

[illegible]

[illegible]

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL IIIII
LLLLLLLLLLL IIIII

SSSSSSSSS
SSSSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
SSSSSSSSS
SSSSSSSSS

```



```
0001  MODULE INIVOL (
0002  MAIN = INIT_VOLUME,
0003  LANGUAGE (BLISS32),
0004  IDENT = 'V04-000'
0005  ) =
0006  BEGIN
0007
0008  |
0009  |*****|
0010  |*|
0011  |*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0012  |*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0013  |*  ALL RIGHTS RESERVED.
0014  |*|
0015  |*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0016  |*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0017  |*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0018  |*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0019  |*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0020  |*  TRANSFERRED.
0021  |*|
0022  |*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0023  |*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0024  |*  CORPORATION.
0025  |*|
0026  |*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0027  |*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0028  |*|
0029  |*|
0030  |*****|
0031  |*|
0032  |**|
0033  |*|
0034  |*  FACILITY:  INIT Utility Structure Level 2
0035  |*|
0036  |*  ABSTRACT:
0037  |*|
0038  |*    This is the main program for the INITIALIZE volume utility.
0039  |*|
0040  |*  ENVIRONMENT:
0041  |*|
0042  |*    STARLET operating system, including privileged system services
0043  |*    and internal exec routines.
0044  |*|
0045  |*  --
0046  |*|
0047  |*|
0048  |*  AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  9-Nov-1977  18:17
0049  |*|
0050  |*  MODIFIED BY:
0051  |*|
0052  |*    V03-003  HH0041      Hai Huang      24-Jul-1984
0053  |*             Remove REQUIRE 'LIBD$:[VMSLIB.OBJ]MOUNTMSG.REQ'.
0054  |*|
0055  |*    V03-002  MCN0140    Maria del C. Nasr  29-Nov-1983
0056  |*             Changes required by new CLI interface.
0057  |*|
```


INIVOL
V04-000

C 2
16-Sep-1984 01:41:24 VAX-11 Bliss-32 V4.0-742 Page 2
14-Sep-1984 12:35:19 DISK\$VMSMASTER:[INIT.SRC]INIVOL.B32;1 (1)

```

58 0058 1 ! V03-001 STJ3038 Steven T. Jeffreys, 14-Oct-1982
59 0059 1 ! If the INIT fails, free up the drive with an IOS_AVAILABLE i/o.
60 0060 1 !
61 0061 1 ! V0103 ACG0072 Andrew C. Goldstein, 15-Oct-1979 16:03
62 0062 1 ! Check both primary and secondary device char
63 0063 1 !
64 0064 1 ! V0102 ACG0069 Andrew C. Goldstein, 9-Oct-1979 16:31
65 0065 1 ! Remove device data table
66 0066 1 !
67 0067 1 ! V0101 ACG25501 Andrew C. Goldstein, 21-Aug-1979 11:34
68 0068 1 ! Clear volume valid only if volume was not mounted
69 0069 1 !
70 0070 1 ! V0100 ACG00001 Andrew C. Goldstein, 10-Oct-1978 21:28
71 0071 1 ! Previous revision history moved to [INIT.SRC]INIT.REV
72 0072 1 ! **
73 0073 1 !
74 0074 1 !
75 0075 1 LIBRARY 'SYSS$LIBRARY:LIB.L32';
76 0076 1 REQUIRE 'SRC$:INIDEF.B32';
77 0367 1 REQUIRE 'LIBD$: [VMSLIB.OBJ]INITMSG.B32';
78 0499 1
79 0500 1
80 0501 1 FORWARD ROUTINE
81 0502 1 INIT_VOLUME, ! main program
82 0503 1 MAIN_HANDLER; ! condition handler
```



```

: 84      0504 1  !+
: 85      0505 1  !
: 86      0506 1  ! General impure storage for the INIT utility.
: 87      0507 1  !
: 88      0508 1  !-
: 89      0509 1  !
: 90      0510 1  GLOBAL LITERAL
: 91      0511 1      NAMEBUF_LEN      = 32,      ! length of device name buffer
: 92      0512 1      USERNAME_LEN     = 12;      ! maximum length of user name string
: 93      0513 1
: 94      0514 1  GLOBAL
: 95      0515 1      CLEANUP_FLAGS    : BITVECTOR [32], ! cleanup action flags
: 96      0516 1      CHANNEL          : channel for all I/O
: 97      0517 1      PROCESS_UIC      : UIC of this process
: 98      0518 1      DEVICE_CHAR      : BBLOCK [DIB$K_LENGTH],
: 99      0519 1      ! buffer for device characteristics
: 100     0520 1      DEVICE_CHAR2      : BBLOCK [DIB$K_LENGTH],
: 101     0521 1      ! buffer for sec. device characteristics
: 102     0522 1      PHYS_NAME         : VECTOR [2], ! physical device name descriptor
: 103     0523 1      NAME_BUFFER       : VECTOR [NAMEBUF_LEN, BYTE],
: 104     0524 1      ! buffer for physical device name
: 105     0525 1      USER_STRING       : VECTOR [USERNAME_LEN, BYTE];
: 106     0526 1      ! Buffer for user name string
: 107     0527 1
: 108     0528 1  GLOBAL
: 109     0529 1      DEVCHAR_DESC      : VECTOR [2] INITIAL (DIB$K_LENGTH, DEVICE_CHAR),
: 110     0530 1      ! descriptor for device characteristics
: 111     0531 1      DEVCHAR_DESC2     : VECTOR [2] INITIAL (DIB$K_LENGTH, DEVICE_CHAR2);
: 112     0532 1      ! descriptor for sec. device characteristics

```



```
114 0533 1 GLOBAL ROUTINE INIT_VOLUME =
115 0534 1
116 0535 1 ++
117 0536 1
118 0537 1 FUNCTIONAL DESCRIPTION:
119 0538 1
120 0539 1 This is the main routine of the INITIALIZE volume utility.
121 0540 1
122 0541 1
123 0542 1 CALLING SEQUENCE:
124 0543 1 INIT_VOLUME (ARG1, ARG2)
125 0544 1
126 0545 1 INPUT PARAMETERS:
127 0546 1 ARG1: program start address
128 0547 1 ARG2: CLI service callback address
129 0548 1
130 0549 1 IMPLICIT INPUTS:
131 0550 1 NONE
132 0551 1
133 0552 1 OUTPUT PARAMETERS:
134 0553 1 NONE
135 0554 1
136 0555 1 IMPLICIT OUTPUTS:
137 0556 1 NONE
138 0557 1
139 0558 1 ROUTINE VALUE:
140 0559 1 assorted status values
141 0560 1
142 0561 1 SIDE EFFECTS:
143 0562 1 volume initialized
144 0563 1
145 0564 1 --
146 0565 1
147 0566 2 BEGIN
148 0567 2
149 0568 2 LOCAL
150 0569 2 P, string scan pointer
151 0570 2 STATUS, system service status
152 0571 2 DEVICE_NAME : BBLOCK [DSC$C_S_BLN]; device name descriptor
153 0572 2
154 0573 2 EXTERNAL
155 0574 2 INIT_OPTIONS : BITVECTOR, parser option flags
156 0575 2 DEVICE_STRING : BBLOCK [DSC$C_S_BLN], device name string descriptor
157 0576 2 USER_NAME : BBLOCK [DSC$C_S_BLN]; user name string descriptor
158 0577 2
159 0578 2 EXTERNAL ROUTINE
160 0579 2 INIT_PARSE, command parser
161 0580 2 TRAN_LOGNAME, translate logical name
162 0581 2 INIT_TAPE, initialize magtape
163 0582 2 INIT_DISK; initialize disk
164 0583 2
165 0584 2 BIND OPTIONS = INIT_OPTIONS : VECTOR;
166 0585 2
167 0586 2 ENABLE MAIN_HANDLER;
168 0587 2
169 0588 2 ! Get the UIC of this process for use in defaulting volume ownership.
170 0589 2 !
```



```

171      0590      2
172      P 0591      2 $GETJPI (ITMLST = UPLIT (WORD (4), WORD (JPI$ UIC),
173      0592      2 LONG (PROCESS_UIC, 0, 0)));
174      0593      2
175      0594      2 USER_NAME [DSC$A_POINTER] = USER_STRING;
176      P 0595      2 $GETJPI (ITMLST = UPLIT (WORD (USERNAME_LEN), WORD (JPI$ USERNAME),
177      0596      2 LONG (USER_STRING, USER_NAME[DSC$W_LENGTH], 0)));
178      0597      2
179      0598      2 ! Parse the command line. Errors are signalled.
180      0599      2 !
181      0600      2
182      0601      2 CHANNEL = 0;
183      0602      2
184      0603      2 INIT_PARSE ();
185      0604      2
186      0605      2 ! First allocate the volume and assign a channel to it. Allocating does most
187      0606      2 ! of the checking that we in fact have a right to touch this device.
188      0607      2 ! Note that we append a zero to the device name if a unit number is not
189      0608      2 ! present. This prevents a generic device search, which is not desirable
190      0609      2 ! in doing an INIT.
191      0610      2 !
192      0611      2
193      0612      2 CH$FILL ( 0, DSC$C_S_BLN, DEVICE_NAME );
194      0613      2 DEVICE_NAME [DSC$B_CLASS] = DSC$R_CLASS_D;
195      0614      2 DEVICE_NAME [DSC$W_LENGTH] = NAMEBUF_LEN;
196      0615      2 DEVICE_NAME [DSC$A_POINTER] = NAME_BUFFER;
197      0616      2
198      0617      2 TRAN_LOGNAME (DEVICE_STRING [DSC$W_LENGTH], DEVICE_NAME [DSC$W_LENGTH]);
199      0618      2
200      0619      2 P = CH$FIND_CH (.DEVICE_NAME [DSC$W_LENGTH], .DEVICE_NAME [DSC$A_POINTER], ':');
201      0620      2
202      0621      2 ! Use the low word, since high word has descriptor information.
203      0622      2 !
204      0623      2 IF CH$FAIL (.P)
205      0624      2 THEN
206      0625      2     P = .DEVICE_NAME [DSC$W_LENGTH] + .DEVICE_NAME [DSC$A_POINTER];
207      0626      2
208      0627      2 IF CH$RCHAR (.P-1) LSSU '0' OR CH$RCHAR (.P-1) GTRU '9'
209      0628      2 THEN
210      0629      2     BEGIN
211      0630      2         CH$WCHAR_A ('0', P);
212      0631      2         CH$WCHAR_A (':', P);
213      0632      2         DEVICE_NAME [DSC$W_LENGTH] = .DEVICE_NAME [DSC$W_LENGTH] + 1;
214      0633      2     END;
215      0634      2
216      0635      2 PHYS_NAME[0] = NAMEBUF_LEN;
217      0636      2 PHYS_NAME[1] = NAME_BUFFER;
218      P 0637      2 STATUS = $ALLOC (DEVNAM = DEVICE_NAME [DSC$W_LENGTH],
219      P 0638      2     PHYLEN = PHYS_NAME[0],
220      0639      2     PHYBUF = PHYS_NAME[0]);
221      0640      2 IF NOT .STATUS
222      0641      2 THEN
223      0642      2     ERR_EXIT (.STATUS);
224      0643      2
225      P 0644      2 STATUS = $ASSIGN (DEVNAM = PHYS_NAME[0],
226      0645      2     CHAN = CHANNEL);
227      0646      2 IF NOT .STATUS
```



```
228 0647 2 THEN
229 0648 2   ERR_EXIT (.STATUS);
230 0649 2
231 0650 2 ! Get the device characteristics and make sure that the device can be
232 0651 2 ! initialized - i.e., that it is file oriented, etc. A mismatch between
233 0652 2 ! primary and secondary device characteristics indicates a spooled device
234 0653 2 ! or something else strange - reject it of so.
235 0654 2 !
236 0655 2
237 0656 2 $GETCHN (CHAN = .CHANNEL, PRIBUF = DEVCHAR_DESC, SCDBUF = DEVCHAR_DESC2);
238 0657 2
239 0658 2 IF CH$NEQ (DIB$K_LENGTH, DEVICE_CHAR, DIB$K_LENGTH, DEVICE_CHAR2, 0)
240 0659 2 OR NOT .DEVICE_CHAR[DEV$V_FOD]
241 0660 2 THEN ERR_EXIT (SS$_NOTFILEDEV);
242 0661 2
243 0662 2 IF NOT .DEVICE_CHAR[DEV$V_AVL]
244 0663 2 THEN ERR_EXIT (SS$_DEVOFFLINE);
245 0664 2
246 0665 2 IF .DEVICE_CHAR[DEV$V_MNT]
247 0666 2 THEN ERR_EXIT (SS$_DEVMOUNT);
248 0667 2
249 0668 2 ! Device is now known not to be mounted.
250 0669 2 !
251 0670 2
252 0671 2 CLEANUP_FLAGS[CLF_CLEARVALID] = 1;
253 0672 2
254 0673 2 IF
255 0674 3 BEGIN
256 0675 3   IF .DEVICE_CHAR[DEV$V_SQD]
257 0676 3   THEN
258 0677 4     ((.OPTIONS[0] AND NOT TAPE_OPTIONS) NEQ 0
259 0678 4     OR (.OPTIONS[1] AND NOT TAPE_OPTIONS2) NEQ 0 )
260 0679 3   ELSE
261 0680 4     ((.OPTIONS[0] AND NOT DISK_OPTIONS) NEQ 0
262 0681 4     OR (.OPTIONS[1] AND NOT DISK_OPTIONS2) NEQ 0 )
263 0682 3   END
264 0683 2 THEN ERR_EXIT (INIT$_ILLOPT);
265 0684 2
266 0685 2 ! Now initialize the volume, depending on its type.
267 0686 2 !
268 0687 2
269 0688 2 IF .DEVICE_CHAR[DEV$V_SQD]
270 0689 2 THEN
271 0690 2   INIT_TAPE ()
272 0691 2 ELSE
273 0692 2   INIT_DISK ();
274 0693 2
275 0694 2 $DASSGN (CHAN = .CHANNEL);
276 0695 2 $DALLOC (DEVNAM = PHYS_NAME);
277 0696 2
278 0697 2 RETURN 1;
279 0698 2
280 0699 1 END;
```

! end of routine INIT_VOLUME

.TITLE INIVOL
.IDENT \V04-000\


```
.PSECT $SPLITS$,NOWRT,NOEXE,2

0004 00000 P.AAA: .WORD 4
0304 00002 .WORD 772
00000000 00000000' 00004 .ADDRESS PROCESS_UIC
00000000 00000000 00008 .LONG 0, 0
000C 00010 P.AAB: .WORD 12
0202 00012 .WORD 514
00000000G 00000000' 00014 .ADDRESS USER_STRING, USER_NAME
00000000 0001C .LONG 0

.PSECT $GLOBAL$,NOEXE,2

00000 CLEANUP_FLAGS::
.BKLB 4
00004 CHANNEL::
.BKLB 4
00008 PROCESS_UIC::
.BKLB 4
0000C DEVICE_CHAR::
.BKLB 116
00080 DEVICE_CHAR2::
.BKLB 116
000F4 PHYS_NAME::
.BKLB 8
000FC NAME_BUFFER::
.BKLB 32
0011C USER_STRING::
.BKLB 12
00000074 00128 DEVCHAR_DESC::
.LONG 116
00000000' 0012C .ADDRESS DEVICE_CHAR
00000074 00130 DEVCHAR_DESC2::
.LONG 116
00000000' 00134 .ADDRESS DEVICE_CHAR2

NAMEBUF LEN== 32
USERNAME LEN== 12
.EXTRN INIT_OPTIONS, DEVICE_STRING
.EXTRN USER_NAME, INIT_PARSE
.EXTRN TRAN_LOGNAME, INIT_TAPE
.EXTRN INIT_DISK, SYSSGETJPI
.EXTRN SYSSALLOC, SYSSASSIGN
.EXTRN SYSSGETCHN, SYSSDASSGN
.EXTRN SYSSDALLOC

.PSECT $CODE$,NOWRT,2

59 0000G CF 9E 00000 .ENTRY INIT_VOLUME, Save R2,R3,R4,R5,R6,R7,R8,R9 : 0533
58 00000000G 00 9E 00002 MOVAB OPTIONS, R9
57 00000000G 00 9E 00007 MOVAB SYSSGETJPI, R8
56 0000' CF 9E 0000E MOVAB LIB$STOP, R7
5E 08 C2 0001A MOVAB DEVICE_CHAR, R6
6D 015F CF DE 0001D SUBL2 #8, SP
7E 7C 00022 MOVAL 17$, (FP) : 0566
CLRQ -(SP) : 0592
```


08	00	0000G	68	CF	0110	7E D4 00024 CF 9F 00026 7E 7C 0002A 7E D4 0002C 07 FB 0002E C6 9E 00031 7E 7C 00038 7E D4 0003A CF 9F 0003C 7E 7C 00040 7E D4 00042 07 FB 00044 F8 A6 D4 00047 00 FB 0004A 00 2C 0004F 6E 00054 02 90 00055 20 B0 00059 C6 9E 0005C 5E DD 00062 CF 9F 00064 02 FB 00068 3A 3A 0006D 02 12 00072 51 D4 00074 51 D5 00076 07 12 00078 6E 3C 0007A AE C0 0007D A1 91 00081 06 1F 00085 A1 91 00087 07 1B 0008B 8F B0 0008D 6E B6 00092 20 D0 00094 C6 9E 00099 7E 7C 000A0 C6 9F 000A2 C6 9F 000A6 AE 9F 000AA 05 FB 000AD 50 D0 000B4 52 E8 000B7 52 DD 000BA 01 FB 000BC 7E 7C 000BF F8 A6 9F 000C1 00E8 C6 9F 000C4 04 FB 000C8 50 D0 000CF 52 E8 000D2 52 DD 000D5 01 FB 000D7 C6 9F 000DA 7E D4 000DE C6 9F 000E0	0000' 0110 0000' F8 00F0 0000G 04 04 00E8 00EC 00000000G 05 67 00000000G 05 67 0124 011C	CLRL -(SP) PUSHAB P.AAA CLRQ -(SP) CLRL -(SP) CALLS #7, SYSS\$GETJPI MOVAB USER_STRING, USER_NAME+4 CLRQ -(SP) CLRL -(SP) PUSHAB P.AAB CLRQ -(SP) CLRL -(SP) CALLS #7, SYSS\$GETJPI CLRL CHANNEL CALLS #0, INIT_PARSE MOVCS #0, (SP); #0, #8, DEVICE_NAME MOVB #2, DEVICE_NAME+3 MOVW #32, DEVICE_NAME MOVAB NAME_BUFFER, DEVICE_NAME+4 PUSHL SP PUSHAB DEVICE_STRING CALLS #2, TRAN_LOGNAME LOCC #58, DEVICE_NAME, @DEVICE_NAME+4 BNEQ 1\$ CLRL R1 TSTL P BNEQ 2\$ MOVZWL DEVICE_NAME, P ADDL2 DEVICE_NAME+4, P CMPB -1(P), #48 BLSSU 3\$ CMPB -1(P), #57 BLEQU 4\$ MOVW #14896, (P)+ INCW DEVICE_NAME MOVL #32, PHYS_NAME MOVAB NAME_BUFFER, PHYS_NAME+4 CLRQ -(SP) PUSHAB PHYS_NAME PUSHAB PHYS_NAME PUSHAB DEVICE_NAME CALLS #5, SYSS\$ALLOC MOVL R0, STATUS BLBS STATUS, 5\$ PUSHL STATUS CALLS #1, LIB\$STOP CLRQ -(SP) PUSHAB CHANNEL PUSHAB PHYS_NAME CALLS #4, SYSS\$ASSIGN MOVL R0, STATUS BLBS STATUS, 6\$ PUSHL STATUS CALLS #1, LIB\$STOP PUSHAB DEVCHAR_DESC2 CLRL -(SP) PUSHAB DEVCHAR_DESC	0594 0596 0601 0603 0612 0613 0614 0615 0617 0619 0623 0625 0627 0630 0632 0635 0636 0639 0640 0642 0645 0646 0648 0656
----	----	-------	----	----	------	---	---	---	--

74	A6	00000000G	00	F8	7E	D4	000E4	CLRL	-(SP)		
			66		A6	DD	000E6	PUSHL	CHANNEL		
	08	01	A6	0074	05	FB	000E9	CALLS	#5, SYSSGETCHN		
			7E		8F	29	000F0	CMPC3	#116, DEVICE_CHAR, DEVICE_CHAR2		0658
			67		05	12	000F7	BNEQ	7\$		
	07	02	A6	01CC	06	E0	000F9	BBS	#6, DEVICE_CHAR+1, 8\$		0659
			7E		8F	3C	000FE	MOVZWL	#460, -(SP)		0660
			67		01	FB	00103	CALLS	#1, LIB\$STOP		
	07	02	A6		02	E0	00106	BBS	#2, DEVICE_CHAR+2, 9\$		0662
			7E	84	8F	9A	0010B	MOVZBL	#132, -(SP)		0663
			67		01	FB	0010F	CALLS	#1, LIB\$STOP		
	07	02	A6		03	E1	00112	BBS	#3, DEVICE_CHAR+2, 10\$		0665
			7E	6C	8F	9A	00117	MOVZBL	#108, -(SP)		0666
			67		01	FB	0011B	CALLS	#1, LIB\$STOP		
		F4	A6		02	88	0011E	BISB2	#2, CLEANUP_FLAGS		0671
13			66		05	E1	00122	BBC	#5, DEVICE_CHAR, 11\$		0675
		30FFDB98	8F		69	D3	00126	BITL	OPTIONS, #822074264		0677
					1D	12	0012D	BNEQ	13\$		
		FFFFFF835	8F	04	A9	D3	0012F	BITL	OPTIONS+4, #-1995		0678
					11	11	00137	BRB	12\$		
		48000022	8F		69	D3	00139	BITL	OPTIONS, #1207959586		0680
					0A	12	00140	BNEQ	13\$		
		FFFFFF3CA	8F	04	A9	D3	00142	BITL	OPTIONS+4, #-3126		0681
					09	13	0014A	BEQL	14\$		
			67	00758034	8F	DD	0014C	PUSHL	#7700532		0683
			66		01	FB	00152	CALLS	#1, LIB\$STOP		
07			CF		05	E1	00155	BBC	#5, DEVICE_CHAR, 15\$		0688
		0000G			00	FB	00159	CALLS	#0, INIT_TAPE		0690
					05	11	0015E	BRB	16\$		
		0000G	CF		00	FB	00160	CALLS	#0, INIT_DISK		0692
				F8	A6	DD	00165	PUSHL	CHANNEL		0694
		00000000G	00		01	FB	00168	CALLS	#1, SYSSDASSGN		
					7E	D4	0016F	CLRL	-(SP)		0695
		00000000G	00	00E8	C6	9F	00171	PUSHAB	PHYS_NAME		
			50		02	FB	00175	CALLS	#2, SYSSDALLOC		0697
					01	D0	0017C	MOVL	#1, R0		0699
					04	00	0017F	RET			0566
					0000	00	00180	.WORD	Save nothing		
					7E	D4	00182	CLRL	-(SP)		
					5E	DD	00184	PUSHL	SP		
		0000V	7E	04	AC	7D	00186	MOVQ	4(AP), -(SP)		
			CF		03	FB	0018A	CALLS	#3, MAIN_HANDLER		
					04	00	0018F	RET			

; Routine Size: 400 bytes, Routine Base: \$CODE\$ + 0000


```
282 0700 1 ROUTINE MAIN_HANDLER (SIGNAL, MECHANISM) =
283 0701 1
284 0702 1 !++
285 0703 1
286 0704 1 FUNCTIONAL DESCRIPTION:
287 0705 1
288 0706 1 This routine is the main level condition handler for the INIT
289 0707 1 utility. It cleans up and returns the error status to the caller
290 0708 1 (the CLI).
291 0709 1
292 0710 1
293 0711 1 CALLING SEQUENCE:
294 0712 1 MAIN_HANDLER (ARG1, ARG2)
295 0713 1
296 0714 1 INPUT PARAMETERS:
297 0715 1 ARG1: address of signal array
298 0716 1 ARG2: address of mechanism array
299 0717 1
300 0718 1 IMPLICIT INPUTS:
301 0719 1 NONE
302 0720 1
303 0721 1 OUTPUT PARAMETERS:
304 0722 1 NONE
305 0723 1
306 0724 1 IMPLICIT OUTPUTS:
307 0725 1 NONE
308 0726 1
309 0727 1 ROUTINE VALUE:
310 0728 1 $$$_RESIGNAL
311 0729 1
312 0730 1 SIDE EFFECTS:
313 0731 1 cleanup done on unwind
314 0732 1
315 0733 1 !--
316 0734 1
317 0735 2 BEGIN
318 0736 2
319 0737 2 MAP
320 0738 2 SIGNAL : REF BBLOCK, ! signal array
321 0739 2 MECHANISM : REF BBLOCK; ! mechanism array
322 0740 2
323 0741 2 EXTERNAL ROUTINE
324 0742 2 CLEAR_VALID; ! clear software volume valid
325 0743 2
326 0744 2
327 0745 2
328 0746 2 IF .BBLOCK [SIGNAL[CHFSL_SIG_NAME], STSV_FAC_NO] EQL 0
329 0747 2 OR .BBLOCK [SIGNAL[CHFSL_SIG_NAME], STSV_FAC_NO] EQL MOUN$ FACILITY
330 0748 2 THEN BBLOCK [SIGNAL[CHFSL_SIG_NAME], STSV_FAC_NO] = INIT$ FACILITY;
331 0749 2
332 0750 2 IF .BBLOCK [SIGNAL[CHFSL_SIG_NAME], STSV_SEVERITY] EQL STSK_SEVERE
333 0751 2 THEN
334 0752 3 BEGIN
335 0753 3
336 0754 3 IF .CHANNEL NEQ 0
337 0755 3 THEN
338 0756 4 BEGIN
```


INIVOL
V04-000

L 2
16-Sep-1984 01:41:24
14-Sep-1984 12:35:19

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[INIT.SRC]INIVOL.B32;1 Page 11
(4)

```

: 339      0757 4      $QIOW (CHAN = .CHANNEL, FUNC = (IOS_AVAILABLE OR IOSM_INHERLOG));
: 340      0758 4      IF .CLEANUP_FLAGS[CLF_CLEARVALID]
: 341      0759 4      THEN KERNEL_CALL (CLEAR_VALID);
: 342      0760 4      $DASSGN (CHAN = .CHANNEL);
: 343      0761 4      CHANNEL = 0;
: 344      0762 4      END;
: 345      0763 4
: 346      0764 4      $DALLOC (DEVNAM = PHYS_NAME);
: 347      0765 4
: 348      0766 4      END;
: 349      0767 4
: 350      0768 2      RETURN SS$_RESIGNAL;
: 351      0769 2
: 352      0770 1      END;

```

! end of routine MAIN_HANDLER

				.EXTRN CLEAR_VALID, SYSS\$QIOW		
				.EXTRN SYSS\$CMKRNL		
0004 00000 MAIN_HANDLER:						
			52	0000'	CF 9E 00002	.WORD Save R2
			50	04	AC D0 00C07	MOVAB CHANNEL, R2
			50		04 C0 0000B	MOVL SIGNAL, R0
		OFFF	8F	02	A0 B3 0000E	ADDL2 #4, R0
					0C 13 00014	BITW 2(R0), #4095
					00 ED 00016	BEQL 1\$
00000072	8F	02	A0		0A 12 00020	CMPZV #0, #12, 2(R0), #114
					8F F0 00022	BNEQ 2\$
02	A0		0C	00000075	00 ED 0002C	INSV #117, #0, #12, 2(R0)
	04		60	03	4B 12 00031	CMPZV #0, #3, (R0), #4
					62 D0 00033	BNEQ 5\$
			50		39 13 00036	MOVL CHANNEL, R0
					7E 7C 00038	BEQL 4\$
					7E 7C 0003A	CLRQ -(SP)
					7E 7C 0003C	CLRQ -(SP)
					7E 7C 0003E	CLRQ -(SP)
					7E D4 00040	CLRQ -(SP)
			7E	0811	8F 3C 00042	CLRL -(SP)
					50 DD 00047	MOVZWL #2065, -(SP)
					7E D4 00049	PUSHL R0
					0C FB 0004B	CLRL -(SP)
		00000000G	00		01 E1 00052	CALLS #12, SYSS\$QIOW
		OF FC	A2		7E D4 00057	BBC #1, CLEANUP_FLAGS, 3\$
					5E DD 00059	CLRL -(SP)
				0000G	CF 9F 0005B	PUSHL SP
		00000000G	9F		03 FB 0005F	PUSHAB CLEAR_VALID
		00000000G	00		62 DD 00066	CALLS #3, @SYSS\$CMKRNL
					01 FB 00068	PUSHL CHANNEL
					62 D4 0006F	CALLS #1, SYSS\$DASSGN
					7E D4 00071	CLRL CHANNEL
				00F0	C2 9F 00073	CLRL -(SP)
		00000000G	00		02 FB 00077	PUSHAB PHYS_NAME
			50	0918	8F 3C 0007E	CALLS #2, SYSS\$DALLOC
					04 00083	MOVZWL #2328, R0
						RET

; Routine Size: 132 bytes, Routine Base: \$CODE\$ + 0190

: 353 0771 1
: 354 0772 1 END
: 355 0773 0 ELUDOM

.EXTRN LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes					
\$GLOBALS	312	NOVEC,	WRT,	RD	NOEXE,NOSHR,	LCL,	REL, CON,NOPIC,ALIGN(2)
\$PLITS	32	NOVEC,NOWRT,	RD	NOEXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)
\$CODE\$	532	NOVEC,NOWRT,	RD	EXE,NOSHR,	LCL,	REL,	CON,NOPIC,ALIGN(2)
. ABS .	0	NOVEC,NOWRT,NORD	NOEXE,NOSHR,	LCL,	ABS,	CON,NOPIC,ALIGN(0)	

Library Statistics

File	----- Symbols -----		Pages Mapped	Processing Time
	Total	Loaded Percent		
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	36 0	1000	00:01.9

COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:INIVOL/OBJ=OBJ\$:INIVOL MSRC\$:INIVOL/UPDATE=(ENH\$:INIVOL)

; Size: 532 code + 344 data bytes
; Run Time: 00:17.3
; Elapsed Time: 00:41.1
; Lines/CPU Min: 2688
; Lexemes/CPU-Min: 39683
; Memory Used: 157 pages
; Compilation Complete

0188 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

INPSMB
MAP

INSDEF
SQL

INPSMBMSG
LIS

RSXLBDF
SQL

INSCREATE
LIS

INITIO
LIS

INSTAL

INSTALL
MAP

INSCMD
CLD

INSPREFIX
REQ

INPSMBCLD
CLD

INPSMB
LIS

INSOLDCMD
CLD

INTUOL
LIS

RDHOME
LIS

INPSMB

INPSMBCLD
LIS

INSCMD
LIS