

UUU	UUU	EEEEEEEEEEEEEEEE	TTTTTTTTTTTTTTTT	PPPPPPPPPPPP	
UUU	UUU	EEEEEEEEEEEEEEEE	TTTTTTTTTTTTTTTT	PPPPPPPPPPPP	
UUU	UUU	EEEEEEEEEEEEEEEE	TTTTTTTTTTTTTTTT	PPPPPPPPPPPP	
UUU	UUU	EEE	TTT	PPP	PPP
UUU	UUU	EEE	TTT	PPP	PPP
UUU	UUU	EEE	TTT	PPP	PPP
UUU	UUU	EEE	TTT	PPP	PPP
UUU	UUU	EEE	TTT	PPP	PPP
UUU	UUU	EEE	TTT	PPP	PPP
UUU	UUU	EEE	TTT	PPP	PPP
UUU	UUU	EEEEEEEEEEEEEEEE	TTT	PPPPPPPPPPPP	
UUU	UUU	EEEEEEEEEEEEEEEE	TTT	PPPPPPPPPPPP	
UUU	UUU	EEEEEEEEEEEEEEEE	TTT	PPPPPPPPPPPP	
UUU	UUU	EEE	TTT	PPP	
UUU	UUU	EEE	TTT	PPP	
UUU	UUU	EEE	TTT	PPP	
UUU	UUU	EEE	TTT	PPP	
UUU	UUU	EEE	TTT	PPP	
UUU	UUU	EEE	TTT	PPP	
UUU	UUU	EEE	TTT	PPP	
UUUUUUUUUUUUUUUU	UUUUUUUUUUUUUUUU	EEEEEEEEEEEEEEEE	TTT	PPP	
UUUUUUUUUUUUUUUU	UUUUUUUUUUUUUUUU	EEEEEEEEEEEEEEEE	TTT	PPP	
UUUUUUUUUUUUUUUU	UUUUUUUUUUUUUUUU	EEEEEEEEEEEEEEEE	TTT	PPP	

_s
Va
--
000
000
000
7F1
7F1
7F1
7F1
7F1
7F1
7F1
7F1

```

SSSSSSSS  AAAAAA  TTTTTTTTTT  SSSSSSSS  SSSSSSSS  FFFFFFFFFF  11  77777777
SSSSSSSS  AAAAAA  TTTTTTTTTT  SSSSSSSS  SSSSSSSS  FFFFFFFFFF  11  77777777
SS        AA      AA      TT        SS        SS        FF        1111      77
SS        AA      AA      TT        SS        SS        FF        1111      77
SS        AA      AA      TT        SS        SS        FF        11        77
SS        AA      AA      TT        SS        SS        FF        11        77
SSSSSSS   AA      AA      TT        SSSSSS   SSSSSS   FFFFFFFF  11        77
SSSSSSS   AA      AA      TT        SSSSSS   SSSSSS   FFFFFFFF  11        77
          SS   AAAAAAAAAA  TT        SS        SS        FF        11        77
          SS   AAAAAAAAAA  TT        SS        SS        FF        11        77
          SS   AA      AA      TT        SS        SS        FF        11        77
          SS   AA      AA      TT        SS        SS        FF        11        77
SSSSSSSS  AA      AA      TT        SSSSSSSS  SSSSSSSS  FF        111111  77
SSSSSSSS  AA      AA      TT        SSSSSSSS  SSSSSSSS  FF        111111  77

```

```

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

```



```
0000 1 .TITLE SATSSF17 - SATS SYSTEM SERVICE TESTS (FAILING 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 SATSSF17 module tests the execution of the following
0000 33 : VMS system services, invoked in such a way as to expect failing
0000 34 : status codes:
0000 35 : $INPUT
0000 36 : $OUTPUT
0000 37 : $QIO
0000 38 : $QIOW
0000 39
0000 40
0000 41 : ENVIRONMENT: User mode image; needs CMKRNL privilege,
0000 42 : dynamically acquires other privileges, as needed.
0000 43
0000 44 : AUTHOR: Larry D. Jones, CREATION DATE: OCTOBER, 1979
0000 45
0000 46 : MODIFIED BY:
0000 47
0000 48 : V03-001 LDJ0001 Larry D. Jones, 17-Sep-1980
0000 49 : Modified to conform to new build command procedures.
0000 50 : **
0000 51 : --
```

```
0000 53 .SBTTL DECLARATIONS
0000 54 :
0000 55 : MACRO LIBRARY CALLS
0000 56 :
0000 57 $PRVDEF ; privilege definitions
0000 58 $UETPDEF ; UETP message definitions
0000 59 $SHR MESSAGES UETP,116,<<TEXT,INFO>> ; UETP$ TEXT definition
0000 60 $PHDDEF ; process header definitions
0000 61 $PCBDEF ; PCB definitions
0000 62 $SSDEF ; SS definitions
0000 63 $STSDEF ; STS definitions
0000 64 :
0000 65 : Equated symbols
0000 66 :
00000000 0000 67 WARNING = 0 ; warning severity value for msgs
00000001 0000 68 SUCCESS = 1 ; success
00000002 0000 69 ERROR = 2 ; error
00000003 0000 70 INFO = 3 ; information
00000004 0000 71 SEVERE = 4 ; fatal
00000001 0000 72 PRVHND_SXV40 = 1 ; page 0 address for SETEXV
0000 73
```

	0000	75	.SBTTL	OWN STORAGE	
	0000	76	.PSECT	RODATA, RD, NOWRT, NOEXE, LONG	
	0000	77	:		
37 31 46 53 53 54 41 53	0000	78	TEST_MOD_NAME:		
08	0000	79	.ASCIC	/SATSSF17/	; needed for SATSMS message
46 53 53 54 41 53	0000011'010E0000'	80	TEST_MOD_NAME D:		
37 31	0009	81	.ASCID	/SATSSF17/	; module name
	0017				
6E 75 67 65 62	0019	82	TEST_MOD_BEGIN:		
05	0019	83	.ASCIC	/begun/	
	0019				
6C 75 66 73 73 65 63 63 75 73	001F	84	TEST_MOD_SUCC:		
0A	001F	85	.ASCIC	/successful/	
	001F				
64 65 6C 69 61 66	002A	86	TEST_MOD_FAIL:		
06	002A	87	.ASCIC	/failed/	
	002A				
54 55 50 4E 49	0031	88	INPUT:		
05	0031	89	.ASCIC	/INPUT/	
	0031				
54 55 50 54 55 4F	0037	90	OUTPUT:		
06	0037	91	.ASCIC	/OUTPUT/	
	0037				
4F 49 51	003E	92	QIO:		
03	003E	93	.ASCIC	/QIO/	
	003E				
57 4F 49 51	0042	94	QIOW:		
04	0042	95	.ASCIC	/QIOW/	
	0042				
00000000'00000000'	0047	96	INADR:		
	0047	97	.LONG	NOACCESS, NOACCESS	; page address of noaccess psect
00000000'	004F	98	PROT:		
	004F	99	.LONG	PRT\$C_NA	; protection code for no access psect
	0053	100	PRVHND_SXV41:		; read only access location
	0053	101	CS1:		
21 20 74 73 65 54	0053	102	.ASCID	\Test !AC service name !AC step !UL failed.\	
6E 20 65 63 69 76 72 65 73 20 43 41	0061				
70 65 74 73 20 43 41 21 20 65 6D 61	006D				
2E 64 65 6C 69 61 66 20 4C 55 21 20	0079				
	0085	103	CS2:		
74 63 65 70 78 45	0085	104	.ASCID	\Expected !AS = !XL received !AS = !XL\	
4C 58 21 20 3D 20 53 41 21 20 64 65	0093				
41 21 20 64 65 76 69 65 63 65 72 20	009F				
4C 58 21 20 3D 20 53	00AB				
	00B2	105	CS3:		
74 63 65 70 78 45	00B2	106	.ASCID	\Expected !AS!UB = !XL received !AS!UB = !XL\	
20 3D 20 42 55 21 53 41 21 20 64 65	00C0				
64 65 76 69 65 63 65 72 20 4C 58 21	00CC				
58 21 20 3D 20 42 55 21 53 41 21 20	00D8				
4C	00E4				
	00E5	107	EXP:		
73 75 74 61 74 73	00E5	108	.ASCID	\status\	
	00F3	109	MBNAM:		
54 54	00F3	110	.ASCID	\TT\	

```

00FD 112 ;
00FD 113 ;
00000000 114 .SBTTL R/W PSECT
          115 .PSECT RWDATA, RD, WRT, NOEXE, LONG
0000 116 ;
00000000 0000 117 PID: ; PID for this process
          0004 118 CURRENT_TC:
00000000 0004 119 .LONG 0 ; ptr to current test case
          0008 120 .ALIGN LONG
00000044 0008 121 REG_SAVE_AREA:
          0044 122 .BLKL 15 ; register save area
007480D9 0044 123 MOD_MSG_CODE:
          0048 124 .LONG UETP$_SATSMS ; test module message code for putmsg
00000000' 0048 125 TMN_ADDR:
          004C 126 .ADDRESS TEST_MOD_NAME
00000019' 004C 127 TMD_ADDR:
          0050 128 .ADDRESS TEST_MOD_BEGIN
          0050 129 PRVPRT:
          00 0050 130 .BYTE 0 ; protection return byte for SETPRT
00000000 00000000 0051 131 PRIVMASK: ; priv. mask
          0059 132 .QUAD 0
00000000 0059 133 CHM_CONT:
          005D 134 .LONG 0 ; change mode continue address
00000065 005D 135 RETADR:
          0065 136 .BLKL 2 ; returned address's from SETPRT
          0065 137 INP:
          0088 138 $INPUT 0,0,0 ; INPUT parameter list
          0088 139 OUT:
          0088 140 $OUTPUT 0,0,0 ; OUTPUT parameter list
          00AD 141 QIOP:
          00AD 142 $QIO -1,0,IO$_READVBLK,0,0,0,MBNAM,0 ; QIO parameter list
          00E1 143 QIOWP:
          00E1 144 $QIOW -1,0,IO$_READVBLK,0,0,0,MBNAM,0 ; QIOW parameter list
          0115 145
74 73 69 67 65 72 0000011D'010E0000' 0115 146 REG:
          52 20 72 65 0123 147 .ASCID \register R\
          0127 148 REGNUM:
00000000 0127 149 .LONG 0 ; register number
          012B 150 MSGL:
00000050 012B 151 .LONG 80 ; buffer desc.
00000133' 012F 152 .ADDRESS BUF
          0133 153 BUF:
00000183 0133 154 .BLKB 80
          0183 155 MESSAGEL:
00000000 0183 156 .LONG 0 ; message desc.
00000133' 0187 157 .ADDRESS BUF
          018B 158 SERV_NAME:
00000000 018B 159 .LONG 0 ; service name pointer
          018F 160 MBCHAN:
          0000 018F 161 .WORD 0 ; channel location

```

```

00000000 163 .PSECT SATS ACCVIO_1,RD,WRT,NOEXE,PAGE
00000200 0000 164 EMPTY: .BLKB 512 ; reserve a page of space
0200 165 :
0200 166 : +
0200 167 : *****
0200 168 : *
0200 169 : * THE ORDER OF STATEMENTS IN THIS PSECT IS CRITICAL. *
0200 170 : * DO NOT RE-ARRANGE THE VARIABLES. CONSULT SATS *
0200 171 : * FUNCTIONAL SPECIFICATION FOR A DESCRIPTION OF THE USE *
0200 172 : * OF THE EMPTY PSECT (AND ITS COMPANION PSECT, NOACCESS). *
0200 173 : *
0200 174 : *****
0200 175 : -
0200 176 :
000001FF 0200 177 PRVHND_SXV42 = . - 1 ; prvhd arg for SETEXV (last byte in the page)
000001F3 0200 178 = . - 13 ; allow room for string descriptor
01F3 179 ; type AAAAA_SSSX5 go here:
00000006 01F3 180 .LONG 6 ; string length (will cross psect boundary)
000001FB' 01F7 181 .ADDRESS .+4 ; string address
01FB 182 ; type AAAAA_SSSX3 go here:
000001FC 01FB 183 .BLKB 1 ; low-order byte of string length
01FC 184 ; type AAAAA_SSSX2 go here:
00000200 01FC 185 .BLKL 1 ; string length
0200 186 :
0200 187 :
0200 188 :
0200 189 :
00000000 190 .PSECT SATS ACCVIO_2,RD,WRT,NOEXE,PAGE
00000200 0000 191 NOACCESS: .BLKB 512 ; reserve a page of space
00000000 0200 192 = . - 512 ; return loc ctr to beginning of psect
00000000' 0000 193 .ADDRESS EMPTY ; address of accessible string
00000000' 0004 194 .ADDRESS EMPTY/^X100 ; address of accessible string
0008 195 : +
0008 196 : *** NOTE -- DO NOT CHANGE LOCATION OR SEQUENCE OF ABOVE STATEMENTS!
0008 197 : *** THIS PSECT (NOACCESS) MUST APPEAR IN MEMORY IMMEDIATELY
0008 198 : *** FOLLOWING THE EMPTY PSECT. PSECT NAMES AND OPTIONS WILL BE
0008 199 : *** CHOSEN TO FORCE THE DESIRED PSECT ORDERING.
0008 200 : -
0008 201 :
0008 202 :
0008 203 :
0008 204 :

```



```

00000000 206      .PSECT SATSSF17, RD, WRT, EXE, LONG
0000      207      .SBTTL SATSSF17
0000      208      :++
0000      209      : FUNCTIONAL DESCRIPTION:
0000      210      :
0000      211      :     After performing some initial housekeeping, such as
0000      212      :     printing the module begin message and acquiring needed privileges,
0000      213      :     the system services are tested in each of their failure conditions.
0000      214      :     Detected failures are identified and an error message is printed
0000      215      :     on the terminal. Upon completion of the test a success or fail
0000      216      :     message is printed on the terminal.
0000      217      :
0000      218      : CALLING SEQUENCE:
0000      219      :
0000      220      :     $ RUN SATSSF17 ... (DCL COMMAND)
0000      221      :
0000      222      : INPUT PARAMETERS:
0000      223      :
0000      224      :     none
0000      225      :
0000      226      : IMPLICIT INPUTS:
0000      227      :
0000      228      :     none
0000      229      :
0000      230      : OUTPUT PARAMETERS:
0000      231      :
0000      232      :     none
0000      233      :
0000      234      : IMPLICIT OUTPUTS:
0000      235      :
0000      236      :     Messages to SYS$OUTPUT are the only output from SATSSF17.
0000      237      :     They are of the form:
0000      238      :
0000      239      :     %UETP-S-SATSMS, TEST MODULE SATSSF17 BEGUN ... (BEGIN MSG)
0000      240      :     %UETP-S-SATSMS, TEST MODULE SATSSF17 SUCCESSFUL ... (END MSG)
0000      241      :     %UETP-E-SATSMS, TEST MODULE SATSSF17 FAILED ... (END MSG)
0000      242      :     %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
0000      243      :
0000      244      : COMPLETION CODES:
0000      245      :
0000      246      :     The SATSSF17 routine terminates with a $EXIT to the
0000      247      :     operating system with a status code defined by UETP$_SATSMS.
0000      248      :
0000      249      : SIDE EFFECTS:
0000      250      :
0000      251      :     none
0000      252      :
0000      253      : --
0000      254      :
0000      255      :
0000      256      :
0000      257      : TEST_START SATSSF17           ; let the test begin

```

```

0000 0000
0004'CF D4 0002
00 DD 0006
0000'CF DF 0008
00000000'GF 02 FB 000C
00000000'GF 00 FB 0013
00009'CF 7F 001A
00000000'GF 01 FB 001E
0A24 30 0025
004C'CF 001F'CF DE 0028
0044'CF 03 00 01 FO 002F
00 DD 0036
0953'CF 01 FB 0038
003D
003D
003D
0056

```

```

STP0:
258
259
260

```

```

.ENTRY SATSSF17,0
CLRL W^CURRENT_TC
PUSHL #0
PUSHAL W^TPID
CALLS #2,G^SYSSWAKE
CALLS #0,G^SYSSHIBER
PUSHAQ W^TEST_MOD_NAME_D
CALLS #1,G^SYSSSETPRN
BSBW W^MOD_MSG_PRINT
MOVAL W^TEST_MOD_SUCC,W^TMD_ADDR
INSV #SUCCESS,#0,#3,W^MOD_MSG_CODE
PUSHL #0
CALLS #1,W^REG_SAVE

```

```

$SETPRT_S INADR=W^INADR, RETADR=W^RETADR, -
PROT=W^PROT, PRVPRT=W^PRVPRT ; set noaccess psect
; ... for no user access

```

```

0056 262 .SBTTL INPUT TESTS
0056 263 :+
0056 264 :
0056 265 : $INPUT tests
0056 266 :
0056 267 : test for an EFN of -1
0056 268 :
0056 269 :-
018B'CF 0031'CF DE 0056 270 MOVAL W^INPUT,W^SERV_NAME ; set the service name
005D 271 $CREMBX_S LOGNAM=W^MBNAM,-
005D 272 CHAN=W^MBCHAN ; get a legal channel number
0074 273 $INPUT CHAN = W^MBCHAN,-
0074 274 BUFFER= W^MBNAM,-
0074 275 LENGTH= #0,-
0074 276 EFN = #-1 ; try EFN = -1
0097 277 FAIL_CHECK SSS_ILLEFC ; check failure
0097 277 PUSHL #SS$ ILLEFC
095D'CF 01 FB 009D 277 CALLS #1,W^REG_CHECK
00A2 278 :+
00A2 279 :
00A2 280 : test for an EFN of 500
00A2 281 :
00A2 282 :-
00A2 283 NEXT_TEST
00A2 283
0004'CF 01 DO 00A2 STP1:
0953'CF 01 FB 00A7 00A7 MOVL #1,W^CURRENT_TC
00A9 00A9 PUSHL #0
00AE 284 $INPUT CHAN = W^MBCHAN,-
00AE 285 BUFFER= W^MBNAM,-
00AE 286 LENGTH= #0,-
00AE 287 EFN= #500 ; try illegal EFN = 500
00D1 288 FAIL_CHECK SSS_ILLEFC ; check failure
00D1 288 PUSHL #SS$ ILLEFC
095D'CF 01 FB 00D7 288 CALLS #1,W^REG_CHECK
00DC 289 :+
00DC 290 :
00DC 291 : test for an EFN of 123 without an associated cluster
00DC 292 :
00DC 293 :-
00DC 294 NEXT_TEST
00DC 294
0004'CF 02 DO 00DC STP2:
0953'CF 01 FB 00E1 00E1 MOVL #2,W^CURRENT_TC
00E3 00E3 PUSHL #0
00E8 295 $INPUT CHAN = W^MBCHAN,-
00E8 296 BUFFER= W^MBNAM,-
00E8 297 LENGTH= #0,-
00E8 298 EFN = #123 ; try EFN = 123
010B 299 FAIL_CHECK SSS_UNASEFC ; check failure
010B 299 PUSHL #SS$ UNASEFC
095D'CF 01 FB 0111 299 CALLS #1,W^REG_CHECK
0116 300 :+
0116 301 :
0116 302 : test unaccessible IOSB parameter = page 0 access

```

```

0116 303 :-
0116 304 :-
0116 305 :-      NEXT_TEST
0116
0116 STP3:
0004'CF 03 DO 0116      MOVL    #3,W^CURRENT_TC
00      00 DD 0116      PUSHL   #0
0953'CF 01 FB 0118      CALLS   #1,W^REG_SAVE
011D      $INPUT  CHAN = W^MBCHAN,-
0122 306      IOSB = W^PRVHND_SXV40,-
0122 307      LENGTH= #0,-
0122 308      BUFFER= W^MBNAM           ; try page 0 access
0122 309      FAIL_CHECK SSS_ACCVIO      ; check failure
0143 310
095D'CF 0C DD 0143
01      01 FB 0145      PUSHL   #SS$ ACCVIO
014A      CALLS   #1,W^REG_CHECK
014A 311 :+
014A 312 :-
014A 313 :-      test unaccessable IOSB parameter = read-only PSECT
014A 314 :-
014A 315 :-
014A 316 :-      NEXT_TEST
014A
014A STP4:
0004'CF 04 DO 014A      MOVL    #4,W^CURRENT_TC
00      00 DD 014A      PUSHL   #0
0953'CF 01 FB 0151      CALLS   #1,W^REG_SAVE
0156 317      $INPUT  CHAN = W^MBCHAN,-
0156 318      IOSB = W^PRVHND_SXV41,-
0156 319      LENGTH= #0,-
0156 320      BUFFER= W^MBNAM           ; try read-only PSECT
0177 321      FAIL_CHECK SSS_ACCVIO      ; check failure
0177
095D'CF 0C DD 0177
01      01 FB 0179      PUSHL   #SS$ ACCVIO
017E      CALLS   #1,W^REG_CHECK
017E 322 :+
017E 323 :-
017E 324 :-      test unaccessable IOSB parameter = noaccess protection
017E 325 :-
017E 326 :-
017E 327 :-      NEXT_TEST
017E
017E STP5:
0004'CF 05 DO 017E      MOVL    #5,W^CURRENT_TC
00      00 DD 0183      PUSHL   #0
0953'CF 01 FB 0185      CALLS   #1,W^REG_SAVE
018A 328      $INPUT  CHAN = W^MBCHAN,-
018A 329      IOSB = W^PRVHND_SXV42,-
018A 330      LENGTH= #0,-
018A 331      BUFFER= W^MBNAM           ; try noaccess BUFFER param.
01AB 332      FAIL_CHECK SSS_ACCVIO      ; check failure
01AB
095D'CF 0C DD 01AB
01      01 FB 01AD      PUSHL   #SS$ ACCVIO
01B2 333 :+
01B2 334 :-
01B2 335 :-      test non-existent channel number
01B2 336 :-
01B2 337 :-
01B2 338 :-      NEXT_TEST

```

```

0004'CF 06 DO 01B2
0000'CF 00 DD 01B2
0953'CF 01 FB 01B7
01BE 339
01CA 340
01CA 341
01CA 342
01E9 343
095D'CF 24 DD 01E9
01 FB 01EB
01F0 344 :+
01F0 345 : test illegal channel number
01F0 346 :-
01F0 347 :-
01F0 348 :-
01F0 349
NEXT_TEST
0004'CF 07 DO 01F0
0000'CF 00 DD 01F5
0953'CF 01 FB 01F7
018F'CF D4 01FC 350
0200 351
0200 352
0200 353
021F 354
0000013C 8F DD 021F
095D'CF 01 FB 0225

STP6:
        MOVL    #6,W^CURRENT_TC
        PUSHL   #0
        CALLS   #1,W^REG_SAVE
        $DASSGN_S CHAN = W^MBCHAN          ; deassign the channel
        $INPUT  CHAN = W^MBCHAN,-
        BUFFER= W^MBNAM,-
        LENGTH= #0                        ; try illegal channel
        FAIL_CHECK SSS_NOPRIV             ; check the failure
        PUSHL   #SS$_NOPRIV
        CALLS   #1,W^REG_CHECK

STP7:
        MOVL    #7,W^CURRENT_TC
        PUSHL   #0
        CALLS   #1,W^REG_SAVE
        CLRL    W^MBCHAN                   ; make an illegal channel number
        $INPUT  CHAN = W^MBCHAN,-
        BUFFER= W^MBNAM,-
        LENGTH= #0                        ; try illegal channel number
        FAIL_CHECK SSS_IVCHAN             ; check failure
        PUSHL   #SS$_IVCHAN
        CALLS   #1,W^REG_CHECK

```

```

022A 356 .SBTTL OUTPUT TESTS
022A 357 :+
022A 358 :
022A 359 : $OUTPUT tests
022A 360 :
022A 361 : test for an EFN of -1
022A 362 :
022A 363 :-
022A 364 NEXT_TEST
022A
022A STP8:
0004'CF 08 DO 022A MOVL #8,W^CURRENT_TC
DD 022F PUSHL #0
0953'CF 01 FB 0231 CALLS #1,W^REG_SAVE
018B'CF 0037'CF DE 0236 MOVAL W^OUTPUT,W^SERV_NAME ; set the service name
023D 366 $CREMBX_S LOGNAM=W^MBNAM,- ; get a legal channel number
023D 367 CHAN=W^MBCHAN
0254 368 $OUTPUT CHAN = W^MBCHAN,-
0254 369 BUFFER = W^MBNAM,-
0254 370 LENGTH = #0,-
0254 371 EFN = #-1 ; try EFN = -1
0279 372 FAIL_CHECK SSS_ILLEFC ; check failure
000000EC 8F DD 0279 PUSHL #SS$ILLEFC
095D'CF 01 FB 027F CALLS #1,W^REG_CHECK
0284 373 :+
0284 374 :
0284 375 : test for an EFN of 500
0284 376 :
0284 377 :-
0284 378 NEXT_TEST
0284
0284 STP9:
0004'CF 09 DO 0284 MOVL #9,W^CURRENT_TC
DD 0289 PUSHL #0
0953'CF 01 FB 028B CALLS #1,W^REG_SAVE
0290 379 $OUTPUT CHAN = W^MBCHAN,-
0290 380 BUFFER = W^MBNAM,-
0290 381 LENGTH = #0,-
0290 382 EFN = #500 ; try illegal EFN = 500
02B5 383 FAIL_CHECK SSS_ILLEFC ; check failure
000000EC 8F DD 02B5 PUSHL #SS$ILLEFC
095D'CF 01 FB 02BB CALLS #1,W^REG_CHECK
02C0 384 :+
02C0 385 :
02C0 386 : test for an EFN of 123 without an associated cluster
02C0 387 :
02C0 388 :-
02C0 389 NEXT_TEST
02C0
02C0 STP10:
0004'CF 0A DO 02C0 MOVL #10,W^CURRENT_TC
DD 02C5 PUSHL #0
0953'CF 01 FB 02C7 CALLS #1,W^REG_SAVE
02CC 390 $OUTPUT CHAN = W^MBCHAN,-
02CC 391 BUFFER = W^MBNAM,-
02CC 392 LENGTH = #0,-
02CC 393 EFN = #123 ; try EFN =123

```

```

00000234 8F DD 02F1 394 FAIL_CHECK SSS_UNASEFC ; check failure
095D'CF 01 FB 02F1 PUSHL #SS$ UNASEFC
02F7 CALLS #1,W*REG_CHECK
02FC 395 :+
02FC 396 : test unaccessable IOSB parameter = page 0 access
02FC 397 :
02FC 398 :-
02FC 399 :-
02FC 400 NEXT_TEST
02FC
0004'CF 0B DO 02FC STP11:
0953'CF 00 DD 0301 MOVL #11,W^CURRENT_TC
01 FB 0303 PUSHL #0
0308 401 $OUTPUT CHAN = W^MBCHAR,-
0308 402 IOSB = W^PRVHND_SXV40,-
0308 403 LENGTH = #0,-
0308 404 BUFFER = W^MBNAM ; try page 0 access
032B 405 FAIL_CHECK SSS_ACCVIO ; check failure
095D'CF 0C DD 032B PUSHL #SS$ ACCVIO
01 FB 032D CALLS #1,W*REG_CHECK
0332 406 :+
0332 407 : test unaccessable IOSB parameter = read-only PSECT
0332 408 :
0332 409 :-
0332 410 :-
0332 411 NEXT_TEST
0332
0004'CF 0C DO 0332 STP12:
0953'CF 00 DD 0337 MOVL #12,W^CURRENT_TC
01 FB 0339 PUSHL #0
033E 412 $OUTPUT CHAN = W^MBCHAR,-
033E 413 IOSB = W^PRVHND_SXV41,-
033E 414 LENGTH = #0,-
033E 415 BUFFER = W^MBNAM ; try read-only PSECT
0361 416 FAIL_CHECK SSS_ACCVIO ; check failure
095D'CF 0C DD 0361 PUSHL #SS$ ACCVIO
01 FB 0363 CALLS #1,W*REG_CHECK
0368 417 :+
0368 418 : test unaccessable IOSB parameter = noaccess protection
0368 419 :
0368 420 :-
0368 421 :-
0368 422 NEXT_TEST
0368
0004'CF 0D DO 0368 STP13:
0953'CF 00 DD 036D MOVL #13,W^CURRENT_TC
01 FB 036F PUSHL #0
0374 423 $OUTPUT CHAN = W^MBCHAR,-
0374 424 IOSB = W^PRVHND_SXV42,-
0374 425 LENGTH = #0,-
0374 426 BUFFER = W^MBNAM ; try noaccess BUFFER param.
0397 427 FAIL_CHECK SSS_ACCVIO ; check failure
095D'CF 0C DD 0397 PUSHL #SS$ ACCVIO
01 FB 0399 CALLS #1,W*REG_CHECK

```

```

039E 428 :+
039E 429 :
039E 430 : test non-existent channel number
039E 431 :
039E 432 :-
039E 433 :
                                NEXT_TEST
039E
039E STP14:
0004'CF 0E  D0 039E          MOVL   #14,W^CURRENT_TC
00          00  DD 03A3          PUSHL  #0
0953'CF 01  FB 03A5          CALLS  #1,W^REG_SAVE
                                ; deassign the channel
03AA 434  $DASSGN S CHAN = W^MBCHAN
03B6 435  $OUTPUT  CHAN = W^MBCHAN,-
03B6 436  BUFFER = W^MBNAM,-
03B6 437  LENGTH = #0
                                ; try illegal channel
03D7 438  FAIL_CHECK SSS$NOPRIV
                                ; check the failure
095D'CF 24  DD 03D7          PUSHL  #SS$ NOPRIV
01          01  FB 03D9          CALLS  #1,W^REG_CHECK
03DE 439 :+
03DE 440 :
03DE 441 : test illegal channel number
03DE 442 :
03DE 443 :-
03DE 444 :
                                NEXT_TEST
03DE
03DE STP15:
0004'CF 0F  D0 03DE          MOVL   #15,W^CURRENT_TC
00          00  DD 03E3          PUSHL  #0
0953'CF 01  FB 03E5          CALLS  #1,W^REG_SAVE
018F'CF  D4  D4 03EA 445  CLRL   W^MBCHAN
                                ; make an illegal channel number
03EE 446  $OUTPUT  CHAN = W^MBCHAN,-
03EE 447  BUFFER = W^MBNAM,-
03EE 448  LENGTH = #0
                                ; try illegal channel number
040F 449  FAIL_CHECK SSS$IVCHAN
                                ; check failure
0000013C 8F DD 040F          PUSHL  #SS$ IVCHAN
095D'CF 01  FB 0415          CALLS  #1,W^REG_CHECK

```

SA

SY
ST
SU
SY
SY
SY
SY
SY
SY
SY
SY
TE
TE
TE
TE
TM
TP
UE
UE
WA

PS
--
\$A
RO
RW
SA
SA

Ph
--
In
Co
Pa
Sy
Pa
Sy
Ps
Cr
As

Th
12


```

041A 451 .SBTTL QIO TESTS
041A 452 :+
041A 453 :
041A 454 : $QIO tests
041A 455 : test for an EFN of 0
041A 456 :
041A 457 :-
018B'CF 003E'CF DE 041A 458 MOVAL W^QIO,W^SERV NAME ; set service name
0421 459 $CREMBX_S LOGNAM=W^MBNAM,-
0421 460 CHAN=W^MBCHAN ; get a legal channel number
00B5'CF 018F'CF 3C 0438 461 MOVZWL W^MBCHAN,W^QIOP+QIOS_CHAN ; set the channel number
043F 462 $QIO_S CHAN=W^MBCHAN,-
043F 463 FUNC=#IOS READVBLK,-
043F 464 P1=W^MBNAM,-
043F 465 P2=#0,-
043F 466 EFN=#-1 ; try EFN=0
0462 467 FAIL_CHECK SSS_ILLEFC ; check failure
000000EC 8F DD 0462 468 PUSHL #SSS_ILLEFC
095D'CF 01 FB 0468 469 CALLS #1,W^REG_CHECK
046D 468 $QIO_G W^QIOP ; try G
0476 469 FAIL_CHECK SSS_ILLEFC ; check failure
000000EC 8F DD 0476 470 :+
095D'CF 01 FB 047C 471 :
0481 472 : test for an EFN of 500
0481 473 :
0481 474 :-
0481 475 NEXT_TEST
0481
0004'CF 10 DO 0481 STP16:
0953'CF 00 DD 0486 MOVL #16,W^CURRENT_TC
0953'CF 01 FB 0488 PUSHL #0
048D 476 $QIO_S CHAN=W^MBCHAN,-
048D 477 FUNC=#IOS READVBLK,-
048D 478 P1=W^MBNAM,-
048D 479 P2=#0,-
048D 480 EFN=#500 ; try EFN=500
04B0 481 FAIL_CHECK SSS_ILLEFC ; check failure
04B0 482 PUSHL #SSS_ILLEFC
000000EC 8F DD 04B0 483 CALLS #1,W^REG_CHECK
095D'CF 01 FB 04B6 482 MOVL #500,W^QIOP+QIOS_EFN ; set illegal EFN
000001F4 8F DO 04BB 483 $QIO_G W^QIOP ; try G
04C4 484 FAIL_CHECK SSS_ILLEFC ; check failure
04CD 484 PUSHL #SSS_ILLEFC
000000EC 8F DD 04CD 485 :+
095D'CF 01 FB 04D3 486 :
04D8 487 : test for an EFN of 123 without an associated cluster
04D8 488 :
04D8 489 :-
04D8 490 NEXT_TEST
04D8
0004'CF 11 DO 04D8 STP17:
00 00 DD 04DD MOVL #17,W^CURRENT_TC
PUSHL #0

```

```

0953'CF 01 FB 04DF 491 CALLS #1,W^REG_SAVE
04E4 491 $QIO_S CHAN=W^MBCHAN,-
04E4 492 FUNC=#IOS$ READVBLK,-
04E4 493 P1=W^MBNAM,-
04E4 494 P2=#0,-
04E4 495 EFN=#123 ; try S
0507 496 FAIL_CHECK SSS_UNASEFC ; check failure
00000234 8F DD 0507 PUSHL #SS$ UNASEFC
095D'CF 01 FB 050D CALLS #1,W^REG_CHECK
COB1'CF 0000007B 8F DO 0512 497 MOVL #123,W^QIOP+QIOS_EFN ; set illegal EFN
051B 498 $QIO_G W^QIOP ; try G
0524 499 FAIL_CHECK SSS_UNASEFC ; check failure
00000234 8F DD 0524 PUSHL #SS$ UNASEFC
095D'CF 01 FB 052A CALLS #1,W^REG_CHECK
00B1'CF D4 052F 500 CLRL W^QIOP+QIOS_EFN ; clean up illegal EFN
0533 501 :+
0533 502 :-
0533 503 : test unaccessible IOSB = page 0 access
0533 504 :-
0533 505 :-
0533 506 NEXT_TEST
0533 STP18:
0004'CF 12 DO 0533 MOVL #18,W^CURRENT_TC
00 DD 0538 PUSHL #0
0953'CF 01 FB 053A CALLS #1,W^REG_SAVE
053F 507 $QIO_S CHAN=W^MBCHAN,-
053F 508 FUNC=#IOS$ READVBLK,-
053F 509 P1=W^MBNAM,-
053F 510 P2=#0,-
053F 511 IOSB=W^PRVHND_SXV40 ; try S
0560 512 FAIL_CHECK SSS_ACCVIO ; check failure
00 DD 0560 PUSHL #SS$ ACCVIO
095D'CF 01 FB 0562 CALLS #1,W^REG_CHECK
00BD'CF 0001'CF DE 0567 513 MOVAL W^PRVHND_SXV40,W^QIOP+QIOS IOSB ; set illegal address
056E 514 $QIO_G W^QIOP ; try G
0577 515 FAIL_CHECK SSS_ACCVIO ; check the failure
00 DD 0577 PUSHL #SS$ ACCVIO
095D'CF 01 FB 0579 CALLS #1,W^REG_CHECK
057E 516 :+
057E 517 :-
057E 518 : test unaccessible IOSB = read-only PSECT
057E 519 :-
057E 520 :-
057E 521 NEXT_TEST
057E STP19:
0004'CF 13 DO 057E MOVL #19,W^CURRENT_TC
00 DD 0583 PUSHL #0
0953'CF 01 FB 0585 CALLS #1,W^REG_SAVE
058A 522 $QIO_S CHAN=W^MBCHAN,-
058A 523 FUNC=#IOS$ READVBLK,-
058A 524 P1=W^MBNAM,-
058A 525 P2=#0,-
058A 526 IOSB=W^PRVHND_SXV41 ; try S
05AB 527 FAIL_CHECK SSS_ACCVIO ; check failure
00 DD 05AB PUSHL #SS$ ACCVIO

```

```

095D'CF 01 FB 05AD          CALLS #1,W^REG_CHECK
00BD'CF 0053'CF DE 05B2 528   MOVAL W^PRVHND_SXV41,W^QIOP+QIOS_IOSB ; set IOSB adr
                                05B9 529   $QIO_G W^QIOP ; try G
                                05C2 530   FAIL_CHECK SSS_ACCVIO ; check failure
                                DD 05C2          PUSHL #SS$ ACCVIO
095D'CF 01 FB 05C4          CALLS #1,W^REG_CHECK
                                05C9 531 :+
                                05C9 532 :
                                05C9 533 : test noaccess protection in IOSB
                                05C9 534 :
                                05C9 535 :-
                                05C9 536   NEXT_TEST
                                05C9          STP20:
0004'CF 14 DO 05C9          MOVL #20,W^CURRENT_TC
0953'CF 00 DC 05CE          PUSHL #0
                                01 FB 05D0          CALLS #1,W^REG_SAVE
                                05D5 537   $QIO_S CHAN=W^MBCHAN,-
                                05D5 538   FUNC=#IOS$ READVBLK,-
                                05D5 539   P1=W^MBNAM,-
                                05D5 540   P2=#0,-
                                05D5 541   IOSB=W^PRVHND_SXV42 ; try S
                                05F6 542   FAIL_CHECK SSS_ACCVIO ; check failure
                                DD 05F6          PUSHL #SS$ ACCVIO
095D'CF 01 FB 05F8          CALLS #1,W^REG_CHECK
                                05FD 543 :+
                                05FD 544 :
                                05FD 545 : test non-existent channel number
                                05FD 546 :
                                05FD 547 :-
                                05FD 548   NEXT_TEST
                                05FD          STP21:
0004'CF 15 DO 05FD          MOVL #21,W^CURRENT_TC
0953'CF 00 DD 0602          PUSHL #0
                                01 FB 0604          CALLS #1,W^REG_SAVE
                                0609 549   $DASSGN_S CHAN=W^MBCHAN ; release the channel
                                0615 550   $QIO_S CHAN=W^MBCHAN,-
                                0615 551   FUNC=#IOS$ READVBLK,-
                                0615 552   P1=W^MBNAM,- ; try S
                                0615 553   P2=#0
                                0634 554   FAIL_CHECK SSS_NOPRIV ; check failure
                                DD 0634          PUSHL #SS$ NOPRIV
095D'CF 01 FB 0636          CALLS #1,W^REG_CHECK
                                0638 555   $QIO_G W^QIOP ; try G
                                0644 556   FAIL_CHECK SSS_NOPRIV ; check failure
                                DD 0644          PUSHL #SS$ NOPRIV
095D'CF 01 FB 0646          CALLS #1,W^REG_CHECK
                                064B 557 :+
                                064B 558 :
                                064B 559 : test illegal channel number
                                064B 560 :
                                064B 561 :-
                                064B 562   NEXT_TEST
                                064B          STP22:
0004'CF 16 DO 064B          MOVL #22,W^CURRENT_TC

```

0953'CF	00	DD	0650			PUSHL #0	
	01	FB	0652			CALLS #1,W^REG_SAVE	
018F'CF		D4	0657	563	CLRL	W^MBCHAN	; set illegal channel number
			0658	564	\$QIO_S	CHAN=W^MBCHAN,-	
			0658	565		FUNC=#IOS\$ READVBLK,-	
			0658	566		P1=W^MBNAM,-	
			0658	567		P2=#0	
			067A	568	FAIL_CHECK	SS\$ _IVCHAN	; check failure
0000013C	8F	DD	067A			PUSHL #SS\$ _IVCHAN	
095D'CF	01	FB	0680			CALLS #1,W^REG_CHECK	
00B5'CF		D4	0685	569	CLRL	W^QIOP+QIOS\$_CHAN	; set illegal channel number
			0689	570	\$QIO_G	W^QIOP	; try G
			0692	571	FAIL_CHECK	SS\$ _IVCHAN	; check failure
0000013C	8F	DD	0692			PUSHL #SS\$ _IVCHAN	
095D'CF	01	FB	0698			CALLS #1,W^REG_CHECK	

```

069D 573 .SBTTL QIOW TESTS
069D 574 :+
069D 575 :
069D 576 : $QIOW tests
069D 577 : test for an EFN of 0
069D 578 :
069D 579 :-
069D 580 NEXT_TEST
069D
069D STP23:
0004'CF 17 DO 069D MOVL #23,W^CURRENT_TC
0953'CF 00 DD 06A2 PUSHL #0
018B'CF 0042'LF DE 06A4 CALLS #1,W^REG_SAVE
06A9 581 MOVAL W^QIOW,W^SERV_NAME ; set service name
06B0 582 $CREMBX_S LOGNAM=W^MBNAM,-
06B0 583 CHAN=W^MBCHAN ; get a legal channel number
00E9'CF 018F'CF 3C 06C7 584 MOVZWL W^MBCHAN,W^QIOWP+QIOWS_CHAN ; set the channel number
06CE 585 $QIOW_S CHAN=W^MBCHAN,-
06CE 586 FUNC=#IOS_READVBLK,-
06CE 587 P1=W^MBNAM,-
06CE 588 P2=#0,-
06CL 589 EFN=#-1 ; try EFN=0
06F1 590 FAIL_CHECK SSS_ILLEFC ; check failure
0000G0EC 8F DD 06F1 PUSHL #SS$ ILLEFC
095D'CF 01 FB 06F7 CALLS #1,W^REG_CHECK
06FC 591 $QIOW G W^QIOWP ; try G
0705 592 FAIL_CHECK SSS_ILLEFC ; check failure
0705 593
095D'CF 01 FB 0708 CALLS #1,W^REG_CHECK
0710 593 :+
0710 594 :
0710 595 : test for an EFN of 500
0710 596 :
0710 597 :-
0710 598 NEXT_TEST
0710
0710 STP24:
0004'CF 18 DO 0710 MOVL #24,W^CURRENT_TC
0953'CF 00 DD 0715 PUSHL #0
0953'CF 01 FB 0717 CALLS #1,W^REG_SAVE
071C 599 $QIOW_S CHAN=W^MBCHAN,-
071C 600 FUNC=#IOS_READVBLK,-
071C 601 P1=W^MBNAM,-
071C 602 P2=#0,-
071C 603 EFN=#500 ; try EFN=500
073F 604 FAIL_CHECK SSS_ILLEFC ; check failure
073F 605 PUSHL #SS$ ILLEFC
095D'CF 01 FB 0745 CALLS #1,W^REG_CHECK
00E5'CF 000001F4 8F DO 074A 605 MOVL #500,W^QIOWP+QIOWS_EFN ; set illegal EFN
0753 606 $QIOW G W^QIOWP ; try G
075C 607 FAIL_CHECK SSS_ILLEFC ; check failure
075C 608
095D'CF 01 FB 0762 PUSHL #SS$ ILLEFC
0767 608 :+
0767 609 :
0767 610 : test for an EFN of 123 without an associated cluster
0767 611 :

```

```

0767 612 :-
0767 613 :- NEXT_TEST
0767
0767 STP25:
0004'CF 19 DO 0767 MOVL #25,W^CURRENT_TC
0953'CF 00 DD 076C PUSHL #0
0953'CF 01 FB 076E CALLS #1,W^REG_SAVE
0773 614 $QIOW_S CHAN=W^MBCHAN,-
0773 615 FUNC=#IOS$ READVBLK,-
0773 616 P1=W^MBNAM,-
0773 617 P2=#0,-
0773 618 EFN=#123 ; try S
0796 619 FAIL_CHECK SSS_UNASEFC ; check failure
095D'CF 01 DD 0796 PUSHL #SS$ UNASEFC
00000234 8F FB 079C CALLS #1,W^REG_CHECK
00E5'CF 0000007B 8F DO 07A1 620 MOVL #123,W^QIOWP+QIOW$_EFN ; set illegal EFN
07AA 621 $QIOW G W^QIOWP ; try G
07B3 622 FAIL_CHECK SSS_UNASEFC ; check failure
095D'CF 01 DD 07B3 PUSHL #SS$ UNASEFC
00000234 8F FB 07B9 CALLS #1,W^REG_CHECK
095D'CF 00E5'CF D4 07BE 623 CLRL W^QIOWP+QIOW$_EFN ; clean up illegal EFN
07C2 624 :+
07C2 625 :-
07C2 626 :- test unaccessable IOSB = page 0 access
07C2 627 :-
07C2 628 :-
07C2 629 :-
07C2 629 NEXT_TEST
07C2 STP26:
0004'CF 1A DO 07C2 MOVL #26,W^CURRENT_TC
0953'CF 00 DD 07C7 PUSHL #0
0953'CF 01 FB 07C9 CALLS #1,W^REG_SAVE
07CE 630 $QIOW_S CHAN=W^MBCHAN,-
07CE 631 FUNC=#IOS$ READVBLK,-
07CE 632 P1=W^MBNAM,-
07CE 633 P2=#0,-
07CE 634 IOSB=W^PRVHND_SXV40 ; try S
07EF 635 FAIL_CHECK SSS_ACCVIO ; check failure
095D'CF 01 DD 07EF PUSHL #SS$ ACCVIO
00F1'CF 0001'CF DE 07F1 CALLS #1,W^REG_CHECK
07F6 636 MOVAL W^PRVHND_SXV40,W^QIOWP+QIOW$_IOSB ; set illegal address
07FD 637 $QIOW G W^QIOWP ; try G
0806 638 FAIL_CHECK SSS_ACCVIO ; check the failure
095D'CF 01 DD 0806 PUSHL #SS$ ACCVIO
0808 FB 0808 CALLS #1,W^REG_CHECK
080D 639 :+
080D 640 :-
080D 641 :- test unaccessable IOSB = read-only PSECT
080D 642 :-
080D 643 :-
080D 644 :-
080D 644 NEXT_TEST
080D STP27:
0004'CF 1B DO 080D MOVL #27,W^CURRENT_TC
0953'CF 00 DD 0812 PUSHL #0
0953'CF 01 FB 0814 CALLS #1,W^REG_SAVE
0819 645 $QIOW_S CHAN=W^MBCHAN,-

```

```
0819 646 FUNC=#IOS$ READVBLK,-
0819 647 P1=W^MBNAM,-
0819 648 P2=#0,-
0819 649 IOSB=W^PRVHND_SXV41 ; try S
083A 650 FAIL_CHECK SSS_ACCVIO ; check failure
095D'CF 0C DD 083A
00F1'CF 01 FB 083C
0053'CF DE 0841 651 MOVAL W^PRVHND_SXV41,W^QIOWP+QIOWS_IOSB ; set IOSB adr
0848 652 $QIOW G W^QIOWP ; try G
0851 653 FAIL_CHECK SSS_ACCVIO ; check failure
095D'CF 0C DD 0851
01 FB 0853
0858 654 :+
0858 655 : test noaccess protection in IOSB
0858 656 :
0858 657 :
0858 658 :-
0858 659 NEXT_TEST
0858
0004'CF 1C DO 0858 STP28:
0953'CF 00 PD 085D
01 FB 085F
0864 660 $QIOW_S CHAN=W^MBCHAN,-
0864 661 FUNC=#IOS$ READVBLK,-
0864 662 P1=W^MBNAM,-
0864 663 P2=#0,-
0864 664 IOSB=W^PRVHND_SXV42 ; try S
0885 665 FAIL_CHECK SSS_ACCVIO ; check failure
095D'CF 0C DD 0885
01 FB 0887
088C 666 :+
088C 667 : test non-existent channel number
088C 668 :
088C 669 :
088C 670 :-
088C 671 NEXT_TEST
088C
0004'CF 1D DO 088C STP29:
0953'CF 01 DD 0891
01 FB 0893
0898 672 $DASSGN_S CHAN=W^MBCHAN ; release the channel
08A4 673 $QIOW_S CHAN=W^MBCHAN,-
08A4 674 FUNC=#IOS$ READVBLK,-
08A4 675 P1=W^MBNAM,-
08A4 676 P2=#0
08C3 677 FAIL_CHECK SSS_NOPRIV ; check failure
095D'CF 24 DD 08C3
01 FB 08C5
08CA 678 $QIOW G W^QIOWP ; try G
08D3 679 FAIL_CHECK SSS_NOPRIV ; check failure
095D'CF 24 DD 08D3
01 FB 08D5
08DA 680 :+
08DA 681 : test illegal channel number
08DA 682 :-
```

			08DA	683	:	
			08DA	684	:-	
			08DA	685		NEXT_TEST
			08DA			
			08DA			STP30:
0004'CF	1E	DD	08DA			MOVL #30,W^CURRENT_TC
	00	DD	08DF			PUSHL #0
0953'CF	01	FB	08E1			CALLS #1,W^REG_SAVE
	018F'LF	D4	08E6	686		CLRL W^MBCHAN ; set illegal channel number
			08EA	687		\$QIOW_S CHAN=W^MBCHAN,-
			08EA	688		FUNC=#IOS_READVBLK,-
			08EA	689		P1=W^MBNAM,-
			08EA	690		P2=#0
			0909	691		FAIL_CHECK SSS_IVCHAN ; check failure
0000013C	8F	DD	0909			PUSHL #SS\$_IVCHAN
095D'CF	01	FB	090F			CALLS #1,W^REG_CHECK
	00E9'CF	D4	0914	692		CLRL W^QIOWP+QIOWS_CHAN ; set illegal channel number
			0918	693		\$QIOW G W^QIOWP ; try G
			0921	694		FAIL_CHECK SSS_IVCHAN ; check failure
0000013C	8F	DD	0921			PUSHL #SS\$_IVCHAN
095D'CF	01	FB	0927			CALLS #1,W^REG_CHECK
			092C	695		TEST_END ; end the test
	004C'CF	DD	092C			PUSHL W^TMD_ADDR
	0048'CF	DD	0930			PUSHL W^TMN_ADDR
		DD	0934			PUSHL #2
	0044'CF	DD	0936			PUSHL W^MOD_MSG_CODE
00000000'GF	04	FB	093A			CALLS #SST1,G^LIB\$SIGNAL
0044'CF	01	FO	0941			INSV #1,#ST\$SV_INHIB_MSG,#1,W^MOD_MSG_CODE
	0044'CF	DD	0948			PUSHL W^MOD_MSG_CODE
00000000'GF	01	FB	094C			CALLS #1,G^SYS\$EXIT

52

54


```

0953 697 .SBTTL REG_SAVE
0953 698 :++
0953 699 : FUNCTIONAL DESCRIPTION:
0953 700 : Subroutine to save R2-R11 in the register save location.
0953 701 :
0953 702 : CALLING SEQUENCE:
0953 703 : PUSHL #0 ; save a dummy parameter
0953 704 : CALLS #1,W^REG_SAVE ; save R2-R11
0953 705 :
0953 706 : INPUT PARAMETERS:
0953 707 : NONE
0953 708 :
0953 709 : OUTPUT PARAMETERS:
0953 710 : NONE
0953 711 :
0953 712 :--
0953 713 :
0953 714 REG_SAVE:
0008'CF 14 AD 28 OFFC 0953 715 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0953 716 MOVCL #4*10,^X14(FP),W^REG_SAVE_AREA ; save the registers in the program
0953 717 RET
0953 718 .SBTTL REG_CHECK
0953 719 :++
0953 720 : FUNCTIONAL DESCRIPTION:
0953 721 : Subroutine to test R0 & R2-R11 for proper content after a service
0953 722 : execution. A snapshot is taken by the REG_SAVE routine at the
0953 723 : beginning of each step and this routine is executed after the
0953 724 : services have been executed.
0953 725 :
0953 726 : CALLING SEQUENCE:
0953 727 : PUSHL #SS$ XXXXX ; push expected R0 contents
0953 728 : CALLS #1,W^REG_CHECK ; execute this routine
0953 729 :
0953 730 : INPUT PARAMETERS:
0953 731 : expected R0 contents on the stack
0953 732 :
0953 733 : OUTPUT PARAMETERS:
0953 734 : possible error messages printed using $PUTMSG
0953 735 :
0953 736 :--
0953 737 :
0953 738 REG_CHECK:
0953 739 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
50 04 AC D1 095F 740 CMPL 4(AP),R0 ; is this the right fail code?
0953 741 BEQL 10$ ; br if yes
0953 742 PUSHL R0 ; push received data
0953 743 PUSHL 4(AP) ; push expected data
0953 744 PUSHAL W^EXP ; push the string variable
09A5'CF 03 FB 096E 745 CALLS #3,W^PRINT_FAIL ; print the error message
0973 746 10$:
0008'CF 14 AD 28 29 0973 747 CMPC3 #4*10,^X14(FP),W^REG_SAVE_AREA ; check all but R0
56 53 00000008'BF C3 097A 748 BEQL 20$ ; br if O.K.
0127'CF 56 04 C6 0984 750 SUBL3 #REG_SAVE_AREA,R3,R6 ; calculate the register number
56 53 00000008'BF C3 097C 749 DIVL2 #4,R6 ;
0127'CF 56 02 B1 0987 751 ADDB3 #^X2,R6,W^REGNUM ; put it in the string
56 53 00000008'BF C3 097C 749 BICL2 #3,R1 ; backup to register boundry
0127'CF 51 03 CA 098D 752 BICL2 #3,R3
0127'CF 53 03 CA 0990 753 BICL2 #3,R3

```

```

0127'CF DD 0993 754          PUSHL  W^REGNUM          ; push register number
        61 DD 0997 755          PUSHL  (R1)              ; push received data
        63 DD 0999 756          PUSHL  (R3)              ; push expected data
0115'CF DF 099B 757          PUSHAL W^REG              ; set string ptr param.
09A5'CF 04 FB 099F 758        CALLS  #4,W^PRINT_FAIL    ; print the error message
        04 09A4 759 20$:      RET
        09A4 760          .SBTTL PRINT_FAIL
        09A5 761          :++
        09A5 762          : FUNCTIONAL DESCRIPTION:
        09A5 763          : Subroutine to report failures using $PUTMSG
        09A5 764          :
        09A5 765          : CALLING SEQUENCE:
        09A5 766          : Mode #1          PUSHL EXPECTED Mode #2          PUSHL REG NUMBER
        09A5 767          :                   PUSHL RECEIVED                PUSHL EXPECTED
        09A5 768          :                   PUSHAL STRING VAR              PUSHL RECEIVED
        09A5 769          :                   CALLS #3,W^PRINT_FAIL          PUSHAL STRING VAR
        09A5 770          :                   CALLS #4,W^PRINT_FAIL          CALLS #4,W^PRINT_FAIL
        09A5 771          :
        09A5 772          : INPUT PARAMETERS:
        09A5 773          : listed above
        09A5 774          :
        09A5 775          : OUTPUT PARAMETERS:
        09A5 776          : an error message is printed using $PUTMSG
        09A5 777          :
        09A5 778          :
        09A5 779          :--
        09A5 780          :
        003C 09A5 781 PRINT_FAIL:
        09A5 782          .WORD  ^M<R2,R3,R4,R5>
        09A7 783          $FAO_S  W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
        09C8 784          PUTMSG  <#UETPS_TEXT,#1,#MESSAGEL>          ; print the message
        04 6C 91 09DD 785          CMPB  (AP),#4                      ; is this a register message?
        21 13 09E0 786          BEQL  10$                          ; br if yes
        25 11 09E2 787          $FAO_S W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
        0A01 788          BRB  20$                                  ; goto output message
        0A03 789 10$:      $FAO_S  W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
        0A03 790          :
        0A28 791 20$:      PUTMSG  <#UETPS_TEXT,#1,#MESSAGEL>          ; print the message
        0A28 792          MOVAL  W^TEST_MOD_FAIL,W^TMD_ADDR          ; set failure message address
0044'CF 03 002A'CF DE 0A3D 793          INSV  #ERROR,#0,#3,W^MOD_MSG_CODE    ; set severity code
0044'CF 03 00 02 04 0A44 794          RET
        04 0A4B 795

```

```
0A4C 797 .SBTTL MOD_MSG_PRINT
0A4C 798 MOD_MSG_PRINT:
0A4C 799 :
0A4C 800 : *****
0A4C 801 : *
0A4C 802 : * PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES *
0A4C 803 : * (USING THE PUTMSG MACRO). *
0A4C 804 : *
0A4C 805 : *****
0A4C 806 :
05 0A4C 807 PUTMSG <W^MOD_MSG_CODE,#2,W^TMN_ADDR,W^TMD_ADDR> ; PRINT MSG
0A61 808 RSB ; ... AND RETURN TO CALLER
0A62 809 :
0A62 810 .SBTTL CHMRTN
0A62 811 CHMRTN:
0A62 812 : *****
0A62 813 : *
0A62 814 : * CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER *
0A62 815 : * A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED *
0A62 816 : * BY THE MODE MACRO ('TO' OPTION). IT MERELY DOES *
0A62 817 : * A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS *
0A62 818 : * THE EFFECT OF RETURNING TO THE END OF THE MODE *
0A62 819 : * MACRO EXPANSION. *
0A62 820 : *
0A62 821 : *****
0A62 822 :
00000059'FF 0000 0A62 823 .WORD 0 ; ENTRY MASK
17 0A64 824 JMP @CHM_CONT ; RETURN TO MODE MACRO IN NEW MODE
0A6A 825 :
0A6A 826 : * RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
0A6A 827 :
0A6A 828 .END SATSSF17
```

SATSSF17
Symbol table

\$\$ARGS	= 0000000C			QIOWS_FUNC	= 0000C00C		
\$\$T1	= 00000004			QIOWS_IOSB	= 00000010		
\$\$T2	= 00000009			QIOWS_NARGS	= 0000000C		
BUF	00000133	R	03	QIOWS_P1	= 0000001C		
CHMRTN	00000A62	R R	06	QIOWS_P2	= 00000020		
CHM_CONT	00000059	R R	03	QIOWS_P3	= 00000024		
CS1	00000053	R R	02	QIOWS_P4	= 00000028		
CS2	00000085	R R	02	QIOWS_P5	= 0000002C		
CS3	000000B2	R R	02	QIOWS_P6	= 00000030		
CURRENT_TC	00000004	R R	03	QIOWP	000000E1	R	03
EMPTY	00000000	R	04	REG	00000115	R R	03
ERROR	= 00000002			REGNUM	00000127	R R	03
EXP	000000E5	R	02	REG_CHECK	0000095D	R R	06
INADR	00000047	R	02	REG_SAVE	00000953	R R	06
INFO	= 00000003			REG_SAVE_AREA	00000008	R R	03
INP	00000065	R	03	RETADR	0000005D	R R	03
INPUT	00000031	R	02	SATSSF17	00000000	R G	06
IOS_READVBLK	= 00000031			SERV_NAME	0000018B	R	03
IOS_WRITEVBLK	= 00000030			SEVERE	= 00000004		
LIB\$SIGNAL	*****	X	06	SHR\$K SHRDEF	= 00000001		
MBCHAN	0000018F	R	03	SHR\$ TEXT	= 00001130		
MBNAM	000000F3	R R	02	SS\$_ACCPIO	= 0000000C		
MESSAGEL	00000183	R R	03	SS\$_ILLEFC	= 000000EC		
MOD_MSG_CODE	00000044	R R	03	SS\$_IVCHAN	= 0000013C		
MOD_MSG_PRINT	00000A4C	R R	06	SS\$_NOPRIV	= 00000024		
MSGC	0000012B	R R	03	SS\$_UNASEFC	= 00000234		
NOACCESS	00000000	R R	05	STEP	= 0000001E		
OUT	00000088	R R	03	STP0	0000003D	R	06
OUTPUT	00000037	R R	02	STP1	000000A2	R R	06
PRINT_FAIL	000009A5	R R	06	STP10	000002C0	R R	06
PRIVMASK	00000051	R R	03	STP11	000002FC	R R	06
PROT	0000004F	R	02	STP12	00000332	R R	06
PRT\$C_NA	*****	X	02	STP13	00000368	R	06
PRVHND_SXV40	= 00000001			STP14	0000039E	R	06
PRVHND_SXV41	= 00000053	R R	02	STP15	000003DE	R R	06
PRVHND_SXV42	= 000001FF	R R	04	STP16	00000481	R R	06
PRVPRT	00000050	R R	03	STP17	000004D8	R R	06
QIO	0000003E	R	02	STP18	00000533	R R	06
QIOS_ASTADR	= 00000014			STP19	0000057E	R	06
QIOS_ASTPRM	= 00000018			STP2	000000DC	R	06
QIOS_CHAN	= 00000008			STP20	000005C9	R	06
QIOS_EFN	= 00000004			STP21	000005FD	R R	06
QIOS_FUNC	= 0000000C			STP22	0000064B	R R	06
QIOS_IOSB	= 0000C010			STP23	0000069D	R R	06
QIOS_NARGS	= 0000000C			STP24	00000710	R	06
QIOS_P1	= 0000001C			STP25	00000767	R	06
QIOS_P2	= 00000020			STP26	000007C2	R	06
QIOS_P3	= 00000024			STP27	0000080D	R R	06
QIOS_P4	= 00000028			STP28	00000858	R R	06
QIOS_P5	= 0000002C			STP29	0000088C	R R	06
QIOS_P6	= 00000030			STP3	00000116	R	06
QIOP	000000AD	R	03	STP30	000008DA	R	06
QIOW	00000042	R	02	STP4	0000014A	R	06
QIOWS_ASTADR	= 00000014			STP5	0000017E	R	06
QIOWS_ASTPRM	= 00000018			STP6	000001B2	R	06
QIOWS_CHAN	= 0000C008			STP7	000001F0	R	06
QIOWS_EFN	= 00000004			STP8	0000022A	R	06

```

STP9          00000284 R    06
STSSV INHIB_MSG = 0000001C
SUCCESS      = 00000001
SYSSCREMBX   ***** GX   06
SYSSDASSGN   ***** GX   06
SYSS$EXIT    ***** GX   06
SYSSFAO      ***** X    06
SYSSHIBER    ***** GX   06
SYSSQIO      *****:GX   06
SYSSQIOW     *****GX    03
SYSS$SETPRN  *****GX   06
SYSS$SETPRT  *****GX   06
SYSSWAKE     *****GX   06
TEST_MOD_BEGIN 00000019 R    02
TEST_MOD_FAIL  0000002A R    02
TEST_MOD_NAME  00000000 R    02
TEST_MOD_NAME_D 00000009 R    02
TEST_MOD_SUCC  0000001F R    02
TMD_ADDR      0000004C R    03
TMN_ADDR      00000048 R    03
TPID          00000000 R    03
UETPS_SATSMS  = 007480D9
UETPS_TEXT    = 00741133
WARNING       = 00000000
    
```

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
. ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
RODATA	000000FD (253.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
RWDATA	00000191 (401.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
SATS_ACCVIO_1	00000200 (512.)	04 (4.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC PAGE
SATS_ACCVIO_2	00000200 (512.)	05 (5.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC PAGE
SATSSF17	00000A6A (2666.)	06 (6.)	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	00:00:00.54
Command processing	138	00:00:00.70	00:00:03.69
Pass 1	488	00:00:20.48	00:00:40.48
Symbol table sort	0	00:00:02.16	00:00:03.54
Pass 2	188	00:00:04.58	00:00:09.53
Symbol table output	17	00:00:00.15	00:00:00.17
Psect synopsis output	2	00:00:00.04	00:00:00.04
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	872	00:00:28.19	00:00:57.99

The working set limit was 1950 pages.
126282 bytes (247 pages) of virtual memory were used to buffer the intermediate code.

SATSSF17
VAX-11 Macro Run Statistics

- SATS SYSTEM SERVICE TESTS (FAILING S. 16-SEP-1984 01:41:08 VAX/VMS Macro V04-00
5-SEP-1984 04:22:23 [UETP.SRC]SATSSF17.MAR;1

Page 27
(3)

SA
VO

There were 80 pages of symbol table space allocated to hold 1387 non-local and 4 local symbols.
828 source lines were read in Pass 1, producing 34 object records in Pass 2.
52 pages of virtual memory were used to define 48 macros.

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

Macro Library name	Macros defined
-----	-----
-\$255\$DUA28:[UETP.OBJ]UETP.MLB;1	10
-\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	2
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	33
TOTALS (all libraries)	45

1663 GETS were required to define 45 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:SATSSF17/OBJ=OBJ\$:SATSSF17 MSRC\$:SATSSF17/UPDATE=(ENH\$:SATSSF17)+EXECMLS/LIB+LIB\$:UETP/LIB

0409 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION CONFIDENTIAL AND PROPRIETARY

RMSTEST4 LIS

RMSTEST7 LIS

RMSTEST8 LIS

RMSTESTR LIS

RMSTEST0 LIS

RMSTEST2 LIS

RMSTEST3 LIS

RMSTEST5 LIS

RMSTEST6 LIS

RMSTESTT LIS

RMSTEST1 LIS

SATSSF17 LIS

0410 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

The image displays a grid of 100 small terminal window screenshots, arranged in a 10x10 grid. Each window shows a different VAX/VMS command and its output. Some windows are clearly legible and contain the following text:

- SATSSSF08 LIS
- SATSSSF09 LIS
- UETCOMS00 LIS
- UETDISK00 LIS
- SATSSSF10 LIS
- UETMPF00 LIS

The other windows show various system messages, file listings, and command prompts.