL   #SS$_NORMAL
          0B08'CF   01   FB  0677                     CALLS   #1,W^REG_CHECK
          0E59'CF   00   FB  067C          CALLS   #0,W^SND_CHECK                ; check results
 0320'CF  01A1'CF   0E   28  0681          MOVC3   #QUENAM1L,W^QUENAM1,W^SMSG+2  ; set que name
                    00   DD  0689          PUSHL   #0                            ; push a dummy param
          0AFE'CF   01   FB  068B          CALLS   #1,W^REG_SAVE                 ; save a reg snapshot
                             0690          $SNDSMB_G W^SNDS                      ; start the next que
                             0699          FAIL_CHECK SS$_NORMAL                 ; check for failures
        00000000'8F      DD  0699                     PUSHL   #SS$_NORMAL
          0B08'CF   01   FB  069F                     CALLS   #1,W^REG_CHECK
          0E59'CF   00   FB  06A4          CALLS   #0,W^SND_CHECK                ; check the results
                             06A9     ;+
                             06A9     ;
                             06A9     ; test SMR$K_STOP
                             06A9     ;
                             06A9     ;-
                             06A9          NEXT_TEST
                             06A9
                             06A9     STP16:
          0004'CF   10   D0  06A9                     MOVL    #16,W^CURRENT_TC
                    00   DD  06AE                     PUSHL   #0
          0AFE'CF   01   FB  06B0                     CALLS   #1,W^REG_SAVE
          031E'CF   07   B0  06B5          MOVW    #SMR$K_STOP,W^SMSG            ; set request code
                             06BA          $SNDSMB_S MSGBUF = W^SMSG_DESC,-
                             06BA                    CHAN = W^MBCHAN             ; try _S STOP
                             06CA          FAIL_CHECK SS$_NORMAL                 ; check failure
        00000000'8F      DD  06CA                     PUSHL   #SS$_NORMAL
          0B08'CF   01   FB  06D0                     CALLS   #1,W^REG_CHECK
          0E59'CF   00   FB  06D5          CALLS   #0,W^SND_CHECK                ; check results
          031E'CF   02   B0  06DA          MOVW    #SMR$K_START,W^SMSG           ; reset request code
                             06DF          $SNDSMB_G W^SNDS                      ; restart the queue
                             06E8          FAIL_CHECK SS$_NORMAL                 ; check failure
        00000000'8F      DD  06E8                     PUSHL   #SS$_NORMAL
          0B08'CF   01   FB  06EE                     CALLS   #1,W^REG_CHECK
          0E59'CF   00   FB  06F3          CALLS   #0,W^SND_CHECK                ; check results
                             06F8     ;+
                             06F8     ;
                             06F8     ; test SMR$K_PAUSE
                             06F8     ;
                             06F8     ;-
                             06F8          NEXT_TEST
                             06F8
                             06F8     STP17:
          0004'CF   11   D0  06F8                     MOVL    #17,W^CURRENT_TC
                    00   DD  06FD                     PUSHL   #0
          0AFE'CF   01   FB  06FF                     CALLS   #1,W^REG_SAVE
          031E'CF   03   B0  0704          MOVW    #SMR$K_PAUSE,W^SMSG           ; set request code
                             0709          $SNDSMB_S MSGBUF = W^SMSG_DESC,-
                             0709                    CHAN = W^MBCHAN             ; try _S PAUSE
                             0719          FAIL_CHECK SS$_NORMAL                 ; check failure
```

```
 00000000'8F   DD   0719                        PUSHL   #SS$_NORMAL
0B08'CF   01   FB   071F                        CALLS   #1,W^REG_CHECK
0E59'CF   00   FB   0724    802          CALLS   #0,W^SND_CHECK                  ; check results
031E'CF   02   B0   0729    803          MOVW    #SMR$K_START,W^SMSG             ; reset the request code
                    072E    804          $SNDSMB_G W^SNDS                        ; reset the queue state
                    0737    805          FAIL_CHECK SS$_NORMAL                   ; check failure
 00000000'8F   DD   0737                        PUSHL   #SS$_NORMAL
0B08'CF   01   FB   073D                        CALLS   #1,W^REG_CHECK
0E59'CF   00   FB   0742    806          CALLS   #0,W^SND_CHECK                  ; check results
                    0747    807 ;+
                    0747    808 ;
                    0747    809 ; test SMR$K_CREJOB, SMR$K_ADDFIL, SMR$K_CLSJOB
                    0747    810 ;
                    0747    811 ;-
                    0747    812          NEXT_TEST
                    0747
                    0747        STP18:
0004'CF   12   D0   0747                        MOVL    #18,W^CURRENT_TC
          00   DD   074C                        PUSHL   #0
0AFE'CF   01   FB   074E                        CALLS   #1,W^REG_SAVE
                    0753    813          $CREATE FAB = W^FAB                     ; open the command file
                    075E    814          $CONNECT RAB = W^RAB                    ; connect up
                    0769    815          $PUT RAB = W^RAB                        ; write the command file
0478'CF   02BC'CF   DE   0774    816          MOVAL   W^REC1,W^RAB+RAB$L_RBF          ; set rec #1 address
     0472'CF   0B   B0   077B    817          MOVW    #REC1_SIZE,W^RAB+RAB$W_RSZ      ; set rec #1 size
                    0780    818          $PUT RAB = W^RAB                        ; write record #1
0478'CF   02C7'CF   DE   078B    819          MOVAL   W^REC2,W^RAB+RAB$L_RBF          ; set rec #2 address
     0472'CF   29   B0   0792    820          MOVW    #REC2_SIZE,W^RAB+RAB$W_RSZ      ; set rec #2 size
                    0797    821          $PUT RAB = W^RAB                        ; write record #2
                    07A2    822          $DISCONNECT RAB = W^RAB                 ; disconnect
                    07AD    823          $CLOSE FAB = W^FAB                      ; file S05.COM now exists
     02F0'CF   DF   07B8    824          PUSHAL  W^OL1                           ; set option list #1
     02FD'CF   DF   07BC    825          PUSHAL  W^JN1                           ; set job name #1
0CFD'CF   02   FB   07C0    826          CALLS   #2,W^CRE_JOB                    ; put a job in the que
                    07C5    827 ;+
                    07C5    828 ;
                    07C5    829 ; test SMR$K_ALTER on job #1 to release it
                    07C5    830 ;
                    07C5    831 ;-
                    07C5    832          NEXT_TEST
                    07C5
                    07C5        STP19:
0004'CF   13   D0   07C5                        MOVL    #19,W^CURRENT_TC
          00   DD   07CA                        PUSHL   #0
0AFE'CF   01   FB   07CC                        CALLS   #1,W^REG_SAVE
031E'CF   0D   B0   07D1    833          MOVW    #SMR$K_ALTER,W^SMSG             ; set request code
53   0320'CF   DE   07D6    834          MOVAL   W^SMSG+2,R3                     ; set message buffer pointer
63   01A1'CF   0E   28   07DB    835          MOVC3   #QUENAM1L,W^QUENAM1,(R3)        ; set the que name
53   0330'CF   DE   07E1    836          MOVAL   W^SMSG1,R3                      ; set to correct end of que name
83   0CE7'CF   B0   07E6    837          MOVW    W^JOBID,(R3)+                   ; set job ID
     83   22   90   07EB    838          MOVB    #SMO$K_JOBPRI,(R3)+             ; set option code
     83   01   90   07EE    839          MOVB    #1,(R3)+                        ; set the job priority
          63   94   07F1    840          CLRB    (R3)                            ; terminate the option list
          00   DD   07F3    841          PUSHL   #0                              ; push a dummy parameter
0AFE'CF   01   FB   07F5    842          CALLS   #1,W^REG_SAVE                   ; save a register snapshot
                    07FA    843          $SNDSMB_G W^SNDS                        ; try _G ALTER
                    0803    844          FAIL_CHECK SS$_NORMAL                   ; check failure
```

```
                00000000'8F    DD   0803                        PUSHL   #SS$_NORMAL
            0B08'CF    01      FB   0809                        CALLS   #1,W^REG_CHECK
            0E59'CF    00      FB   080E    845                CALLS   #0,W^SND_CHECK                  ; check results
                               0813    846 ;+
                               0813    847 ;
                               0813    848 ; test SMR$K_RELEASE on job #1
                               0813    849 ;
                               0813    850 ;-
                               0813    851                NEXT_TEST
                               0813
                               0813        STP20:
            0004'CF    14      D0   0813                        MOVL    #20,W^CURRENT_TC
                       00      DD   0818                        PUSHL   #0
            0AFE'CF    01      FB   081A                        CALLS   #1,W^REG_SAVE
            031E'CF    0F      B0   081F    852                MOVW    #SMR$K_RELEASE,W^SMSG           ; set request code
    0330'CF    0CE7'CF    B0   0824    853                MOVW    W^JOBID,W^SMSG1                 ; set job ID
               0332'CF    94   082B    854                CLRB    W^SMSG1+2                       ; set no options
                               082F    855                $SNDSMB_G W^SNDS                        ; try _G RELEASE
                               0838    856                FAIL_CHECK SS$_NORMAL                   ; check failure
                00000000'8F    DD   0838                        PUSHL   #SS$_NORMAL
            0B08'CF    01      FB   083E                        CALLS   #1,W^REG_CHECK
            0E59'CF    00      FB   0843    857                CALLS   #0,W^SND_CHECK                  ; check results
                               0848    858 ;+
                               0848    859 ;
                               0848    860 ; test SMR$K_SYNCJOB on job #1
                               0848    861 ;
                               0848    862 ;-
                               0848    863                NEXT_TEST
                               0848
                               0848        STP21:
            0004'CF    15      D0   0848                        MOVL    #21,W^CURRENT_TC
                       00      DD   084D                        PUSHL   #0
            0AFE'CF    01      FB   084F                        CALLS   #1,W^REG_SAVE
            031E'CF    11      B0   0854    864                MOVW    #SMR$K_SYNCJOB,W^SMSG           ; set request code
    0330'CF    0CE7'CF    B0   0859    865                MOVW    W^JOBID,W^SMSG1                 ; set job ID
               0332'CF    94   0860    866                CLRB    W^SMSG1+2                       ; set option list end
                               0864    867                $SNDSMB_G W^SNDS                        ; try _G SYNCJOB
                               086D    868                FAIL_CHECK SS$_NORMAL                   ; check failure
                00000000'8F    DD   086D                        PUSHL   #SS$_NORMAL
            0B08'CF    01      FB   0873                        CALLS   #1,W^REG_CHECK
     56     00000000'8F    D0   0878    869                MOVL    #SS$_NORMAL,R6                  ; set expected status return
            0E59'CF    00      FB   087F    870                CALLS   #0,W^SND_CHECK                  ; check results
     56     00040001   8F    D0   0884    871                MOVL    #JBC$_NORMAL,R6                 ; set expected status return
                               088B    872                $TRNLOG_S LOGNAM = W^YES_DESC,-
                               088B    873                          RSLBUF = W^SYM_DESC,-
                               088B    874                          DSBMSK = #5                   ; look for the group symbol
     50     00000000'8F    D1   08A2    875                CMPL    #SS$_NORMAL,R0                  ; is it there?
                       09      13   08A9    876                BEQL    10$                             ; br if OK
                0164'CF    DF   08AB    877                PUSHAL  W^BAT_IMP_EXC                   ; push error message address
            0B4A'CF    01      FB   08AF    878                CALLS   #1,W^PRINT_FAIL                 ; print the failure
                               08B4    879 10$:
                               08B4    880                $DELLOG_S LOGNAM = W^YES_DESC           ; dump the logical name
                               08C3    881                $ERASE FAB = W^FAB1                     ; delete the log file
                               08CE    882 ;+
                               08CE    883 ;
                               08CE    884 ; test SMR$K_RMVJOB on job #2
                               08CE    885 ;
```

```
                                08CE    886 ;-
                                08CE    887         NEXT_TEST
                                08CE
                                08CE        STP22:
    0004'CF    16    D0         08CE                MOVL    #22,W^CURRENT_TC
               00    DD         08D3                PUSHL   #0
    0AFE'CF    01    FB         08D5                CALLS   #1,W^REG_SAVE
        02F8'CF      DF         08DA    888         PUSHAL  W^OL2                   ; set option list #2
        0307'CF      DF         08DE    889         PUSHAL  W^JN2                   ; set job name #2
    0CFD'CF    02    FB         08E2    890         CALLS   #2,W^CRE_JOB            ; put job #2 in the que
    031E'CF    0C    B0         08E7    891         MOVW    #SMR$K_RMVJOB,W^SMSG    ; set request code
0330'CF 0CE7'CF      B0         08EC    892         MOVW    W^JOBID,W^SMSG1         ; set job ID
        0332'CF      94         08F3    893         CLRB    W^SMSG1+2               ; set no options
               00    DD         08F7    894         PUSHL   #0                      ; push a dummy parameter
    0AFE'CF    01    FB         08F9    895         CALLS   #1,W^REG_SAVE           ; save a reg snapshot
                                08FE    896         $SNDSMB_G W^SNDS                ; try _G and nail the last job
                                0907    897         FAIL_CHECK SS$_NORMAL           ; check failure
     00000000'8F    DD         0907                PUSHL   #SS$_NORMAL
    0B08'CF    01    FB         090D                CALLS   #1,W^REG_CHECK
    0E59'CF    00    FB         0912    898         CALLS   #0,W^SND_CHECK          ; check results
                                0917    899 ;+
                                0917    900 ;
                                0917    901 ; test SMR$K_MERGE on job #3
                                0917    902 ;
                                0917    903 ;-
                                0917    904         NEXT_TEST
                                0917
                                0917        STP23:
    0004'CF    17    D0         0917                MOVL    #23,W^CURRENT_TC
               00    DD         091C                PUSHL   #0
    0AFE'CF    01    FB         091E                CALLS   #1,W^REG_SAVE
        02F0'CF      DF         0923    905         PUSHAL  W^OL1                   ; set option list #3
        0311'CF      DF         0927    906         PUSHAL  W^JN3                   ; set job name #3
    0CFD'CF    02    FB         092B    907         CALLS   #2,W^CRE_JOB            ; put job 3 in the que
    53  031E'CF      DE         0930    908         MOVAL   W^SMSG,R3               ; set address
        83   04      B0         0935    909         MOVW    #SMR$K_MERGE,(R3)+      ; set request code
63  01AF'CF   0E     28         0938    910         MOVC3   #QUENAM2L,W^QUENAM2,(R3) ; set queue name 1
    53  0330'CF      DE         093E    911         MOVAL   W^SMSG1,R3              ; get to correct end of name
63  01A1'CF   0E     28         0943    912         MOVC3   #QUENAM1L,W^QUENAM1,(R3) ; set queue name 2
          02 A3      94         0949    913         CLRB    2(R3)                   ; set no options(*watch que name len
               00    DD         094C    914         PUSHL   #0                      ; push a dummy parameter
    0AFE'CF    01    FB         094E    915         CALLS   #1,W^REG_SAVE           ; save a reg snapshot
                                0953    916         $SNDSMB_G W^SNDS                ; try _G MERGE
                                095C    917         FAIL_CHECK SS$_NORMAL           ; check failure
     00000000'8F    DD         095C                PUSHL   #SS$_NORMAL
    0B08'CF    01    FB         0962                CALLS   #1,W^REG_CHECK
    0E59'CF    00    FB         0967    918         CALLS   #0,W^SND_CHECK          ; check results
    53  031E'CF      DE         096C    919         MOVAL   W^SMSG,R3               ; set message address
        83   0F      B0         0971    920         MOVW    #SMR$K_RELEASE,(R3)+    ; set request code
63  01AF'CF   0E     28         0974    921         MOVC3   #QUENAM2L,W^QUENAM2,(R3) ; set the que name
    53  0330'CF      DE         097A    922         MOVAL   W^SMSG1,R3              ; get to the end of the quenam
    83  0CE7'CF      B0         097F    923         MOVW    W^JOBID,(R3)+           ; set the job ID
               63    94         0984    924         CLRB    (R3)                    ; set no options
               00    DD         0986    925         PUSHL   #0                      ; push a dummy parameter
    0AFE'CF    01    FB         0988    926         CALLS   #1,W^REG_SAVE           ; save a register snapshot
                                098D    927         $SNDSMB_G W^SNDS                ; release the job
                                0996    928         FAIL_CHECK SS$_NORMAL           ; check for failures
```

```
            00000000'8F  DD  0996             PUSHL   #SS$_NORMAL
               0B08'CF  01  FB  099C             CALLS   #1,W^REG_CHECK
               0E59'CF  00  FB  09A1   929     CALLS   #0,W^SND_CHECK              ; check the results
               031E'CF  11  B0  09A6   930     MOVW    #SMR$K_SYNCJOB,W^SMSG       ; set request code
                            09AB   931     $SNDSMB_G W^SNDS                ; sync on the job
                            09B4   932     FAIL_CHECK SS$_NORMAL           ; check for failures
            00000000'8F  DD  09B4             PUSHL   #SS$_NORMAL
               0B08'CF  01  FB  09BA             CALLS   #1,W^REG_CHECK
         56 00000000'8F  D0  09BF   933     MOVL    #SS$_NORMAL,R6              ; set the expected status return
               0E59'CF  00  FB  09C6   934     CALLS   #0,W^SND_CHECK              ; check the results
         56 00040001 8F  D0  09CB   935     MOVL    #JBC$_NORMAL,R6             ; set the expected status return
                            09D2   936     $TRNLOG_S LOGNAM = W^YES_DESC,-
                            09D2   937             RSLBUF = W^SYM_DESC,-
                            09D2   938             DSBMSK = #5             ; look for the group symbol
         50 00000000'8F  D1  09E9   939     CMPL    #SS$_NORMAL,R0              ; is it there?
                        09  13  09F0   940     BEQL    10$                         ; br if OK
                  0164'CF  DF  09F2   941     PUSHAL  W^BAT_IMP_EXC               ; push error message address
               0B4A'CF  01  FB  09F6   942     CALLS   #1,W^PRINT_FAIL             ; print the failure
                            09FB   943 10$:
                            09FB   944     $DELLOG_S LOGNAM = W^YES_DESC   ; dump the logical name
                            0A0A   945 ;+
                            0A0A   946 ;
                            0A0A   947 ; test SMR$K_DELETE
                            0A0A   948 ;
                            0A0A   949 ;-
                            0A0A   950     NEXT_TEST
                            0A0A
                            0A0A       STP24:
               0004'CF  18  D0  0A0A             MOVL    #24,W^CURRENT_TC
                        00  DD  0A0F             PUSHL   #0
               0AFE'CF  01  FB  0A11             CALLS   #1,W^REG_SAVE
               031E'CF  07  B0  0A16   951     MOVW    #SMR$K_STOP,W^SMSG          ; set request code
      0320'CF  01A1'CF  0E  28  0A1B   952     MOVC3   #QUENAM1L,W^QUENAM1,W^SMSG+2 ; set the que name
                  0330'CF  94  0A23   953     CLRB    W^SMSG1                     ; set no options
                        00  DD  0A27   954     PUSHL   #0                          ; push a dummy parameter
               0AFE'CF  01  FB  0A29   955     CALLS   #1,W^REG_SAVE               ; save a reg snapshot
                            0A2E   956     $SNDSMB_G W^SNDS                ; stop que 1
                            0A37   957     FAIL_CHECK SS$_NORMAL           ; check for failures
            00000000'8F  DD  0A37             PUSHL   #SS$_NORMAL
               0B08'CF  01  FB  0A3D             CALLS   #1,W^REG_CHECK
               0E59'CF  00  FB  0A42   958     CALLS   #0,W^SND_CHECK              ; check the results
      0320'CF  01AF'CF  0E  28  0A47   959     MOVC3   #QUENAM2L,W^QUENAM2,W^SMSG+2 ; set the que name
                        00  DD  0A4F   960     PUSHL   #0                          ; push a dummy param
               0AFE'CF  01  FB  0A51   961     CALLS   #1,W^REG_SAVE               ; save a reg snapshot
                            0A56   962     $SNDSMB_G W^SNDS                ; stop que 2
                            0A5F   963     FAIL_CHECK SS$_NORMAL           ; check for failures
            00000000'8F  DD  0A5F             PUSHL   #SS$_NORMAL
               0B08'CF  01  FB  0A65             CALLS   #1,W^REG_CHECK
               0E59'CF  00  FB  0A6A   964     CALLS   #0,W^SND_CHECK              ; check the results
               031E'CF  01  B0  0A6F   965     MOVW    #SMR$K_DELETE,W^SMSG        ; set the request code
                            0A74   966     $SNDSMB_G W^SNDS                ; delete the que
                            0A7D   967     FAIL_CHECK SS$_NORMAL           ; check for failures
            00000000'8F  DD  0A7D             PUSHL   #SS$_NORMAL
               0B08'CF  01  FB  0A83             CALLS   #1,W^REG_CHECK
               0E59'CF  00  FB  0A88   968     CALLS   #0,W^SND_CHECK              ;  check the results
      0320'CF  01A1'CF  0E  28  0A8D   969     MOVC3   #QUENAM1L,W^QUENAM1,W^SMSG+2 ; set the new que name
                        00  DD  0A95   970     PUSHL   #0                          ; push a dummy parameter
```

```
        0AFE'CF   01   FB  0A97   971         CALLS   #1,W^REG_SAVE                    ; save a reg snapshot
                           0A9C   972         $SNDSMB_G W^SNDS                         ; delete the last que
                           0AA5   973         FAIL_CHECK SS$_NORMAL                    ; check for failures
     00000000'8F       DD  0AA5                       PUSHL   #SS$_NORMAL
        0B08'CF   01   FB  0AAB                       CALLS   #1,W^REG_CHECK
        0E59'CF   00   FB  0AB0   974         CALLS   #0,W^SND_CHECK                   ; check the last results
                           0AB5   975         $ERASE FAB=W^FAB                         ; delete the .COM file
                           0AC0   976         $ERASE FAB=W^FAB1                        ; delete the .LOG file
                           0ACB   977         $DASSGN_S CHAN=W^MBCHAN                  ; drop the mailbox
                           0AD7   978         TEST_END
             004C'CF   DD  0AD7                       PUSHL   W^TMD_ADDR
             0048'CF   DD  0ADB                       PUSHL   W^TMN_ADDR
                  02   DD  0ADF                       PUSHL   #2
             0044'CF   DD  0AE1                       PUSHL   W^MOD_MSG_CODE
     00000000'GF  04   FB  0AE5                       CALLS   #$$T1,G^LIB$SIGNAL
0044'CF   01   1C  01  F0  0AEC                       INSV    #1,#STS$V_INHIB_MSG,#1,W^MOD_MSG_CODE
             0044'CF   DD  0AF3                       PUSHL   W^MOD_MSG_CODE
     00000000'GF  01   FB  0AF7                       CALLS   #1,G^SYS$EXIT
```

```
                                 0AFE          .SBTTL REG_SAVE
                                 0AFE  ;++
                                 0AFE  ; FUNCTIONAL DESCRIPTION:
                                 0AFE  ;       Subroutine to save R2-R11 in the register save location.
                                 0AFE  ;
                                 0AFE  ; CALLING SEQUENCE:
                                 0AFE  ;       PUSHL   #0                  ; save a dummy parameter
                                 0AFE  ;       CALLS   #1,W^REG_SAVE       ; save R2-R11
                                 0AFE  ;
                                 0AFE  ; INPUT PARAMETERS:
                                 0AFE  ;       NONE
                                 0AFE  ;
                                 0AFE  ; OUTPUT PARAMETERS:
                                 0AFE  ;       NONE
                                 0AFE  ;
                                 0AFE  ;--
                                 0AFE
                                 0AFE  REG_SAVE:
                           0FFC  0AFE          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
   0008'CF    14 AD    28    28  0B00          MOVC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; save the registers in the program
                             04  0B07          RET
                                 0B08          .SBTTL  REG_CHECK
                                 0B08  ;++
                                 0B08  ; FUNCTIONAL DESCRIPTION:
                                 0B08  ;       Subroutine to test R0 & R2-R11 for proper content after a service
                                 0B08  ;       execution. A snapshot is taken by the REG_SAVE routine at the
                                 0B08  ;       beginning of each step and this routine is executed after the
                                 0B08  ;       services have been executed.
                                 0B08  ;
                                 0B08  ; CALLING SEQUENCE:
                                 0B08  ;       PUSHL   #SS$_XXXXXX        ; push expected R0 contents
                                 0B08  ;       CALLS   #1,W^REG_CHECK     ; execute this routine
                                 0B08  ;
                                 0B08  ; INPUT PARAMETERS:
                                 0B08  ;       expected R0 contents on the stack
                                 0B08  ;
                                 0B08  ; OUTPUT PARAMETERS:
                                 0B08  ;       possible error messages printed using $PUTMSG
                                 0B08  ;
                                 0B08  ;--
                                 0B08
                                 0B08  REG_CHECK:
                           0FFC  0B08          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
             50    04 AC     D1  0B0A          CMPL    4(AP),R0                        ; is this the right fail code?
                       0E    13  0B0E          BEQL    10$                             ; br if yes
                       50    DD  0B10          PUSHL   R0                              ; push received data
                    04 AC    DD  0B12          PUSHL   4(AP)                           ; push expected data
                  0156'CF    DF  0B15          PUSHAL  W^EXP                           ; push the string variable
        0B4A'CF        03    FB  0B19          CALLS   #3,W^PRINT_FAIL                 ; print the error message
                                 0B1E  10$:
   0008'CF    14 AD    28    29  0B1E          CMPC3   #4*10,^X14(FP),W^REG_SAVE_AREA  ; check all but R0
                       22    13  0B25          BEQL    20$                             ; br if O.K.
   56    53    00000008'8F   C3  0B27          SUBL3   #REG_SAVE_AREA,R3,R6            ; calculate the register number
                 56    04    C6  0B2F          DIVL2   #4,R6
         7E      56    02    81  0B32          ADDB3   #^X2,R6,-(SP)           ; set number past R0-R1 and save
                 51    03    CA  0B36          BICL2   #3,R1                   ; backup to register boundrys
                 53    03    CA  0B39          BICL2   #3,R3
```

```
              61  DD  0B3C              PUSHL   (R1)                                    ; push received data
              63  DD  0B3E              PUSHL   (R3)                                    ; push expected data
         00C9'CF  DF  0B40              PUSHAL  W^REG                                   ; set string pntr param.
 0B4A'CF      04  FB  0B44              CALLS   #4,W^PRINT_FAIL                         ; print the error message
                      0B49      20$:
                  04  0B49              RET
                      0B4A              .SBTTL  PRINT_FAIL
                      0B4A      ;++
                      0B4A      ; FUNCTIONAL DESCRIPTION:
                      0B4A      ;       Subroutine to report failures using $PUTMSG
                      0B4A      ;
                      0B4A      ; CALLING SEQUENCE:
                      0B4A      ; Mode  #1      PUSHL EXPECTED  Mode    #2      PUSHL REG_NUMBER
                      0B4A      ;               PUSHL RECEIVED                  PUSHL EXPECTED
                      0B4A      ;               PUSHAL STRING_VAR               PUSHL RECEIVED
                      0B4A      ;               CALLS #3,W^PRINT_FAIL           PUSHAL STRING_VAR
                      0B4A      ;                                               CALLS #4,W^PRINT_FAIL
                      0B4A      ; Mode  #3      PUSHAL STRING_VAR
                      0B4A      ;               CALLS #1,W^PRINT_FAIL
                      0B4A      ;
                      0B4A      ; INPUT PARAMETERS:
                      0B4A      ;       listed above
                      0B4A      ;
                      0B4A      ; OUTPUT PARAMETERS:
                      0B4A      ;       an error message is printed using $PUTMSG
                      0B4A      ;
                      0B4A      ;--
                      0B4A
                      0B4A      PRINT_FAIL:
                003C  0B4A              .WORD   ^M<R2,R3,R4,R5>
                      0B4C              $FAO_S  W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
                      0B6D              $PUTMSG_S W^MSGVEC                              ; print the message
          04  6C  91  0B7E              CMPB    (AP),#4                                 ; is this a register message?
              26  13  0B81              BEQL    10$                                     ; br if yes
          01  6C  91  0B83              CMPB    (AP),#1                                 ; is this just a message?
              48  13  0B86              BEQL    20$                                     ; br if yes
                      0B88              $FAO_S  W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
              40  11  0BA7              BRB     30$                                     ; goto output message
                      0BA9      10$:
                      0BA9              $FAO_S  W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
              19  11  0BCE              BRB     30$                                     ; goto output message
                      0BD0      20$:
 01D7'CF   04 AC  D0  0BD0              MOVL    4(AP),W^MSGVEC1+12                      ; save string address
                      0BD6              $PUTMSG_S W^MSGVEC1                             ; print the message
              11  11  0BE7              BRB     40$                                     ; skip the other message
                      0BE9      30$:
                      0BE9              $PUTMSG_S W^MSGVEC                              ; print the message
                      0BFA      40$:
    0ECF'CF   00  FB  0BFA              CALLS   #0,W^MODE_ID                            ; identify the mode
 004C'CF  002A'CF DE  0BFF              MOVAL   W^TEST_MOD_FAIL,W^TMD_ADDR              ; set failure message address
0044'CF  03   00  02  F0  0C06          INSV    #ERROR,#0,#3,W^MOD_MSG_CODE             ; set severity code
                  04  0C0D              RET
                      0C0E              .SBTTL  READ_CHECK
                      0C0E      ;++
                      0C0E      ; FUNCTIONAL DESCRIPTION:
                      0C0E      ;       Subroutine to read a mailbox and check the status returned
                      0C0E      ;       from the $SNDACC system service.
```

```
                                0C0E  1094 ;
                                0C0E  1095 ; CALLING SEQUENCE:
                                0C0E  1096 ;       CALLS #0,W^READ_CHECK
                                0C0E  1097 ;
                                0C0E  1098 ; INPUT PARAMETERS:
                                0C0E  1099 ;       NONE
                                0C0E  1100 ;
                                0C0E  1101 ; OUTPUT PARAMETERS:
                                0C0E  1102 ;       NONE
                                0C0E  1103 ;
                                0C0E  1104 ;--
                                0C0E  1105
                                0C0E  1106 READ_CHECK:
                         003C   0C0E  1107         .WORD   ^M<R2,R3,R4,R5>
                                0C10  1108         $QIOW_S FUNC=#IO$_READVBLK,-
                                0C10  1109                 CHAN=W^MBCHAN,-
                                CC10  1110                 IOSB=W^STATUSM,-
                                0C10  1111                 P1  =W^MBUF,-
                                0C10  1112                 P2  =#80                        ; read the mail
      0000'8F   017B'CF   B1    0C37  1113         CMPW    W^MBUF,#MSG$_ACCRSP             ; correct response type?
                    13    13    0C3E  1114         BEQL    10$                             ; br if yes
                017B'CF   DD    0C40  1115         PUSHL   W^MBUF                          ; push received
              00000000'8F   DD    0C44  1116         PUSHL   #MSG$_ACCRSP                    ; push expected
                0156'CF   DF    0C4A  1117         PUSHAL  W^EXP                           ; push string variable
             FEF7 CF    03   FB    0C4E  1118         CALLS   #3,W^PRINT_FAIL                 ; print the failure
                                0C53  1119 10$:
  00040000'8F   017F'CF   D1    0C53  1120         CMPL    W^MBUF+4,#4@16!SS$_NORMAL       ; check the results
                    13    13    0C5C  1121         BEQL    20$                             ; br if OK
                017F'CF   DD    0C5E  1122         PUSHL   W^MBUF+4                        ; push received
              00040000'8F   DD    0C62  1123         PUSHL   #4@16!SS$_NORMAL                ; push expected
                0156'CF   DF    0C68  1124         PUSHAL  W^EXP                           ; push the string variable
             FED9 CF    03   FB    0C6C  1125         CALLS   #3,W^PRINT_FAIL                 ; print the failure
                                0C71  1126 20$:
                         04    0C71  1127         RET                                     ; return
```

```
                                                M 9
                     0C72  1129          .SBTTL CRE_JOB
                     0C72  1130 ;++
                     0C72  1131 ; FUNCTIONAL DESCRIPTION:
                     0C72  1132 ;       Routine to enter a job in queue #1
                     0C72  1133 ;
                     0C72  1134 ; CALLING SEQUENCE:
                     0C72  1135 ;       PUSHAL  W^OPTION_LIST           ; counted option list ending with a
                     0C72  1136 ;                                       ; byte of 0
                     0C72  1137 ;       PUSHAL  W^JOB_NAME              ; counted job name ending with a byte of 0
                     0C72  1138 ;       CALLS #0,W^CRE_JOB              ; check buffer
                     0C72  1139 ;
                     0C72  1140 ; INPUT PARAMETERS:
                     0C72  1141 ;       Listed above plus inited NAMBLK to proper command file and
                     0C72  1142 ;       location MBCHAN inited to the mailbox channel.
                     0C72  1143 ;
                     0C72  1144 ; OUTPUT PARAMETERS:
                     0C72  1145 ;       Location JOBID contains the job ID of the created job and
                     0C72  1146 ;       the job is placed in QUE #1
                     0C72  1147 ;
                     0C72  1148 ;--
                     0C72  1149
                     0C72  1150 CREATE:                                 ; create a job message buffer
          00000032'  0C72  1151          .LONG   CR_MSGSIZ
          00000C7A'  0C76  1152          .ADDRESS .+4
                0009  0C7A  1153          .WORD   SMR$K_CREJOB
55 51 5F 54 41 42 5F 50 54 45 55 00'  0C7C  1154          .ASCIC  /UETP_BAT_QUE1/
                  31 45  0C88
                  0D  0C7C
          00000C8C  0C8A  1155          .BLKB   2
                     0C8C  1156 OPTIONS:
          00000CAC  0C8C  1157          .BLKB   32
          00000032  0CAC  1158          CR_MSGSIZ=.-CREATE-8
                     0CAC  1159 ADDFILE:                                ; add a file message buffer
          0000003E'  0CAC  1160          .LONG   AD_MSGSIZ
          00000CB4'  0CB0  1161          .ADDRESS .+4
                000A  0CB4  1162          .WORD   SMR$K_ADDFIL
                     0CB6  1163 DEVICE:
          00000CC6  0CB6  1164          .BLKB   16
                     0CC6  1165 FID:
          00000CCC  0CC6  1166          .BLKB   6
                     0CCC  1167 DID:
          00000CD2  0CCC  1168          .BLKB   6
4D 4F 43 2E 35 30 53 00'  0CD2  1169          .ASCIC  /S05.COM/
                  07  0CD2
          00000CE7  0CDA  1170          .BLKB   13
                     0CE7  1171 JOBID:
                0000  0CE7  1172          .WORD   0
                     0CE9  1173 JOB_NAME:
          00000CF1  0CE9  1174          .BLKB   8
                  00  0CF1  1175          .BYTE   0
          0000003E  0CF2  1176          AD_MSGSIZ=.-ADDFILE-8
                     0CF2  1177 CLOSE:                                  ; close a job message buffer
          00000003'  0CF2  1178          .LONG   CL_MSGSIZ
          00000CFA'  0CF6  1179          .ADDRESS .+4
                000B  0CFA  1180          .WORD   SMR$K_CLSJOB
                  00  0CFC  1181          .BYTE   0
          00000003  0CFD  1182          CL_MSGSIZ=.-CLOSE-8
```

```
                          0CFD  ;
                          0CFD  CRE_JOB:
                    07FC  0CFD          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
       56    08 AC    D0  0CFF          MOVL    8(AP),R6                        ; get the option list pointer
          57    86    9A  0D03          MOVZBL  (R6)+,R7                        ; get the option list size
  FF80 CF    66    57    28  0D06       MOVC3   R7,(R6),W^OPTIONS               ; set the option list
       56    04 AC    D0  0D0C          MOVL    4(AP),R6                        ; get the job name pointer
          57    86    9A  0D10          MOVZBL  (R6)+,R7                        ; get the job name size
  FFD0 CF    66    57    28  0D13       MOVC3   R7,(R6),W^JOB_NAME              ; set the job name
  FFA5 CF  03C4'CF   06    28  0D19     MOVC3   #FIDSIZ,W^NAMBLK+NAM$W_FID,W^FID ; set the FID
  FFA3 CF  03CA'CF   06    28  0D21     MOVC3   #DIDSIZ,W^NAMBLK+NAM$W_DID,W^DID ; set the DID
       56    03B4'CF   9A  0D29          MOVZBL  W^NAMBLK+NAM$T_DVI,R6           ; get device name size
                56    D6  0D2E          INCL    R6                              ; include the count byte
  FF7E CF  03B4'CF   56    28  0D30     MOVC3   R6,W^NAMBLK+NAM$T_DVI,W^DEVICE ; set the device name
    56    00040001 8F    D0  0D38       MOVL    #JBC$_NORMAL,R6                 ; set expected status return
                00    DD  0D3F          PUSHL   #0                              ; set a dummy parameter
     FDB8 CF    01    FB  0D41          CALLS   #1,W^REG_SAVE                   ; save a reg snapshot
                          0D46          $SNDSMB_S MSGBUF = W^CREATE,-
                          0D46                  CHAN = W^MBCHAN                 ; create a job
                          0D56          FAIL_CHECK SS$_NORMAL                   ; check for failure
       00000000'8F    DD  0D56                  PUSHL   #SS$_NORMAL
       FDA7 CF    01    FB  0D5C                CALLS   #1,W^REG_CHECK
       0E59'CF    00    FB  0D61          CALLS   #0,W^SND_CHECK                  ; check the results
  FF7A CF    017D'CF   B0  0D66          MOVW    W^MBUF+2,W^JOBID                ; save the job ID
                          0D6D          $SNDSMB_S MSGBUF = W^ADDFILE,-
                          0D6D                  CHAN = W^MBCHAN                 ; add the file
                          0D7D          FAIL_CHECK SS$_NORMAL                   ; check for failure
       00000000'8F    DD  0D7D                  PUSHL   #SS$_NORMAL
       FD80 CF    01    FB  0D83                CALLS   #1,W^REG_CHECK
       0E59'CF    00    FB  0D88          CALLS   #0,W^SND_CHECK                  ; check the results
                          0D8D          $SNDSMB_S MSGBUF = W^CLOSE,-
                          0D8D                  CHAN = W^MBCHAN                 ; close the job
                          0D9D          FAIL_CHECK SS$_NORMAL                   ; check for failures
       00000000'8F    DD  0D9D                  PUSHL   #SS$_NORMAL
       FD60 CF    01    FB  0DA3                CALLS   #1,W^REG_CHECK
       0E59'CF    00    FB  0DA8          CALLS   #0,W^SND_CHECK                  ; check the results
                04    0DAD          RET                                     ; thats all folks
```

```
                     0DAE  1215          .SBTTL BUF_CHECK
                     0DAE  1216 ;++
                     0DAE  1217 ; FUNCTIONAL DESCRIPTION:
                     0DAE  1218 ;       Routine to check the contents of a buffer against known good
                     0DAE  1219 ;       data.
                     0DAE  1220 ;
                     0DAE  1221 ; CALLING SEQUENCE:
                     0DAE  1222 ;       CALLS #0,W^BUF_CHECK            ; check buffer
                     0DAE  1223 ;
                     0DAE  1224 ; INPUT PARAMETERS:
                     0DAE  1225 ;       R6 = buffer address
                     0DAE  1226 ;       R7 = good data address
                     0DAE  1227 ;       R8 = byte count
                     0DAE  1228 ;
                     0DAE  1229 ; OUTPUT PARAMETERS:
                     0DAE  1230 ;       NONE
                     0DAE  1231 ;
                     0DAE  1232 ;--
                     0DAE  1233
                     0DAE  1234 BCSD:
00000050             0DAE  1235          .LONG   80
00000DB6'            0DB2  1236          .ADDRESS BCBUF
                     0DB6  1237 BCBUF:
00000E06             0DB6  1238          .BLKB   80
                     0E06  1239 BCOSD:
00000000             0E06  1240          .LONG   0
00000DB6'            0E0A  1241          .ADDRESS BCBUF
                     0E0E  1242 PARAM1:
00000E1A             0E0E  1243          .BLKL   3
                     0E1A  1244 ;
                     0E1A  1245 BUF_CHECK:
                07FC 0E1A  1246          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
          59    56   D0   0E1C  1247          MOVL    R6,R9                   ; save a copy of the buffer address
    66    67    58   29   0E1F  1248          CMPC3   R8,(R7),(R6)            ; check the buffer
                33   13   0E23  1249          BEQL    10$                     ; br if good
      5A    E6  AF   DE   0E25  1250          MOVAL   B^PARAM1,R10            ; set parameter pointer
          8A    63   9A   0E29  1251          MOVZBL  (R3),(R10)+             ; save bad data
          8A    61   9A   0E2C  1252          MOVZBL  (R1),(R10)+             ; save good data
  8A      53    59   C3   0E2F  1253          SUBL3   R9,R3,(R10)+            ; save byte offset
      5A    D8  AF   DE   0E33  1254          MOVAL   B^PARAM1,R10            ; reset address pointer
                         0E37  1255          $FAO_S  CTRSTR = W^CS6,-
                         0E37  1256                  OUTLEN = W^BCOSD,-
                         0E37  1257                  OUTBUF = W^BCSD,-
                         0E37  1258                      P1 = (R10)+,-
                         0E37  1259                      P2 = (R10)+,-
                         0E37  1260                      P3 = (R10)+         ; make the string
          B3  AF   DF   0E50  1261          PUSHAL  B^BCOSD                 ; push the string variable
FCF2  CF    01   FB   0E53  1262          CALLS   #1,W^PRINT_FAIL         ; print the failure
                         0E58  1263 10$:
                    04   0E58  1264          RET                             ; return
```

```
                                0E59  1266          .SBTTL  SND_CHECK
                                0E59  1267 ;++
                                0E59  1268 ; FUNCTIONAL DESCRIPTION:
                                0E59  1269 ;       Routine to check the contents of a buffer against known good
                                0E59  1270 ;       data.
                                0E59  1271 ;
                                0E59  1272 ; CALLING SEQUENCE:
                                0E59  1273 ;       CALLS #0,W^SND_CHECK            ; check buffer
                                0E59  1274 ;
                                0E59  1275 ; INPUT PARAMETERS:
                                0E59  1276 ;       R6 = expected status code
                                0E59  1277 ;
                                0E59  1278 ; OUTPUT PARAMETERS:
                                0E59  1279 ;       NONE
                                0E59  1280 ;
                                0E59  1281 ;--
                                0E59  1282
                                0E59  1283 SND_CHECK:
                         003C   0E59  1284          .WORD   ^M<R2,R3,R4,R5>
                                0E5B  1285          $QIOW_S FUNC=#IO$_READVBLK,-
                                0E5B  1286                  CHAN=W^MBCHAN,-
                                0E5B  1287                  IOSB=W^STATUSM,-
                                0E5B  1288                  P1  =W^MBUF,-
                                0E5B  1289                  P2  =#80                ; read the mail
00000046'8F    0171'CF    D1    0E82  1290          CMPL    W^SERV_NAME,#SNDSMB     ; SNDSMB or SNDOPR
                     0D   13    0E8B  1291          BEQL    10$                     ; br if SNDSMB
               56    017D'CF    B1    0E8D  1292          CMPW    W^MBUF+OPC$W_MS_STATUS,R6 ; correct response type?
                     1C   13    0E92  1293          BEQL    30$                     ; br if yes
                     017D'CF    DD    0E94  1294          PUSHL   W^MBUF+OPC$W_MS_STATUS  ; push received
                     0B   11    0E98  1295          BRB     20$                     ; br to common code
                                0E9A  1296 10$:
               56    017F'CF    D1    0E9A  1297          CMPL    W^MBUF+4,R6             ; correct status return?
                     0F   13    0E9F  1298          BEQL    30$                     ; br if yes
                     017F'CF    DD    0EA1  1299          PUSHL   W^MBUF+4                ; push received
                                0EA5  1300 20$:
                        56      DD    0EA5  1301          PUSHL   R6                      ; push expected
                     0156'CF    DF    0EA7  1302          PUSHAL  W^EXP                   ; push string variable
FC9A CF                 03      FB    0EAB  1303          CALLS   #3,W^PRINT_FAIL         ; print the failure
                                0EB0  1304 30$:
                        04      0EB0  1305          RET
                                0EB1  1306          .SBTTL GENREQ
                                0EB1  1307 ;++
                                0EB1  1308 ; FUNCTIONAL DESCRIPTION:
                                0EB1  1309 ; routine to generate a pending request for $SNDOPR
                                0EB1  1310 ;
                                0EB1  1311 ; CALLING SEQUENCE:
                                0EB1  1312 ;       CALLS   #0,W^GENREQ     ; generate a pending request
                                0EB1  1313 ;
                                0EB1  1314 ; INPUT PARAMETERS:
                                0EB1  1315 ;       NONE
                                0EB1  1316 ;
                                0EB1  1317 ; OUTPUT PARAMETERS:
                                0EB1  1318 ;       NONE
                                0EB1  1319 ;
                                0EB1  1320 ;--
                                0EB1  1321
                                0EB1  1322 GENREQ:
```

```
                                                D 10
                    OFFC  0EB1  1323          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                          0EB3  1324          $SNDOPR_S MSGBUF = W^OP_MSG1,-
                          0EB3  1325                    CHAN = W^MBCHAN     ; generate a request
                          0EC3  1326          FAIL_CHECK SS$_NORMAL        ; check for failure
                                              PUSHL   #SS$_NORMAL
 00000000'8F    DD  0EC3
FC3A CF    01    FB  0EC9                    CALLS   #1,W^REG_CHECK
                    04  0ECE  1327          RET
```

```
                0ECF  1329          .SBTTL  MODE_ID
                0ECF  1330 ;++
                0ECF  1331 ; FUNCTIONAL DESCRIPTION:
                0ECF  1332 ;        Subroutine to identify the mode that an exit handler is in.
                0ECF  1333 ;
                0ECF  1334 ; CALLING SEQUENCE:
                0ECF  1335 ;        CALLS   #0,W^MODE_ID
                0ECF  1336 ;
                0ECF  1337 ; INPUT PARAMETERS:
                0ECF  1338 ;        MODE contains an address pointing to an ascii string desc.
                0ECF  1339 ;        of the current CPU mode.
                0ECF  1340 ;
                0ECF  1341 ; OUTPUT PARAMETERS:
                0ECF  1342 ;        NONE
                0ECF  1343 ;
                0ECF  1344 ;--
                0ECF  1345
                0ECF  1346 MODE_ID:
        003C    0ECF  1347          .WORD   ^M<R2,R3,R4,R5>
                0ED1  1348          $FAO_S  W^CS5,W^MESSAGEL,W^MSGL,MODE ; format the error message
                0EEA  1349          $PUTMSG_S W^MSGVEC                   ; print the mode message
          04    0EFB  1350          RET
```

```
                                F 10
                0EFC  1352 MOD_MSG_PRINT:
                0EFC  1353 ;
                0EFC  1354 ;  ***************************************************************************
                0EFC  1355 ;  *                                                                         *
                0EFC  1356 ;  *  PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES                 *
                0EFC  1357 ;  *      (USING THE PUTMSG MACRO).                                          *
                0EFC  1358 ;  *                                                                         *
                0EFC  1359 ;  ***************************************************************************
                0EFC  1360 ;
                0EFC  1361          PUTMSG  <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> ; PRINT MSG
          05    0F17  1362          RSB                                         ; ... AND RETURN TO CALLER
                0F18  1363 ;
                0F18  1364 CHMRTN:
                0F18  1365 ;  ***************************************************************************
                0F18  1366 ;  *                                                                         *
                0F18  1367 ;  *     CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER              *
                0F18  1368 ;  *     A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED                 *
                0F18  1369 ;  *     BY THE MODE MACRO ('TO' OPTION).  IT MERELY DOES                    *
                0F18  1370 ;  *     A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS                   *
                0F18  1371 ;  *     THE EFFECT OF RETURNING TO THE END OF THE MODE                      *
                0F18  1372 ;  *     MACRO EXPANSION.                                                    *
                0F18  1373 ;  *                                                                         *
                0F18  1374 ;  ***************************************************************************
                0F18  1375 ;
          0000  0F18  1376          .WORD   0                                   ; ENTRY MASK
00000059'FF     17  0F1A  1377          JMP     @CHM_CONT                       ; RETURN TO MODE MACRO IN NEW MODE
                0F20  1378 ;
                0F20  1379 ;  *     RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
                0F20  1380 ;
                0F20  1381          .END    SATSSS05
```

```
$$.TAB                  = 00000494 R     03
$$.TABEND               = 000004E4 R     03
$$.TMP                  = 00000000
$$.TMP1                 = 00000001
$$.TMP2                 = 000000CF
$$ARGS                  = 00000002
$$T1                    = 00000004
$$T2                    = 00000004
A                       = 00000064
ACC$K_BATTRM            = 00000002
ACC$K_DISAACC           = 00000004
ACC$K_DISASEL           = 00000006
ACC$K_ENABACC           = 00000003
ACC$K_ENABSEL           = 00000005
ACC$K_INSMESG           = 00000001
ACC$K_INSMSG            = 00000011
ACC$K_INTTRM            = 00000003
ACC$K_LOGTRM            = 00000004
ACC$K_PRCTRM            = 00000001
ACC$K_PRTJOB            = 00000010
ACC_DESC                  0000023E R     03
ACC_MSG                   000001E3 R     03
ACC_MSG1                  00000235 R     03
ADDFILE                   00000CAC R     04
AD_MSGSIZ               = 0000003E
ALL_OPR                 = 00FFF01F
BAT_IMP_EXC               00000164 R     02
BCBOF                     00000DB6 R     04
BCOSD                     00000E06 R     04
BCSD                      00000DAE R     04
BUF                       000000E7 R     03
BUF_CHECK                 00000E1A R     04
BUF_SIZE                = 00000064
CHMRTN                    00000F18 R     04
CHM_CONT                  00000059 R     03
CLOSE                     00000CF2 R     04
CL_MSGSIZ               = 00000003
COM_FILE                  00000297 R     02
COM_FIL_SIZ             = 00000007
CR                      = 0000000D
CREATE                    00000C72 R     04
CRE_JOB                   00000CFD R     04
CR_MSGSIZ               = 00000032
CST                       0000004D R     02
CS2                       0000007F R     02
CS3                       000000AC R     02
CS5                       000000DF R     02
CS6                       000000F4 R     02
CTL$GL_PHD                ******** X     04
CURRENT_TC                00000004 R     03
DEVICE                    00000CB6 R     04
DIB$W_UNIT              = 0000000C
DID                       00000CCC R     04
DIDSIZ                  = 00000006
EMB$C_SS                = 00000027
EMB$W_HD_ENTRY          = 00000004
ERROR                   = 00000002
EXP                       00000156 R     02
FAB                       00000400 R     03
FAB$C_BID               = 00000003
FAB$C_BLN               = 00000050
FAB$C_SEQ               = 00000000
FAB$C_VAR               = 00000002
FAB$L_ALQ               = 00000010
FAB$L_FOP               = 00000004
FAB$V_CHAN_MODE         = 00000002
FAB$V_CR                = 00000001
FAB$V_FILE_MODE         = 00000004
FAB$V_LNM_MODE          = 00000000
FAB$V_PUT               = 00000000
FAB$W_GBC               = 00000048
FAB1                      00000494 R     03
FID                       00000CC6 R     04
FIDSIZ                  = 00000006
FILE_NAME                 0000027C R     02
FILE_NAME1                00000290 R     02
FILNAMSIZ               = 00000014
GENREQ                    00000EB1 R     04
GET1                      00000221 R     04
GET2                      00000299 R     04
INFO                    = 00000003
IO$M_NOW                  ******** X     04
IO$_READVBLK              ******** X     03
JBC$_NORMAL             = 00040001
JN1                       000002FD R     02
JN2                       00000307 R     02
JN3                       00000311 R     02
JOBID                     00000CE7 R     04
JOB_NAME                  00000CE9 R     04
LF                      = 0000000A
LIB$SIGNAL                ******** X     04
MBCHAN                    00000175 R     03
MBNAM                     00000140 R     02
MBUF                      0000017B R     03
MESSAGEL                  00000169 R     03
MODE                      00000177 R     03
MODE_ID                   00000ECF R     04
MOD_MSG_CODE              00000044 R     03
MOD_MSG_PRINT             00000EFC R     04
MSG$_ACCRSP               ******** X     04
MSG1_                   = 00000036
MSG1_SIZE               = 00000009
MSGL                      000000DF R     03
MSGVEC                    000001BD R     02
MSGVEC1                   000001CB R     03
MSG_LEN                 = 00000080
MSG_SIZE                = 00000052
NAM$B_ESS               = 0000000A
NAM$B_NOP               = 00000008
NAM$B_RSS               = 00000002
NAM$C_BID               = 00000002
NAM$C_BLN               = 00000060
NAM$L_ESA               = 0000000C
NAM$L_RSA               = 00000004
```

```
NAM$T_DVI                       =  00000014
NAM$W_DID                       =  0000002A
NAM$W_FID                       =  00000024
NAMBLK                             000003A0 R      03
NAME_SIZE                       =  00000008
OL1                                000002F0 R      02
OL1S                            =  00000007
OL2                                000002F8 R      02
OL2S                            =  00000004
OPC$B_MS_ENAB                   =  00000001
OPC$L_MS_MASK                   =  00000004
OPC$L_MS_OTEXT                  =  0000001A
OPC$L_MS_RPLYID                 =  00000004
OPC$L_MS_RQSTID                 =  00000004
OPC$L_MS_TEXT                   =  00000008
OPC$M_NM_CENTRL                 =  00000001
OPC$M_NM_DEVICE                 =  00000010
OPC$M_NM_DISKS                  =  00000008
OPC$M_NM_OPER1                  =  00001000
OPC$M_NM_OPER10                 =  00200000
OPC$M_NM_OPER11                 =  00400000
OPC$M_NM_OPER12                 =  00800000
OPC$M_NM_OPER2                  =  00002000
OPC$M_NM_OPER3                  =  00004000
OPC$M_NM_OPER4                  =  00008000
OPC$M_NM_OPER5                  =  00010000
OPC$M_NM_OPER6                  =  00020000
OPC$M_NM_OPER7                  =  00040000
OPC$M_NM_OPER8                  =  00080000
OPC$M_NM_OPER9                  =  00100000
OPC$M_NM_PRINT                  =  00000002
OPC$M_NM_TAPES                  =  00000004
OPC$T_MS_ONAME                  =  0000000A
OPC$W_MS_OUNIT                  =  00000008
OPC$W_MS_STATUS                 =  00000002
OPC$_RQSTABORT                  =  0005801C
OPC$_RQSTCAN                    =  00058084
OPC$_RQSTCMPLTE                 =  00058029
OPC$_RQSTPEND                   =  00058021
OPC$_RQ_CANCEL                  =  00000005
OPC$_RQ_REPLY                   =  00000004
OPC$_RQ_RQST                    =  00000003
OPC$_RQ_TERME                   =  00000001
OPMSG                              00000296 R      03
OPMSG_DESC                         0000028E R      03
OPNAME                             00000239 R      02
OPTIONS                            00000C8C R      04
OPTYPE                             00000246 R      03
OP_MESG                            0000024E R      02
OP_MESG_LEN                     =  0000002E
OP_MSG1                            0000023E R      02
PARAM1                             00000E0E R      04
PHD$Q_PRIVMSK                   =  00000000
PRINT_FAIL                         00000B4A R      04
PRIVMASK                           00000051 R      03
PRIV_ARGS                       =  00000002
PRV$V_BUGCHK                    =  00000017
PRV$V_OPER                      =  00000012
PRVPRT                             00000050 R      03
QIO                                00000069 R      03
QIO$_ASTADR                     =  00000014
QIO$_ASTPRM                     =  00000018
QIO$_CHAN                       =  00000008
QIO$_EFN                        =  00000004
QIO$_FUNC                       =  0000000C
QIO$_IOSB                       =  00000010
QIO$_NARGS                      =  0000000C
QIO$_P1                         =  0000001C
QIO$_P2                         =  00000020
QIO$_P3                         =  00000024
QIO$_P4                         =  00000028
QIO$_P5                         =  0000002C
QIO$_P6                         =  00000030
QUENAM1                            000001A1 R      02
QUENAM1L                        =  0000000E
QUENAM2                            000001AF R      02
QUENAM2L                        =  0000000E
RAB                                00000450 R      03
RAB$B_RAC                       =  0000001E
RAB$C_BID                       =  00000001
RAB$C_BLN                       =  00000044
RAB$C_SEQ                       =  00000000
RAB$L_CTX                       =  00000018
RAB$L_RBF                       =  00000028
RAB$L_ROP                       =  00000004
RAB$W_RSZ                       =  00000022
READ_CHECK                         00000C0E R      04
RECO_SIZE                       =  00000025
REC1                               000002BC R      02
REC1_SIZE                       =  0000000B
REC2                               000002C7 R      02
REC2_SIZE                       =  00000029
REG                                000000C9 R      03
REGNUM                             000000DB R      03
REG_CHECK                          00000B08 R      04
REG_SAVE                           00000AFE R      04
REG_SAVE_AREA                      00000008 R      03
RETADR                             0000005D R      03
SATSSS05                           00000000 RG     04
SERV_NAME                          00000171 R      03
SEVERE                          =  00000004
SHR$K_SHRDEF                    =  00000001
SHR$_TEXT                       =  00001130
SMO$K_DETJOB                    =  00000043
SMO$K_DISWAP                    =  0000004E
SMO$K_HOLD                      =  00000021
SMO$K_INIPRI                    =  0000004D
SMO$K_JOBLIM                    =  0000004C
SMO$K_JOBPRI                    =  00000022
SMO$K_PARAMS                    =  00000026
SMR$K_ADDFIL                    =  0000000A
SMR$K_ALTER                     =  0000000D
SMR$K_CLSJOB                    =  0000000B
SMR$K_CREJOB                    =  00000009
```

SMR$K_DELETE	=	00000001			STP5		0000019F	R	04
SMR$K_INITIAL	=	00000000			STP6		0000025D	R	04
SMR$K_MERGE	=	00000004			STP7		000002C6	R	04
SMR$K_PAUSE	=	00000003			STP8		00000318	R	04
SMR$K_RELEASE	=	0000000F			STP9		00000366	R	04
SMR$K_RMVJOB	=	0000000C			STS$V_INHIB_MSG	=	0000001C		
SMR$K_START	=	00000002			SUCCESS	=	00000001		
SMR$K_STOP	=	00000007			SYM		0000038A	R	03
SMR$K_SYNCJOB	=	00000011			SYM_DESC		00000195	R	02
SMSG		0000031E	R	03	SYM_NAME		00000192	R	02
SMSG1		00000330	R	03	SYS$CLOSE		********	GX	04
SMSG_DESC		00000316	R	03	SYS$CMKRNL		********	GX	04
SMSG_LEN	=	0000006C			SYS$CONNECT		********	GX	04
SNDA		0000009D	R	03	SYS$CREATE		********	GX	04
SNDACC		00000031	R	02	SYS$CREMBX		********	GX	04
SNDACC$_CHAN	=	00000008			SYS$DASSGN		********	GX	04
SNDACC$_MSGBUF	=	00000004			SYS$DELLOG		********	GX	04
SNDACC$_NARGS	=	00000002			SYS$DERLMB		********	X	04
SNDE		000000A9	R	03	SYS$DISCONNECT		********	GX	04
SNDERR		00000038	R	02	SYS$ERASE		********	GX	04
SNDERR$_MSGBUF	=	00000004			SYS$EXIT		********	GX	04
SNDERR$_NARGS	=	00000001			SYS$FAO		********	X	04
SNDO		000000B1	R	03	SYS$GETCHN		********	GX	04
SNDOPR		0000003F	R	02	SYS$HIBER		********	GX	04
SNDOPR$_CHAN	=	00000008			SYS$PUT		********	GX	04
SNDOPR$_MSGBUF	=	00000004			SYS$PUTMSG		********	GX	04
SNDOPR$_NARGS	=	00000002			SYS$QIO		********	GX	04
SNDS		000000BD	R	03	SYS$QIOW		********	GX	04
SNDSMB		00000046	R	02	SYS$SETPRN		********	GX	04
SNDSMB$_CHAN	=	00000008			SYS$SETPRV		********	GX	04
SNDSMB$_MSGBUF	=	00000004			SYS$SNDACC		********	GX	04
SNDSMB$_NARGS	=	00000002			SYS$SNDERR		********	GX	04
SND_CHECK		00000E59	R	04	SYS$SNDOPR		********	GX	04
SS$_NORMAL		********	X	04	SYS$SNDSMB		********	GX	04
STATUS		000001DB	R	03	SYS$TRNLOG		********	GX	04
STATUSM		00000065	R	03	SYS$WAITFR		********	GX	04
STEP	=	00000018			SYS$WAKE		********	GX	04
STP0		0000003D	R	04	TEST_ERROR		000001CD	R	02
STP1		000000DB	R	04	TEST_MOD_BEGIN		00000019	R	02
STP10		000003DC	R	04	TEST_MOD_FAIL		0000002A	R	02
STP11		00000450	R	04	TEST_MOD_NAME		00000000	R	02
STP12		00000543	R	04	TEST_MOD_NAME_D		00000009	R	02
STP13		00000575	R	04	TEST_MOD_SUCC		0000001F	R	02
STP14		000005B9	R	04	TMD_ADDR		0000004C	R	03
STP15		00000653	R	04	TMN_ADDR		00000048	R	03
STP16		000006A9	R	04	TPID		00000000	R	03
STP17		000006F8	R	04	TTNAM		0000014F	R	02
STP18		00000747	R	04	TTUNIT		00000154	R	02
STP19		000007C5	R	04	UETP$_SATSMS	=	007480D9		
STP2		0000010C	R	04	UETP$_TEXT	=	00741133		
STP20		00000813	R	04	UM		00000134	R	02
STP21		00000848	R	04	WARNING	=	00000000		
STP22		000008CE	R	04	YES		0000019D	R	02
STP23		00000917	R	04	YES_DESC		0000018A	R	02
STP24		00000A0A	R	04					
STP3		0000013D	R	04					
STP4		0000016E	R	04					

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
RODATA	0000031B (795.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
RWDATA	000004E4 (1252.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
SATSSS05	00000F20 (3872.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	37	00:00:00.08	00:00:00.43
Command processing	152	00:00:00.73	00:00:03.13
Pass 1	589	00:00:24.73	00:00:38.18
Symbol table sort	0	00:00:02.46	00:00:02.81
Pass 2	297	00:00:05.77	00:00:07.90
Symbol table output	39	00:00:00.27	00:00:00.28
Psect synopsis output	3	00:00:00.02	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1119	00:00:34.06	00:00:52.76

The working set limit was 2000 pages.
144825 bytes (283 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1646 non-local and 19 local symbols.
1381 source lines were read in Pass 1, producing 40 object records in Pass 2.
98 pages of virtual memory were used to define 88 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SHRLIB]UETP.MLB;1	12
_255DUA28:[SYS.OBJ]LIB.MLB;1	6
_255DUA28:[SYSLIB]STARLET.MLB;2	67
TOTALS (all libraries)	85

2203 GETS were required to define 85 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:SATSSS05/OBJ=OBJ$:SATSSS05 MSRC$:SATSSS05/UPDATE=(ENH$:SATSSS05)+EXECML$/LIB+SHRLIB$:UETP/LIB
```

SATSSS08
LIS

SATSSS05
LIS

SATSSS22
LIS

SATSSS07
LIS

SATSSS01
LIS