

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	
UUUUUUUUUUUUUUUU	UUU	EEEEEEEEEEEEEEEE	TTT	PPP	
UUUUUUUUUUUUUUUU	UUU	EEEEEEEEEEEEEEEE	TTT	PPP	
UUUUUUUUUUUUUUUU	UUU	EEEEEEEEEEEEEEEE	TTT	PPP	

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

```

SSSSSSSS  AAAAAA  TTTTTTTTTT  SSSSSSSS  SSSSSSSS  SSSSSSSS  11  00000G
SSSSSSSS  AAAAAA  TTTTTTTTTT  SSSSSSSS  SSSSSSSS  SSSSSSSS  11  000000
SS         AA      AA      TT         SS         SS         SS         1111  00      00
SS         AA      AA      TT         SS         SS         SS         1111  00      00
SS         AA      AA      TT         SS         SS         SS         11    00    0000
SS         AA      AA      TT         SS         SS         SS         11    00    0000
SSSSSSS   AA      AA      TT         SSSSSSS   SSSSSSS   SSSSSSS   11    00    00
SSSSSSS   AA      AA      TT         SSSSSSS   SSSSSSS   SSSSSSS   11    00    00
          SS     AA      AA      TT         SS         SS         SS         11    0000  00
          SS     AA      AA      TT         SS         SS         SS         11    0000  00
          SS     AA      AA      TT         SS         SS         SS         11    00    00
          SS     AA      AA      TT         SS         SS         SS         11    00    00
SSSSSSSS  AA      AA      TT         SSSSSSSS  SSSSSSSS  SSSSSSSS  111111  000000  .....
SSSSSSSS  AA      AA      TT         SSSSSSSS  SSSSSSSS  SSSSSSSS  111111  000000  .....

```

```

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

```

SATSSS10
Table of contents

(1)	56	DECLARATIONS
(1)	171	R/W PSECT
(1)	240	SATSSS10
(1)	289	GETMSG TESTS
(1)	342	PUTMSG TESTS
(1)	408	REG_SAVE
(1)	429	REG_CHECK
(1)	472	PRINT_FAIL
(1)	508	CHECK_RESULTS
(1)	564	ACT

```

0000 1 .TITLE SATSSS10 - 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 SATSSS10 module tests the execution of the following
0000 33 : VMS system services:
0000 34
0000 35 : $GETMSG
0000 36 : $PUTMSG
0000 37
0000 38
0000 39 : ENVIRONMENT: User mode image.
0000 40 : Needs CMKRNL privilege and dynamically acquires other
0000 41 : privileges, as needed.
0000 42
0000 43 : AUTHOR: Larry D. Jones, CREATION DATE: JULY, 1978
0000 44
0000 45 : MODIFIED BY:
0000 46
0000 47 : V03-002 LDJ0002 Larry D. Jones, 14-Dec-1981
0000 48 : Modified to conform to new $PUTMSG argument modification which
0000 49 : made a BR out of range error in routine CHECK_RESULTS.
0000 50
0000 51 : V03-001 LDJ0001 Larry D. Jones, 17-Sep-1980
0000 52 : Modified to conform to new build command procedures.
0000 53 : **
0000 54 : --

```

```
0000 56 .SBTTL DECLARATIONS
0000 57 :
0000 58 : MACRO LIBRARY CALLS
0000 59 :
0000 60 $PRVDEF ; privilege definitions
0000 61 $SHR MESSAGES UETP,116,<<TEXT,INFO>> ; UETP$ TEXT definition
0000 62 $STSDEF ; STS definitions
0000 63 $UETPDEF ; UETP message definitions
0000 64 :
0000 65 : Equated symbols
0000 66 :
00000000 0000 67 WARNING = 0 ; warning severity value for msgs
00000001 0000 68 SUCCESS = 1 ; success
00000002 0000 69 ERROR = 2 ; error
00000003 0000 70 INFO = 3 ; information
00000004 0000 71 SEVERE = 4 ; fatal
0000 72 :
0000 73 : MACROS
0000 74 :
```

```

00000000 76 .PSECT RODATA,RD,NOWRT,NOEXE,LONG
0000 77 ;
0000 78 TEST_MOD_NAME:
30 31 53 53 53 54 41 53 00' 0000 79 .ASCIC /SATSSS10/ ; needed for SATSMS message
08 0000
0009 80 TEST_MOD_NAME_D:
53 53 53 54 41 53 00000L'1'010E0000' 0009 81 .ASCIC /SATSSS10/ ; module name
30 31 0017
0019 82 TEST_MOD_BEGIN:
6E 75 57 65 62 00' 0019 83 .ASCIC /begun/
05 0019
001F 84 TEST_MOD_SUCC:
6C 75 66 73 73 65 63 63 75 73 00' 001F 85 .ASCIC /successful/
0A 001F
002A 86 TEST_MOD_FAIL:
64 65 6C 69 61 66 00' 002A 87 .ASCIC /failed/
06 002A
0031 88 GETMSG:
47 53 4D 54 45 47 00' 0031 89 .ASCIC /GETMSG/
06 0031
0038 90 PUTMSG:
47 53 4D 54 55 50 00' 0038 91 .ASCIC /PUTMSG/
06 0038
003F 92 CS1:
21 20 74 73 65 54 00000047'010E0000' 003F 93 .ASCIC \Test !AC service name !AC step !UL failed.\
6E 20 65 63 69 76 72 65 73 20 43 41 004D
70 65 74 73 20 43 41 21 20 65 6D 61 0059
2E 64 65 6C 69 61 66 20 4C 55 21 20 0065
0071 94 CS2:
74 63 65 70 78 45 00000079'010E0000' 0071 95 .ASCIC \Expected !AS = !XL received !AS = !XL\
4C 58 21 20 3D 20 53 41 21 20 64 65 007F
41 21 20 64 65 76 69 65 63 65 72 20 008B
4C 58 21 20 3D 20 53 0097
009E 96 CS3:
74 63 65 70 78 45 000000A6'010E0000' 009E 97 .ASCIC \Expected !AS!UB = !XL received !AS!UB = !XL\
20 3D 20 42 55 21 53 41 21 20 64 65 00AC
64 65 76 69 65 63 65 72 20 4C 58 21 008B
58 21 20 3D 20 42 55 21 53 41 21 20 00C4
4C 00D0
00D1 98 CS4:
74 63 65 70 78 45 000000D9'010E0000' 00D1 99 .ASCIC \Expected= !AS!/-UETP-I-TEXT, Received= !AS\
45 55 2D 2F 21 53 41 21 2D 3D 64 65 00DF
52 2D 2C 54 58 45 54 2D 49 2D 50 54 00EB
53 41 21 20 3D 64 65 76 69 65 63 65 00F7
0103 100 EXP:
73 75 74 61 74 73 0000010B'010E0000' 0103 101 .ASCIC \status\
0111 102 LEN:
68 74 67 6E 65 6C 00000119'010E0000' 0111 103 .ASCIC \length\
011F 104 OUT:
52 44 41 54 55 4F 00000127'010E0000' 011F 105 .ASCIC \OUTADR\
012D 106 MSGID:
007480E1 012D 107 .LONG UETPS_DDB ; message ID used in the tests
0131 108 MSGVEC:
00000003 0131 109 .LONG 3 ; PUTMSG message vector
00741133 0135 110 .LONG UETPS_TEXT
00000001 0139 111 .LONG 1
000002B3' 013D 112 .ADDRESS MESSAGEL

```

```

00000218'000001EC'000001E7'000001C1' 0141 113 MSGTBL:
0000027A'0000024C'00000245'0000021B' 0141 114 .ADDRESS M1,M2,M3,M4,M5,M6,M7,M8,M9,M10,M11,M12,M13,M14,M15
000002E8'000002B7'000002AD'00000280' 0151
0000032B'0000031F'000002F0' 0161
00000384'000001E7'0000035E'00000495' 0171
000003DA'00000245'00000380'00000218' 017D 115 MSGTBL1:
0000035'000002AD'00000408'0000027A' 017D 116 .ADDRESS M23,M16,M2,M17,M4,M18,M6,M19,M8,M20,M10,M21,M12,M22,M14,M23,M24
00000455'0000031F'00000466'000002E8' 018D
000004CB' 019D
01AD
01BD
01C1 117 .NLIST BINARY
01C1 118 M1:
44 20 30 30 54 49 4E 49 54 45 55 00' 01C1 119 .ASCIC /UETINIT00 DDB 0 !AS 00000000 00000000/
30 30 30 20 53 41 21 20 30 20 42 44 01CD
30 30 30 30 30 30 20 30 30 30 30 30 01D9
30 30 30 30 30 30 30 30 30 30 30 30 01E5
25 01E7
42 44 44 25 00' 01E7 120 M2:
04 01E7 121 .ASCIC /%DDB/
01EC 122 M3:
4E 49 54 45 55 20 2C 42 44 44 25 00' 01EC 123 .ASCIC /%DDB, UETINIT00 DDB 0 !AS 00000000 00000000/
21 20 30 20 42 44 44 20 30 30 54 49 01F8
20 30 30 30 30 30 30 30 30 30 20 53 41 0204
30 30 30 30 30 30 30 30 30 30 30 30 0210
28 01EC
53 25 00' 0218 124 M4:
02 0218 125 .ASCIC /%S/
0218
54 49 4E 49 54 45 55 20 2C 53 25 00' 0218 126 M5:
53 41 21 20 30 20 42 44 44 20 30 30 0218 127 .ASCIC /%S, UETINIT00 DDB 0 !AS 00000000 00000000/
30 30 20 30 30 30 30 30 30 30 30 20 0227
30 30 30 30 30 30 30 30 30 30 30 30 0233
30 30 30 30 30 30 30 30 30 30 30 30 023F
29 0218
42 44 44 2D 53 25 00' 0245 128 M6:
06 0245 129 .ASCIC /%S-DDB/
0245
54 45 55 20 2C 42 44 44 2D 53 25 00' 024C 130 M7:
30 20 42 44 44 20 30 30 54 49 4E 49 024C 131 .ASCIC /%S-DDB, UETINIT00 DDB 0 !AS 00000000 00000000/
30 30 30 30 30 30 30 30 20 53 41 21 20 0258
30 30 30 30 30 30 30 30 30 30 20 30 0264
30 30 30 30 30 30 30 30 30 30 20 30 0270
2D 024C
50 54 45 55 25 00' 027A 132 M8:
05 027A 133 .ASCIC /%UETP/
027A
49 54 45 55 20 2C 50 54 45 55 25 00' 0280 134 M9:
20 30 20 42 44 44 20 30 30 54 49 4E 0280 135 .ASCIC /%UETP, UETINIT00 DDB 0 !AS 00000000 00000000/
30 30 30 30 30 30 30 30 20 53 41 21 028C
30 30 30 30 30 30 30 30 30 30 20 30 0298
30 30 30 30 30 30 30 30 30 30 20 30 02A4
2C 0280
42 44 44 2D 50 54 45 55 25 00' 02AD 136 M10:
09 02AD 137 .ASCIC /%UETP-DDB/
02AD

```



```

20 20 42 44 44 2D 50 54 45 55 25 00' 0435
44 44 20 30 30 54 49 4E 49 54 45 55 0441
30 30 20 30 20 41 44 54 20 30 20 42 044D
30 30 30 30 30 30 20 30 30 30 30 30 0459
30 0465
30 0435
30 0466
45 55 20 20 53 2D 50 54 45 55 25 00' 0466
20 42 44 44 20 30 30 54 49 4E 49 54 0472
30 30 30 30 30 30 20 41 44 54 20 30 047E
30 30 30 30 30 30 30 30 20 30 30 048A
2E 0466
0495
42 44 44 2D 53 2D 50 54 45 55 25 00' 0495
20 30 30 54 49 4E 49 54 45 55 20 20 04A1
30 30 20 41 44 54 20 30 20 42 44 44 04AD
30 30 30 30 30 20 30 30 30 30 30 30 04B9
30 30 30 04C5
32 0495
04C8
42 44 44 2D 53 2D 54 53 45 54 25 00' 04C8
20 30 30 54 49 4E 49 54 45 55 20 20 04D4
30 30 20 41 44 54 20 30 20 42 44 44 04E0
30 30 30 30 30 20 30 30 30 30 30 30 04EC
30 04FB
32 04C8
04FB
54 53 45 54 00000503'010E0000' 04FB
0507

```

```

159 .ASCIC /XUETP-DDB, UETINIT00 DDB 0 TDA 00000000 00000000/
160 M22:
161 .ASCIC /XUETP-S, UETINIT00 DDB 0 TDA 00000000 00000000/
162 M23:
163 .ASCIC /XUETP-S-DDB, UETINIT00 DDB 0 TDA 00000000 00000000/
164 M24:
165 .ASCIC /XTEST-S-DDB, UETINIT00 DDB 0 TDA 00000000 00000000/
166 TEST:
167 .ASCID /TEST/
168 .LIST BINARY

```

```

0507 170 ;
0507 171 .SBTTL R/W PSECT
00000000 172 .PSECT RWDATA, RD, WRT, NOEXE, LONG
0000 173 ;
0000 174 IPID:
00000000 0000 175 .LONG 0 ; PID for this process
00000000 0004 176 CURRENT_IC: ; ptr to current test case
00000000 0004 177 .LONG 0
0008 178 .ALIGN LONG
00000044 0008 179 REG_SAVE_AREA: ; register save area
0004 180 .BLKL 15
007480D9 0044 181 MOD_MSG_CODE: ; test module message code for putmsg
0048 182 .LONG UETPS_SATSMS
00000000' 0048 183 TMN_ADDR:
004C 184 .ADDRESS TEST_MOD_NAME
00000019' 004C 185 TMD_ADDR:
004C 186 .ADDRESS TEST_MOD_BEGIN
0050 187 PRVPRT:
00 0050 188 .BYTE 0 ; protection return byte for SETPRT
00000000 00000000 0051 189 PRIVMASK: ; priv. mask
0059 190 .QUAD 0
00000000 0059 191 CHM_CONT: ; change mode continue address
005D 192 .LONG 0
00000065 005D 193 RETADR: ; returned address's from SETPRT
0065 194 .BLKL 2
00000000 0065 195 STATUS: ; returned address's from SETPRT
0065 196 .LONG 0
0069 197 GET:
0069 198 $GETMSG UETPS_DDB, MSGLEN, BUFADR, 0, OUTADR ; GETMSG parameter list
00000005' 0069 .ADDRESS 5
007480E1' 006D .ADDRESS UETPS_DDB
000002BF' 0071 .ADDRESS MSGLEN
000002C7' 0075 .ADDRESS BUFADR
00000000' 0079 .ADDRESS 0
000002C3' 007D .ADDRESS OUTADR
0081 199 PUT:
0081 200 $PUTMSG MSGVEC1, ACT, 0 ; PUTMSG parameter list
00000004' 0081 .ADDRESS 4
000003DF' 0085 .ADDRESS MSGVEC1
00000355' 0089 .ADDRESS ACT
00000000' 008D .ADDRESS 0
00000000' 0091 .ADDRESS 0
74 73 69 67 65 72 0000009D' 010E0000' 0095 201 REG:
52 20 72 65 00A3 202 .ASCID \register R\
00A7 203 REGNUM:
00000000 00A7 204 .LONG 0 ; register number
00AB 205 MSGL: ; buffer desc.
00000200 00AB 206 .LONG 512
000000B3' 00AF 207 .ADDRESS BUF
00B3 208 BUF:
000002B3 00B3 209 .BLKB 512
02B3 210 MESSAGEL: ; message desc.
00000000 02B3 211 .LONG 0
000000B3' 02B7 212 .ADDRESS BUF
02BB 213 SERV_NAME:
00000000 02BB 214 .LONG 0 ; service name pointer

```

SA
Sy
SS
SS
SS
AC
BA
BU
BU
CH
CH
CH
CS
CS
CS
CS
CU
DE
ER
EX
GE
GE
GE
GE
GE
GE
GO
IN
LE
LI
M1
M1
M1
M1
M1
M1
M1
M1
M2
M2
M2
M2
M2
M2
M3
M4
M5
M6
M7
M8
M9
ME
MO

00000000	02BF	215	MSGLEN:			
	02BF	216		.LONG	0	; message length parameter
00000000	02C3	217	OUTADR:			
	02C3	218		.LONG	0	; output address parameter
00000100	02C7	219	BUFADR:			
	02C7	220		.LONG	256	; GETMSG & PUTMSG buffer
000002CF	02CB	221		.ADDRESS	+4	
000003CF	02CF	222		.BLKB	256	; maximum message length
	03CF	223	BADATA:			
00000000	03CF	224		.LONG	0	; bad string desc.
000002CF	03D3	225		.ADDRESS	BUFADR+8	
	03D7	226	GOODATA:			
00000000	03D7	227		.LONG	0	; expected string desc.
00000000	03DB	228		.ADDRESS	0	
	03DF	229	MSGVEC1:			
00000003	03DF	230		.LONG	3	; PUTMSG test message vector
007480E1	03E3	231		.LONG	UETPS_DDB	
00000001	03E7	232		.LONG	1	
000003EF	03EB	233		.ADDRESS	DEV_NAM	
	03EF	234	DEV_NAM:			
41 44 54 000003F7'010E0000'	03EF	235		.ASCID	/TDA/	; test device name
	03FA	236	MSG:			
00000000	03FA	237		.LONG	0	; correct message pointer

SA
Sy
TP
UE
UE
UE
WA

PS
--
SA
RO
RU
SA

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

Th
41
Th
62
38

Ma
--
--
--
TO
SO
Th
MA

```

00000000 239      .PSECT  SATSSS10, RD, WRT, EXE, LONG
0000      240      .SBT:IL  SATSSS10
0000      241      :++
0000      242      : FUNCTIONAL DESCRIPTION:
0000      243      :
0000      244      :     After performing some initial housekeeping, such as
0000      245      :     printing the module begin message and acquiring needed privileges,
0000      246      :     the system services are tested in each of their normal conditions.
0000      247      :     Detected failures are identified and an error message is printed
0000      248      :     on the terminal. Upon completion of the test a success or fail
0000      249      :     message is printed on the terminal.
0000      250      :
0000      251      : CALLING SEQUENCE:
0000      252      :
0000      253      :     $ RUN SATSSS10 ... (DCL COMMAND)
0000      254      :
0000      255      : INPUT PARAMETERS:
0000      256      :
0000      257      :     none
0000      258      :
0000      259      : IMPLICIT INPUTS:
0000      260      :
0000      261      :     none
0000      262      :
0000      263      : OUTPUT PARAMETERS:
0000      264      :
0000      265      :     none
0000      266      :
0000      267      : IMPLICIT OUTPUTS:
0000      268      :
0000      269      :     Messages to SYS$OUTPUT are the only output from SATSSS10.
0000      270      :     They are of the form:
0000      271      :
0000      272      :     %UETP-S-SATSMS, TEST MODULE SATSSS10 BEGUN ... (BEGIN MSG)
0000      273      :     %UETP-S-SATSMS, TEST MODULE SATSSS10 SUCCESSFUL ... (END MSG)
0000      274      :     %UETP-E-SATSMS, TEST MODULE SATSSS10 FAILED ... (END MSG)
0000      275      :     %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
0000      276      :
0000      277      : COMPLETION CODES:
0000      278      :
0000      279      :     The SATSSS10 routine terminates with a $EXIT to the
0000      280      :     operating system with a status code defined by UETP$_SATSMS.
0000      281      :
0000      282      : SIDE EFFECTS:
0000      283      :
0000      284      :     none
0000      285      :
0000      286      : --
0000      287      :
0000      288      : TEST_START SATSSS10           ; 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
0009'CF 7F 001A
00000000'GF 01 FB 001E
034F 30 0025
004C'CF 001F'CF DE 0028
0044'CF 03 00 01 FO 002F
00 DD 0036
019B'CF 01 FB 0038

```

.LIST ME

```

.ENTRY SATSSS10.0
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

```

STP0:

.SBTTL GETMSG TESTS

```

003D 289
003D 290
003D 291
003D 292
003D 293
003D 294
003D 295
003D 296
003D 297
003D 298
003D 299
003D 300
003D 301
003D 302
003D 303
003D 304
003D 305
003D 306
003D 307
003D 308
003D 309
003D 310
003D 311
003D 312
003D 313
003D 314
003D 315
003D 316

```

\$GETMSG tests

The tests are executed in the following order:

STP #	FLAG value	message content
STP0	0	no message
STP1	1	only text
STP2	2	only mesg ID
STP3	3	text and mesg ID
STP4	4	only severity code
STP5	5	severity code & text
STP6	6	severity code & mesg ID
STP7	7	severity code, mesg ID, & text
STP8	8	only facility name
STP9	9	facility name & text
STP10	10	facility name & mesg ID
STP11	11	facility name, mesg ID, & text
STP12	12	facility name & severity code
STP13	13	facility name, severity code, & text
STP14	14	facility name, severity code, & mesg ID
STP15	15	every thing

```

02BB'CF 0031'CF DE 003D 318      MOVAL  W^GETMSG,W^SERV_NAME      ; set service name
56 0141'CF DE 0044 319      MOVAL  W^MSGTBL,R6              ; set table pointer
57 01      DO 0049 320      MOVL   #1,R7                   ; set initial flag value
                                GETMSG_LOOP:
03FA'CF 86  DO 004C 322      MOVL   (R6)+,W^MSG              ; get string pointer from table
00      DD 0051 323      PUSHL  #0                       ; save a dummy parameter
019B'CF 01  FB 0053 324      CALLS  #1,W^REG_SAVE            ; save a register snapshot
                                0058 325      $GETMSG_S MSGID=W^MSGID,-
                                0058 326      MSGLEN=W^MSGLEN,-
                                0058 327      BUFADR=W^BUFADR,-
                                0058 328      FLAGS=R7,-
                                0058 329      OUTADR=W^OUTADR
                                0071 330      FAIL_CHECK SSS_NORMAL      ; check for success
                                PUSHL  #SS$NORMAL
00000000'8F DD 0071      CALLS  #1,W^REG_CHECK
01A5'CF 01  FB 0077      CALLS  #0,W^CHECK_RESULTS      ; check returned message
0286'CF 00  FB 007C 331      MOVL   R7,W^GETMSG$FLAGS      ; set flag value
0079'CF 57  DO 0081 332      PUSHL  #0                       ; save a dummy
019B'CF 01  FB 0088 334      CALLS  #1,W^REG_SAVE            ; save a register snapshot
                                008D 335      $GETMSG G W^GET
                                0096 336      FAIL_CHECK SSS_NORMAL      ; check for success
                                PUSHL  #SS$NORMAL
00000000'8F DD 0096      CALLS  #1,W^REG_CHECK
01A5'CF 01  FB 009C      CALLS  #0,W^CHECK_RESULTS      ; check returned message
0286'CF 00  FB 00A1 337      INCL  W^CURRENT_IC           ; bump the test number
0004'CF D6 00A6 338      AOBLEQ #14,R7,B^GETMSG_LOOP ; bump the flag & do it again
9E 57 0E F3 00AA 339      STEP=STEP+15
0000000F 00AE 340

```

```

00AE 342 .SBTTL PUTMSG TESTS
00AE 343 :+
00AE 344 :
00AE 345 $PUTMSG tests
00AE 346 :
00AE 347 The tests are executed if the following order:
00AE 348 :
00AE 349 STP # FLAG value message content
00AE 350 -----
00AE 351 STP17 0 no message
00AE 352 STP18 1 only text
00AE 353 STP19 2 only mesg ID
00AE 354 STP20 3 text and mesg ID
00AE 355 STP21 4 only severity code
00AE 356 STP22 5 severity code & text
00AE 357 STP23 6 severity code & mesg ID
00AE 358 STP24 7 severity code, mesg ID, & text
00AE 359 STP25 8 only facility name
00AE 360 STP26 9 facility name & text
00AE 361 STP27 10 facility name & mesg ID
00AE 362 STP28 11 facility name, mesg ID, & text
00AE 363 STP29 12 facility name & severity code
00AE 364 STP30 13 facility name, severity code, & text
00AE 365 STP31 14 facility name, severity code, & mesg ID
00AE 366 STP32 15 every thing
00AE 367 :
00AE 368 :
00AE 369 :-
00AE 370 NEXT_TEST

```

```

00AE STP16:
00AE 0004'CF 10 DO 00AE MOVL #16,W^CURRENT_TC
00AE 00 00 DD 00B3 PUSHL #0
019B'CF 01 FB 00B5 CALLS #1,W^REG_SAVE
02BB'CF 0038'CF DE 00BA 371 MOVAL W^PUTMSG,W^SERV_NAME ; set service name
56 017D'CF DE 00C1 372 MOVAL W^MSGTBL1,R6 ; set table pointer
57 D4 00C6 373 CLRL R7 ; set initial flag value
03FA'CF 86 DO 00C8 374 PUTMSG_LOOP:
03E9'CF 57 BO 00CD 375 MOVL (R6)+,W^MSG ; get string pointer from table
019B'CF 01 FB 00D4 376 MOVW R7,W^MSGVEC1+10 ; set a new flag value
00D9 377 PUSHL #0 ; save a dummy parameter
00D9 378 CALLS #1,W^REG_SAVE ; save a register snapshot
00EC 379 $PUTMSG_S MSGVEC=W^MSGVEC1,-
00EC 380 ACTRTN=W^ACT ; try it
00EC 381 FAIL_CHECK SS$NORMAL ; check for success
00000000'8F DD 00EC
01A5'CF 01 FB 00F2 PUSHL #SS$NORMAL
019B'CF 01 FB 00F7 382 CALLS #1,W^REG_CHECK ; save a dummy
00FE 383 CALLS #1,W^REG_SAVE ; save a register snapshot
C107 384 $PUTMSG G W^PUT ; try G form
00000000'8F DD 0107 385 FAIL_CHECK SS$NORMAL ; check for success
01A5'CF 01 FB 010D CALLS #1,W^REG_CHECK
0004'CF D6 0112 386 INCL W^CURRENT_TC ; bump the test number
AE 57 OF F3 0116 387 AOBLEQ #15,R7,B^PUTMSG_LOOP ; bump the flag & do it again
011A 388 :+
011A 389 :

```

```

011A 390 : test the facility name parameter
011A 391 :-
011A 392 :-
011A 393 :
                                NEXT_TEST
011A
011A STP17:
00C4'CF 11 DD 011A          MOVL #17,W^CURRENT_TC
                                PUSHL #0
019B'CF 01 DD 011F          PUSHL #0
03FA'CF 86 DD 0121          CALLS #1,W^REG_SAVE
                                (R6)+,W^MSG ; get string pointer from table
019B'CF 01 DD 0126          MOVL #0 ; save a dummy
                                CALLS #1,W^REG_SAVE ; save a reg snapshot
                                $PUTMSG_S MSGVEC=W^MSGVEC1,-
                                ACTRTN=W^ACT,-
                                FACNAM=W^TEST ; try the _S form
                                FAIL_CHECK SSS_NORMAL ; check for success
                                PUSHL #SS$NORMAL
00000000'8F DD 0147          CALLS #1,W^REG_CHECK
01A5'CF 01 FB 014D          PUSHL #0 ; save a dummy
                                DE 0152          401 ; save a reg snapshot
019B'CF 01 FB 0154          402 ; set facility name adr
008D'CF 04FB'CF DE 0159          403 ; try the _G form
                                MOVAL W^TEST,W^PUT+PUTMSG$FACNAM ; check for success
                                $PUTMSG G W^PUT
                                FAIL_CHECK SSS_NORMAL
                                PUSHL #SS$NORMAL
                                CALLS #1,W^REG_CHECK
                                TEST_END
                                PUSHL W^TMD_ADDR
                                PUSHL W^TMN_ADDR
                                PUSHL #2
                                PUSHL W^MOD_MSG_CODE
                                CALLS #$$T1,G^LIB$SIGNAL
                                INSV #1,#STSSV_INHIB_MSG,#1,W^MOD_MSG_CODE
                                PUSHL W^MOD_MSG_CODE
                                CALLS #1,G^SYS$EXIT
0160 404
0169 405
016F
0174 406
0174
0178
017C
017E
0044'CF DD 017E
00000000'GF 04 FB 0182
0044'CF 01 1C 01 FO 0189
0044'CF DD 0190
00000000'GF 01 FB 0194

```



```

019B 408 .SBTTL REG_SAVE
019B 409 :++
019B 410 : FUNCTIONAL DESCRIPTION:
019B 411 : Subroutine to save R2-R11 in the register save location.
019B 412 :
019B 413 : CALLING SEQUENCE:
019B 414 : PUSHL #0 ; save a dummy parameter
019B 415 : CALLS #1,W^REG_SAVE ; save R2-R11
019B 416 :
019B 417 : INPUT PARAMETERS:
019B 418 : NONE
019B 419 :
019B 420 : OUTPUT PARAMETERS:
019B 421 : NONE
019B 422 :
019B 423 :--
019B 424
019B 425 REG_SAVE:
019B 426 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0008'CF 14 AD 28 OFFC 019D 427 MOV C3 #4*10,^X14(FP),W^REG_SAVE_AREA ; save the registers in the program
01A4 428 RET
01A5 429 .SBTTL REG_CHECK
01A5 430 :++
01A5 431 : FUNCTIONAL DESCRIPTION:
01A5 432 : Subroutine to test R0 & R2-R11 for proper content after a service
01A5 433 : execution. A snapshot is taken by the REG_SAVE routine at the
01A5 434 : beginning of each step and this routine is executed after the
01A5 435 : services have been executed.
01A5 436 :
01A5 437 : CALLING SEQUENCE:
01A5 438 : PUSHL #SS$ XXXXXX ; push expected R0 contents
01A5 439 : CALLS #1,W^REG_CHECK ; execute this routine
01A5 440 :
01A5 441 : INPUT PARAMETERS:
01A5 442 : expected R0 contents on the stack
01A5 443 :
01A5 444 : OUTPUT PARAMETERS:
01A5 445 : possible error messages printed using $PUTMSG
01A5 446 :
01A5 447 :--
01A5 448
01A5 449 REG_CHECK:
50 04 AC D1 OFFC 01A5 450 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
01A7 451 CMPL 4(AP),R0 ; is this the right fail code?
01AB 452 BEQL 10$ ; br if yes
01AD 453 PUSHL R0 ; push received data
01AF 454 PUSHL 4(AP) ; push expected data
01E7'CF 03 FB 01B2 455 PUSHAL W^EXP ; push the string variable
01B6 456 CALLS #3,W^PRINT_FAIL ; print the error message
01BB 457 10$:
0008'CF 14 AD 28 29 01BB 458 CMPC3 #4*10,^X14(FP),W^REG_SAVE_AREA ; check all but R0
56 53 00000008'BF C3 01C2 459 BEQL 20$ ; br if O.K.
56 56 04 C6 01CC 460 SUBL3 #REG_SAVE_AREA,R3,R6 ; calculate the register number
7E 56 02 81 01CF 461 DIVL2 #4,R6
51 03 CA 01D3 462 ADDB3 #^X2,R6,-(SP) ; set number past R0-R1 and save
53 03 CA 01D6 463 BICL2 #3,R1 ; backup to register boundrys
464 BICL2 #3,R3

```

SATSSS10
V04-000

- SATS SYSTEM SERVICE TESTS (SUCC S.C.)
REG_CHECK

N 7

16-SEP-1984 01:43:51
5-SEP-1984 04:22:41

VAX/VMS Macro V04-00
[UETP.SRC]SATSSS10.MAR;1

Page 15
(1)

	61	DD	01D9	465		PUSHL	(R1)	
	63	DD	01DB	466		PUSHL	(R3)	
0095	CF	DF	01DD	467		PUSHAL	W^REG	
01E7	CF	FB	01E1	468		CALLS	#4,W^FRINT_FAIL	
	04		01E6	469	20\$:			
		04	01E6	470		RET		

: push received data
: push expected data
: set string pntr param.
: print the error message

UE
V04
53
40
50
41
4E
21
2A
65
72
6E
63

```

01E7 472 .SBTTL PRINT_FAIL
01E7 473 :++
01E7 474 : FUNCTIONAL DESCRIPTION:
01E7 475 : Subroutine to report failures using $PUTMSG
01E7 476 :
01E7 477 : CALLING SEQUENCE:
01E7 478 : Mode #1          PUSHL EXPECTED Mode #2          PUSHL REG NUMBER
01E7 479 :                   PUSHL RECEIVED                   PUSHL EXPECTED
01E7 480 :                   PUSHAL STRING VAR                   PUSHL RECEIVED
01E7 481 :                   CALLS #3,W^PRINT_FAIL                PUSHAL STRING VAR
01E7 482 :                                                           CALLS #4,W^PRINT_FAIL
01E7 483 : INPUT PARAMETERS:
01E7 484 : Listed above
01E7 485 :
01E7 486 : OUTPUT PARAMETERS:
01E7 487 : an error message is printed using $PUTMSG
01E7 488 :
01E7 489 :--
01E7 490
01E7 491 PRINT_FAIL:
003C 01E7 492 .WORD ^M<R2,R3,R4,R5>
01E9 493 $FAO S W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
020A 494 $PUTMSG_S W^MSGVEC ; print the message
04 6C 91 021B 495 CMPB (AP),#4 ; is this a register message?
21 13 021E 496 BEQL 10$ ; br if yes
25 11 0220 497 $FAO_S W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
023F 498 BRB 20$ ; goto output message
0241 499 10$:
0241 500 $FAO_S W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
0266 501 20$:
0266 502
0266 503 $PUTMSG_S W^MSGVEC ; print the message
0044'CF 03 002A'CF 02 DE 0277 504 MOVAL W^TEST_MOD_FAIL,W^TMD_ADDR ; set failure message address
0044'CF 03 00 02 FO 027E 505 INSV #ERROR,#0,#3,W^MOD_MSG_CODE ; set severity code
0285 506 RET

```

UE
VO
20
6C
72
61
4E
69
20
2E
61
72
20
41
66
69
61
44
20
54
64
41
66

```

0286 508 .SBTTL CHECK_RESULTS
0286 509 :++
0286 510 : FUNCTIONAL DESCRIPTION:
0286 511 : Routine to check message content, message length, and OUTADR.
0286 512 :
0286 513 : CALLING SEQUENCE:
0286 514 : CALLS #0,W^CHECK_RESULTS ; check returned message
0286 515 :
0286 516 : INPUT PARAMETERS:
0286 517 : MSG = pointer to correct resultant counted string
0286 518 : MSGBUF = returned string
0286 519 : OUTADR = outadr parameter returned by the service
0286 520 : MSGLEN = contains returned message length
0286 521 :
0286 522 : OUTPUT PARAMETERS:
0286 523 : NONE
0286 524 :
0286 525 :--
0286 526
0286 527 CHECK_RESULTS:
0286 528 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
58 03FA'DF 9A 0288 MOVZBL @W^MSG,R8 ; get the length
58 02BF'CF 91 028D CMPB W^MSGLEN,R8 ; is the length OK?
02BF'CF 0F 13 0292 BEQL 10$ ; br if yes
02BF'CF 58 DD 0294 PUSHL W^MSGLEN ; push received
0111'CF DF 0298 PUSHL R8 ; push expected
FF44 CF 03 FB 029A 534 PUSHAL W^LEN ; push string variable
02A3 535 CALLS #3,W^PRINT_FAIL ; print the failure
0000100 8F 02C3'CF D1 02A3 536 10$:
02C3'CF 13 13 02AC 537 CMPL W^OUTADR,#^X100 ; is OUTADR param OK?
02C3'CF DD 02AE 538 BEQL 20$ ; br if OK
0000100 8F DD 02B2 539 PUSHL W^OUTADR ; push received
011F'CF DF 02B8 540 PUSHL #^X100 ; push expected
FF26 CF 03 FB 02BC 541 PUSHAL W^OUT ; push string variable
02CF'CF 59 03FA'CF D0 02C1 542 20$:
01 A9 58 29 02C6 543 MOVL W^MSG,R9 ; get message pointer
03 12 02CD 544 CMPC3 R8,B^1(R9),W^BUFADR+8 ; check the string
0082 31 02CF 545 BNEQ 25$ ; br if not OK
02D2 546 BRW 30$ ; BR if OK
03DB'CF 01 A9 DE 02D2 547 25$:
03D7'CF 58 D0 02D8 548 MOVAL B^1(R9),W^GOODDATA+4 ; fill in good data desc.
03CF'CF 02BF'CF D0 02DD 549 MOVL R8,W^GOODDATA
02E4 550 MOVL W^MSGLEN,W^BADDATA ; fill in bad data desc.
02E4 551 $FAO_S W^CS1,W^MESSAGEL,W^MSGL,-
02E4 552 #TEST_MOD_NAME,W^SERV_NAME,-
0305 553 W^CURRENT_TC ; form 1st line and
0316 554 $PUTMSG S W^MSGVEC ; print it
0316 555 $FAO_S W^CS4,W^MESSAGEL,W^MSGL,-
0335 556 #GOODDATA,#BADDATA ; form 2nd line and
004C'CF 002A'CF DE 0335 557 $PUTMSG S W^MSGVEC ; print it
0044'CF 03 00 02 FO 0346 558 MOVAL W^TEST_MOD_FAIL,W^TMD_ADDR ; set failure message address
0354 559 INSV #ERROR,#0,#?,W^MOD_MSG_CODE ; set severity code
0354 561 30$:
04 0354 562 RET

```

64
3A

76
65
64
20
20

20
53
20
74

20
65
6F
62

65
72
72
75

6C
20
67
69
20

20
40
73
64

20
40
63
65

```

0355 564 .SBTTL ACT
0355 565 :++
0355 566 : FUNCTIONAL DESCRIPTION:
0355 567 : Routine to take action on a PUTMSG call. The message is checked
0355 568 : for proper content and printing is disabled.
0355 569 :
0355 570 : CALLING SEQUENCE:
0355 571 : Entered via ACTRTN parameter specified with PUTMSG.
0355 572 :
0355 573 : INPUT PARAMETERS:
0355 574 : MSG = pointer to correct resultant counted string
0355 575 : MSGBUF = returned string
0355 576 :
0355 577 : OUTPUT PARAMETERS:
0355 578 : NONE
0355 579 :
0355 580 :--
0355 581
0355 582 ACT:
0355 583 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0357 584 MOVL 4(AP),R6 ; get the desc. address
0358 585 MOVL (R6),W^MSGLEN ; set length for CHECK_RESULTS
0360 586 MOVL 4(R6),R7 ; get string pointer
0364 587 MOVCL (R6),(R7),W^BUFADR+8 ; get string for CHECK_RESULTS
036A 588 CALLS #0,W^CHECK_RESULTS ; check the PUTMSG results
036F 589 MOVL #SS$ _ABORT,R0 ; don't really print it
0376 590 RET

```

56	04	AC	DO	0357	584
02BF	'CF	66	DO	0358	585
57	04	A6	DO	0360	586
02CF	'CF	67	28	0364	587
FF17	CF	00	FB	036A	588
50	00000000	'8F	DO	036F	589
			04	0376	590

```

0377 592 MOD_MSG_PRINT:
0377 593 :
0377 594 : *****
0377 595 : *
0377 596 : * PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES *
0377 597 : * (USING THE PUTMSG MACRO). *
0377 598 : *
0377 599 : *****
0377 600 :
05 0377 601 PUTMSG <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> ; PRINT MSG
0392 602 RSB ; ... AND RETURN TO CALLER
0393 603 :
0393 604 CHMRTN:
0393 605 : *****
0393 606 : *
0393 607 : * CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER *
0393 608 : * A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED *
0393 609 : * BY THE MODE MACRO ('TO' OPTION). IT MERELY DOES *
0393 610 : * A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS *
0393 611 : * THE EFFECT OF RETURNING TO THE END OF THE MODE *
0393 612 : * MACRO EXPANSION. *
0393 613 : *
0393 614 : *****
0393 615 :
0000059'FF 0000 0393 616 .WORD 0 ; ENTRY MASK
17 0395 617 JMP @CHM_CONT ; RETURN TO MODE MACRO IN NEW MODE
039B 618 :
039B 619 : * RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
039B 620 :
039B 621 .END SATSSS10

```

SATSSS10
Symbol table

\$\$ARGS	= 00000004		MOD_MSG_PRINT	00000377	R	04	
\$\$T1	= 00000004		MSG	000003FA	R	03	
\$\$T2	= 00000005		MSGID	0000012D	R	02	
ACT	00000355	R	04	MSG_L	000000AB	R	03
BADATA	000003CF	R	03	MSGLEN	000002BF	R	03
BUF	000000B3	R	03	MSGTBL	00000141	R	02
BUFADR	000002C7	R	03	MSGTBL1	0000017D	R	02
CHECK_RESULTS	00000286	R	04	MSGVEC	00000131	R	02
CHMRTN	00000393	R	04	MSGVEC1	000003DF	R	03
CHM_CONT	00000059	R	03	OUT	0000011F	R	02
CS1	0000003F	R	02	OUTADR	000002C3	R	03
CS2	00000071	R	02	PRINT_FAIL	000001E7	R	04
CS3	0000009E	R	02	PRIVMASK	00000051	R	03
CS4	000000D1	R	02	PRVPR	00000050	R	03
CURRENT_TC	00000004	R	03	PUT	00000081	R	03
DEV_NAM	000003EF	R	03	PUTMSG	00000038	R	02
ERRR	= 00000002		PUTMSG\$ACTPRM	= 00000010			
EXP	00000103	R	02	PUTMSG\$ACTRTN	= 00000008		
GET	00000069	R	03	PUTMSG\$FACNAM	= 0000000C		
GETMSG	00000031	R	02	PUTMSG\$MSGVEC	= 00000004		
GETMSG\$BUFADR	= 0000000C		PUTMSG\$NARGS	= 00000004			
GETMSG\$FLAGS	= 00000010		PUTMSG\$LOOP	000000C8	R	04	
GETMSG\$MSGID	= 00000004		REG	00000095	R	03	
GETMSG\$MSGLEN	= 00000008		REGNUM	000000A7	R	03	
GETMSG\$NARGS	= 00000005		REG_CHECK	000001A5	R	04	
GETMSG\$OUTADR	= 00000014		REG_SAVE	0000019B	R	04	
GETMSG\$LOOP	0000004C	R	04	REG_SAVE_AREA	00000008	R	03
GOODATA	000003D7	R	03	RETADR	0000005D	R	03
INFO	= 00000003		SATSSS10	00000000	RG	04	
LEN	00000111	R	02	SERV_NAME	000002BB	R	03
LIB\$SIGNAL	*****	X	04	SEVERE	= 00000004		
M1	000001C1	R	02	SHR\$K_SHRDEF	= 00000001		
M10	000002AD	R	02	SHR\$TEXT	= 00001130		
M11	000002B7	R	02	SS\$ABORT	*****	X	04
M12	000002E8	R	02	SS\$NORMAL	*****	X	04
M13	000002F0	R	02	STATUS	00000065	R	03
M14	0000031F	R	02	STEP	= 00000011		
M15	0000032B	R	02	STPO	0000003D	R	04
M16	0000035E	R	02	STP16	000000AE	R	04
M17	00000384	R	02	STP17	0000011A	R	04
M18	000003B0	R	02	ST\$V_INHIB_MSG	= 0000001C		
M19	000003DA	R	02	SUCCESS	= 00000001		
M2	000001E7	R	02	SYS\$EXIT	*****	GX	04
M20	00000408	R	02	SYS\$FAO	*****	X	04
M21	00000435	R	02	SYS\$GETMSG	*****	GX	04
M22	00000466	R	02	SYS\$HIBER	*****	GX	04
M23	00000495	R	02	SYS\$PUTMSG	*****	GX	04
M24	000004C8	R	02	SYS\$SETPRN	*****	GX	04
M3	000001EC	R	02	SYS\$WAKE	*****	GX	04
M4	00000218	R	02	TEST	000004FB	R	02
M5	0000021B	R	02	TEST_MOD_BEGIN	00000019	R	02
M6	00000245	R	02	TEST_MOD_FAIL	0000002A	R	02
M7	0000024C	R	02	TEST_MOD_NAME	00000000	R	02
M8	0000027A	R	02	TEST_MOD_NAME_D	00000009	R	02
M9	00000280	R	02	TEST_MOD_SUCC	0000001F	R	02
MESSAGEL	000002B3	R	03	TMD_ADDR	0000004C	R	03
MOD_MSG_CODE	00000044	R	03	TMN_ADDR	00000048	R	03

SATSSS10
Symbol table

TPID = 00000000 R 03
UETPS_DDB = 007480E1
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	00000507 (1287.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
RWDATA	000003FE (1022.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
SATSSS10	00000398 (923.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.10	00:00:00.41
Command processing	107	00:00:00.68	00:00:04.94
Pass 1	263	00:00:06.48	00:00:16.24
Symbol table sort	0	00:00:00.43	00:00:00.54
Pass 2	145	00:00:02.18	00:00:04.37
Symbol table output	15	00:00:00.11	00:00:00.32
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	563	00:00:10.02	00:00:26.86

The working set limit was 1350 pages.
41621 bytes (82 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 349 non-local and 8 local symbols.
621 source lines were read in Pass 1, producing 27 object records in Pass 2.
38 pages of virtual memory were used to define 34 macros.

! Macro library statistics !

Macro library name	Macros defined
-\$255\$DUA28:[UETP.OBJ]UETP.MLB;1	10
-\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	21
TOTALS (all libraries)	31

508 GETS were required to define 31 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:SATSSS10/OBJ=OBJ\$:SATSSS10 MSRC\$:SATSSS10/UPDATE=(ENH\$:SATSSS10)+EXECML\$/LIB+LIB\$:UETP/LIB

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

The image displays a grid of 100 terminal windows, arranged in 10 rows and 10 columns. Each window contains text-based output from a VAX/VMS system. The outputs vary significantly, including:

- System status and configuration information.
- Large blocks of error messages, such as "NO FILE" and "NO DEVICE".
- Diagnostic messages like "UNRECOGNIZED COMMAND" and "INVALID COMMAND".
- System logs and command histories.
- Specific program outputs, including "SATSSSF08 LIS", "SATSSSF09 LIS", "NETCOMS00 LIS", "NETDISK00 LIS", "SATSSSF10 LIS", and "NETMPF00 LIS".
- Memory dumps and stack traces.
- File listings and directory structures.

The text is rendered in a monospaced font, typical of early computer terminals. The overall appearance is that of a multi-user system environment with various users and processes running simultaneously.