

```
UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP  PPP
UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP  PPP
UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP  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  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEEEEEEEEEEEEEE  TTT      TTT      PPPPPPPPPPPP  PPP
UUU      UUU  EEEEEEEEEEEEEEE  TTT      TTT      PPPPPPPPPPPP  PPP
UUU      UUU  EEEEEEEEEEEEEEE  TTT      TTT      PPPPPPPPPPPP  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
UUUUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT      TTT      PPP
UUUUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT      TTT      PPP
UUUUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT      TTT      PPP
```

```
SSSSSSSS  AAAAAA  TTTTTTTTTT  SSSSSSSS  SSSSSSSS  SSSSSSSS  000000  999999
SSSSSSSS  AAAAAA  TTTTTTTTTT  SSSSSSSS  SSSSSSSS  SSSSSSSS  000000  999999
SS        AA      AA      TT        SS        SS        SS        00      00  99      99
SS        AA      AA      TT        SS        SS        SS        00      00  99      99
SS        AA      AA      TT        SS        SS        SS        00      00  99      99
SS        AA      AA      TT        SS        SS        SS        00      00  99      99
SSSSSSS  AA      AA      TT        SSSSSS  SSSSSS  SSSSSS  00  00  00  99999999
SSSSSSS  AA      AA      TT        SSSSSS  SSSSSS  SSSSSS  00  00  00  99999999
SS        AA      AA      TT        SS        SS        SS        0000  00  99
SS        AA      AA      TT        SS        SS        SS        0000  00  99
SS        AA      AA      TT        SS        SS        SS        0000  00  99
SSSSSSSS  AA      AA      TT        SSSSSSSS  SSSSSSSS  SSSSSSSS  000000  999999
SSSSSSSS  AA      AA      TT        SSSSSSSS  SSSSSSSS  SSSSSSSS  000000  999999

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
```


(1)	52	DECLARATIONS
(1)	265	R/W PSECT
(1)	344	SATSSS09
(1)	393	FAO TESTS
(2)	524	FAOL TESTS
(2)	643	BUF_CHECK
(2)	694	REG_SAVE
(2)	715	REG_CHECK
(2)	757	PRINT_FAIL
(2)	804	MODE_ID

```
0000 1 .TITLE SATSSS09 - 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 SATSSS09 module tests the execution of the following
0000 33 : VMS system services:
0000 34
0000 35 : $FAO
0000 36 : $FAOL
0000 37
0000 38
0000 39 : ENVIRONMENT: User and Kernal 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: SEP, 1979
0000 44
0000 45 : MODIFIED BY:
0000 46
0000 47 : V03-001 LDJ0001 Larry D. Jones, 17-Sep-1980
0000 48 : Modified to conform to new build command procedures.
0000 49 : **
0000 50 : --
```



```
0000 52 .SBTTL DECLARATIONS
0000 53 :
0000 54 : MACRO LIBRARY CALLS
0000 55 :
0000 56 .LIBRARY /SYSS$LIBRARY:STARLET.MLB/
0000 57 $SHR MESSAGES UETP,116,<<TEXT,INFO>> ; UETP$_TEXT definition
0000 58 $STSDEF
0000 59 $UETPDEF ; UETP message definitions
0000 60 :
0000 61 : Equated symbols
0000 62 :
00000000 0000 63 WARNING = 0 ; warning severity value for msgs
00000001 0000 64 SUCCESS = 1 ; success
00000002 0000 65 ERROR = 2 ; error
00000003 0000 66 INFO = 3 ; information
00000004 0000 67 SEVERE = 4 ; fatal
0000 68 :
00000000 0000 69 NUL = 00
0000000D 0000 70 CR = 13
0000000A 0000 71 LF = 10
00000009 0000 72 TAB = 09
0000000C 0000 73 FF = 12
0000 74 :
0000 75 : MACROS
0000 76 :
0000 77 .MACRO FAONTST CS,RS,RSL,TYPE
0000 78 .IF NOT_EQUAL $LENGTH(TYPE)
0000 79 $FAOL_S CTRSTR = W^CS,-
0000 80 OUTLEN = W^LEN,-
0000 81 OUTBUF = W^BUFFER,-
0000 82 PRMLST = W^PARAM ; try _S
0000 83 .IF FALSE
0000 84 MOVAL W^PL6+8,R10 ; set the arg pntr
0000 85 PUSHL #0 ; push a dummy parameter
0000 86 CALLS #1,W^REG SAVE ; save a reg snapshot
0000 87 $FAO_S CTRSTR = W^CS,-
0000 88 OUTLEN = W^LEN,-
0000 89 OUTBUF = W^BUFFER,-
0000 90 P1 = -(R10),-
0000 91 P2 = -(R10),-
0000 92 P3 = -(R10),-
0000 93 P4 = -(R10),-
0000 94 P5 = -(R10),-
0000 95 P6 = -(R10) ; try _S
0000 96 MOVAL W^PL6+8,R10 ; reset the parameter pntr
0000 97 .ENDC
0000 98 FAIL_CHECK $$$_NORMAL ; check success
0000 99 .LIST MEB
0000 100 MOVAL W^RS,R7 ; set good data adr
0000 101 MOVL #RSL,R8 ; set byte count
0000 102 CALLS #0,W^BUF_CHECK ; check results
0000 103 MOVAL W^CS,W^FAO'TYPE'P+FAO$_CTRSTR ; set new cntrl str
0000 104 PUSHL #0 ; push a dummy parameter
0000 105 CALLS #1,W^REG SAVE ; save a reg snapshot
0000 106 $FAO'TYPE'G W^FAO'TYPE'P ; try _G
0000 107 FAIL_CHECK $$$_NORMAL ; check success
0000 108 .LIST MEB
```


SATSSS09
V04-000

- SATS SYSTEM SERVICE TESTS (SUCC^I S.C.⁴) 16-SEP-1984 01:43:09 VAX/VMS Macro V04-00 Page 3
DECLARATIONS 5-SEP-1984 04:22:34 [UETP.SRC]SATSSS09.MAR;1 (1)

0000 109
0000 110
0030 111

CALLS #0,W^BUF_CHECK ; check results
.NLIST MEB
.ENDM FAONTST


```
00000009 113 .PSECT RODATA,RD,NOWRT,NOEXE, LONG
0000 114
0000 115 TEST_MOD_NAME:
39 30 53 53 53 54 41 53 00' 0000 116 .ASCIC /SATSSS09/ ; needed for SATSMS message
08 0000
0009 117 TEST_MOD_NAME_D:
53 53 53 54 41 53 00000011'010E0000' 0009 118 .ASCID /SATSSS09/ ; module name
39 30 0017
0019 119 TEST_MOD_BEGIN:
6E 75 67 65 62 00' 0019 120 .ASCIC /begun/ ; start end and fail messages
05 0019
001F 121 TEST_MOD_SUCC:
6C 75 66 73 73 65 63 63 75 73 00' 001F 122 .ASCIC /successful/
0A 001F
002A 123 TEST_MOD_FAIL:
64 65 6C 69 61 66 00' 002A 124 .ASCIC /failed/
06 002A
0031 125 CS1: ; failure messages
21 20 74 73 65 54 00000039'010E0000' 0031 126 .ASCID \Test !AC service name !AC step !UL failed.\
6E 20 65 63 69 76 72 65 73 20 43 41 003F
70 65 74 73 20 43 41 21 20 65 6D 61 004B
2E 64 65 6C 69 61 66 20 4C 55 21 20 0057
0063 127 CS2:
74 63 65 70 78 45 00000068'010E0000' 0063 128 .ASCID \Expected !AS = !XL received !AS = !XL\
4C 58 21 20 3D 20 53 41 21 20 64 65 0071
41 21 20 64 65 76 69 65 63 65 72 20 007D
4C 58 21 20 3D 20 53 0089
0090 129 CS3:
74 63 65 70 78 45 00000098'010E0000' 0090 130 .ASCID \Expected !AS!UB = !XL received !AS!UB = !XL\
20 3D 20 42 55 21 53 41 21 20 64 65 009E
64 65 76 69 65 63 65 72 20 4C 58 21 00AA
58 21 20 3D 20 42 55 21 53 41 21 20 00B6
4C 00C2
00C3 131 CS4:
72 69 75 71 65 52 000000CB'010E0000' 00C3 132 .ASCID \Required channel not received.\
6E 20 6C 65 6E 6E 61 68 63 20 64 65 00D1
2E 64 65 76 69 65 63 65 72 20 74 6F 00DD
00E9 133 CS5:
77 20 65 64 6F 4D 00C000F1'010E0000' 00E9 134 .ASCID \Mode was !AS.\
2E 53 41 21 20 73 61 00F7
00FE 135 CS6:
74 63 65 70 78 45 00000106'010E0000' 00FE 136 .ASCID \Expected byte offset !UB(10) = !XB(16) received !XB(16).\
73 66 66 6F 20 65 74 79 62 20 64 65 010C
3D 20 29 30 31 28 42 55 21 20 74 65 0118
63 65 72 20 29 36 31 28 42 58 21 20 0124
36 31 28 42 58 21 20 64 65 76 69 65 0130
2E 29 013C
013E 137 UM: ; mode message
72 65 73 75 00000146'010E0000' 013E 138 .ASCID \user\
014A 139 FA0:
4F 41 46 00' 014A 140 .ASCIC /FA0/ ; service names
03 014A
014E 141 FAOL:
4C 4F 41 46 00' 014E 142 .ASCIC /FAOL/
04 014E
0153 143 EXP:
73 75 74 61 74 73 0000015B'010E0000' 0153 144 .ASCID \status\
```



```
00000003 0161 145 MSGVEC:
00741133 0161 146 .LONG 3 ; PUTMSG message vector
00000001 0165 147 .LONG UETPS_TEXT
0000029B' 0169 148 .LONG 1
0171 149 .ADDRESS MESSAGEL
54 53 45 54 00' 0171 150 STR1: ; string variables
04 0171 151 .ASCII /TEST/
00 54 53 45 54 0176 152 STR2:
00000005 0176 153 .ASCII /TEST/<NUL>
017B 154 STR2L=.-STR2
54 53 45 54 00000183'010E0000' 017B 155 STR3:
017B 156 .ASCII /TEST/
0187 157 PL1: ; parameter list
00000171' 0187 158 .ADDRESS STR1
00000171' 018B 159 .ADDRESS STR1
018F 160 PL2:
00000005 018F 161 .LONG STR2L
00000176' 0193 162 .ADDRESS STR2
00000005 0197 163 .LONG STR2L
00000176' 019B 164 .ADDRESS STR2
019F 165 PL3:
0000017B' 019F 166 .ADDRESS STR3
0000017B' 01A3 167 .ADDRESS STR3
01A7 168 PL4:
00000000 01A7 169 .LONG 0 ; smallest byte value
000000FF 01AB 170 .LONG ^XFF ; largest byte value
01AF 171 PL5:
00000000 01AF 172 .LONG 0 ; smallest word value
0000FFFF 01B3 173 .LONG ^XFFFF ; largest word value
01B7 174 PL6:
00000000 01B7 175 .LONG 0 ; smallest long word value
FFFFFFFF 01BB 176 .LONG -1 ; largest long word value
01BF 177 PL7:
00000001 01BF 178 .LONG 1 ; a one
00000000 01C3 179 .LONG 0 ; a zero
01C7 180 ;
01C7 181 FCS1: ; control strings
00000096' 01C7 182 .LONG FCS1L
000001CF' 01CB 183 .ADDRESS .+4
01CF 184 .ASCII /A!ACB!-!2(AC)C!2(-)!3ACD!-!2(3AC)/
01DB 185
01E7
01F0 185 .ASCII /E!ADF!2(-)!2(AD)G!4(-)!3ADH!2(-)!2(3AD)!4(-)/
01FC
0208
0214
021C 186 .ASCII /I!AFJ!2(-)!2(AF)K!4(-)!3AFL!2(-)!2(3AF)/
0228
0234
0240
0243 187 .ASCII /M!ASN!-!2(AS)O!2(-)!3ASP!-!2(3AS)Q/
024F
025B
00000096 0265 188 FCS1L=.-FCS1-8
0265 189 FCS2:
00000064' 0265 190 .LONG FCS2L
```


42 4F 28 32 21 2D 21 42 42	0000026D'	0269	191	.ADDRESS .+4
44 42 4F 32 21 29 2D 28 32	4F 21 41	026D	192	.ASCII /A!OBB!-!2(OB)C!2(-)!2OBD!-!2(2OB)/
57 4F 28 32 21 2D 21 46 57	21 43 29	0279		
48 57 4F 32 21 2D 28 32 21	2D 21	0285		
57 4F 28 32 21 2D 21 46 57	4F 21 45	028E	193	.ASCII /E!OWF!-!2(OW)G!2(-)!3OWH!-!2(3OW)/
48 57 4F 32 21 2D 28 32 21	47 29	029A		
4C 4F 28 32 21 2D 21 4A 4C	4F 21 49	02A6		
4C 4C 4F 32 21 2D 28 32 21	4B 29	02AF	194	.ASCII /I!OLJ!-!2(OL)K!2(-)!3OLL!-!2(3OL)M/
4D 29 4C 4F 32 28 32	21 2D 21	02BB		
	00000064	02C7		
	00000064	02D1	195	FCS2L=-FCS2-8
		02D1	196	FCS3:
	00000064'	02D1	197	.LONG FCS3L
	000002D9'	02D5	198	.ADDRESS .+4
42 58 28 32 21 2D 21 42 42	58 21 41	02D9	199	.ASCII /A!XBB!-!2(XB)C!2(-)!1XBD!-!2(1XB)/
44 42 58 31 21 29 2D 28 32	21 43 29	02E5		
57 58 28 32 21 2D 21 46 57	58 21 45	02F1		
48 57 58 32 21 2D 28 32 21	47 29	02FA	200	.ASCII /E!XWF!-!2(XW)G!2(-)!3XWH!-!2(3XW)/
4C 58 28 32 21 2D 21 4A 4C	58 21 49	0306		
4C 4C 58 32 21 2D 28 32 21	4B 29	0312		
	00000064	031B	201	.ASCII /I!XLJ!-!2(XL)K!2(-)!3XLL!-!2(3XL)M/
		0327		
		0333		
	00000064	033D	202	FCS3L=-FCS3-8
		033D	203	FCS4:
	00000064'	033D	204	.LONG FCS4L
	00000345'	0341	205	.ADDRESS .+4
42 5A 28 32 21 2D 21 42 42	5A 21 41	0345	206	.ASCII /A!ZBB!-!2(ZB)C!2(-)!3ZBD!-!2(3ZB)/
44 42 5A 32 21 29 2D 28 32	21 43 29	0351		
57 5A 28 32 21 2D 21 46 57	5A 21 45	035D		
48 57 5A 32 21 2D 28 32 21	47 29	0366	207	.ASCII /E!ZWF!-!2(ZW)G!2(-)!3ZWH!-!2(3ZW)/
4C 5A 28 32 21 2D 21 4A 4C	5A 21 49	0372		
4C 4C 5A 32 21 2D 28 32 21	4B 29	037E		
	00000064	0387	208	.ASCII /I!ZLJ!-!2(ZL)K!2(-)!3ZLL!-!2(3ZL)M/
		0393		
		039F		
	00000064	03A9	209	FCS4L=-FCS4-8
		03A9	210	FCS5:
	00000064'	03A9	211	.LONG FCS5L
	000003B1'	03AD	212	.ADDRESS .+4
42 55 28 32 21 2D 21 42 42	55 21 41	03B1	213	.ASCII /A!UBB!-!2(UB)C!2(-)!3UBD!-!2(3UB)/
44 42 55 32 21 29 2D 28 32	21 43 29	03BD		
57 55 28 32 21 2D 21 46 57	55 21 45	03C9		
48 57 55 32 21 2D 28 32 21	47 29	03D2	214	.ASCII /E!UWF!-!2(UW)G!2(-)!3UWH!-!2(3UW)/
4C 55 28 32 21 2D 21 4A 4C	55 21 49	03DE		
4C 4C 55 32 21 2D 28 32 21	4B 29	03EA		
	00000064	03F3	215	.ASCII /I!ULJ!-!2(UL)K!2(-)!3ULL!-!2(3UL)M/
		03FF		
		040B		
	00000064	0415	216	FCS5L=-FCS5-8
		0415	217	FCS6:
	00000064'	0415	218	.LONG FCS6L
	0000041D'	0419	219	.ADDRESS .+4
42 53 28 32 21 2D 21 42 42	53 21 41	041D	220	.ASCII /A!SBB!-!2(SB)C!2(-)!3SBD!-!2(3SB)/
44 42 53 32 21 29 2D 28 32	21 43 29	0429		
57 53 28 32 21 2D 21 46 57	53 21 45	0435		
		043E	221	.ASCII /E!SWF!-!2(SW)G!2(-)!3SWH!-!2(3SW)/


```
48 57 53 33 21 29 2D 28 32 21 47 29 044A
29 57 53 33 28 32 21 2D 21 0456
4C 53 28 32 21 2D 21 4A 4C 53 21 49 045F 222 .ASCII /I!SLJ!-!2(SL)K!2(-)!3SLL!-!2(3SL)M/
4C 4C 53 33 21 29 2D 28 32 21 4B 29 046B
4D 29 4C 53 33 28 32 21 2D 21 0477
00000064 0481 223 FCS6L=-FCS6-8
0481 224 FCS7:
225 .ASCID \A!/B!_C!^D!!E!%SF!UB!-G!%SH!5*AI!#*B!-J!+!XBK\
5F 21 42 2F 21 41 00000489'010E0000' 0481
21 46 53 25 21 45 21 21 44 5E 21 43 048F
2A 35 21 48 53 25 21 47 2D 21 42 55 049B
21 2B 21 4A 2D 21 42 2A 23 21 49 41 04A7
4B 42 58 04B3
21 44 25 33 32 21 000004BE'010E0000' 04B6 226 FCS8:
54 25 31 31 21 2D 04B6 227 .ASCID /!23%D!-!11%T/
04C4 228 :
04CA 229 RS1:
04CA 230 .ASCII /ATESTBTESTTESTCTESDTESTES/
: resultant strings
45 54 54 53 45 54 42 54 53 45 54 41 04CA
45 54 53 45 54 44 53 45 54 43 54 53 04D6
53 04E2
00 54 53 45 54 46 00 54 53 45 54 45 04E3
45 54 48 53 45 54 47 00 54 53 45 54 04EF
53 45 54 53 04FB
2E 54 53 45 54 4A 2E 54 53 45 54 49 04FF 232 .ASCII /ITEST.JTEST.TEST.KTESLTESTES/
45 54 4C 53 45 54 4B 2E 54 53 45 54 050B
53 45 54 53 0517
45 54 54 53 45 54 4E 54 53 45 54 4D 051B 233 .ASCII /MTESTNTESTTESTOTESPTESTESQ/
45 54 53 45 54 50 53 45 54 4F 54 53 0527
51 53 0533
0000006B 0535 234 RS1L=-RS1
0535 235 RS2:
0541 236 .ASCII /A000B000377C00D0077/
0548 237 .ASCII /E000000F000000177777G000H000777/
0554
0560
0567 238 .ASCII /I00000000000J0000000000037777777777K000L000777M/
0573
057F
058B
00000061 0596 239 RS2L=-RS2
0596 240 RS3:
0596 241 .ASCII /A00B00FFC0D0F/
05A2
05A3 242 .ASCII /E0000F0000FFFFG00CH000FFF/
05AF
05BB
05BC 243 .ASCII /I00000000J00000000FFFFFFFFFFK000L000FFFFM/
05C8
05D4
05E0
0000004C 05E2 244 RS3L=-RS3
05E2 245 RS4:
05E2 246 .ASCII /A0B0255C000D000255/
05EE
05F4 247 .ASCII /E0F065535G00CH000***/
0600
```


Line	Address	Hex	ASCII
248	32 37 36 39 34 39 32 34 30 4A 30 49	0608	.ASCII /IOJ04294967295K000L000***M/
	2A 2A 30 30 30 4C 30 30 30 4B 35 39	0614	
		4D 2A 0620	
		00000040	
249		0622	RS4L=.-RS4
250	44 30 20 20 43 35 35 32 30 42 30 41	0622	
251		0622	RS5:
		062E	.ASCII /A0B0255C 0D 0255/
252	20 20 47 35 33 35 35 36 30 46 30 45	0634	.ASCII /E0F065535G 0H 0***/
		0640	
253	32 37 36 39 34 39 32 34 30 4A 30 49	0648	.ASCII /IOJ04294967295K 0L 0***M/
	2A 2A 30 20 20 4C 30 20 20 4B 35 39	0654	
		4D 2A 0660	
		00000040	
254		0662	RS5L=.-RS5
255	20 44 30 20 20 43 31 2D 30 42 30 41	0662	RS6:
256		066E	.ASCII /A0B0-1C 0D 0 -1/
		0673	
257	20 48 30 20 20 47 31 2D 30 46 30 45	0673	.ASCII /E0F0-1G 0H 0 -1/
		067F	
258	20 4C 30 20 20 4B 31 2D 30 4A 30 49	0684	.ASCII /IOJ0-1K 0L 0 -1M/
		0690	
		00000034	
259		0696	RS6L=.-RS6
260	46 53 45 21 44 0C 43 09 42 0A 0D 41	0696	RS7:
261	30 4A 42 49 41 41 41 41 41 48 47 31	06A2	.ASCII /A/<CR><LF>/B/<TAB>/C/<FF>/D!ESF1GHAAAAAIBJ00K/
		06AE	
		0000001A	
262		06B0	RS7L=.-RS7


```
06B0 264 ;
06B0 265 .SBTTL R/W PSECT
00000000 266 .PSECT RWDATA, RD, WRT, NOEXE, LONG
0000 267 ;
0000 268 IPID: ; PID for this process
00000000 0000 269 .LONG 0 ; ptr to current test case
00000000 0004 270 CURRENT_TC: ; put it on a long word boundry
00000000 0004 271 .LONG 0
0008 272 .ALIGN LONG
0008 273 REG_SAVE_AREA: ; register save area
00000044 0008 274 .BLKL 15
0044 275 MOD_MSG_CODE: ; test module message code for putmsg
007480D9 0044 276 .LONG UETP$_SATSMS
0048 277 TMN_ADDR:
00000000 0048 278 .ADDRESS TEST_MOD_NAME
004C 279 TMD_ADDR:
00000019 004C 280 .ADDRESS TEST_MOD_BEGIN
0050 281 PRVPRT:
00 0050 282 .BYTE 0 ; protection return byte for SETPRT
00000000 00000000 0051 283 PRIVMASK: ; priv. mask
0059 284 .QUAD 0
00000000 0059 285 CHM_CONT: ; change mode continue address
005D 286 .LONG 0
00000065 005D 287 RETADR: ; returned address's from SETPRT
0065 288 .BLKL 2
00000000 0065 289 STATUS:
0069 290 .LONG 0
00000000 0069 291 MODE: ; current mode string pointer
006D 292 .LONG 0
74 73 69 67 65 72 00000075 010E0000 006D 293 REG: \register R\
52 20 72 65 007B 294 .ASCID
007F 295 REGNUM: ; register number
00000000 007F 296 .LONG 0
0083 297 MSGL: ; buffer desc.
00000050 0083 298 .LONG 80
0000018B 0087 299 .ADDRESS BUF1
008B 300 BUF:
0000018B 008B 301 .BLKB 256
018B 302 BUF1:
000001DB 018B 303 .BLKB 80
01DB 304 LEN:
00000000 01DB 305 .LONG 0
0000008B 01DF 306 .ADDRESS BUF
01E3 307 TLEN:
00000000 01E3 308 .LONG 0
01E7 309 BUFFER:
00000100 01E7 310 .LONG 256
0000008B 01EB 311 .ADDRESS BUF
01EF 312 PARAM:
00000187 01EF 313 A=PL1
01EF 314 .REPT 8
01EF 315 .ADDRESS A
01EF 316 A=A+4
00000187 01EF 317 .ENDR
020F 318 CTRSIR:
00000084 020F 319 .LONG 132 ; same as above
```



```
00000217' 0213 320 .ADDRESS .+4
0000029B 0217 321 .BLKB 132
00000000 029B 322 MESSAGEL:
0000018B' 029F 323 .LONG 0 ; message desc.
02A3 324 .ADDRESS BUF1
00000000 02A3 325 SERV_NAME:
02A7 326 .LONG 0 ; service name pointer
00000003 02A7 327 MSGVEC1: ; PUTMSG message vector
00741133 02AB 328 .LONG 3
00000001 02AF 329 .LONG UETP$_TEXT
00000000 02B3 330 .LONG 1
02B7 331 .LONG 0
02B7 332 FAOP:
02E7 333 $FAO FCS1,LEN,BUFFER,PL1,PL1+4,PL2,PL2+4,PL2+8,PL2+12,PL3,PL3+4
02E7 334 FAOLP:
02FB 335 $FAOL FCS1,LEN,BUFFER,PARAM
00000000 00000000 02FB 336 PL8:
0303 337 .QUAD 0 ; time parameter
00000018 0303 338 RS8:
0000030B' 0307 339 .LONG 24
00000323 030B 340 .ADDRESS .+4
341 .BLKB 24
```



```
00000000 343      .PSECT SATSSS09, RD, WRT, EXE, LONG
0000      344      .SBTTL SATSSS09
0000      345      :++
0000      346      : FUNCTIONAL DESCRIPTION:
0000      347      :
0000      348      :     After performing some initial housekeeping, such as
0000      349      :     printing the module begin message and acquiring needed privileges,
0000      350      :     the system services are tested in each of their normal conditions.
0000      351      :     Detected failures are identified and an error message is printed
0000      352      :     on the terminal. Upon completion of the test a success or fail
0000      353      :     message is printed on the terminal.
0000      354      :
0000      355      : CALLING SEQUENCE:
0000      356      :
0000      357      :     $ RUN SATSSS09 ... (DCL COMMAND)
0000      358      :
0000      359      : INPUT PARAMETERS:
0000      360      :
0000      361      :     none
0000      362      :
0000      363      : IMPLICIT INPUTS:
0000      364      :
0000      365      :     none
0000      366      :
0000      367      : OUTPUT PARAMETERS:
0000      368      :
0000      369      :     none
0000      370      :
0000      371      : IMPLICIT OUTPUTS:
0000      372      :
0000      373      :     Messages to SYS$OUTPUT are the only output from SATSSS09.
0000      374      :     They are of the form:
0000      375      :
0000      376      :     %UETP-S-SATSMS, TEST MODULE SATSSS09 BEGUN ... (BEGIN MSG)
0000      377      :     %UETP-S-SATSMS, TEST MODULE SATSSS09 SUCCESSFUL ... (END MSG)
0000      378      :     %UETP-E-SATSMS, TEST MODULE SATSSS09 FAILED ... (END MSG)
0000      379      :     %UETP-I-TEXT, ... (VARIABLE INFORMATION ABOUT A TEST MODULE FAILURE)
0000      380      :
0000      381      : COMPLETION CODES:
0000      382      :
0000      383      :     The SATSSS09 routine terminates with a $EXIT to the
0000      384      :     operating system with a status code defined by UETP$_SATSMS.
0000      385      :
0000      386      : SIDE EFFECTS:
0000      387      :
0000      388      :     none
0000      389      :
0000      390      : --
0000      391      :
0000      392      : TEST_START SATSSS09                                ; let the test begin
```



```
0000 0000
0004'CF 00 DD 0002
0000'CF 00 DF 0006
00000000'GF 02 FB 000C
00000000'GF 00 FB 0013
0009'CF 01 7F 001A
00000000'GF 01 FB 001E
004C'CF 001F'CF 09CB 30 0025
0044'CF 03 00 01 FO 002F
08B6'CF 01 FB 0036
003D 0038
003D 003D
003D 393
003D 394
003D 395
003D 396
003D 397
003D 398
003D 399
003D 400
02A3'CF 014A'CF DE 003D 401
0069'CF 013E'CF DE 0044 402
5A 01A7'CF DE 004B 403
08B6'CF 01 FB 0050 404
0052 405
0057 406
0057 407
0057 408
0057 409
0057 410
0057 411
0057 412
0057 413
0057 414
0057 415
0057 416
0028'CF 0187'CF DE 007A 417
0081 418
00000000'8F DD 0081
08C0'CF 01 FB 0087
56 008B'CF DE 008C 419
57 04CA'CF DE 0091 420
58 0000006B 8F DO 0096 421
0877'CF 00 FB 009D 422
02C7'CF 0187'CF 20 28 00A2 423
00 DD 00AA 424
08B6'CF 01 FB 00AC 425
00B1 426
00BA 427
00000000'8F DD 00BA
08C0'CF 01 FB 00C0
0877'CF 00 FB 00C5 428
00CA 429
00CA 430

.ENTRY SATSSS09,0
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 FAO TESTS
:+
$FAO tests
test ascii directives
:-
MOVAL W^FAO,W^SERV_NAME ; set service name
MOVAL W^UM,W^MODE ; set the mode
MOVAL W^PL3+8,R10 ; set param pointer
PUSHL #0 ; push a dummy parameter
CALLS #1,W^REG_SAVE ; save a reg snapshot
$FAO_S CTRSTR = W^FCS1,-
OUTLEN = W^LEN,-
OUTBUF = W^BUFFER,-
P1 = -(R10),-
P2 = -(R10),-
P3 = -(R10),-
P4 = -(R10),-
P5 = -(R10),-
P6 = -(R10),-
P7 = -(R10),-
P8 = -(R10),-
MOVAL W^PL1,W^REG_SAVE_AREA+32 ; try S
FAIL_CHECK SS$NORMAL ; fix the reg_save_area for R10
PUSHL #SS$NORMAL ; check success
CALLS #1,W^REG_CHECK
MOVAL W^BUF,R6 ; set buffer adr
MOVAL W^RS1,R7 ; set good data adr
MOVL #RS1L,R8 ; set byte count
CALLS #0,W^BUF_CHECK ; check results
MOVC3 #32,W^PLT,W^FAOP+FAO$_P1 ; set the param list
PUSHL #0 ; push a dummy param
CALLS #1,W^REG_SAVE ; save a reg snapshot
$FAO_G W^FAOP ; try G
FAIL_CHECK SS$NORMAL ; check success
PUSHL #SS$NORMAL
CALLS #1,W^REG_CHECK
CALLS #0,W^BUF_CHECK ; check results
```



```
00CA 431 : test octal conversion, zero filled directives
00CA 432 :
00CA 433 :-
00CA 434 :
                                NEXT_TEST
                                STP1:
0004'CF 01 DO 00CA
00      00 DD 00CF
02C7'CF 08B6'CF 01 FB 00D1
01A7'CF 18 28 00D6 435 MOV C3 #24,W^PL4,W^FAOP+FAOS_P1 ; set new parameters
                                .LIST MEB
                                FAONTST FCS2,RS2,RS2L
                                MOVAL W^PL6+8,R10 ; set the arg pntr
                                PUSH L #0 ; push a dummy parameter
                                CALLS #1,W^REG_SAVE ; save a reg snapshot
                                PUSH L -(R10)
                                PUSH L -(R10)
                                PUSH L -(R10)
                                PUSH L -(R10)
                                PUSH L -(R10)
                                PUSH L -(R10)
                                PUSH L -(R10)
                                PUSH AQ W^BUFFER
                                PUSH AW W^LEN
                                PUSH AQ W^FCS2
                                CALLS #SST2,G^SYSS$FAO
                                MOVAL W^PL6+8,R10 ; reset the parameter pntr
                                PUSH L #SS$ NORMAL
                                CALLS #1,W^REG_CHECK
                                MOVAL W^RS2,R7 ; set good data adr
                                MOV L #RS2L,R8 ; set byte count
                                CALLS #0,W^BUF_CHECK ; check results
                                MOVAL W^FCS2,W^FAOP+FAOS_CTRSTR ; set new cntrl str
                                PUSH L #0 ; push a dummy parameter
                                CALLS #1,W^REG_SAVE ; save a reg snapshot
                                CALLG W^FAOP,G^SYSS$FAO
                                PUSH L #SS$ NORMAL
                                CALLS #1,W^REG_CHECK
                                CALLS #0,W^BUF_CHECK ; check results
                                438 :+
                                439 :
                                0151 440 : test hex conversion, zero filled directives
                                0151 441 :
                                0151 442 :-
                                0151 443 :
                                NEXT_TEST
                                STP2:
0004'CF 02 DO 0151
00      00 DD 0156
08B6'CF 01 FB 0158
                                444 .LIST MEB
                                445 FAONTST FCS3,RS3,RS3L
                                MOVAL #2,W^CURRENT_TC
                                PUSH L #0
                                CALLS #1,W^REG_SAVE
                                .LIST MEB
                                FAONTST FCS3,RS3,RS3L
                                MOVAL W^PL6+8,R10 ; set the arg pntr
                                PUSH L #0 ; push a dummy parameter
                                CALLS #1,W^REG_SAVE ; save a reg snapshot
                                PUSH L -(R10)
                                PUSH L -(R10)
                                PUSH L -(R10)
```



```

7A DD 016F
7A DD 0171
7A DD 0173
01E7'CF 7F 0175
01DB'CF 3F 0179
02D1'CF 7F 017D
00000000'GF 09 FB 0181
5A 01BF'CF DE 0188
00000000'8F DD 018D
08C0'CF 01 FB 0193
57 0596'CF DE 0198
58 0000004C'8F DO 019D
0877'CF 00 FB 01A4
02BB'CF 02D1'CF DE 01A9
00 DD 01B0
08B6'CF 01 FB 01B2
00000000'GF 02B7'CF FA 01B7
00000000'8F DD 01C0
08C0'CF 01 FB 01C6
0877'CF 00 FB 01CB
01D0
01D0
01D0
01D0
01D0
01D0
01D0
01D0
01D0
0004'CF 03 DO 01D0
00 DD 01D5
08B6'CF 01 FB 01D7
01DC
01DC
5A 01BF'CF DE 01DC
00 DD 01E1
08B6'CF 01 FB 01E3
7A DD 01E8
7A DD 01EA
7A DD 01EC
7A DD 01EE
7A DD 01F0
7A DD 01F2
01E7'CF 7F 01F4
01DB'CF 3F 01F8
033D'CF 7F 01FC
00000000'GF 09 FB 0200
5A 01BF'CF DE 0207
00000000'8F DD 020C
08C0'CF 01 FB 0212
57 05E2'CF DE 0217
58 00000040'8F DO 021C
0877'CF 00 FB 0223
02BB'CF 033D'CF DE 0228
00 DD 022F
08B6'CF 01 FB 0231
00000000'GF 02B7'CF FA 0236
00000000'8F DD 023F
PUSHL -(R10)
PUSHL -(R10)
PUSHL -(R10)
PUSHAQ W^BUFFER
PUSHAW W^LEN
PUSHAQ W^FCS3
CALLS $$T2,G^SYSS$FAO
MOVAL W^PL6+8,R10 ; reset the parameter ptr
PUSHL #SS$ NORMAL
CALLS #1,W^REG_CHECK
MOVAL W^RS3,R7 ; set good data adr
MOVL #RS3L,R8 ; set byte count
CALLS #0,W^BUF_CHECK ; check results
MOVAL W^FCS3,W^FAOP+FAO$_CTRSTR ; set new cntrl str
PUSHL #0 ; push a dummy parameter
CALLS #1,W^REG_SAVE ; save a reg snapshot
CALLG W^FAOP,G^SYSS$FAO
PUSHL #SS$ NORMAL
CALLS #1,W^REG_CHECK
CALLS #0,W^BUF_CHECK ; check results
446 :+
447 :
448 : test unsigned decimal, zero filled directives
449 :
450 :-
451
NEXT_TEST
STP3:
MOVL #3,W^CURRENT_TC
PUSHL #0
CALLS #1,W^REG_SAVE
452 .LIST MEB
453 FAONTST FCS4,RS4,RS4L
MOVAL W^PL6+8,R10 ; set the arg ptr
PUSHL #0 ; push a dummy parameter
CALLS #1,W^REG_SAVE ; save a reg snapshot
PUSHL -(R10)
PUSHL -(R10)
PUSHL -(R10)
PUSHL -(R10)
PUSHL -(R10)
PUSHL -(R10)
PUSHL -(R10)
PUSHAQ W^BUFFER
PUSHAW W^LEN
PUSHAQ W^FCS4
CALLS $$T2,G^SYSS$FAO
MOVAL W^PL6+8,R10 ; reset the parameter ptr
PUSHL #SS$ NORMAL
CALLS #1,W^REG_CHECK
MOVAL W^RS4,R7 ; set good data adr
MOVL #RS4L,R8 ; set byte count
CALLS #0,W^BUF_CHECK ; check results
MOVAL W^FCS4,W^FAOP+FAO$_CTRSTR ; set new cntrl str
PUSHL #0 ; push a dummy parameter
CALLS #1,W^REG_SAVE ; save a reg snapshot
CALLG W^FAOP,G^SYSS$FAO
PUSHL #SS$ NORMAL

```



```
08C0'CF 01 FB 0245 CALLS #1,W^REG_CHECK
0877'CF 00 FB 024A CALLS #0,W^BUF_CHECK ; check results
024F 454 :+
024F 455 :-
024F 456 : test unsigned decimal, blank filled directives
024F 457 :-
024F 458 :-
024F 459 :
NEXT_TEST
STP4:
0004'CF 04 DO 024F MOVL #4,W^CURRENT_TC
00 DD 0254 PUSHL #0
08B6'CF 01 FB 0256 CALLS #1,W^REG_SAVE
025B 460 .LIST MEB
025B 461 FAONTST FCS5,RS5,RS5L
025B DE 025B MOVAL W^PL6+8,R10 ; set the arg pntr
00 DD 0260 PUSHL #0 ; push a dummy parameter
08B6'CF 01 FB 0262 CALLS #1,W^REG_SAVE ; save a reg snapshot
7A DD 0267 PUSHL -(R10)
7A DD 0269 PUSHL -(R10)
7A DD 026B PUSHL -(R10)
7A DD 026D PUSHL -(R10)
7A DD 026F PUSHL -(R10)
7A DD 0271 PUSHL -(R10)
01E7'CF 7F 0273 PUSHAQ W^BUFFER
01DB'CF 3F 0277 PUSHAQ W^LEN
03A9'CF 7F 027B PUSHAQ W^FCS5
00000000'GF 09 FB 027F CALLS #1,W^REG_CHECK
5A 01BF'CF DE 0286 MOVAL W^PL6+8,R10 ; reset the parameter pntr
00000000'8F DD 028B PUSHL #SS$ NORMAL
08C0'CF 01 FB 0291 CALLS #1,W^REG_CHECK
57 0622'CF DE 0296 MOVAL W^RS5,R7 ; set good data adr
58 00000040 8F DO 029B MOVL #RS5L,R8 ; set byte count
0877'CF 00 FB 02A2 CALLS #0,W^BUF_CHECK ; check results
02BB'CF 03A9'CF DE 02A7 MOVAL W^FCS5,W^FAOP+FAO$_CTRSTR ; set new cntrl str
00 DD 02AE PUSHL #0 ; push a dummy parameter
08B6'CF 01 FB 02B0 CALLS #1,W^REG_SAVE ; save a reg snapshot
00000000'GF 02B7'CF FA 02B5 CALLG W^FAOP,G^SYSS$FAO
00000000'8F DD 02BE PUSHL #SS$ NORMAL
08C0'CF 01 FB 02C4 CALLS #1,W^REG_CHECK
0877'CF 00 FB 02C9 CALLS #0,W^BUF_CHECK ; check results
02CE 462 :+
02CE 463 :-
02CE 464 : test signed decimal, blank filled directives
02CE 465 :-
02CE 466 :-
02CE 467 :
NEXT_TEST
STP5:
0004'CF 05 DO 02CE MOVL #5,W^CURRENT_TC
00 DD 02D3 PUSHL #0
08B6'CF 01 FB 02D5 CALLS #1,W^REG_SAVE
02DA 468 .LIST MEB
02DA 469 FAONTST FCS6,RS6,RS6L
5A 01BF'CF DE 02DA MOVAL W^PL6+8,R10 ; set the arg pntr
00 DD 02DF PUSHL #0 ; push a dummy parameter
08B6'CF 01 FB 02E1 CALLS #1,W^REG_SAVE ; save a reg snapshot
```



```

7A DD 02E6 PUSH - (R10)
7A DD 02E8 PUSH - (R10)
7A DD 02EA PUSH - (R10)
7A DD 02EC PUSH - (R10)
7A DD 02EE PUSH - (R10)
7A DD 02F0 PUSH - (R10)
01E7'CF 7F 02F2 PUSHAQ W^BUFFER
01DB'CF 3F 02F6 PUSHAQ W^LEN
0415'CF 7F 02FA PUSHAQ W^FCS6
00000000'GF 09 FB 02FE CALLS #SST2,G^SYSSFAO
5A 01BF'CF DE 0305 MOVAL W^PL6+8,R10 ; reset the parameter pntr
00000000'8F DD 030A PUSH #SS$ NORMAL
08C0'CF 01 FB 0310 CALLS #1,W^REG_CHECK
57 0662'CF DE 0315 MOVAL W^RS6,R7 ; set good data adr
58 34 DO 031A MOVL #RS6L,R8 ; set byte count
0877'CF 00 FB 031D CALLS #0,W^BUF_CHECK ; check results
02BB'CF 0415'CF DE 0322 MOVAL W^FCS6,W^FAOP+FAOS_CTRSTR ; set new cntrl str
00 DD 0329 PUSH #0 ; push a dummy parameter
08B6'CF 01 FB 032B CALLS #1,W^REG_SAVE ; save a reg snapshot
00000000'GF 02B7'CF FA 0330 CALLG W^FAOP,G^SYSSFAO
00000000'8F DD 0339 PUSH #SS$ NORMAL
08C0'CF 01 FB 033F CALLS #1,W^REG_CHECK
0877'CF 00 FB 0344 CALLS #0,W^BUF_CHECK ; check results
0349 470 :+
0349 471 :-
0349 472 : test output string formatting and parameter interpretation directives
0349 473 :-
0349 474 :-
0349 475
NEXT_TEST
0349
STP6:
0004'CF 06 DO 0349 MOVL #6,W^CURRENT_TC
00 DD 034E PUSH #0
08B6'CF 01 FB 0350 CALLS #1,W^REG_SAVE
0355 476 $FAO_S CTRSTR = W^FCS7,-
0355 477 OUTLEN = W^LEN,-
0355 478 OUTBUF = W^BUFFER,-
0355 479 P1 = W^PL7,-
0355 480 P2 = W^PL7+4 ; try _S
0370 481 FAIL_CHECK SS$ NORMAL
00000000'8F DD 0370 PUSH #SS$ NORMAL
08C0'CF 01 FB 0376 CALLS #1,W^REG_CHECK
57 0696'CF DE 037B 482 MOVAL W^RS7,R7 ; set good data address
58 1A DO 0380 483 MOVL #RS7L,R8 ; set byte count
0877'CF 00 FB 0383 484 CALLS #0,W^BUF_CHECK ; check results
02BB'CF 0481'CF DE 0388 485 MOVAL W^FCS7,W^FAOP+FAOS_CTRSTR ; set new cntrl string
02C7'CF 01BF'CF DO 038F 486 MOVL W^PL7,W^FAOP+FAOS_P1 ; set new parameters
02CB'CF 01C3'CF DO 0396 487 MOVL W^PL7+4,W^FAOP+FAOS_P2
00 DD 039D 488 PUSH #0 ; push a dummy parameter
08B6'CF 01 FB 039F 489 CALLS #1,W^REG_SAVE ; save a reg snapshot
03A4 490 $FAO_G W^FAOP ; try _G
03AD 491 FAIL_CHECK SS$ NORMAL ; check success
00000000'8F DD 03AD PUSH #SS$ NORMAL
08C0'CF 01 FB 03B3 CALLS #1,W^REG_CHECK
0877'CF 00 FB 03B8 492 CALLS #0,W^BUF_CHECK ; check results
03BD 493 :+
03BD 494 :-
```



```
03BD 495 : test %D and %T directives
03BD 496 :-
03BD 497 :-
03BD 498 NEXT_TEST
03BD
03BD STP7:
0004'CF 07 DO 03BD
00 DD 03C2
08B6'CF 01 FB 03C4
03C9 499 $GETTIM_S TIMADR=W^PL8 ; get a binary time
03D4 500 $ASCTIM_S TIMBUF=W^RS8,-
03D4 501 TIMADR=W^PL8,-
03D4 502 TIMLEN=W^TLEN ; make it ascii for result string
0322'CF 0317'CF 0B 28 03E9 503 MOV C3 #11,W^RS8+8+12,W^RS8+8+23 ; copy the time parameter
0303'CF 0B CO 03F1 504 ADD L2 #11,W^RS8 ; fix the descriptor
08B6'CF 01 FB 03F6 505 PUSH L #0 ; push a dummy parameter
03F8 506 CALLS #1,W^REG_SAVE ; save a reg snapshot
03FD 507 $FAO_S CTRSTR = W^FCS8,-
03FD 508 OUTLEN = W^LEN,-
03FD 509 OUTBUF = W^BUFFER,-
03FD 510 P1 = #PL8 ; try S
0416 511 FAIL_CHECK SSS_NORMAL ; check success
00000000'8F DD 0416
08C0'CF 01 FB 041C
57 030B'CF DE 0421 512 MOVAL W^RS8+8,R7 ; set good data adr
58 01E3'CF DO 0426 513 MOVL W^TLEN,R8 ; set byte count
0877'CF 00 FB 042B 514 CALLS #0,W^BUF_CHECK ; check results
02BB'CF 04B6'CF DE 0430 515 MOVAL W^FCS8,W^FAOP+FAOS_CTRSTR ; set cntrl str address
02C7'CF 02FB'CF DE 0437 516 MOVAL W^PL8,W^FAOP+FAOS_P1 ; set parameter list
00 DD 043E 517 PUSH L #0 ; push dummy param
08B6'CF 01 FB 0440 518 CALLS #1,W^REG_SAVE ; save a reg snapshot
0445 519 $FAO_G W^FAOP ; try G
044E 520 FAIL_CHECK SSS_NORMAL ; check success
00000000'8F DD 044E
08C0'CF 01 FB 0454
0877'CF 00 FB 0459 521 CALLS #0,W^BUF_CHECK ; check results
01EF'CF 0187'CF 20 28 045E 522 MOV C3 #32,W^PLT,W^PARAM ; set FAOL parameters
```



```
0466 524 .SBTTL FAOL TESTS
0466 525 :+
0466 526 :
0466 527 : $FAOL tests
0466 528 :
0466 529 : test ascii directives
0466 530 :
0466 531 :-
0466 532 NEXT_TEST
0466
0466 STP8:
0004'CF 08 DO 0466 MOVL #8,W^CURRENT_TC
00 DD 046B PUSHL #0
08B6'CF 01 FB 046D CALLS #1,W^REG_SAVE
02A3'CF 014E'CF DE 0472 533 MOVAL W^FAOL,W^SERV_NAME ; set service name
0479 534 $FAOL_S CTRSTR = W^FCS1,-
0479 535 OUTLEN = W^LEN,-
0479 536 OUTBUF = W^BUFFER,-
0479 537 PRMLST = W^PARAM ; S
0490 538 FAIL_CHECK $$$_NORMAL ; check success
0490 539 PUSHL #$$$_NORMAL
0496 540 CALLS #1,W^REG_CHECK
049B 541 MOVAL W^RS1,R7 ; set good data adr
04A0 542 MOVL #RS1L,R8 ; set byte count
04A7 543 CALLS #0,W^BUF_CHECK ; check results
04AC 544 PUSHL #0 ; push a dummy param
04AE 545 CALLS #1,W^REG_SAVE ; save a reg snapshot
04B3 546 $FAOL_G W^FAOLP ; try G
04BC 547 FAIL_CHECK $$$_NORMAL ; check success
04BC 548 PUSHL #$$$_NORMAL
04C2 549 CALLS #1,W^REG_CHECK
04C7 550 CALLS #0,W^BUF_CHECK ; check results
04CC 551 :+
04CC 552 : test octal conversion, zero filled directives
04CC 553 :
04CC 554 :
04CC 555 :-
04CC 556 NEXT_TEST
04CC
04CC STP9:
0004'CF 09 DO 04CC MOVL #9,W^CURRENT_TC
00 DD 04D1 PUSHL #0
08B6'CF 01 FB 04D3 CALLS #1,W^REG_SAVE
01EF'CF 01A7'CF 18 28 04D8 553 MOVC3 #24,W^PL4,W^PARAM ; set new parameters
00 DD 04E0 554 PUSHL #0 ; push a dummy param
08B6'CF 01 FB 04E2 555 CALLS #1,W^REG_SAVE ; save a reg snapshot
04E7 556 .LIST MEB
04E7 557 FAONTST FCS2,RS2,RS2L,L
04E7 558 PUSHAL W^PARAM
04EB 559 PUSHAQ W^BUFFER
04EF 560 PUSHAQ W^LEN
04F3 561 PUSHAQ W^FCS2
04F7 562 CALLS #4,G^SYSS$FAOL
04FE 563 PUSHL #$$$_NORMAL
0504 564 CALLS #1,W^REG_CHECK
0509 565 MOVAL W^RS2,R7 ; set good data adr
050E 566 MOVL #RS2L,R8 ; set byte count
01EF'CF DF 04E7
01E7'CF 7F 04EB
01DB'CF 3F 04EF
0265'CF 7F 04F3
00000000'GF 04 FB 04F7
00000000'8F DD 04FE
08C0'CF 01 FB 0504
57 0535'CF DE 0509
58 00000061 8F DO 050E
```



```
0877'CF 00 FB 0515 CALLS #0,W^BUF_CHECK ; check results
02EB'CF 0265'CF 00 DE 051A MOVAL W^FCS2,W^FAOLP+FAOS_CTRSTR ; set new cntrl str
00000000'GF 02E7'CF 01 FB 0523 PUSHL #0 ; push a dummy parameter
08B6'CF 01 FA 0528 CALLS #1,W^REG_SAVE ; save a reg snapshot
00000000'8F DD 0531 CALLG W^FAOLP,G^SYSS$FAOL
08C0'CF 01 FB 0537 PUSHL #SS$ NORMAL
0877'CF 00 FB 053C CALLS #1,W^REG_CHECK
CALLS #0,W^BUF_CHECK ; check results

558 :+
559 :
560 : test hex conversion, zero filled directives
561 :
562 :-
563 :
NEXT_TEST
STP10:
0004'CF 0A DO 0541 MOVL #10,W^CURRENT_TC
0000 DD 0546 PUSHL #0
08B6'CF 01 FB 0548 CALLS #1,W^REG_SAVE
564 .LIST MEB
565 FAONTST FCS3,RS3,RS3L,L
PUSHAL W^PARAM
PUSHAQ W^BUFFER
PUSHAW W^LEN
PUSHAQ W^FCS3
CALLS #4,G^SYSS$FAOL
PUSHL #SS$ NORMAL
CALLS #1,W^REG_CHECK
MOVAL W^RS3,R7 ; set good data adr
MOVL #RS3L,R8 ; set byte count
CALLS #0,W^BUF_CHECK ; check results
MOVAL W^FCS3,W^FAOLP+FAOS_CTRSTR ; set new cntrl str
PUSHL #0 ; push a dummy parameter
CALLS #1,W^REG_SAVE ; save a reg snapshot
CALLG W^FAOLP,G^SYSS$FAOL
PUSHL #SS$ NORMAL
CALLS #1,W^REG_CHECK
CALLS #0,W^BUF_CHECK ; check results

566 :+
567 :
568 : test unsigned decimal, zero filled directives
569 :
570 :-
571 :
NEXT_TEST
STP11:
0004'CF 0B DO 05A7 MOVL #11,W^CURRENT_TC
0000 DD 05AC PUSHL #0
08B6'CF 01 FB 05AE CALLS #1,W^REG_SAVE
572 .LIST MEB
573 FAONTST FCS4,RS4,RS4L,L
PUSHAL W^PARAM
PUSHAQ W^BUFFER
PUSHAW W^LEN
PUSHAQ W^FCS4
CALLS #4,G^SYSS$FAOL
PUSHL #SS$ NORMAL
```



```

08C0'CF 01 FB 05D0 CALLS #1,W^REG_CHECK
57 57 05E2'CF DE 05D5 MOVAL W^RS4,R7 ; set good data adr
58 00000040 8F DO 05DA MOVL #RS4L,R8 ; set byte count
0877'CF 00 FB 05E1 CALLS #0,W^BUF_CHECK ; check results
02EB'CF 033D'CF DE 05E6 MOVAL W^FCS4,W^FAOLP+FA0$_CTRSTR ; set new cntrl str
00 00 DD 05ED PUSHL #0 ; push a dummy parameter
08B6'CF 01 FB 05EF CALLS #1,W^REG_SAVE ; save a reg snapshot
00000000'GF 02E7'CF FA 05F4 CALLG W^FAOLP,G^SYSS$FAOL
00000000'8F DD 05FD PUSHL #SS$ NORMAL
08C0'CF 01 FB 0603 CALLS #1,W^REG_CHECK
0877'CF 00 FB 0608 CALLS #0,W^BUF_CHECK ; check results
060D 574 ;+
060D 575 ;:
060D 576 ;: test unsigned decimal, blank filled directives
060D 577 ;:
060D 578 ;:-
060D 579 NEXT_TEST
060D STP12:
0004'CF 0C DO 060D MOVL #12,W^CURRENT_TC
08B6'CF 01 DD 0612 PUSHL #0
0619 580 CALLS #1,W^REG_SAVE
0619 581 .LIST MEB
FAONTST FCS5,RS5,RS5L,L
PUSHAL W^PARAM
PUSHAQ W^BUFFER
PUSHAW W^LEN
PUSHAQ W^FCS5
CALLS #4,G^SYSS$FAOL
PUSHL #SS$ NORMAL
CALLS #1,W^REG_CHECK
0630 MOVAL W^RS5,R7 ; set good data adr
0636 MOVL #RS5L,R8 ; set byte count
57 0622'CF DE 063B CALLS #0,W^BUF_CHECK ; check results
58 00000040 8F DO 0640 MOVAL W^FCS5,W^FAOLP+FA0$_CTRSTR ; set new cntrl str
0877'CF 00 FB 0647 PUSHL #0 ; push a dummy parameter
02EB'CF 03A9'CF DE 064C CALLS #1,W^REG_SAVE ; save a reg snapshot
00 00 DD 0653 CALLG W^FAOLP,G^SYSS$FAOL
08B6'CF 01 FB 0655 PUSHL #SS$ NORMAL
00000000'GF 02E7'CF FA 065A CALLS #1,W^REG_CHECK
00000000'8F DD 0663 CALLS #0,W^BUF_CHECK ; check results
08C0'CF 01 FB 0669
0877'CF 00 FB 066E
0673 582 ;+
0673 583 ;:
0673 584 ;: test signed decimal, blank filled directives
0673 585 ;:
0673 586 ;:-
0673 587 NEXT_TEST
0673 STP13:
0004'CF 0D DO 0673 MOVL #13,W^CURRENT_TC
08B6'CF 01 DD 0678 PUSHL #0
067F 588 CALLS #1,W^REG_SAVE
067F 589 .LIST MEB
FAONTST FCS6,RS6,RS6L,L
PUSHAL W^PARAM
PUSHAQ W^BUFFER
PUSHAW W^LEN

```



```
00000000'0415'CF 7F 068B      PUSHAQ  W^FCS6
00000000'GF 04 FB 068F      CALLS   #4,G^SYSS$FAOL
00000000'8F DD 0696      PUSHL   #SS$ NORMAL
08C0'CF 01 FB 069C      CALLS   #1,W^REG_CHECK
57 0662'CF DE 06A1      MOVAL   W^RS6,R7 ; set good data adr
58 34 DO 06A6      MOVL    #RS6L,R8 ; set byte count
0877'CF 00 FB 06A9      CALLS   #0,W^BUF_CHECK ; check results
02EB'CF 0415'CF DE 06AE      MOVAL   W^FCS6,W^FAOLP+FAOL$_CTRSTR ; set new cntrl str
00 DD 06B5      PUSHL   #0 ; push a dummy parameter
08B6'CF 01 FB 06B7      CALLS   #1,W^REG_SAVE ; save a reg snapshot
00000000'GF 02E7'CF FA 06BC      CALLG   W^FAOLP,G^SYSS$FAOL
00000000'8F DD 06C5      PUSHL   #SS$ NORMAL
08C0'CF 01 FB 06CB      CALLS   #1,W^REG_CHECK
0877'CF 00 FB 06D0      CALLS   #0,W^BUF_CHECK ; check results
01EF'CF 01BF'CF DO 06D5 590      MOVL    W^PL7,W^PARAM ; set the new params
01F3'CF 01C3'CF DO 06DC 591      MOVL    W^PL7+4,W^PARAM+4
592 :+
593 :
594 : test output string formatting and parameter interpretation directives
595 :
596 :-
597 :
NEXT_TEST
STP14:
0004'CF 0E DO 06E3      MOVL    #14,W^CURRENT_TC
00 00 DD 06E8      PUSHL   #0
08B6'CF 01 FB 06EA      CALLS   #1,W^REG_SAVE
06EF 598 $FAOL_S CTRSTR = W^FCS7,-
06EF 599 OUTLEN = W^LEN,-
06EF 600 OUTBUF = W^BUFFER,-
06EF 601 PRMLST = W^PARAM ; try S
0706 602 FAIL_CHECK SS$ NORMAL ; check success
DD 0706      PUSHL   #SS$ NORMAL
08C0'CF 01 FB 070C      CALLS   #1,W^REG_CHECK
57 0696'CF DE 0711 603      MOVAL   W^RS7,R7 ; set good data adr
58 1A DO 0716 604      MOVL    #RS7L,R8 ; set byte count
0877'CF 00 FB 0719 605      CALLS   #0,W^BUF_CHECK ; check results
00 DD 071E 606      PUSHL   #0 ; push a dummy param
08B6'CF 01 FB 0720 607      CALLS   #1,W^REG_SAVE ; save a reg snapshot
02EB'CF 0481'CF DE 0725 608      MOVAL   W^FCS7,W^FAOLP+FAOL$_CTRSTR ; set new control string
072C 609 $FAOL_G W^FAOLP ; try G
0735 610 FAIL_CHECK SS$ NORMAL ; check success
DD 0735      PUSHL   #SS$ NORMAL
08C0'CF 01 FB 073B      CALLS   #1,W^REG_CHECK
0877'CF 00 FB 0740 611      CALLS   #0,W^BUF_CHECK ; check results
0745 612 :+
0745 613 :
0745 614 : test %D and %T directives
0745 615 :
0745 616 :-
0745 617 :
NEXT_TEST
STP15:
0004'CF 0F DO 0745      MOVL    #15,W^CURRENT_TC
00 00 DD 074A      PUSHL   #0
08B6'CF 01 FB 074C      CALLS   #1,W^REG_SAVE
0751 618 $GETTIM_S TIMADR=W^PL8 ; get a binary time
```



```
0322'CF 0317'CF 0B 28 075C 619
          0303'CF 0B CO 075C 620
          08B6'CF 00 DD 075C 621
01EF'CF 02FB'CF 01 FB 0771 622
          00 DD 0779 623
          00 DD 077E 624
          01 FB 0780 625
          01 DE 0785 626
          00 DD 078C 627
          00 DD 078C 628
          00 DD 078C 629
          00 DD 078C 630
          00 DD 07A3 631
          00000000'8F DD 07A3
          08C0'CF 01 FB 07A9
          57 030B'CF DE 07AE 632
          58 01E3'CF DO 07B3 633
          0877'CF 00 FB 07B8 634
02EB'CF 04B6'CF DE 07BD 635
          00 DD 07C4 636
          08B6'CF 01 FB 07C6 637
          00 DD 07CB 638
          00000000'8F DD 07D4 639
          08C0'CF 01 FB 07DA
          0877'CF 00 FB 07DF 640
          00 DD 07E4 641
          004C'CF DD 07E4
          0048'CF DD 07E8
          02 DD 07EC
          0044'CF DD 07EE
          00000000'GF 04 FB 07F2
0044'CF 01 1C 01 FO 07F9
          0044'CF DD 0800
          00000000'GF 01 FB 0804
```

```
$ASCTIM_S TIMBUF=W^RS8,-
          TIMADR=W^PL8,-
          TIMLEN=W^TLEN
MOV C3 #11,W^RS8+8+12,W^RS8+8+23 ; make it ascii for result string
ADD L2 #11,W^RS8 ; copy the time parameter
PUSHL #0 ; fix the descriptor
CALLS #1,W^REG_SAVE ; push a dummy parameter
MOVAL W^PL8,W^PARAM ; save a reg snapshot
$FAOL_S CTRSTR = W^FCS8,- ; set the param list
        OUTLEN = W^LEN,-
        OUTBUF = W^BUFFER,-
        PRMLST = W^PARAM
FAIL_CHECK SSS_NORMAL ; try S
        PUSHL #SSS_NORMAL ; check success
        CALLS #1,W^REG_CHECK
MOVAL W^RS8+8,R7 ; set good data adr
MOVL W^TLEN,R8 ; set byte count
CALLS #0,W^BUF_CHECK ; check results
MOVAL W^FCS8,W^FAOLP+FAOLS_CTRSTR ; set cntrl str address
PUSHL #0 ; push dummy param
CALLS #1,W^REG_SAVE ; save a reg snapshot
$FAOL_G W^FAOLP ; try G
FAIL_CHECK SSS_NORMAL ; check success
        PUSHL #SSS_NORMAL
        CALLS #1,W^REG_CHECK
CALLS #0,W^BUF_CHECK ; check results
TEST_END
        PUSHL W^TMD_ADDR
        PUSHL W^TMN_ADDR
        PUSHL #2
        PUSHL W^MOD_MSG_CODE
        CALLS #SS$T1,G^LIB$SIGNAL
        INSV #1,#ST$SV_INHIB_MSG,#1,W^MOD_MSG_CODE
        PUSHL W^MOD_MSG_CODE
        CALLS #1,G^SYS$EXIT
```



```
080B 643 .SBTTL BUF_CHECK
080B 644 :++
080B 645 : FUNCTIONAL DESCRIPTION:
080B 646 : Routine to check the contents of a buffer against known good
080B 647 : data.
080B 648 :
080B 649 : CALLING SEQUENCE:
080B 650 : CALLS #0,W^BUF_CHECK ; check buffer
080B 651 :
080B 652 : INPUT PARAMETERS:
080B 653 : R6 = buffer address
080B 654 : R7 = good data address
080B 655 : R8 = byte count
080B 656 :
080B 657 : OUTPUT PARAMETERS:
080B 658 : NONE
080B 659 :
080B 660 :--
080B 661 :
080B 662 BCSD:
00000050 080B 663 .LONG 80
00000813' 080F 664 .ADDRESS BCBUF
0813 665 BCBUF:
00000863 0813 666 .BLKB 80
0863 667 BCOSD:
00000000 0863 668 .LONG 0
00000813' 0867 669 .ADDRESS BCBUF
086B 670 PARAM1:
00000877 086B 671 .BLKL 3
0877 672 :
0877 673 BUF_CHECK:
0877 674 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
66 59 56 07FC 0879 675 MOVL R6,R9 ; save a copy of the buffer address
67 67 58 29 087C 676 CMPC3 R8,(R7),(R6) ; check the buffer
5A E6 AF DE 0880 677 BEQL 10$ ; br if good
8A 8A 63 9A 0882 678 MOVAL B^PARAM1,R10 ; set parameter pointer
8A 53 59 C3 0886 679 MOVZBL (R3),(R10)+ ; save bad data
5A D8 AF DE 0889 680 MOVZBL (R1),(R10)+ ; save good data
088C 681 SUBL3 R9,R3,(R10)+ ; save byte offset
0890 682 MOVAL B^PARAM1,R10 ; reset address pointer
0894 683 $FAO_S CTRSTR = W^CS6,-
0894 684 OUTLEN = W^BCOSD,-
0894 685 OUTBUF = W^BCSD,-
0894 686 P1 = (R10)+,-
0894 687 P2 = (R10)+,-
0894 688 P3 = (R10)+
0902'CF B3 AF DF 08AD 689 PUSHAL B^BCOSD ; make the string
01 FB 08B0 690 CALLS #1,W^PRINT_FAIL ; push the string variable
08B5 691 10$: ; print the failure
08B5 692 RET ; return
```



```
08B6 694 .SBTTL REG_SAVE
08B6 695 :++
08B6 696 : FUNCTIONAL DESCRIPTION:
08B6 697 : Subroutine to save R2-R11 in the register save location.
08B6 698 :
08B6 699 : CALLING SEQUENCE:
08B6 700 : PUSHL #0 ; save a dummy parameter
08B6 701 : CALLS #1,W^REG_SAVE ; save R2-R11
08B6 702 :
08B6 703 : INPUT PARAMETERS:
08B6 704 : NONE
08B6 705 :
08B6 706 : OUTPUT PARAMETERS:
08B6 707 : NONE
08B6 708 :
08B6 709 :--
08B6 710 :
08B6 711 REG_SAVE:
08B6 712 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
08B8 713 MOVCL3 #4*10,^X14(FP),W^REG_SAVE_AREA ; save the registers in the program
08BF 714 RET
08C0 715 .SBTTL REG_CHECK
08C0 716 :++
08C0 717 : FUNCTIONAL DESCRIPTION:
08C0 718 : Subroutine to test R0 & R2-R11 for proper content after a service
08C0 719 : execution. A snapshot is taken by the REG_SAVE routine at the
08C0 720 : beginning of each step and this routine is executed after the
08C0 721 : services have been executed.
08C0 722 :
08C0 723 : CALLING SEQUENCE:
08C0 724 : PUSHL #SS$ XXXXXX ; push expected R0 contents
08C0 725 : CALLS #1,W^REG_CHECK ; execute this routine
08C0 726 :
08C0 727 : INPUT PARAMETERS:
08C0 728 : expected R0 contents on the stack
08C0 729 :
08C0 730 : OUTPUT PARAMETERS:
08C0 731 : possible error messages printed using $PUTMSG
08C0 732 :
08C0 733 :--
08C0 734 :
08C0 735 REG_CHECK:
08C0 736 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
08C2 737 CMPL 4(AP),R0 ; is this the right fail code?
08C6 738 BEQL 10$ ; br if yes
08C8 739 PUSHL R0 ; push received data
08CA 740 PUSHL 4(AP) ; push expected data
08CD 741 PUSHAL W^EXP ; push the string variable
08D1 742 CALLS #3,W^PRINT_FAIL ; print the error message
08D6 743 10$:
08D6 744 CMPC3 #4*10,^X14(FP),W^REG_SAVE_AREA ; check all but R0
08DD 745 BEQL 20$ ; br if O.K.
08DF 746 SUBL3 #REG_SAVE_AREA,R3,R6 ; calculate the register number
08E7 747 DIVL2 #4,R6 ;
08EA 748 ADDB3 #^X2,R6,-(SP) ; set number past R0-R1 and save
08EE 749 BICL2 #3,R1 ; backup to register boundrys
08F1 750 BICL2 #3,R3
```

0008'CF 14 AD 28 OFFC 28 04

50 04 AC D1 08C2 737

04 AC DD 08C8 739

0153'CF DF 08CD 741

0902'CF 03 FB 08D1 742

0008'CF 14 AD 28 29 08D6 744

56 53 00000008'8F C3 08DF 746

7E 56 04 C6 08E7 747

51 03 CA 08EE 749

53 03 CA 08F1 750


```

61 DD 08F4 751          PUSHL (R1)          ; push received data
63 DD 08F6 752          PUSHL (R3)          ; push expected data
006D'CF DF 08F8 753          PUSHAL W^REG    ; set string pntr param.
0902'CF 04 FB 08FC 754      CALLS #4,W^PRINT_FAIL ; print the error message
                                20$:
                                04 0901 755      RET
                                0901 756      .SBTTL PRINT_FAIL
                                0902 757
                                0902 758      :++
                                0902 759      : FUNCTIONAL DESCRIPTION:
                                0902 760      : Subroutine to report failures using $PUTMSG
                                0902 761
                                0902 762      : CALLING SEQUENCE:
                                0902 763      : Mode #1          PUSHL EXPECTED Mode #2          PUSHL NUMBER
                                0902 764      :                   PUSHL RECEIVED          PUSHL EXPECTED
                                0902 765      :                   PUSHAL STRING VAR      PUSHAL RECEIVED
                                0902 766      :                   CALLS #3,W^PRINT_FAIL    PUSHAL STRING VAR
                                0902 767      :                                           CALLS #4,W^PRINT_FAIL
                                0902 768      : Mode #3          PUSHAL STRING VAR
                                0902 769      :                   CALLS #1,W^PRINT_FAIL
                                0902 770
                                0902 771      : INPUT PARAMETERS:
                                0902 772      : listed above
                                0902 773
                                0902 774      : OUTPUT PARAMETERS:
                                0902 775      : an error message is printed using $PUTMSG
                                0902 776
                                0902 777      :--
                                0902 778
                                003C 0902 779 PRINT_FAIL:
                                0902 780      .WORD ^M<R2,R3,R4,R5>
                                0904 781      $FAO S W^CS1,W^MESSAGEL,W^MSGL,#TEST_MOD_NAME,W^SERV_NAME,W^CURRENT_TC
                                0925 782      $PUTMSG_S W^MSGVEC          ; print the message
                                04 6C 91 0936 783      CMPB (AP),#4          ; is this a register message?
                                01 26 13 0939 784      BEQL 10$              ; br if yes
                                01 6C 91 093B 785      CMPB (AP),#1          ; is this just a message?
                                48 13 093E 786      BEQL 20$              ; br if yes
                                40 11 0940 787      $FAO_S W^CS2,W^MESSAGEL,W^MSGL,4(AP),8(AP),4(AP),12(AP)
                                095F 788      BRB 30$              ; goto output message
                                0961 789 10$:
                                0961 790      $FAO_S W^CS3,W^MESSAGEL,W^MSGL,4(AP),16(AP),8(AP),4(AP),16(AP),12(AP)
                                19 11 0986 791      BRB 30$              ; goto output message
                                0988 792 20$:
                                02B3'CF 04 AC D0 0988 793      MOVL 4(AP),W^MSGVEC1+12 ; save string address
                                098E 794      $PUTMSG_S W^MSGVEC1          ; print the message
                                11 11 099F 795      BRB -40$           ; skip the other message
                                09A1 796 30$:
                                09A1 797      $PUTMSG_S W^MSGVEC          ; print the message
                                09B2 798 40$:
                                09B2 799      CALLS #0,W^MODE ID          ; identify the mode
                                004C'CF 002A'CF DE 09B7 800      MOVAL W^TEST_MOD_FAIL,W^TMD_ADDR ; set failure message address
                                0044'CF 03 00 02 FO 09BE 801      INSV #ERROR,#0,#3,W^MOD_MSG_CODE ; set severity code
                                04 09C5 802      RET
```



```

09C6 804 .SBTTL MODE_ID
09C6 805 :++
09C6 806 : FUNCTIONAL DESCRIPTION:
09C6 807 : Subroutine to identify the mode that an exit handler is in.
09C6 808 :
09C6 809 : CALLING SEQUENCE:
09C6 810 : CALLS #0,W^MODE_ID
09C6 811 :
09C6 812 : INPUT PARAMETERS:
09C6 813 : MODE contains an address pointing to an ascii string desc.
09C6 814 : of the current CPU mode.
09C6 815 :
09C6 816 : OUTPUT PARAMETERS:
09C6 817 : NONE
09C6 818 :
09C6 819 :--
09C6 820
003C 09C6 821 MODE_ID:
09C6 822 .WORD ^M<R2,R3,R4,R5>
09C8 823 $FAO S W^CS5,W^MESSAGEL,W^MSGL,MODE ; format the error message
09E1 824 $PUTMSG_S W^MSGVEC ; print the mode message
04 09F2 825 RET

```



```
09F3 828 MOD_MSG_PRINT:
09F3 829 :
09F3 830 : *****
09F3 831 : *
09F3 832 : * PRINTS THE TEST MODULE BEGUN/SUCCESSFUL/FAILED MESSAGES *
09F3 833 : * (USING THE PUTMSG MACRO). *
09F3 834 : *
09F3 835 : *****
09F3 836 :
05 09F3 837 PUTMSG <MOD_MSG_CODE,#2,TMN_ADDR,TMD_ADDR> : PRINT MSG
0A0E 838 RSB ; ... AND RETURN TO CALLER
0A0F 839 :
0A0F 840 CHMRTN:
0A0F 841 : *****
0A0F 842 : *
0A0F 843 : * CHANGE MODE ROUTINE. THIS ROUTINE GETS CONTROL WHENEVER *
0A0F 844 : * A CMKRNL, CMEXEC, OR CMSUP SYSTEM SERVICE IS ISSUED *
0A0F 845 : * BY THE MODE MACRO ('TO' OPTION). IT MERELY DOES *
0A0F 846 : * A JUMP INDIRECT ON A FIELD SET UP BY MODE. IT HAS *
0A0F 847 : * THE EFFECT OF RETURNING TO THE END OF THE MODE *
0A0F 848 : * MACRO EXPANSION. *
0A0F 849 : *
0A0F 850 : *****
0A0F 851 :
00000059'FF 0000 0A0F 852 .WORD 0 ; ENTRY MASK
17 0A11 853 JMP @CHM_CONT ; RETURN TO MODE MACRO IN NEW MODE
0A17 854 :
0A17 855 : * RET INSTR WILL BE ISSUED IN EXPANSION OF 'MODE FROM, ....' MACRO
0A17 856 :
0A17 857 .END SATSSS09
```


SATSSS09
Symbol table

- SATS SYSTEM SERVICE TESTS (SUCC S.C.) H 6
16-SEP-1984 01:43:09 VAX/VMS Macro V04-00
5-SEP-1984 04:22:34 [UETP.SRC]SATSSS09.MAR;1

Page 28
(3)

\$\$ARGS	= 00000004		FCS2L	= 00000064	
\$\$T1	= 00000004		FCS3	= 000002D1 R	02
\$\$T2	= 00000004		FCS3L	= 00000064	
A	= 000001A7 R	02	FCS4	= 0000033D R	02
BCBUF	00000813 R	04	FCS4L	= 00000064	
BCOSD	00000863 R	04	FCS5	= 000003A9 R	02
BCSD	0000080B R	04	FCS5L	= 00000064	
BUF	0000008B R	03	FCS6	= 00000415 R	02
BUF1	0000018B R	03	FCS6L	= 00000064	
BUFFER	000001E7 R	03	FCS7	= 00000481 R	02
BUF_CHECK	00000877 R	04	FCS8	= 000004B6 R	02
CHMRTN	00000A0F R	04	FF	= 0000000C	
CHM_CONT	00000059 R	03	INFO	= 00000003	
CR	= 0000000D		LEN	= 000001DB R	03
CS1	00000031 R	02	LF	= 0000000A	
CS2	00000063 R	02	LIB\$SIGNAL	***** X	04
CS3	00000090 R	02	MESSAGEL	0000029B R	03
CS4	000000C3 R	02	MODE	00000069 R	03
CS5	000000E9 R	02	MODE_ID	000009C6 R	04
CS6	000000FE R	02	MOD_MSG_CODE	00000044 R	03
CTRSTR	0000020F R	03	MOD_MSG_PRINT	000009F3 R	04
CURRENT_TC	00000004 R	03	MSGC	00000083 R	03
ERROR	= 00000002		MSGVEC	00000161 R	02
EXP	00000153 R	02	MSGVEC1	000002A7 R	03
FAO	0000014A R	02	NUL	= 00000000	
FAOS_CTRSTR	= 00000004		PARAM	000001EF R	03
FAOS_NARGS	= 00000014		PARAM1	0000086B R	04
FAOS_OUTBUF	= 0000000C		PL1	00000187 R	02
FAOS_OUTLEN	= 00000008		PL2	0000018F R	02
FAOS_P1	= 00000010		PL3	0000019F R	02
FAOS_P10	= 00000034		PL4	000001A7 R	02
FAOS_P11	= 00000038		PL5	000001AF R	02
FAOS_P12	= 0000003C		PL6	000001B7 R	02
FAOS_P13	= 00000040		PL7	000001BF R	02
FAOS_P14	= 00000044		PL8	000002FB R	03
FAOS_P15	= 00000048		PRINT_FAIL	00000902 R	04
FAOS_P16	= 0000004C		PRIVMASK	00000051 R	03
FAOS_P17	= 00000050		PRVPRT	00000050 R	03
FAOS_P2	= 00000014		REG	0000006D R	03
FAOS_P3	= 00000018		REGNUM	0000007F R	03
FAOS_P4	= 0000001C		REG_CHECK	000008C0 R	04
FAOS_P5	= 00000020		REG_SAVE	000008B6 R	04
FAOS_P6	= 00000024		REG_SAVE_AREA	00000008 R	03
FAOS_P7	= 00000028		RETADR	0000005D R	03
FAOS_P8	= 0000002C		RS1	000004CA R	02
FAOS_P9	= 00000030		RS1L	= 0000006B	
FAOL	0000014E R	02	RS2	= 00000535 R	02
FAOL\$CTRSTR	= 00000004		RS2L	= 00000061	
FAOL\$NARGS	= 00000004		RS3	= 00000596 R	02
FAOL\$OUTBUF	= 0000000C		RS3L	= 0000004C	
FAOL\$OUTLEN	= 00000008		RS4	= 000005E2 R	02
FAOL\$PRMLST	= 00000010		RS4L	= 00000040	
FAOLP	000002E7 R	03	RS5	= 00000622 R	02
FAOP	000002B7 R	03	RS5L	= 00000040	
FCS1	000001C7 R	02	RS6	= 00000662 R	02
FCS1L	= 00000096		RS6L	= 00000034	
FCS2	00000265 R	02	RS7	= 00000696 R	02

SATSSS09
Symbol table

- SATS SYSTEM SERVICE TESTS (SUCC S.C.) 16-SEP-1984 01:43:09 VAX/VMS Macro V04-00
5-SEP-1984 04:22:34 [UETP.SRC]SATSSS09.MAR;1

Page 29
(3)

RS7L	= 0000001A		
RS8	00000303	R	03
SATSSS09	00000000	RG	04
SERV_NAME	000002A3	R	03
SEVERE	= 00000004		
SHRSK_SHRDEF	= 00000001		
SHRS_TEXT	= 00001130		
SSS_NORMAL	*****	X	04
STATUS	00000065	R	03
STEP	= 0000000F		
STP0	0000003D	R	04
STP1	000000CA	R	04
STP10	00000541	R	04
STP11	000005A7	R	04
STP12	0000060D	R	04
STP13	00000673	R	04
STP14	000006E3	R	04
STP15	00000745	R	04
STP2	00000151	R	04
STP3	000001D0	R	04
STP4	0000024F	R	04
STP5	000002CE	R	04
STP6	00000349	R	04
STP7	000003BD	R	04
STP8	00000466	R	04
STP9	000004CC	R	04
STR1	00000171	R	02
STR2	00000176	R	02
STR2L	= 00000005		
STR3	0000017B	R	02
STSSV_INHIB_MSG	= 0000001C		
SUCCESS	= 00000001		
SYSSASCTIM	*****	GX	04
SYSEXIT	*****	GX	04
SYSSFAO	*****	GX	04
SYSSFAOL	*****	GX	04
SYSSGETTIM	*****	GX	04
SYSSHIBER	*****	GX	04
SYSSPUTMSG	*****	GX	04
SYSSSETPRN	*****	GX	04
SYSSWAKE	*****	GX	04
TAB	= 00000009		
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
TLEN	000001E3	R	03
TMD_ADDR	0000004C	R	03
TMN_ADDR	00000048	R	03
TPID	00000000	R	03
UETPS_SATSMS	= 007480D9		
UETPS_TEXT	= 00741133		
UM	0000013E	R	02
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
\$AB\$\$	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
RODATA	000006B0 (1712.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
RWDATA	00000323 (803.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
SATSSS09	00000A17 (2583.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	37	00:00:00.07	00:00:00.71
Command processing	139	00:00:00.62	00:00:02.44
Pass 1	351	00:00:10.31	00:00:24.34
Symbol table sort	0	00:00:00.43	00:00:00.66
Pass 2	219	00:00:03.46	00:00:08.74
Symbol table output	21	00:00:00.15	00:00:00.44
Psect synopsis output	2	00:00:00.02	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	771	00:00:15.08	00:00:37.38

The working set limit was 1800 pages.
86582 bytes (170 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 324 non-local and 7 local symbols.
857 source lines were read in Pass 1, producing 34 object records in Pass 2.
44 pages of virtual memory were used to define 36 macros.

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

Macro library name	Macros defined
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	22
-\$255\$DUA28:[UETP.OBJ]UETP.MLB;1	10
-\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	0
TOTALS (all libraries)	32

498 GETS were required to define 32 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:SATSSS09/OBJ=OBJ\$:SATSSS09 MSRC\$:SATSSS09/UPDATE=(ENH\$:SATSSS09)+EXECML\$/LIB+LIB\$:UETP/LIB

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY