

[illegible]

Syl
Syl
SS
SS
BE
BO
CL
CL
CL
CL
CTI
DEI
DEI
DEI
DEI
DEI
DFI
DII
DO
DO
ERI
EXI
EXI
FAI
FAI
FAI
FII
HDI
IAI
IAI
IHI
IMK
IMK
IMK
IMK
IMK
IMK
IMK
IMK
IMK
LNI
LNI
MAI
MAI
NAI
PRI
SAI
SAI
SCI
SEI
SSI
SSI
SSI
SSI

(2)	63	DECLARATIONS
(3)	139	DEBUG_BOOT - DEBUG/TRACEBACK/IMAGE ACTIVATOR
(4)	194	Start the Program or DEBUG
(5)	229	Exception Handler for TRACE
(6)	312	Map in and Start DEBUG or TRACE

```
0000 1 .TITLE SYSIMGSTA - Image Startup System Service
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: EXECUTIVE
0000 31
0000 32 Abstract:
0000 33 This code is part of the System image. It is entered directly
0000 34 as the 'DEBUG' transfer address, even when the real DEBUG
0000 35 will not be invoked, since it is this 'bootstrap' which
0000 36 decides whether to
0000 37 1) simply start up the user image directly,
0000 38 2) map the bulk of DEBUG into the users' P0 space
0000 39 and transfers control to DEBUG,
0000 40 3) catch an SSS DEBUG signal and arrange to have
0000 41 a mapped DEBUgger attached to a running process,
0000 42 or
0000 43 4) catch any other signal, report it, and try to
0000 44 invoke the mapped image which prints a 'traceback' or
0000 45 symbolic dump of where the program was when it faulted.
0000 46
0000 47 Environment:
0000 48 Runs at user mode, non-AST level in the System address space.
0000 49
0000 50 Author: Carol Peters, Creation date: 21 July 1977
0000 51
0000 52 Modified by:
0000 53
0000 54 V03-002 TMK0001 Todd M. Katz 04-Apr-1984
0000 55 Increase the maximum size of a LIB$DEBUG/LIB$TRACE translation
0000 56 from 64 to LNMSC_NAMLENGTH.
0000 57
```


SYSIMGSTA
V04-000

- Image Startup System Service

H 2

16-SEP-1984 02:20:47 VAX/VMS Macro V04-00
5-SEP-1984 03:54:48 [SYS.SRC]SYSIMGSTA.MAR;1

Page 2
(1)

0000 58 :
0000 59 :
0000 60 :
0000 61 :--

V03-001 WMC0001 Wayne Cardoza 04-Apr-1983
Try to take an image dump if appropriate.


```
0000 63 .SBTTL DECLARATIONS
0000 64
0000 65 ;
0000 66 ; Include files:
0000 67 ;
0000 68 $CLIDEF ; Command interpreter flags
0000 69 $FABDEF ; file access block definitions
0000 70 $IACDEF ; Image activation control flags
0000 71 $IHDDEF ; image header definitions
0000 72 $LNMDEF ; logical name system service definitions
0000 73 $NAMDEF ; name block definitions
0000 74 $PHDDEF ; process header block definitions
0000 75 $SECDEF ; process/global section definitions
0000 76 $SSDEF ; system service status definitions
0000 77 $STSDEF ; status code definitions
0000 78
0000 79 $IMGACTDEF ; Image activator argument definitions
0000 80 ;
0000 81 ; Macros:
0000 82 ;
0000 83 ;
0000 84 ; Equated symbols:
0000 85 ;
00000001 0000 86 IMGHDRBLKCNT=1 ; Number of image header blocks on
0000 87 ; the front of the DEBUG/TRACE images
0000 88 ;
0000 89 ; Offsets into the vector at the front of the DEBUG and TRACE images
0000 90 ;
0000 91 $OFFSET 0,POSITIVE,<-
0000 92 DEBUG_WRT_PAGES,-
0000 93 DEBUG_WRT_ENDPG,-
0000 94 DEBUG_START_ADR -
0000 95 >
0000 DEBUG_WRT_PAGES:
0004 DEBUG_WRT_ENDPG:
0008 DEBUG_START_ADR:
0000 96
0000 97 ;
0000 98 ; Offsets from R6 used by the mapin logic
0000 99 ;
0000 100 $OFFSET 0,POSITIVE,<-
0000 101 <,<<IMGACT$ NARGS+1>*4>>,- ; Argument list for $IMGACT call
0000 102 <IMGACT_INADR,8>,- ; Input address range to map image at
0000 103 <IMGACT_RETADR,8>,- ; Ret adr range where image was mapped
0000 104 <NAMDSC,8>,- ; Descriptor for untranslated file name
0000 105 <DFLTNAMDSC,8>,- ; Descriptor for default file string
0000 106 <TRNNAMDSC,8>,- ; Descriptor for translated file spec
0000 107 <TRNNAMESTRING,LNM$C_NAMLENGTH>,- ; Buffer for translated file spec
0000 108 <HDRBUF,512>,- ; Image header buffer for $IMGACT
0000 109 <SCRATCHSIZE,0>,- ; size of area address off of FP
0000 110 >
0024 IMGACT_INADR:
002C IMGACT_RETADR:
0034 NAMDSC:
003C DFLTNAMDSC:
0044 TRNNAMDSC:
004C TRNNAMESTRING:
```



```
014B      HDRBUF:
034B      SCRATCHSIZE:
0000 111 :
0000 112 : An 'arg' vector is passed to TRACE using the following
0000 113 : locations thru SECT_BOUNDS. FAULT_PC must remain the first
0000 114 : of these locations (we use it to address the vector), and
0000 115 : the order of these variables must not change.
0000 116 : The storage for this vector and any additional own data is allocated
0000 117 : off the end of the call frame generated by the call to BOOT_START.
0000 118 : The FP needed to address this own data is found at exception time
0000 119 : as the "establisher frame" in the "mechanism array."
0000 120 :
0000 121 : SOFFSET 0, POSITIVE, <-
0000 122 :     FAULT_PC, -           ; Pass the PC on to TRACE.EXE
0000 123 :     FAULT_FP, -           ; Pass to TRACE the FP when the fault occurs
0000 124 :     <PROG_START, 0>, -    ; Program start address
0000 125 :     FIRST_FP, -          ; What FP was when user image was invoked in
0000 126 :     SIGNAL_ARRAY, -      ; Address of signal array.
0000 127 :     <SECT_BOUNDS, 8>, -   ; Beginning of TRACE's symbol table,
0000 128 -:
0000 129 -: end of TRACE's 'arg' vector.
0000 130 -: beginning of TRACE's additional own data
0000 131 -:
0000 132 -: Additional own storage variables are to be added here
0000 133 -:
0000 134 :     SAVED_AP, -           ; Saved AP that image was invoked with
0000 135 :     MAP_SEC_XADDR, -      ; Where to JMP to invoke TRACE, 0 if not map
0000 136 :     <TRACE_CTX_SIZ, 0>, - ; Size of own storage reserved at end of fra
0000 137 :
0000      FAULT_PC:
0004      FAULT_FP:
0008      PROG_START:
0008      FIRST_FP:
000C      SIGNAL_ARRAY:
0010      SECT_BOUNDS:
0018      SAVED_AP:
001C      MAP_SEC_XADDR:
0020      TRACE_CTX_SIZ:
```



```
0000 139 .SBTTL DEBUG_BOOT - DEBUG/TRACEBACK/IMAGE ACTIVATOR
0000 140
0000 141 :++
0000 142 : Functional description:
0000 143 : This 'bootstrap' invokes one of DEBUG, TRACEback, or the
0000 144 : users' image, depending on flags passed on by/from LINK
0000 145 : and CLI. In all cases, we set up the argument list, etc,
0000 146 : in such a way that the image being invoked does not know
0000 147 : that DBGBOOT was there. If DEBUG or TRACE is selected,
0000 148 : this routine finds out where the user program's P0 space ends.
0000 149 : It then maps the DEBUG/TRACE global section into the space following
0000 150 : the user's P0 space, marks the DEBUG/TRACE impure storage area
0000 151 : as 'copy on reference', and transfers control to the initialization
0000 152 : code in the DEBUG/TRACE global section.
0000 153
0000 154 : Calling sequence:
0000 155 : CALLS #6, EXE$IMGSTA
0000 156
0000 157 : Input parameters:
0000 158 : 4(AP) - transfer vector address
0000 159 : 8(AP) - address of CLI parse information (not used)
0000 160 : 12(AP) - address of image header information
0000 161 : 16(AP) - address of image file information
0000 162 : 20(AP) - LINK status bits
0000 163 : 24(AP) - CLI status bits
0000 164
0000 165 : Implicit inputs:
0000 166 : We may use one of two image files, DEBUG.EXE or TRACE.EXE, both
0000 167 : located in the logical device SYS$LIBRARY.
0000 168
0000 169 : Output parameters:
0000 170 : R0 - system status code.
0000 171 : = $$$_RESIGNAL if error code from $CRMPSC or $MGBLSC
0000 172 : or anything else in this driver module fails
0000 173 : = $$$_NORMAL or other DBG$... returns from the rest
0000 174 : of DEBUG in the other case.
0000 175
0000 176 : Implicit outputs:
0000 177 : The stack frame created when this BOOT is called (by CLI)
0000 178 : remains around to contain the frame handler we need to
0000 179 : catch programs which fault when there is no DEBUG
0000 180 : around.
0000 181
0000 182 : Completion codes:
0000 183 : $$$_NORMAL - successfully mapped in D:\BUG/TRACE
0000 184 : $$$_RESIGNAL - unable to map in DEBUG/TRACE
0000 185
0000 186 : Side effects:
0000 187 : Lets user run DEBUG, his image directly, or TRACEback.
0000 188 :--
0000 189
00000000 190 .PSECT YF$$$SYSIMGSTA B"TE,EXE
0000 191
0000 192 .LIST MEB
```



```
0000 194 .SBTTL Start the Program or DEBUG
0000 195
0000 196 .ENTRY EXESIMGSTA,0
0002 197
0002 198 ASSUME TRACE_CTX_SIZ-4 EQ MAP_SEC_XADDR
0002 199 CLRL -(SP) ; no mapped section transfer address
18 5E 7E D4 0004 200 SUBL #TRACE_CTX_SIZ-4,SP ; reserve traceback own storage
50 AE 1C C2 0007 201 MOVL AP,SAVED_AP(SP) ; Save AP that program is called with
08 AE 04 AC D0 000B 202 MOVL 4(AP),R0 ; Address of transfer vector
08 AE 04 A0 D0 000F 203 MOVL 4(R0),PROG_START(SP) ; Save program start address and avoid
18 AC 00 E0 0014 204 ; referencing the transfer vector after purg
18 AC 05 E0 0014 205 BBS S^#CLISV DEBUG,24(AP),- ; See if command-line override given
14 AC 00 E0 0018 206 CLI_OVERRIDE
14 AC 51 E0 0019 207 BBS S^#IHD$V LNKDEBUG,20(AP),- ; No override - check for $LINK/DEBUG
18 AC 01 E0 001D 208 DEBUG_MAPIN
18 AC 4C E0 001E 209 CLI_OVERRIDE:
001E 210 BBS S^#CLISV DBGTRU,24(AP),- ; Check for $RUN/DEBUG
0022 211 DEBUG_MAPIN
0023 212 ; DEBUG will not come up - either because "$RUN program" was given along with
0023 213 ; $LINK program (i.e. no $LINK/DEBUG or $RUN/DEBUG), or because "$RUN/NODEBUG"
0023 214 ; as given.
0023 215 ;
0023 216
0023 217 BE_SILENT:
0023 218 ; Set up a frame handler to look
0023 219 ; after invoking traceback if the
0023 220 ; 'standalone' image faults if
0023 221 ; the user types ^Y DEBUG to attach
0023 222 ; DEBUG to an already-running image.
6D 30 AF DE 0023 223 MOVAL B^BOOT_HANDLER,(FP)
04 AC 04 C0 0027 224 ADDL #4,4(AP) ; Adjust transfer vector to ignore
08 BE 6C FA 002B 225 ; the fact that DEBUG is around
08 BE 04 04 002B 226 CALLG (AP),@PROG_START(SP) ; and invoke the user program directly.
002F 227 RET
```



```
0030 229 .SBTTL Exception Handler for TRACE
0030 230
0030 231 ;
0030 232 ; Control is passed to DEBUG or TRACEBACK with a JMP instruction. The
0030 233 ; register save mask at BOOT_HANDLER thus serves as the register save
0030 234 ; mask for these procedures. Both DEBUG and TRACEBACK assume that all
0030 235 ; registers have been saved.
0030 236 ;
0030 237 ;
0030 238 BOOT_HANDLER:
0030 239 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; save mask
50 04 AC 7D 0032 240 MOVQ 4(AP),R0 ; Pick up pointer to signal arg list.
58 04 A1 20 C3 0036 241 ; R1 = address of mechanism array
0036 242 SUBL3 #TRACE_CTX_SIZ,4(R1),R8 ; Address of own storage
003B 243 ;
003B 244 ; The debugger is started up if the exception name is SS$_DEBUG.
003B 245 ; Traceback happens if error level is WARNING, SEVERE, or ERROR.
003B 246 ; Otherwise we let DCL's frame handler decide
003B 247 ; what to do with this exception.
003B 248 ;
0000046C 8F 04 A0 D1 003B 249 Cmpl 4(R0),#SS$_DEBUG ; if DEBUG exception,
52 60 03 20 EF 0043 250 BEQL DO_DEBUG ; then start up the debugger
004A 251 EXTZV #ST$$_SEVERITY+32,#ST$$_SEVERITY,(R0),R2 ;
004A 252 ; Check for exact match to the
004A 253 ; severity levels which cause TRACE
52 00 91 004A 254 CMPB #ST$$_WARNING,R2
52 02 91 004D 255 BEQL DO_TRACE
52 04 91 004F 256 CMPB #ST$$_ERROR,R2
52 04 91 0052 257 BEQL DO_TRACE
50 0918 8F 3C 0054 258 CMPB #ST$$_SEVERE,R2
0057 259 BEQL DO_TRACE
0059 260 MOVZWL #SS$_RESIGNAL,R0 ; This signal is not for us.
005E 261 RET
005F 262 ;
005F 263 ; Start up the debugger as a result of a DEBUG exception.
005F 264 ;
005F 265 DO_DEBUG:
5C 52 5C D0 005F 266 MOVL AP,R2 ; save condition handler AP
5C 18 A8 D0 0062 267 MOVL SAVED_AP(R8),AP ; AP when image was first invoked
04 AC 52 D0 0066 268 MOVL R2,4(AP) ; replace start vector address with
00 18 AC 10 E2 006A 269 ; condition handler AP
006A 270 BBSS #CLISV_DBGEXCP,24(AP),10$ ; DEBUG exception triggered
006F 271 ; the starting of the debugger
006F 272 10$:
006F 273 ;
006F 274 ; Map the debugger in and transfer control to it
006F 275 ;
006F 276 DEBUG_MAPIN:
57 C4'AF 9E 006F 277 MOVAB B^DEBUGNAME,R7 ; Address of debugger name string
0080 30 0073 278 BSBW MAPIN ; Map it in
61 17 0076 279 JMP (R1) ; and transfer control to it
0078 280 ;
0078 281 ; Start up TRACE
0078 282 ;
0078 283 DO_TRACE:
0078 284 ;
0078 285 ; If status is SEVERE, check to see if image dump was requested.
```



```
04 52 91 0078 286 ;
0C 12 0078 287 ; CMPB R2,#STSSK_SEVERE
7E 50 7D 007B 288 ; BNEQ 5$
00000000 EF 16 007D 289 ; MOVQ R0,-(SP)
50 8E 7D 0080 290 ; JSB EX$IMGDMP_MERGE ; Take dump if it was requested
; We will ignore errors
0086 291 ;
0089 292 ;
0089 293 ; Build a vector of args to TRACEback and pass on a pointer to this vector
0089 294 ; in the same way as DEBUG usually gets a pointer to the transfer vector.
0089 295 ;
08 0C A8 50 D0 0089 296 5$: MOVL R0,SIGNAL_ARRAY(R8) ; Put signal arg ptr into traceback vector.
08 A8 04 A1 D0 008D 297 ; MOVL 4(R1),FIRST_FP(R8) ; Put first FP into traceback vector.
04 A8 0C AD D0 0092 298 ; MOVL 12(FP),FAULT_FP(R8) ; FP at time of the exception
5C 60 9A 0097 299 ; MOVZBL (R0),AP ; Signal argument count
68 70 4C D0 009A 300 ; MOVL -(R0)[AP],FAULT_PC(R8) ; Put the fault PC in the vector
5C 18 A8 D0 009E 301 ; MOVL SAVED_AP(R8),AP ; AP when image was first invoked
04 AC 58 D0 00A2 302 ; MOVL R8,4(AP) ; Pass address of traceback vector
; in place of the transfer vector
51 1C A8 D0 00A6 303 ; MOVL MAP_SEC_XADDR(R8),R1 ; Is TRACE already mapped in?
0D 12 00AA 304 ; BNEQ 10$ ; Branch if it is
10 A8 7C 00AC 305 ; CLRQ SECT_BOUNDS(R8) ; Init section bounds for TRACE
57 BB AF 9E 00AF 306 ; MOVAB B*TRACENAME,R7 ; Mapping in TRACE
41 10 00B3 307 ; BSBB MAPIN ; Go map the image in
1C A8 51 D0 00B5 308 ; MOVL R1,MAP_SEC_XADDR(R8) ; Save its transfer address
61 17 00B9 309 ; JMP (R1) ; and transfer control to TRACE/DEBUG
10$: JMP (R1)
```



```
00BB 312 .SBTTL Map in and Start DEBUG or TRACE
00BB 313 :
00BB 314 : DEBUG and TRACE name strings
00BB 315 :
00BB 316 TRACENAME:
45 43 41 52 54 24 42 49 4C 00BB 317 .ASCII /LIB$TRACE/ ; Logical name to translate into file
; spec and file name prefixed by LIB$
00C4 318
00C4 319 LNAME$IZ=.-TRACENAME
00C4 320
00C4 321 DEBUGNAME:
47 55 42 45 44 24 42 49 4C 00C4 322 .ASCII /LIB$DEBUG/ ; Logical name to translate into file
; spec and file name prefixed by LIB$
00CD 323
00CD 324 ASSUME LNAME$IZ EQ .-DEBUGNAME
00CD 325
00CD 326 DEFAULTNAME$DSC:
49 4C 24 53 59 53 000000D5'010E0000' 00CD 327 .ASCID /SYS$LIBRARY:.EXE/ ; RMS default name string for $IMGACT
45 58 45 2E 3A 59 52 41 52 42 00DB
00E5 328
00E5 329 FACNAME$DES: ; str dsc for facility name for err msgs
42 47 55 42 45 44 000000ED'010E0000' 00E5 330 .ASCID /DEBUGBOOT/
54 4F 4F 00F3
00F6 331 :
00F6 332 : Map DEBUG or TRACE in as a shared GLOBAL section or as a process section.
00F6 333 :
00F6 334 : CALLING SEQUENCE:
00F6 335 :
00F6 336 : BSBW MAPIN
00F6 337 :
00F6 338 : INPUTS:
00F6 339 :
00F6 340 : R7 = Address of name string for DEBUG or TRACE
00F6 341 :
00F6 342 : OUTPUTS:
00F6 343 :
00F6 344 : R0 = SS$NORMAL or SS$RESIGNAL
00F6 345 : R1 = Transfer address for entering the desired program
00F6 346 : R2,R3,R4,R5,R6 SCRATCH
00F6 347 :
00F6 348 : If any error occurs, this routine exits via RET with status in R0
00F6 349 :
00F6 350 MAPIN:
5E FC85 CE DE 00F6 351 MOVAL -SCRATCHSIZE(SP),SP ; reserve scratch storage on the stack
00FB 352 MOVL SP,R6 ; R6 = base of scratch area
6E 034B 8F 00 6E 00 2C 00FE 353 MOVCS #0,(SP),#0,SCRATCHSIZE,(SP) ; zero the scratch storage
50 00000000'GF DO 0106 354 MOVL G^CTL$GL_PHD,R0 ; address of process header window
010D 355 :
010D 356 : Initialize the argument list for $IMGACT merge request system service call.
010D 357 :
010D 358 MOVL #IMGACT$NARGS,(R6) ; Set argument count for $IMGACT call
0110 359 MOVAB NAME$DSC(R6),IMGACT$NAME(R6) ; Set adr of input file name desc
0115 360 MOVAB DEFAULTNAME$DSC,IMGACT$DFLNAM(R6) ; Set adr of default name str
011A 361 MOVL #<IACSM MERGE ! IACSM-EXPREG>,IMGACT$IMGCTL(R6) ; Set ctl flags
0C A6 014B C6 9E 011E 362 MOVAB HDRBUF(R6),IMGACT$HDRBUF(R6) ; Set adr of image header buffer
14 A6 24 A6 9E 0124 363 MOVAB IMGACT_INADR(R6),IMGACT$INADR(R6) ; Set adr of input va range
18 A6 2C A6 9E 0129 364 MOVAB IMGACT_RETADR(R6),IMGACT$RETADR(R6) ; Set adr of return range
1C A6 D4 012E 365 CLRL IMGACT$IDENT(R6) ; No match ident specified
24 A6 0200 8F 3C 0131 366 MOVZWL #^X200,IMGACT_INADR(R6) ; Set a blueprint P0 address range for
```



```
28 A6 3FFFFFFF 8F D0 0137 367      MOVL #1230-1,IMGACT_INADR+4(R6) ; mapping to first free va space
      34 A6 09 9A 013F 368      MOVZBL #LNAMESIZ,NAMDISC(R6) ; Init size of name to translate
      38 A6 57 D0 0143 369      MOVL R7,NAMDISC+4(R6) ; Init adr of nam to translate
      44 A6 FF 8F 9A 0147 370      MOVZBL #LNMSZ,NAMLENGTH,TRNNAMDISC(R6) ; Set size of translated name buffer
      48 A6 4C A6 9E 014C 371      MOVAB TRNNAMSTRING(R6),TRNNAMDISC+4(R6) ; Set adr of tran name buffer
      0151 372
      0151 373 ; Now do a logical name translation of LIB$DEBUG/LIB$TRACE to get the file
      0151 374 ; specification to pass to $IMGACT. If there is no translation, then strip
      0151 375 ; the LIB$ off the name and pass that.
      0151 376
      0151 377      STRNLOG_S - ; Translate to get file spec
      0151 378      LOGNAM=NAMDISC(R6),- ; Name string to be translated
      0151 379      RSLLEN=TRNNAMDISC(R6),- ; Resultant size of translated string
      0151 380      RSLBUF=TRNNAMDISC(R6) ; ADR of translated string buffer desc
      00 DD 0151      PUSHL #0
      00 DD 0153      PUSHL #0
      00 DD 0155      PUSHL #0
      44 A6 7F 0157      PUSHAQ TRNNAMDISC(R6)
      44 A6 3F 015A      PUSHAQ TRNNAMDISC(R6)
      34 A6 7F 015D      PUSHAQ NAMDISC(R6)
      00000000'GF 06 FB 0160      CALLS #6,G^SYSS$STRNLOG
      50 01 D1 0167 381      CMPL #SS$_NORMAL,R0 ; Successful translation?
      34 A6 44 A6 7D 016A 382      BNEQ 40$ ; Branch if not successful
      0F 11 0171 384      MOVQ TRNNAMDISC(R6),NAMDISC(R6) ; Pass translated string to $IMGACT
      50 0629 8F B1 0173 385 40$:      BRB 60$ ; Go call image activator
      2B 12 0178 386      CMPW #SS$_NOTRAN,R0 ; Was there no logical translation?
      34 A6 04 C2 017A 387      BNEQ ERROR_RET ; No, there was an error, ret w/status
      38 A6 04 C0 017E 388      SUBL2 #4,NAMDISC(R6) ; Strip the LIB$ off the file name that
      0182 389 60$:      ADDL2 #4,NAMDISC+4(R6) ; is passed to the image activator
      00000000'GF 66 FA 0182      $IMGACT_G (R6) ; Let image activator map in the image
      19 50 E9 0189 390      CALLG (R6),G^SYSS$IMGACT
      0F 00 FB 018C 391      BLBC R0,ERROR_RET ; Branch on error to return w/ status
      0F 50 E9 018C 391      $IMGFIX_S ; Do address relocation fixups
      0193 392      CALLS #0,G^SYSS$IMGFIX
      0196 393      BLBC R0,ERROR_RET ; Branch on error to return w/ status
      0196 394
      0196 395 ;+
      0196 396 ; A vector in the DEBUG global section contains the start of impure
      0196 397 ; storage, the end of impure storage, and the transfer address of DEBUG
      0196 398 ; itself. This vector is in PSECT DBG$ABS_ZERO, and is always the first
      0196 399 ; thing in DEBUG.EXE. Since DEBUG.EXE is linked with a base of zero, the
      0196 400 ; vector is the first thing in the global section.
      0196 401
      0196 402 ; The third address in the vector is DEBUG's transfer address.
      0196 403 ; Add the DEBUG transfer address to the base address of the
      0196 404 ; first mapped page and jump to it.
      0196 405
      0196 406 TRANS_CONTROL:
      51 51 2C A6 D0 0196 407      MOVL IMGACT_RETADR(R6),R1 ; Get mapped address of section
      51 51 08 A1 C1 019A 408      ADDL3 DEBUG_START_ADR(R1),R1,R1 ; compute DEBUG start address
      5E 034B C6 DE 019F 409      MOVAL SCRATCHSIZE(R6),SP ; remove scratch storage from stack
      05 01A4 410      RSB ; return transfer address to caller
      01A5 411
      01A5 412 ERROR_RET:
      7E 50 07 CB 01A5 413      BICL3 #STSSM_SEVERITY,R0,-(SP) ; Reduce error code severity to warning
      7E 0F B0 01A9 414      MOVW #15,-(SP) ; Print all pieces of err msg
```


7E	01	B0	01AC	415	MOVW	#1,-(SP)	; Set the number of parameters
50	5E	D0	01AF	416	MOVL	SP,R0	; Save adr of argument list
			01B2	417	SPUTMSG	S(R0),,FACNAMDES	; Report error to user
	00	DD	01B2		PUSHL	#0	
FF2D	CF	7F	01B4		PUSHAQ	FACNAMDES	
	00	DD	01B8		PUSHL	#0	
	60	DF	01BA		PUSHAL	(R0)	
00000000'GF	04	FB	01BC		CALLS	#4,G^SYSSPUTMSG	
50	0918	8F	3C	01C3	418	MOVZWL	#SSS_RESIGNAL,R0
			04	01C8	419	RET	; Resignal the error
				01C9	420		; Return with error status
				01C9	421	.END	

SYSGSTA
Symbol table

- Image Startup System Service

E 3

16-SEP-1984 02:20:47 VAX/VMS Macro V04-00
5-SEP-1984 03:54:48 [SYS.SRC]SYSGSTA.MAR;1

Page 12
(6)

```

$$ARGS          = 00000008
$$T1            = 00000024
BE_SILENT       = 00000023 R    02
BOOT_HANDLER    = 00000030 R    02
CLISV_DBGEXCP   = 00000010
CLISV_DBGTRU    = 00000001
CLISV_DEBUG     = 00000000
CLI_OVERRIDE    = 0000001E R    02
CTL$GL_PHD      = ***** X    02
DEBUGNAME       = 000000C4 R    02
DEBUG_MAPIN     = 0000006F R    02
DEBUG_START_ADR = 00000008
DEBUG_WRT_ENDPG = 00000004
DEBUG_WRT_PAGES = 00000000
DEFAULTNAME$DSC = 000000CD R    02
DFTNAME$DSC     = 0000003C
DIR...          = 00000001
DO_DEBUG        = 0000005F R    02
DO_TRACE        = 00000078 R    02
ERROR_RET       = 000001A5 R    02
EXESIM$GDMF_MERGE = ***** X    02
EXESIM$STA      = 00000000 RG   02
FACNAME$DES     = 000000E5 R    02
FAULT_FP        = 00000004
FAULT_PC        = 00000000
FIRST_FP        = 00000008
HDRBUF          = 0000014B
IACSM_EXPREG    = 00000020
IACSM_MERGE     = 00000010
IHDSV_LNKDEBUG  = 00000000
IMGACT$_ACMODE  = 00000020
IMGACT$_DFLNAM  = 00000008
IMGACT$_HDRBUF  = 0000000C
IMGACT$_IDENT   = 0000001C
IMGACT$_IMGCTL  = 00000010
IMGACT$_INADR   = 00000014
IMGACT$_NAME     = 00000004
IMGACT$_NARGS   = 00000008
IMGACT$_RETADR  = 00000018
IMGACT_INADR    = 00000024
IMGACT_RETADR   = 0000002C
IMGHDRBLKCNT    = 00000001
LNAMESIZ        = 00000009
LNMSC_NAMLENGTH = 000000FF
MAPIN           = 000000F6 R    02
MAP_SEC_XADDR   = 0000001C
NAME$DSC        = 00000034
PROG_START      = 00000008
SAVABS...       = 00000020
SAVED_AP        = 00000018
SCRATCHSIZE     = 0000034B
SECT_BOUNDS     = 00000010
SIGNAL_ARRAY    = 0000000C
SS$_DEBUG       = 0000046C
SS$_NORMAL      = 00000001
SS$_NOTRAN      = 00000629
SS$_RESIGNAL    = 00000918

```

```

STSSK_ERROR     = 00000002
STSSK_SEVERE    = 00000004
STSSK_WARNING   = 00000000
STSSM_SEVERITY  = 00000007
STSS$SEVERITY   = 00000003
STSSV_SEVERITY  = 00000000
SYSSIM$ACT      = ***** GX   02
SYSSIM$FIX      = ***** GX   02
SYSSPUTMSG      = ***** GX   02
SYSTRNLOG       = ***** GX   02
TRACENAME       = 000000BB R    02
TRACE_CTX_SIZ   = 00000020
TRANS_CONTROL    = 00000196 R    02
TRNNAME$DSC     = 00000044
TRNNAMESTRING   = 0000004C

```

SY
VO

+-----+
! 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	00000348 (843.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
YF\$\$\$SYSIMGSTA	000001C9 (457.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.09	00:00:00.56
Command processing	106	00:00:00.57	00:00:02.32
Pass 1	368	00:00:12.70	00:00:27.96
Symbol table sort	0	00:00:02.01	00:00:04.32
Pass 2	97	00:00:02.43	00:00:04.16
Symbol table output	10	00:00:00.11	00:00:00.13
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	614	00:00:17.93	00:00:39.47

The working set limit was 1500 pages.

73891 bytes (145 pages) of virtual memory were used to buffer the intermediate code.

There were 70 pages of symbol table space allocated to hold 1330 non-local and 5 local symbols.

421 source lines were read in Pass 1, producing 17 object records in Pass 2.

28 pages of virtual memory were used to define 26 macros.

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

Macro library name	Macros defined
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	2
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	21
TOTALS (all libraries)	23

1474 GETS were required to define 23 macros.

There were no errors, warnings or information messages.

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

0386 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

