



```
SSSSSSSS  YY  YY  SSSSSSSS  IIIIII  MM  MM  GGGGGGGG  FFFFFFFF  IIIIII  XX  XX
SSSSSSSS  YY  YY  SSSSSSSS  IIIIII  MM  MM  GGGGGGGG  FFFFFFFF  IIIIII  XX  XX
SS  YY  YY  SS  SS  II  II  GG  FF  II  II  XX  XX
SS  YY  YY  SS  SS  II  II  GG  FF  II  II  XX  XX
SS  YY  YY  SS  SS  II  II  GG  FF  II  II  XX  XX
SSSSSSS  YY  YY  SSSSSS  II  II  MM  MM  GG  FF  II  II  XX  XX
SSSSSSS  YY  YY  SSSSSS  II  II  MM  MM  GG  FF  II  II  XX  XX
SS  YY  YY  SS  SS  II  II  MM  MM  GG  FF  II  II  XX  XX
SS  YY  YY  SS  SS  II  II  MM  MM  GG  FF  II  II  XX  XX
SSSSSSSS  YY  YY  SSSSSSSS  IIIIII  MM  MM  GGGGGG  FF  IIIIII  XX  XX
SSSSSSSS  YY  YY  SSSSSSSS  IIIIII  MM  MM  GGGGGG  FF  IIIIII  XX  XX
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)	42	History
(2)	101	Declarations
(3)	115	EXESIMGFIX Address Relocation Fixup System Service
(4)	172	GET_BASE_ADDRESSES - Locate Each Shareable Image
(5)	253	IMG\$IS_IT_MAPPED - Search ICB List for Shareable Image
(6)	331	PROCESS_FIXUP_LIST - Perform Post-Activation Fixups
(7)	410	FIXUP_G_HAT Fixup G-hat exit vector
(8)	461	SHIMG_BASVA Convert a shareable image index to an address
(9)	499	FIXUP_ADDRESS Fixup .ADDRESS entries throughout the image
(10)	541	FIXUP_PROT Alter page protection to read only
(11)	601	IMG\$PRVSHRIMG Fixup Routine for Privileged Shareable Images
(12)	660	INISHRIMG - Look for and Call Shareable Image Initialization Code

```
0000 1 .TITLE SYSSIMGFIX - Address Fixup System Service
0000 2 .IDENT 'V04-000'
0000 3 :
0000 4 :*****
0000 5 :
0000 6 :* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 :* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 :* ALL RIGHTS RESERVED.
0000 9 :*
0000 10 :* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 :* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 :* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 :* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 :* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 :* TRANSFERRED.
0000 16 :*
0000 17 :* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 :* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 :* CORPORATION.
0000 20 :*
0000 21 :* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 :* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23 :*
0000 24 :*
0000 25 :*****
0000 26 :
0000 27 :++
0000 28 : Facility:
0000 29 :
0000 30 : Executive - Image Activator Completion Routines
0000 31 :
0000 32 : Abstract:
0000 33 :
0000 34 : This module contains subroutines used by the image activator
0000 35 : to perform address relocation after images have been activated.
0000 36 :
0000 37 : Environment:
0000 38 :
0000 39 : Most of the code in this module runs in user mode but some routines
0000 40 : may also be called from exec mode.
0000 41 :
0000 42 : .SUBTITLE History
0000 43 :
0000 44 : Author:
0000 45 :
0000 46 : Lawrence J. Kenah
0000 47 :
0000 48 : Creation Date:
0000 49 :
0000 50 : 19 March 198
0000 51 :
0000 52 : Modified by:
0000 53 :
0000 54 : V03-010 LJK0279 Lawrence J. Kenah 8-May-1984
0000 55 : Miscellaneous cleanup. Remove temporary definition of
0000 56 : SHL$B_SHL_SIZE. Put all code into YF$$$SYSIMGACT program
0000 57 : section.
```

```

0000 58 :
0000 59 :
0000 60 :
0000 61 :
0000 62 :
0000 63 :
0000 64 :
0000 65 :
0000 66 :
0000 67 :
0000 68 :
0000 69 :
0000 70 :
0000 71 :
0000 72 :
0000 73 :
0000 74 :
0000 75 :
0000 76 :
0000 77 :
0000 78 :
0000 79 :
0000 80 :
0000 81 :
0000 82 :
0000 83 :
0000 84 :
0000 85 :
0000 86 :
0000 87 :
0000 88 :
0000 89 :
0000 90 :
0000 91 :
0000 92 :
0000 93 :
0000 94 :
0000 95 :
0000 96 :
0000 97 :
0000 98 :
0000 99 :--

```

V03-009 LJK0270 Lawrence J. Kenah 31-Mar-1984  
Add code to call shareable image initialization routines.

V03-008 LJK0275 Lawrence J. Kenah 25-Mar-1984  
The size of SHL elements is variable. It depends on when  
the image was linked.

V03-007 LJK0238 Lawrence J. Kenah 26-Jul-1983  
Use new concept of image base address instead of first address  
into which image is mapped.

V03-006 LJK0218 Lawrence J. Kenah 28-Jun-1983  
Minor cleanup.

V03-005 LJK0200 Lawrence J. Kenah 14-Jun-1983  
Make changes that support new image activator

Base addresses of shareable images are now located by searching  
the ICB list, a much simpler list than the master fixup vector  
list. Routine COPY\_SHL is no longer needed. All code that  
existed to support a previous design for mapping shareable  
images permanently into P1 space is also eliminated. Use  
IMG\$ prefix for global entry point names. Eliminate prefix  
from routines that are local.

V03-004 LJK0195 Lawrence J. Kenah 9-Mar-1983  
Make so-called recursive activation capable of activating  
more than one image without dropping some fixups on the floor.

V03-003 LJK0192 Lawrence J. Kenah 7-Jan-1983  
Do poor man's recursive activation to support shareable  
images that reference other shareable images not known  
to the image header of the executable image.

V03-002 MLJ0099 Martin L. Jack, 20-Oct-1982 19:40  
Fix broken BSBWs.

V03-001 KDM0002 Kathleen D. Morse 28-Jun-1982  
Added \$\$SDEF.

```
0000 101      .SUBTITLE
0000 102
0000 103 ; Include Files:
0000 104
0000 105      $IACDEF
0000 106      $IAFDEF
0000 107
0000 108      $ICBDEF
0000 109      $IMAGCTXDEF
0000 110      $IMGDEF
0000 111      $PSLDEF
0000 112      $SETPRTDEF
0000 113      $SHLDEF
```

Declarations

```
: Image activator control flags
: Offsets into image activator fixup
:   area within image file
: Image control block offsets
: Context of currently executing image
: Image activator status codes
: PSL field definitions and constants
: Argument list offsets for $SETPRT system service
: Offsets into shareable image list element
```

```

0000 115      .SBTTL  EXESIMGFIX      Address Relocation Fixup System Service
0000 116      :+
0000 117      : Functional Description:
0000 118      :
0000 119      : This procedure is called after an image is activated but before it
0000 120      : is called in order to allow address fixups to be performed in user
0000 121      : access mode if the caller of the image so wishes. This prevents
0000 122      : process destruction or security breaches if the image that has just
0000 123      : been activated contains garbage or selected cleverness.
0000 124      :
0000 125      : Calling Sequence:
0000 126      :
0000 127      : CALLS  #0,EXESIMGFIX
0000 128      :
0000 129      : Implicit Input:
0000 130      :
0000 131      : Address space of image just activated that contains fixup vectors
0000 132      : that describe the address fixups that must be done.
0000 133      :
0000 134      : Implicit Output:
0000 135      :
0000 136      : All appropriate locations are relocated to reflect the locations
0000 137      : of each shareable image during this activation of the image.
0000 138      :
0000 139      : Completion Codes:
0000 140      :
0000 141      : R0 low bit set => successful completion (SS$_NORMAL)
0000 142      :
0000 143      : R0 low bit clear => error occurred
0000 144      :
0000 145      : Actual error status codes are returned by GET_BASE_ADDRESSES and
0000 146      : PROCESS_FIXUP_LIST.
0000 147      :
0000 148      : Side Effects:
0000 149      :
0000 150      : See the routine headers for the two functional routines for a
0000 151      : description of the effects of this procedure.
0000 152      :-
0000 153      :
0000 154      : Put all of this module into a separate pageable program section
0000 155      :
0000 156      .PSECT  YF$$$SYSIMGACT
0000 157      :
0000 158      EXESIMGFIX::
0000 159      .WORD  ^M<R2,R3,R4,R5>      ; Save some registers
0000 160      BSBB  GET_BASE_ADDRESSES  ; Get base address of each image
0000 161      BLBC  R0,T0$              ; Skip hard part if an error occurred
0000 162      BSBW  PROCESS_FIXUP_LIST ; Do the actual relocation
0000 163      BBC   #IMAGCTX$V SETVECTOR,- ;
0000 164      G^IAC$GL IMAGCTX,10$      ; Any vectors to set?
0000 165      $IMGACT_S 0,0,0,-
0000 166      -IMGCTL=#IAC$M SETVECTOR ; Let image activator set them
0000 167      10$: BBC   #IMAGCTX$V INITIALIZE,- ;
0000 168      G^IAC$GL IMAGCTX,20$      ; Any routines to be called?
0000 169      BSBW  INISHRIMG          ; Find them and call them
0000 170      20$: RET                ; Return with final status

```

```

0039 172 .SBTTL GET_BASE_ADDRESSES - Locate Each Shareable Image
0039 173 :+
0039 174 : Functional Description:
0039 175 :
0039 176 : This routine is called before the actual fixup operations are performed
0039 177 : to determine the base address of each shareable image that has been
0039 178 : mapped. If a shareable image in the fixup list has no corresponding
0039 179 : entry of the same name in the master ICB list, an error is reported.
0039 180 :
0039 181 : Note that the image activator has filled in the base address for SHL
0039 182 : entry 0, the SHL associated with the image itself.
0039 183 :
0039 184 : Calling Sequence:
0039 185 :
0039 186 : JSB GET_BASE_ADDRESSES
0039 187 :
0039 188 : Input Parameters:
0039 189 :
0039 190 : none
0039 191 :
0039 192 : Implicit Input:
0039 193 :
0039 194 : Listheads for fixup vector list and ICB list
0039 195 :
0039 196 : Output Parameters:
0039 197 :
0039 198 : none
0039 199 :
0039 200 : Implicit Output:
0039 201 :
0039 202 : All SHL entries in the linked list of fixup vectors have base addresses
0039 203 : of their associated shareable images stored in SHL$BASEVA.
0039 204 :
0039 205 : Completion Codes:
0039 206 :
0039 207 : R0 = SSS_NORMAL
0039 208 :
0039 209 : All base addresses were successfully stored.
0039 210 :
0039 211 : R0 = IMG$IMAGE_NOT_FOUND
0039 212 :
0039 213 : A shareable image name in a SHL entry had no corresponding
0039 214 : ICB. This means that the shareable image was not mapped,
0039 215 : which indicates an inconsistency between SHL entries and
0039 216 : image section descriptors in the image header of one of the
0039 217 : images that was mapped.
0039 218 :
0039 219 : Side Effects.
0039 220 :
0039 221 : R0 and R1 are destroyed
0039 222 : -
0039 223 :
0039 224 GET_BASE_ADDRESSES:
55 FFFFFFFC'GF DE 0039 225 MOVAL G^<CTL$GL_FIXUPLNK-IAF$L_FIXUPLNK>,R5 ; Pick up listhead address
0040 226
55 04 A5 DO 0040 227 10$: MOVL IAF$L_FIXUPLNK(R5),R5 ; Get address of next fixup vector
27 13 0044 228 BEQL 30$ ; Return success if done

```





```

0073 253 .SUBTITLE IMG$IS_IT_MAPPED - Search ICB List for Shareable Image
0073 254 :+
0073 255 : Functional Description:
0073 256 :
0073 257 : This routine searches the shareable image list associated with the
0073 258 : executable image to determine whether a shareable image with a given
0073 259 : name exists in the list. This routine is used to determine whether a
0073 260 : shareable image has already been mapped. It is also used to relate the
0073 261 : relative shareable image list within a shareable image to the shareable
0073 262 : image list associated with the executable image.
0073 263 :
0073 264 : Calling Sequence:
0073 265 :
0073 266 : JSB IMG$IS_IT_MAPPED
0073 267 :
0073 268 : Input Parameters:
0073 269 :
0073 270 : R0 = address of counted (ASCII) string of shareable image name
0073 271 :
0073 272 : Implicit Input:
0073 273 :
0073 274 : IAC$GL_IMAGE_LIST - Doubly linked list of ICBs describing images
0073 275 : that have already been mapped
0073 276 :
0073 277 : Output Parameters:
0073 278 :
0073 279 : If successful, R1 contains the address of the image control block
0073 280 : that describes the named image.
0073 281 :
0073 282 : Completion Codes:
0073 283 :
0073 284 : R0 low bit set indicates success (SS$_NORMAL)
0073 285 :
0073 286 : R0 low bit clear indicates failure (IMG$_IMAGE_NOT_FOUND)
0073 287 :
0073 288 : This status indicates that no match occurred, implying that
0073 289 : the shareable image in question has not yet been mapped.
0073 290 :
0073 291 : The severity of this status depends on the caller. This routine
0073 292 : is called by the image activator to determine whether an image
0073 293 : has been mapped. If the image name is not found, then the image
0073 294 : activator maps the image. If this routine returns this status
0073 295 : to the fixup code located earlier in this module, that is a
0073 296 : fatal error indicating an inconsistency between shareable image
0073 297 : lists inside fixup vectors and ISD lists in image headers.
0073 298 :-
0073 299 :-
0073 300 IMG$IS_IT_MAPPED::
0073 301 POSHR #*M<R2,R3,R4,R5,R6,R7> ; Save some registers
0077 302 MOVZBL (R0)+,R4 ; Save character count in R4
007A 303 MOVL R0,R5 ; Save string address in R5
007D 304
007D 305 ASSUME ICB$_FLINK EQ 0
007D 306
007D 307 MOVAL G^IAC$GL_IMAGE_LIST,R7 ; Get address of ICB listhead
0084 308 MOVL R7,R6 ; Copy it to a working register
0087 309
  
```

```

00FC 8F BB
54 80 9A
55 50 D0
57 00000000'GF DE
56 57 D0
  
```

SYS  
 VAX  
  
 Pha  
 ---  
 Ini  
 Com  
 Pas  
 Syn  
 Pas  
 Syn  
 Pse  
 Crc  
 Ass  
  
 The  
 219  
 The  
 713  
 20  
  
 Mac  
 ---  
 \$2  
 - \$2  
 - \$2  
 TOT  
 329  
 The  
 MAC

```

    56 66 D0 0087 310 10$: MOVL ICB$L_FLINK(R6),R6 ; Get address of next ICB
    57 56 D1 008A 311      CMPL R6,R7      ; Check for end of list
      1C 13 008D 312      BEQL 30$      ; Equality indicates no more ICBs
      008F 313
    14 A6 54 91 008F 314      CMPB R4,ICBST_IMAGE_NAME(R6) ; Do string sizes agree?
      F2 12 0093 315      BNEQ 10$      ; No, go get next ICB
    15 A6 65 54 29 0C95 316      CMPC3 R4,(R5),ICBST_IMAGE_NAME+1(R6) ; Check strings for equality
      EB 12 009A 317      BNEQ 10$      ; Go get next ICB if no match
      51 56 D0 009C 318      MOVL R6,R1      ; Store ICB address
      009F 319
    50 00000000'8F D0 009F 320      MOVL #SS$ NORMAL,R0 ; Indicate success to caller
      00FC 8F BA 00A6 321 20$: POPR #^M<R2,R3,R4,R5,R6,R7> ; Restore registers
      05 00AA 322      RSB ; and return
      00AB 323
      00AB 324 ; If we loop through the entire ICB list without matching the image name, then
      00AB 325 ; the shareable image has not yet been mapped. Indicate that to caller.
      00AB 326
    50 084D8962 8F D0 00AB 327 30$: MOVL #IMG$_IMAGE_NOT_FOUND,R0
      51 D4 00B2 328      CLRL R1
      F0 11 00B4 329      BRB 20$
  
```

```

00B6 331      .SBTTL PROCESS_FIXUP_LIST - Perform Post-Activation Fixups
00B6 332      :+
00B6 333      : Functional Description:
00B6 334      :
00B6 335      : This routine processes a linked list of fixup vectors and performs
00B6 336      : the specific fixup operations listed in each vector. There are three
00B6 337      : forms of fixup.
00B6 338      :
00B6 339      :     o Each exit vector has the base address of the shareable
00B6 340      :       image added to each entry.
00B6 341      :
00B6 342      :     o Each .ADDRESS directive has the base address of the
00B6 343      :       appropriate shareable image added to it.
00B6 344      :
00B6 345      :     o The protection of each fixup vector is changed to prevent
00B6 346      :       its being written when the image executes.
00B6 347      :
00B6 348      : Calling Sequence:
00B6 349      :
00B6 350      :     JSB     PROCESS_FIXUP_LIST
00B6 351      :
00B6 352      : Input Parameters:
00B6 353      :
00B6 354      :     none
00B6 355      :
00B6 356      : Implicit Input:
00B6 357      :
00B6 358      :     CTL$GL_FIXUPLNK      Listhead of linked list of fixup vectors for
00B6 359      :                          a set of shareable images
00B6 360      :
00B6 361      : Output Parameters:
00B6 362      :
00B6 363      :     none
00B6 364      :
00B6 365      : Implicit Output:
00B6 366      :
00B6 367      :     Elements in fixup vector G-hat offset area have base address
00B6 368      :     of appropriate shareable image added to them.
00B6 369      :
00B6 370      :     .ADDRESS directives throughout the address space have base
00B6 371      :     addresses added in.
00B6 372      :
00B6 373      :     Pages that should eventually be read-only but were set to
00B6 374      :     writable while the image activator works are set back to read-only.
00B6 375      :
00B6 376      : Completion Codes:
00B6 377      :
00B6 378      :     none
00B6 379      :
00B6 380      : Side Effects:
00B6 381      :
00B6 382      :     CTL$GL_FIXUPLNK cleared after fixups are completed.
00B6 383      :
00B6 384      : --
00B6 385      :
00B6 386      : PROCESS_FIXUP_LIST:
00B6 387      : MOVAL     G^<CTL$GL_FIXUPLNK-IAF$&L_FIXUPLNK>,R5 ; Pick up listhead address

```

55 FFFFFFFC'GF DE

55	04	A5	D0	00BD	388						
		2E	13	00C1	389	10\$:	MOVL	IAF\$\$_FIXUPLNK(R5),R5	:	Get address of next fixup vector	
		51	D4	00C3	390		BEQL	40\$	:	Quit if no more to process	
		0042	30	00C5	391		CLRL	R1	:	Need base address of this shareable	
	53	51	D0	00C8	392		BSBW	SHIMG_BASVA	:	image (with index 0)	
		54	0C	A5	393		MOVL	R1,R3	:	Load correct input register	
		06	13	00CB	394		MOVL	IAF\$\$_G_FIXOFF(R5),R4	:	Get offset to G-hat fixup data	
		54	55	C0	395		BEQL	20\$	:	Skip this step if none	
		001B	30	00D1	396		ADDL2	R5,R4	:	Make an address	
		54	10	A5	397		BSBW	FIXUP_G_HAT	:	Go do the actual work	
		06	13	00D7	398	20\$:	MOVL	IAF\$\$_DOTADROFF(R5),R4	:	Get offset to .ADDRESS fixup data	
		54	55	C0	399		BEQL	30\$	:	Skip this step if none	
		0039	30	00DD	400		ADDL2	R5,R4	:	Make an address	
		54	14	A5	401		BSBW	FIXUP_ADDRESS	:	Fixup all .ADDRESS data	
		D4	13	00E0	402	30\$:	MOVL	IAF\$\$_CHGPRTOFF(R5),R4	:	Get offset to page protection data	
		54	55	C0	403		BEQL	10\$	:	Skip this step if none	
		0049	30	00E7	404		ADDL2	R5,R4	:	Make an address	
		CC	11	00E9	405		BSBW	FIXJP_PROT	:	Change page protection	
				00EF	406		BRB	10\$	:	All done with this fixup vector	
				00F1	407				:	See if there are any more	
			05	00F1	408	40\$:	RSB		:	Return to caller	

```

00F2 410      .SBTTL  FIXUP_G_HAT      Fixup G-hat exit vector
00F2 411      :+
00F2 412      : Functional Description:
00F2 413      :
00F2 414      : This routine performs the G-hat fixup for a specific exit vector.
00F2 415      : specifically, the base address of the appropriate shareable image
00F2 416      : is added to each entry in the exit vector.
00F2 417      :
00F2 418      : Calling Sequence:
00F2 419      :
00F2 420      :     BSBW  FIXUP_G_HAT
00F2 421      :
00F2 422      : Input Parameters:
00F2 423      :
00F2 424      :     R4 = Address of G-hat fixup area within fixup vector
00F2 425      :
00F2 426      : Implicit Input:
00F2 427      :
00F2 428      :     Contents of G-hat fixup area
00F2 429      :
00F2 430      : Output Parameters:
00F2 431      :
00F2 432      :     none
00F2 433      :
00F2 434      : Implicit Output:
00F2 435      :
00F2 436      :     Elements in fixup vector G-hat offset area have base address
00F2 437      :     of appropriate shareable image added to them.
00F2 438      :
00F2 439      : Completion Codes:
00F2 440      :
00F2 441      :     none
00F2 442      :
00F2 443      : Side Effects:
00F2 444      :
00F2 445      :     R0, R1, and R2 are destroyed
00F2 446      : -
00F2 447      :
00F2 448      : FIXUP_G_HAT:
52  84  D0 00F2 449      MOVL  (R4)+,R2      ; R2 contains a count of fixups
   OD  13 00F5 450      BEQL  20$      ; A zero indicates the end of the G-hat data
51  84  D0 00F7 451      MOVL  (R4)+,R1      ; Store shareable image number in R1
   OE  10 00FA 452      BSBB  SHIMG_BASVA    ; and then load R1 with base address
   00FC 453      ; of next shareable image.
84  51  C0 00FC 454 10$: ADDL2  R1,(R4)+    ; Bias next exit vector entry
   FA  52  F5 00FF 455      SOBGR  R2,10$    ; Do next entry
   EE  11 0102 456      BRB   FIXUP_G_HAT    ; Now do next shareable image
   0104 457
50  0000'8F 3C 0104 458 20$: MOVZWL #SS$_NORMAL,R0 ; Indicate success
   05  0109 459      RSB   ; Return

```

```

010A 461      .SBTTL SHIMG_BASVA      Convert a shareable image index to an address
010A 462      :+
010A 463      : Functional Description.
010A 464      :
010A 465      : This routine converts a relative shareable image number into the
010A 466      : absolute base address at which that shareable image is mapped. It
010A 467      : assumes that the base address of each shareable image has already
010A 468      : been stored in its associated SHL entry.
010A 469      :
010A 470      : Calling Sequence:
010A 471      :
010A 472      :     BSBW      SHIMG_BASVA
010A 473      :
010A 474      : Input Parameters:
010A 475      :
010A 476      :     R1 = Relative number of shareable image
010A 477      :     R5 = Base address of fixup vector
010A 478      :
010A 479      : Implicit Input:
010A 480      :
010A 481      :     Contents of SHL$$_BASEVA for shareable image indexed by R1.
010A 482      :
010A 483      : Output Parameters:
010A 484      :
010A 485      :     R1 = Base address of shareable image indicated by input parameter
010A 486      :
010A 487      : Side Effects:
010A 488      :
010A 489      :     R0 is destroyed
010A 490      :-
010A 491
010A 492 SHIMG_BASVA:
50    18 A5 55 C1 010A 493      ADDL3  R5,IAF$$_SHLSTOFF(R5),R0 ; Base address of shareable image list
      7E 10 A0 9A 010F 494      MOVZBL SHL$$_SHL_SIZE(R0),-(SP) ; Get size of each SHL element
50    50 51 8E 7A 0113 495      EMUL   (SP)+,R1,R0,R0 ; R0 points to correct SHL entry
      51 60 D0 0118 496      MOVL   SHL$$_BASEVA(R0),R1 ; Store associated base address
05    011B 497      RSB ; and return

```

```

011C 499 .SBTTL FIXUP_ADDRESS Fixup .ADDRESS entries throughout the image
011C 500 :+
011C 501 : Functional Description:
011C 502 :
011C 503 : This routine performs the .ADDRESS fixup for a specific exit vector.
011C 504 : Specifically, the base address of the appropriate shareable image
011C 505 : is added to each .ADDRESS entry in this shareable image.
011C 506 :
011C 507 : Calling Sequence:
011C 508 :
011C 509 : BSBW FIXUP_ADDRESS
011C 510 :
011C 511 : Input Parameters:
011C 512 :
011C 513 : R3 = Base address of shareable image whose .ADDRESS directives
011C 514 : are being fixed
011C 515 : R4 = Address of .ADDRESS fixup area within fixup vector
011C 516 :
011C 517 : Implicit Input:
011C 518 :
011C 519 : Contents of .ADDRESS fixup area
011C 520 :
011C 521 : Implicit Output:
011C 522 :
011C 523 : .ADDRESS directives within this shareable image have the base addresses
011C 524 : of the appropriate shareable images added to them.
011C 525 :-
011C 526 :
011C 527 FIXUP_ADDRESS:
52 84 D0 011C 528 MOVL (R4)+,R2 ; R2 contains a count of fixups
11 13 011F 529 BEQL 20$ ; A zero indicates the end of the G-hat data
51 84 D0 0121 530 MOVL (R4)+,R1 ; Store shareable image number in R1
E4 10 0124 531 BSBB SHIMG_BASVA ; and then load R1 with base address
0126 532 ; of next shareable image.
50 84 53 C1 0126 533 10$: ADDL3 R3,(R4)+,R0 ; Get address of .ADDRESS directive
60 51 C0 012A 534 ADDL2 R1,(R0) ; Bias by base address of shareable image
F6 52 F5 012D 535 SOBGTR R2,10$ ; Do next entry
EA 11 0130 536 BRB FIXUP_ADDRESS ; Now do next shareable image
50 0000'8F 3C 0132 537 20$: MOVZWL #SS$_NORMAL,R0 ; Indicate success
05 0137 538 RSB ; Return
0137 539

```



```

0138 541      .SBTTL  FIXUP_PROT      Alter page protection to read only
0138 542      :+
0138 543      : Functional Description:
0138 544      :
0138 545      : This routine alters the page protection of various sections within
0138 546      : the image to read only. These pages were initially writable so the
0138 547      : image activator could fixup all of the relative references. The pages
0138 548      : cannot be writable while the image is executing.
0138 549      :
0138 550      : Calling Sequence:
0138 551      :
0138 552      :     BSBW  FIXUP_PROT
0138 553      :
0138 554      : Input Parameters:
0138 555      :
0138 556      :     R3 = Base address of image whose pages' protection is being altered
0138 557      :     R4 = Address of protection data within fixup vector
0138 558      :
0138 559      : Implicit Input:
0138 560      :
0138 561      :     Contents of protection data in fixup vector
0138 562      :
0138 563      : Implicit Output:
0138 564      :
0138 565      :     Pages in address ranges specified in fixup vector have their protections
0138 566      :     changed to the protections also specified in that data area. The
0138 567      :     protection is usually no write access for any access mode.
0138 568      :
0138 569      : Side Effects:
0138 570      :
0138 571      :     R0, R1, and R2 are destroyed
0138 572      :-
0138 573
0138 574  FIXUP_PROT:
0138 575      PUSHL  R6                ; Need one more register here
0138 576      SUBL2  #<4*SETPRT$ NARGS>,SP ; Set up space for argument list
0138 577      PUSHL  #SETPRT$_NARGS      ; Push argument count
0138 578      MOVL   SP,R6                ; Use R6 as argument pointer
0138 579      CLRQ   -(SP)                 ; Initialize input address array
0138 580      MOVL   SP,SETPRT$ INADR(R6) ; Put its address into argument list
0138 581      CLRL   SETPRT$ RETADR(R6)   ; Not interested in this argument
0138 582      MOVL   #PSL$C EXEC,SETPRT$_ACMODE(R6) ; The image activator owns these page
0138 583      CLRL   SETPRT$ PRVPRT(R6)  ; Not interested in this either
0138 584      MOVZWL #SS$ NORMAL,R0     ; Establish initial status
0138 585      MOVL   (R4)+,R2            ; Get count of number of protection changes
0138 586      BEQL   20$                  ; Do not even start if nothing here
0138 587 10$: ADDL3  R3,(R4)+,(SP)     ; Get starting address
0138 588      MOVZWL (R4)+,R1            ; Ending address must be calculated
0138 589      ASHL  #9,R1,R1             ; ... from page count in image section
0138 590      DECL  R1                    ; Make byte count an inclusive count
0138 591      ADDL3  R1,(SP),4(SP)        ; Put ending address in second longword
0138 592      MOVZWL (R4)+,SETPRT$ PRCT(R6) ; Get new protection from fixup vector
0138 593      CALLG (R6),G^SYS$SETPRT    ; Call the system service
0138 594      ; Ignore errors
0138 595      SOBGR  R2,10$              ; Go get next image section
0138 596
0138 597 20$: ADDL2  #<8+4+<4*SETPRT$_NARGS>>,SP ;Reset stack pointer,

```

SYSSIMGFIX  
V04-000

- Address Fixup System Service L 1  
FIXUP\_PROT Alter page protection to read

16-SEP-1984 02:20:23 VAX/VMS Macro V04-00  
5-SEP-1984 03:54:43 [SYS.SRC]SYSSIMGFIX.MAR;1

Page 15  
(10)

SYSS  
V04

56 BED0 017F 598 POPL R6 ; restore that extra register,  
05 0182 599 RSB ; and return

```

0183 601      .SBTTL  IMG$PRVSHRIMG  Fixup Routine for Privileged Shareable Images
0183 602      :+
0183 603      : Functional Description:
0183 604      :
0183 605      : This routine checks that a privileged shareable image has no
0183 606      : outbound calls. For images passing this test, remaining
0183 607      : .ADDRESS fixups are performed.
0183 608      :
0183 609      : Calling Sequence:
0183 610      :
0183 611      :     BSBW  IMG$PRVSHRIMG
0183 612      :
0183 613      : Input Parameters:
0183 614      :
0183 615      :     R0    Address of fixup vector
0183 616      :     R1    Base address of privileged shareable image currently
0183 617      :           being mapped
0183 618      :
0183 619      : Implicit Output:
0183 620      :
0183 621      : If the fixup vector indicates no outbound calls, the base address
0183 622      : of the privileged shareable image is stored in the fixup vector
0183 623      : and the .ADDRESS fixups are performed.
0183 624      :
0183 625      : Side Effects:
0183 626      :
0183 627      :     R0 and R1 are destroyed
0183 628      :
0183 629      : Completion Codes:
0183 630      :
0183 631      :     $$$_NORMAL    Fixups were completed for privileged shareable image
0183 632      :
0183 633      :     $$$_NOSHRIMG  Shareable image has outbound calls
0183 634      :-
0183 635
0183 636 IMG$PRVSHRIMG::
50 1C 55 3C BB 0183 637      PUSHR    #*M<R2,R3,R4,R5>      ; Save some registers
50 1C A5 50 DO 0185 638      MOVL     R0,R5          ; Store fixup vector address in R5
50 1C A5 01 C3 0188 639      SUBL3   #1,IAF$S_SHRIMGCNT(R5),R0      ; Is shareable image count 1?
50 1C A5 2B 12 018D 640      BNEQ    30$          ; If not, report error
50 1C A5 0C A5 D5 018F 641      TSTL   IAF$S_G_FIXOFF(R5)      ; Also report error if G^ fixup data
50 1C A5 26 12 0192 642      BNEQ    30$          ;
50 18 A5 53 51 DO 0194 643      MOVL     R1,R3          ; Store base address of image in R3
50 18 A5 55 C1 0197 644      ADDL3   R5,IAF$S_SHLSTOFF(R5),R0      ; Also store base address in
50 18 A5 60 51 DO 019C 645      MOVL     R1,SHL$S_BASEVA(R0)      ; SHL entry for SHIMG_BASVA
50 18 A5 10 A5 DO 019F 646      MOVL     IAF$S_DOTADROFF(R5),R4      ; Any .ADDRESS fixups?
50 18 A5 06 13 01A3 647      BEQL    10$          ; Branch if none
50 18 A5 54 55 C0 01A5 648      ADDL2   R5,R4          ; Convert R4 offset to address
50 18 A5 FF 71 30 01A8 649      BSBW    FIXUP_ADDRESS      ; Fixup all .ADDRESS data
50 18 A5 14 A5 DO 01AB 650 10$: MOVL     IAF$S_CHGPRTOFF(R5),R4      ; Get offset to protection data
50 18 A5 06 13 01AF 651      BEQL    20$          ; All done if none
50 18 A5 54 55 C0 01B1 652      ADDL2   R5,R4          ; Make R4 an address
50 18 A5 FF 81 30 01B4 653      BSBW    FIXUP_PROT      ; Change page protection
50 18 A5 3C BA 01B7 654 20$: POPR    #*M<R2,R3,R4,R5>      ; Restore registers
50 18 A5 05 01B9 655      RSB          ; and return
50 0000'8F 3C 01BA 656
50 0000'8F 3C 01BA 657 30$: MOVZWL #$$$_NOSHRIMG,R0      ; No outbound calls allowed

```

SYSSIMGFIX  
V04-000

- Address Fixup System Service N 1  
IMG\$PRVSHRIMG Fixup Routine for Privileg

16-SEP-1984 02:20:23 VAX/VMS Macro V04-00  
5-SEP-1984 03:54:43 [SYS.SRC]SYSSIMGFIX.MAR;1

Page 17  
(11)

F6 11 01BF 658 BRB 20\$

; Return error status

SY  
V04

```

01C1 660 .SBTTL INISHRIMG - Look for and Call Shareable Image Initialization Code
01C1 661 :+
01C1 662 : Functional Description:
01C1 663 :
01C1 664 : This routine searches the shareable image list for images that have
01C1 665 : included initialization code.
01C1 666 :
01C1 667 : Calling Sequence:
01C1 668 :
01C1 669 : BSBW INISHRIMG
01C1 670 :
01C1 671 : Input Parameters:
01C1 672 :
01C1 673 : none
01C1 674 :
01C1 675 : Implicit Input:
01C1 676 :
01C1 677 : IAC$GL_IMAGE_LIST - List of ICBs describing shareable images that
01C1 678 : are currently mapped.
01C1 679 : IAC$GL_FIRST_ICB - Address of ICB representing main image in the
01C1 680 : most recent image activation.
01C1 681 :
01C1 682 : Implicit Output:
01C1 683 :
01C1 684 : If there are any images with ICBs containing shareable image
01C1 685 : initialization code, these procedures are called at their entry
01C1 686 : points. Note that the ICB list is traversed backwards.
01C1 687 :
01C1 688 : Side Effects:
01C1 689 :
01C1 690 : R0 and R1 are destroyed
01C1 691 :
01C1 692 : Completion Codes:
01C1 693 :
01C1 694 : none
01C1 695 :-
01C1 696
01C1 697 INISHRIMG:
52 00000000'GF 7E 52 7D 01C1 698 MOVQ R2,-(SP) ; Save some registers
53 00000000'GF 53 04 A2 D0 01C4 699 MOVAL G^IAC$GL_IMAGE_LIST,R2 ; Get the listhead address
01D2 700 MOVL G^IAC$GL_FIRST_ICB,R3 ; This is the stopper
01D2 701
10$ 01D2 702 MOVL ICB$BLINK(R2),R2 ; Get the next ICB
01D6 703 BBC #ICB$V_INITIALIZE,- ; Does this image need to be called?
01D8 704 ICB$FLAGS(R2),20$ ; Branch if no initialization routine
01DB 705 ADDL3 ICB$INITIALIZE(R2),- ; Form the address of the entry point
01DE 706 ICB$BASE_ADDRESS(R2),R1
01E1 707 CALLS #0,(RT) ; Call the routine
01E4 708 20$ 20$ Cmpl R2,R3 ; Is this the end of the line?
01E7 709 BNEQ 10$ ; Back to the top is there's more
01E9 710 MOVQ (SP)+,R2 ; Restore R2 and R3
01EC 711 RSB ; All done. Return to caller.
01ED 712
01ED 713 .END

```

```

52 00000000'GF 7E 52 7D
53 00000000'GF 53 04 A2 D0
    09 10 A2 E1
    50 A2 C1
51 5C A2 01DE
    61 00 FB 01E1
    53 52 D1 01E4
    52 8E 12 01E7
    7D 01E9
    C5 01EC
    01ED
    01ED

```

SYSSIMGFIX  
Symbol table

- Address Fixup System Service

C 2

16-SEP-1984 02:20:23 VAX/VMS Macro V04-00  
5-SEP-1984 03:54:43 [SYS.SRC]SYSSIMGFIX.MAR;1

Page 19  
(12)

SY  
V04

```

SSARGS          = 00000005
SST1           = 00000000
CTL$GL_FIXUPLNK ***** X 02
EXESIMGFIX     = 00000000 RG 02
FIXUP_ADDRESS  = 0000011C R 02
FIXUP_G_HAT    = 000000F2 R 02
FIXUP_PROT     = 00000138 R 02
GET_BASE_ADDRESSES
IAC$GL_FIRST_ICB ***** X 02
IAC$GL_IMAGCTX ***** X 02
IAC$GL_IMAGE_LIST ***** X 02
IAC$M_SETVECTOR = 00200000
IAF$SL_CHGPRTOFF = 00000014
IAF$SL_DOTADROFF = 00000010
IAF$SL_FIXUPLNK = 00000004
IAF$SL_G_FIXOFF = 0000000C
IAF$SL_SHLSTOFF = 00000018
IAF$SL_SHRIMGCNT = 0000001C
ICBSL_BASE_ADDRESS = 0000005C
ICBSL_BLINK    = 00000004
ICBSL_FLAGS    = 00000010
ICBSL_FLINK    = 00000000
ICBSL_INITIALIZE = 00000060
ICBST_IMAGE_NAME = 00000014
ICBSV_INITIALIZE = 00000005
IMAGCTX$V_INITIALIZE = 00000011
IMAGCTX$V_SETVECTOR = 00000010
IMGS_IS_IT_MAPPED = 00000073 RG 02
IMGS_PRVSHRING = 00000183 RG 02
IMGS_IMAGE_NOT_FOUND = 084D8962
INISHRING     = 000001C1 R 02
PROCESS_FIXUP_LIST = 000000B6 R 02
PSL$C_EXEC    = 00000001
SETPRT$_ACMODE = 0000000C
SETPRT$_INADR = 00000004
SETPRT$_NARGS = 00000005
SETPRT$_PROT  = 00000010
SETPRT$_PRVPRT = 00000014
SETPRT$_RETADR = 00000008
SHIMG_BASVA   = 0000010A R 02
SHL$B_SHL_SIZE = 00000010
SHL$SL_BASEVA = 00000000
SHL$T_IMGNAME = 00000018
SS$NORMAL     ***** X 02
SS$NOSHRING   ***** X 02
SYS$IMCACT    ***** GX 02
SYS$SETPRT    ***** 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
YF\$\$SYSIMGACT	000001ED ( 493.)	02 ( 2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

-----  
! Performance in s !  
-----

Phase	Page faults	CPU Time	Elapsed Time
Initialization	34	00:00:00.07	00:00:00.40
Command processing	136	00:00:00.73	00:00:03.65
Pass 1	192	00:00:04.03	00:00:10.67
Symbol table sort	0	00:00:00.26	00:00:00.34
Pass 2	133	00:00:01.52	00:00:03.66
Symbol table output	7	00:00:00.05	00:00:00.05
Psect synopsis output	2	00:00:00.03	00:00:00.04
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	506	00:00:06.69	00:00:18.82

The working set limit was 1500 pages.  
 21961 bytes (43 pages) of virtual memory were used to buffer the intermediate code.  
 There were 20 pages of symbol table space allocated to hold 218 non-local and 25 local symbols.  
 713 source lines were read in Pass 1, producing 14 object records in Pass 2.  
 20 pages of virtual memory were used to define 19 macros.

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

Macro library name	Macros defined
_\$255\$DUA28:[SYS.OBJ]IMGACT.MLB;1	3
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	2
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	16

329 GETS were required to define 16 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:SY\$IMGFIX/OBJ=OBJ\$:SY\$IMGFIX MSRC\$:SY\$IMGFIX/UPDATE=(ENH\$:SY\$IMGFIX)+EXECMLS/LIB+LIB\$:IMGACT/LIB

0385 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

This image displays a grid of 144 small terminal window screenshots, arranged in 12 rows and 12 columns. Each window shows a different view of system logs, diagnostic data, or command-line output. The text is dense and small, typical of a terminal display. Several windows are clearly labeled with the following text:

- SYSGETSYI LIS** (row 4, column 4)
- SYSGETPTI LIS** (row 5, column 3)
- SYSGETTIM LIS** (row 6, column 6)
- SYSGETLKI LIS** (row 7, column 1)
- SYSGETMSG LIS** (row 8, column 3)
- SYSTEMGACT LIS** (row 10, column 6)
- SYSTEMGFIX LIS** (row 5, column 12)

The screenshots show various data points, including system identifiers, error codes, and performance metrics, all rendered in a monospaced font.



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

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

