


```

SSSSSSSS  YY      YY  SSSSSSSS  GGGGGGGG  TTTTTTTTTT  EEEEEEEEEE  RRRRRRRR  MM      MM
SSSSSSSS  YY      YY  SSSSSSSS  GGGGGGGG  TTTTTTTTTT  EEEEEEEEEE  RRRRRRRR  MM      MM
SS        YY      YY  SS        GG        TT        EE        RR      RR  MMMM  MMMM
SS        YY      YY  SS        GG        TT        EE        RR      RR  MMMM  MMMM
SS        YY      YY  SS        GG        TT        EE        RR      RR  MM   MM  MM
SSSSSSS   YY      YY  SSSSSS   GG        TT        EEEEEEEE  RRRRRRRR  MM   MM  MM
SSSSSSS   YY      YY  SSSSSS   GG        TT        EEEEEEEE  RRRRRRRR  MM   MM  MM
SS        YY      YY  SS        GG  GGGGGG  TT        EE        RR  RR  MM   MM  MM
SS        YY      YY  SS        GG  GGGGGG  TT        EE        RR  RR  MM   MM  MM
SS        YY      YY  SS        GG        TT        EE        RR  RR  MM   MM  MM
SSSSSSSS  YY      YY  SSSSSSSS  GGGGGG   TT        EE        RR      RR  MM   MM  MM
SSSSSSSS  YY      YY  SSSSSSSS  GGGGGG   TT        EEEEEEEEEE  RR      RR  MM   MM  MM
SSSSSSSS  YY      YY  SSSSSSSS  GGGGGG   TT        EEEEEEEEEE  RR      RR  MM   MM  MM

```

```

...
...
...
...

```

```

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
LLLLLLLLLL IIIIII  SSSSSSSS
LLLLLLLLLL IIIIII  SSSSSSSS

```

(1) 58
(2) 160

DECLARATIONS
SYSG&LOAD_TT_STR - Load terminal driver echo strings

```
0000 1 .TITLE SYSGTERM - SYSGEN TERMINAL commands
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 :
0000 31 : FACILITY:
0000 32 :
0000 33 : SYSGEN
0000 34 :
0000 35 : ABSTRACT:
0000 36 :
0000 37 : Process terminal driver customization commands.
0000 38 :
0000 39 :
0000 40 : ENVIRONMENT:
0000 41 :
0000 42 : VMS, Kernel mode. Raised IPL.
0000 43 :
0000 44 :--
0000 45 :
0000 46 : AUTHOR: Jake VanNoy, CREATION DATE: 14-Jul-1983
0000 47 :
0000 48 : MODIFIED BY:
0000 49 :
0000 50 : V03-001 JLV0383 Jake VanNoy 23-JUL-1984
0000 51 : Attempt to signal string that FAO fails on.
0000 52 : Add support for TTYEXIT STRING, a logical name
0000 53 : that the RTL uses for it's echoing.
0000 54 :
0000 55 :++
0000 56 :
0000 57
```

```

0000 58          .SBTTL  DECLARATIONS
0000 59          :
0000 60          : INCLUDE FILES:
0000 61          :
0000 62          $DPTDEF          : Driver prologue table
0000 63          $DYNDEF          : pool tags
0000 64          $IPLDEF          : interrupt levels
0000 65          $IRPDEF          : IRP offsets
0000 66          $LNMDEF          : Logical name definitions
0000 67          $PSLDEF          : PSL definitions
0000 68          $STSDEF          : status definition
0000 69          $TTYVECDEF       : terminal driver symbols
0000 70          :
0000 71          : MACROS:
0000 72          :
0000 73          :
0000 74          :
0000 75          : EQUATED SYMBOLS:
0000 76          :
0000 77          :
000003E8 0000 78 BYTES_TO_ALLOCATE = 1000
00000064 0000 79 BUFFER_LEN = 100
0000 80          :
0000 81          :
0000 82          : OWN STORAGE:
0000 83          :
00000000 84          .PSECT  NONPAGED_DATA
0000 85          :
0A 0D 00000008'010E0000' 0000 86 CRLF:          .ASCID  <13><10>          : carriage return/line feed
00000000' 000A 87 FAO_PRMLST:  .ADDRESS          CRLF          : list of parameters for FAOL
00000000' 000E 88          .ADDRESS          CRLF          :
00000000' 0012 89          .ADDRESS          CRLF          :
00000000' 0016 90          .ADDRESS          CRLF          :
00000000' 001A 91          .ADDRESS          CRLF          :
00000000' 001E 92          .ADDRESS          CRLF          :
00000000' 0022 93          .ADDRESS          CRLF          :
00000000' 0026 94          .ADDRESS          CRLF          :
00000000' 002A 95          .ADDRESS          CRLF          :
00000000' 002E 96          .ADDRESS          CRLF          :
00000000' 0032 97          .ADDRESS          CRLF          :
0000 98          :
00000000 0036 99 VEC_TABLE:      .LONG    0
00000000 003A 100 STR_BLOCK:    .LONG    0
000003E8 003E 101 NUM_BYTES:    .LONG    BYTES_TO_ALLOCATE
0000 102          :
00000092 0042 103 OFFSET_TABLE:  .BLKL   20          : Number of longword vectors
0000 104          :
00000000 0092 105 MSGLEN:      .LONG    0
00000000 0096 106 MSG_OUTADR:  .BYTE    0          : first byte of outadr
0000009A 0097 107 FAO_COUNT:    .BLKB   3          : fao_count is second byte
0000 108          :
00000064 009A 109 STR_DESC:      .LONG    BUFFER_LEN
00000000 009E 110          .LONG    0          : to be filled in
0000 111          :
00000064 00A2 112 BUFFER_DESC:    .LONG    BUFFER_LEN
000000AA 00A6 113          .LONG    BUFFER_ADDR
0000010E 00AA 114 BUFFER_ADDR:    .BLKB   BUFFER_LEN          : data goes here from $GETMSG

```

```

010E 115
010E 116 MSG_TABLE: ; List of message codes
010E 117
00000000 010E 118 .LONG 0 ; Break
00000000' 0112 119 .LONG SYSG_TTS_CTRLY ; standard ^Y
00000000' 0116 120 .LONG SYSG_TTS_CTRLC ; standard ^C
00000000' 011A 121 .LONG SYSG_TTS_CTRLY_REG ; ^Y for REGIS
00000000' 011E 122 .LONG SYSG_TTS_CTRLC_REG ; ^C for REGIS
00000000' 0122 123 .LONG SYSG_TTS_CTRLY_DEC ; ^Y for DEC_CRT
00000000' 0126 124 .LONG SYSG_TTS_CTRLC_DEC ; ^C for DEC_CRT
00000000' 012A 125 .LONG SYSG_TTS_CTRLY_DECREG ; ^Y for DEC_CRT and REGIS
00000000' 012E 126 .LONG SYSG_TTS_CTRLC_DECREG ; ^C for DEC_CRT and REGIS
00000000 0132 127 .LONG 0 ; Break in table
0136 128
00000000' 0136 129 .LONG SYSG_TTS_CTRLZ ; Standard ^Z
00000000' 013A 130 .LONG SYSG_TTS_CTRLZ_DEC ; ^Z for DEC_CRT
00000000 013E 131 .LONG 0 ; Break
0142 132
00000000' 0142 133 .LONG SYSG_TTS_CTRL0 ; output off
00000000' 0146 134 .LONG SYSG_TTS_CTRL0_DEC ; output off for DEC_CRT
00000000' 014A 135 .LONG SYSG_TTS_OUTON ; output on
00000000' 014E 136 .LONG SYSG_TTS_OUTON_DEC ; output on for DEC_CRT
00000000 0152 137 .LONG 0 ; Break
0156 138
00000000 0156 139 .LONG 0 ; End of table
015A 140
015A 141 LNM_ITMLST:
0000 015A 142 LNM_CTRLZ_DEC: .WORD 0
0002 015C 143 .WORD LNMS_STRING
00000000 015E 144 .LONG 0
00000000 0162 145 .LONG 0
0166 146
0000 0166 147 LNM_CTRLZ: .WORD 0
0002 0168 148 .WORD LNMS_STRING
00000000 016A 149 .LONG 0
00000000 016E 150 .LONG 0
0172 151
00000000 0172 152 .LONG 0 ; End of item list
0176 153
59 53 24 4D 4E 4C 000017E'010E0000' 0176 154 LNM_SYSTEM: .ASCID /LNM$SYSTEM_TABLE/ ; table to create name in
45 4C 42 41 54 5F 4D 45 54 53 0184
58 45 24 59 54 54 0000196'010E0000' 018E 155 LNM_LOGNAM: .ASCID /TTY$EXIT_STRING/ ; name of logical name
47 4E 49 52 54 53 5F 54 49 019C
01A5 156
01A5 157
00000000 158 .PSECT NONPAGED_CODE

```

```

0000 160 .SBTTL SYSG$LOAD_TT_STR - Load terminal driver echo strings
0000 161
0000 162 :++
0000 163
0000 164 : FUNCTIONAL DESCRIPTION:
0000 165
0000 166 : Use $GETMSG to fetch message defintions, then load into non-page
0000 167 : pool so that TTDRIVER echos them correctly.
0000 168
0000 169
0000 170 : CALLING SEQUENCE:
0000 171
0000 172 : CALLS #0,SYSG$LOAD_TT_STR
0000 173
0000 174 : INPUT PARAMETERS:
0000 175 : NONE
0000 176
0000 177 : IMPLICIT INPUTS:
0000 178
0000 179 : SYSG_IT$_ message symbols
0000 180
0000 181 : OUTPUT PARAMETERS:
0000 182 : NONE
0000 183
0000 184 : IMPLICIT OUTPUTS:
0000 185 : NONE
0000 186
0000 187 : COMPLETION CODES:
0000 188
0000 189 : SSS_NORMAL
0000 190 : error status from:
0000 191 : $GETMSG
0000 192 : LIB$GET_VM
0000 193 : exec pool allocation
0000 194
0000 195 : SIDE EFFECTS:
0000 196
0000 197 : Non-paged pool is allocated and left with tables set up.
0000 198 : Pointer to tables is modified in TTDRIVER.
0000 199
0000 200 :--
0000 201
0000 202
03FC 0000 203 :.ENTRY SYSG$LOAD_TT_STR ,^M<R2,R3,R4,R5,R6,R7,R8,R9>
0002 204
0002 205 : Set tables up in P0 space first
0002 206
000003A'EF DF 0002 207 PUSHAL STR_BLOCK ; Block addres (returned)
000003E'EF DF 0008 208 PUSHAL NUM_BYTES ; Number of bytes (input)
0000000'GF 02 FB 000E 209 CALLS #2,G^LIB$GET_VM ; Fetch memory
01 50 E8 0015 210 BLBS R0,10$ ; Continue if ok
04 0018 211 RET ; Return with error
0019 212 10$:
0000036'EF DF 0019 213 PUSHAL VEC_TABLE ; Block addres (returned)
000003E'EF DF 001F 214 PUSHAL NUM_BYTES ; Number of bytes (input)
0000000'GF 02 FB 0025 215 CALLS #2,G^LIB$GET_VM ; Fetch memory
01 50 E8 002C 216 BLBS R0,15$ ; Continue if ok

```

```

04 002F 217 RET ; Return with error
      0030 218 15$:
52 0000010E'EF 9E 0030 219 MOVAB MSG_TABLE,R2 ; Message table
53 00000036'EF D0 0037 220 MOVL VEC_TABLE,R3 ; address of memory for vectors
54 00000042'EF 9E 003E 221 MOVAB OFFSET_TABLE,R4 ; table of indirect pointers offsets
55 0000003A'EF D0 0045 222 MOVL STR_BLOCK,R5 ; where text goes
      004C 223
      004C 224 ; Loop through MSG_TABLE, calling $GETMSG for strings
      004C 225
      004C 226 20$:
      56 82 D0 004C 227 MOVL (R2)+,R6 ; Get next message
      12 12 004F 228 BNEQ 30$ ; Zero marks break in table
      0051 229
      0051 230 ; Save offset to table vector
      0051 231
      84 64 53 D0 0051 232 MOVL R3,(R4) ; record next pointer
      00000036'EF 12 0054 233 SUBL VEC_TABLE,(R4)+ ; minus base address
      005B 234
      56 82 D0 005B 235 MOVL (R2)+,R6 ; get real message
      03 12 005E 236 BNEQ 30$ ; If zero, then done
      00CC 31 0060 237 BRW 100$ ; exit loop
      0063 238 30$:
      83 63 55 D0 0063 239 MOVL R5,(R3) ; set string address
      0000003A'EF C2 0066 240 SUBL STR_BLOCK,(R3)+ ; minus base address
      006D 241
      57 000000A2'EF 9E 006D 242 MOVAB BUFFER_DESC,R7 ; address
      67 0064 3F 3C 0074 243 MOVZWL #BUFFER_LEN,(R7) ; set length
04 A7 000010AA'EF 9E 0079 244 MOVAB BUFFER_ADDR,4(R7) ; Set address
      0081 245
      0081 246 $GETMSG_S -
      0081 247 MSGID = R6,- ; message ID
      0081 248 MSGLEN = (R7),- ; return message length
      0081 249 BUFADR = (R7),- ; return message string
      0081 250 FLAGS = #1,- ; text only
      0081 251 OUTADR = MSG_OUTADR ; output counts
      09 50 E9 0096 252 BLBC R0,35$ ; check for error
50 00000000'8F D1 0099 253 CMPL #SS$-MSGNOTFND,R0 ; continue if OK
      03 12 00A0 254 BNEQ 40$ ; must be normal, exit if MSGNOTFND
      0089 31 00A2 255 BRW 90$
      00A5 256 35$:
00000092'EF 67 3C 00A5 257 MOVZWL (R7),MSGLEN ; set length
51 00000097'EF 9A 00AC 258 MOVZBL FAO_COUNT,R1 ; fao count for CRLF
      38 13 00B3 259 BEQL 50$ ; Branch if none
      00B5 260
      00B5 261 ; Set up FAO to insert <CR><LF> pairs
      00B5 262
      58 0000009A'EF 9E 00B5 263 MOVAB STR_DESC,R8 ; descriptor
      68 0064 8F 3C 00BC 264 MOVZWL #BUFFER_LEN,(R8) ; length
      04 A8 01 A5 9E 00C1 265 MOVAB 1(R5),4(R8) ; address
      00C6 266
      00C6 267 $FAOL_S CTRSTR = (R7),- ; control string
      00C6 268 OUTLEN = MSGLEN,- ; length of output
      00C6 269 OUTBUF = (R8),- ; descriptor
      00C6 270 PRMLST = FAO_PRMLST ; list of CR LF's
      OD 50 E8 00DD 271 BLBS R0,50$ ; branch if ok
      56 DD 00E0 272 PUSHL R6 ; message that went bad
      50 DD 00E2 273 PUSHL R0 ; error from FAO

```



```

00000000'GF 02 FB 00E4 274 CALLS #2,G^LIB$SIGNAL ; signal
                39 11 00EB 275 BRB 80$
                00ED 276
51 00000092'EF 9A 00ED 277 50$: MOVZBL MSGLEN,R1 ; Get message length
    65 51 90 00F4 278 MOVB R1,(R5) ; set length
    55 D6 00F7 279 INCL R5 ; add to address
    59 55 D0 00F9 280 MOVL R5,R9 ; save address
    55 51 C0 00FC 281 ADDL2 R1,R5 ; add to address
                00FF 282
                00FF 283 ; Build ^Z strings for TTY$EXIT_STRING logical name (for RTL use)
                00FF 284
50 0000015A'EF 9E 00FF 285 MOVAB LNM_CTRLZ_DEC,R0 ; assume DEC crt
00000000'BF 56 D1 0106 286 CMPL R6,#SYSG_TTS_CTRLZ_DEC ; ^Z for DEC CRT
    10 13 010D 287 BEQL 60$ ; br if equal
50 00000166'EF 9E 010F 288 MOVAB LNM_CTRLZ,R0 ; set regular
00000000'BF 56 D1 0116 289 CMPL R6,#SYSG_TTS_CTRLZ ; Standard ^Z
    07 12 011D 290 BNEQ 80$ ; br if not
                011F 291 60$:
    60 51 B0 011F 292 MOVW R1,(R0) ; set length
    04 A0 59 D0 0122 293 MOVL R9,4(R0) ; set address
                0126 294
                FF23 31 0126 295 80$: BRW 20$ ; Loop
                0129 296
                0129 297 ; Error - exit
                0129 298
                0129 299 85$:
    04 F0 0129 300 INSV #STSS$ SEVERE,-
    00 012B 301 #STSS$ SEVERITY,-
    50 03 04 012C 302 #STSS$ SEVERITY,R0 ; make it fatal
                012E 303 90$: RET ; return
                012F 304
                012F 305 ; Table is complete, define logical name and set up in pool
                012F 306
                012F 307 100$:
                012F 308 $CRELNM_S -
                012F 309 TABNAM = LNM_SYSTEM,-
                012F 310 LOGNAM = LNM_LOGNAM,-
                012F 311 ACMODE = #PS[$C EXEC,-
                012F 312 ITMLST = LNM_ITMLST
    OF 50 E9 U14D 313 BLBC R0,110$
                0150 314
                0150 315 $CMKRNL_S LOAD_TT_SYMS ; Do the loading in kernel mode
    04 015F 316 110$: RET ; Return with status
                0160 317
    0000 0160 318 LOAD_TT_SYMS: .WORD 0
                0162 319
56 53 00000036'EF C3 0162 320 SUBL3 VEC_TABLE,R3,R6 ; length of VECTOR table
57 55 0000003A'EF C3 016A 321 SUBL3 STR_BLOCK,R5,R7 ; length of text
    51 57 56 C1 0172 322 ADDL3 R6,R7,R1 ; length of both
                0176 323
    51 0C C0 0176 324 ADDL #12,R1 ; pool overhead
                0179 325
                0179 326 SETIPL #IPL$ ASTDEL ; Prevent deletion
    00000000'GF 16 017C 327 JSB G^EXE$ALONONPAGED ; Fetch Pool
    6C 50 E9 0182 328 BLBC R0,190$ ; Branch on error
                0185 329
62 20595454 8F D0 0185 330 MOVL #^A/TTY /,(R2) ; flag pool with text

```

```

04 A2 54584554 8F D0 018C 331 MOVL #^A/TEXT/,4(R2) ; just for grins
      08 A2 51 B0 0194 332 MOVW R1,IRPSW $SIZE(R2) ; size (IRP offset is as good as any)
      80 8F 9B 0198 333 MOVZBW #DYN$C_SPECIAL,- ; Set ^ype ?
      0A A2 0198 334 IRPSB_TYPE(R2)
      019D 335
      019D 336 ; Relocate offsets to these addresses
      019D 337
      52 DD 019D 338 PUSHL R2 ; Save structure address
68 00000036'FF 58 0C A2 9E 019F 339 MOVAB 12(R2),R8 ; Base of vectors
      56 28 01A3 340 MOVCS R6,@VEC_TABLE,(R8) ; copy data
      01AB 341
      01AB 342 ; R3 is now where the string block will start
      01AB 343 ; Vector offsets just loaded must be relocated relative to this
      01AB 344
      88 53 C0 01AB 345 30$: ADDL2 R3,(R8)+ ; add address to offset
      58 53 D1 01AE 346 ; addresses the same?
      F8 12 01B1 347 BNEQU 30$ ; no, loop
      01B3 348
      01B3 349 ; Now copy strings into block
      01B3 350
63 0000003A'FF 57 28 01B3 351 MOVCS R7,@STR_BLOCK,(R3) ; copy strings
      52 8ED0 01BB 352 POPL R2 ; Restore structure address
      01BE 353
      01BE 354 ; Block is now set-up, fix up pointers in TTDRIVER
      01BE 355
50 00000000'GF 50 01BE 356 MOVL G^TTY$GL_DPT,R0 ; Fetch TTDRIVER address
      51 1E A0 3C 01C5 357 MOVZWL DPT$W_VECTOR(R0),R1 ; offset to table
      51 50 C0 01C9 358 ADDL2 R0,R1 ; base + offset
      50 24 A1 D0 01CC 359 MOVL CLASS_TABLES(R1),R0 ; address of vectors
54 00000042'EF 9E 01D0 360 MOVAB OFFSET_TABLE,R4 ; address of new vectors
      01D7 361
      01D7 362 SETIPL #IPL$POWER ; block out interrupts
      01DA 363
      58 0C A2 9E 01DA 364 MOVAB 12(R2),R8 ; base address
      80 84 58 C1 01DE 365 ADDL3 R8,(R4)+,(R0)+ ; new interrupt echos
      80 84 58 C1 01E2 366 ADDL3 R8,(R4)+,(R0)+ ; new exit echos
      80 84 58 C1 01E6 367 ADDL3 R8,(R4)+,(R0)+ ; new ctrl 0 echos
      01EA 368
50 00000000'8F 50 01EA 369 MOVL #SS$NORMAL,R0 ; Set status
      01F1 370 190$:
      01F1 371 SETIPL #0
      04 01F4 372 RET
      01F5 373
      01F5 374 .END

```

SYSGTERM
Symbol table

- SYSGEN TERMINAL commands

M 13

16-SEP-1984 00:15:26 VAX/VMS Macro V04-00
4-SEP-1984 23:07:23 [BOOTS.SRC]SYSGTERM.MAR;1

BUFFER_ADDR	000000AA	R	02	TTY\$GL DPT	*****	X	03
BUFFER_DESC	000000A2	R	02	VEC_TABLE	00000036	R	02
BUFFER_LEN	= 00000064						
BYTES_TO_ALLOCATE	= 000003E8						
CLASS_TABLES	= 00000024						
CRLF	00000000	R	02				
DPT\$W_VECTOR	= 0000001E						
DYN\$C_SPECIAL	= 00000080						
EXESA(ONONPAGED	*****	X	03				
FAO_COUNT	00000097	R	02				
FAO_PRMLST	0000000A	R	02				
IPL\$ASTDEL	= 00000002						
IPL\$POWER	= 0000001F						
IRPSB_TYPE	= 0000000A						
IRPSW_SIZE	= 00000008						
LIB\$GET_VM	*****	X	03				
LIB\$SIGNAL	*****	X	03				
LNMS_STRING	= 00000002						
LNM_CTRLZ	00000166	R	02				
LNM_CTRLZ_DEC	0000015A	R	02				
LNM_ITMLST	0000015A	R	02				
LNM_LOGNAM	0000018E	R	02				
LNM_SYSTEM	00000176	R	02				
LOAD TT_SYMS	00000160	R	03				
MSGLN	00000092	R	02				
MSG_OUTADR	00000096	R	02				
MSG_TABLE	0000010E	R	02				
NUM_BYTES	0000003E	R	02				
OFFSET_TABLE	00000042	R	02				
PR\$IPC	*****	X	03				
PSL\$C_EXEC	= 00000001						
SS\$MSGNOTFND	*****	X	03				
SS\$NORMAL	*****	X	03				
STR_BLOCK	0000003A	R	02				
STR_DESC	0000009A	R	02				
ST\$K_SEVERE	= 00000004						
ST\$S_SEVERITY	= 00000003						
ST\$V_SEVERITY	= 00000000						
SY\$C\$KRNL	*****	GX	03				
SY\$C\$RELNM	*****	GX	03				
SY\$C\$FAOL	*****	GX	03				
SY\$C\$GETMSG	*****	GX	03				
SY\$G\$LOAD TT STR	00000000	RG	03				
SY\$G_TTS_CTRLZ	*****	X	02				
SY\$G_TTS_CTRLZ_DEC	*****	X	02				
SY\$G_TTS_CTRLZ_DECREG	*****	X	02				
SY\$G_TTS_CTRLZ_REG	*****	X	02				
SY\$G_TTS_CTRLZ	*****	X	02				
SY\$G_TTS_CTRLZ_DEC	*****	X	02				
SY\$G_TTS_CTRLZ_DECREG	*****	X	02				
SY\$G_TTS_CTRLZ_REG	*****	X	02				
SY\$G_TTS_CTRLZ	*****	X	02				
SY\$G_TTS_CTRLZ_DEC	*****	X	02				
SY\$G_TTS_OUTON	*****	X	02				
SY\$G_TTS_OUTON_DEC	*****	X	02				

! 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
NONPAGED_DATA	000001A5 (421.)	J2 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
NONPAGED_CODE	000001F5 (501.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.07	00:00:00.43
Command processing	110	00:00:00.68	00:00:03.04
Pass 1	224	00:00:05.81	00:00:12.37
Symbol table sort	0	00:00:00.71	00:00:01.03
Pass 2	79	00:00:01.38	00:00:02.72
Symbol table output	7	00:00:00.07	00:00:00.27
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	454	00:00:08.75	00:00:19.89

The working set limit was 1350 pages.
31137 bytes (61 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 513 non-local and 15 local symbols.
374 source lines were read in Pass 1, producing 21 object records in Pass 2.
21 pages of virtual memory were used to define 20 macros.

! Macro library statistics !

Macro library name	Macros defined
-\$255\$DUA28:[BOOTS.OBJ]BOOTS.MLB;1	0
-\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	6
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	17

599 GETS were required to define 17 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$SYSGTERM/OBJ=OBJ\$SYSGTERM MSRC\$SYSGTERM/UPDATE=(ENHS\$SYSGTERM)+EXECMLS/LIB+LIB\$BOOTS.MLB/LIB

