

[illegible][illegible]

```
1 0001 0 MODULE MOUNTIMG (  
2 0002 0     MAIN = PARSE COMMAND,  
3 0003 0     ADDRESSING MODE (EXTERNAL = GENERAL),  
4 0004 0     LANGUAGE (BLISS32),  
5 0005 0     IDENT = 'V04-000'  
6 0006 0 ) =  
7 0007 1 BEGIN  
8 0008 1  
9 0009 1  
10 0010 1 *****  
11 0011 1 *  
12 0012 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY  
13 0013 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.  
14 0014 1 *  ALL RIGHTS RESERVED.  
15 0015 1 *  
16 0016 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  
17 0017 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE  
18 0018 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER  
19 0019 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  
20 0020 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY  
21 0021 1 *  TRANSFERRED.  
22 0022 1 *  
23 0023 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE  
24 0024 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT  
25 0025 1 *  CORPORATION.  
26 0026 1 *  
27 0027 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS  
28 0028 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.  
29 0029 1 *  
30 0030 1 *  
31 0031 1 *****  
32 0032 1  
33 0033 1 ++  
34 0034 1  
35 0035 1 FACILITY: MOUNT Utility Structure Level 1  
36 0036 1  
37 0037 1 ABSTRACT:  
38 0038 1  
39 0039 1     This module contains the data base and utilities used to acquire the  
40 0040 1     MOUNT command line from the CLI parser.  
41 0041 1  
42 0042 1 ENVIRONMENT:  
43 0043 1  
44 0044 1     STARLET operating system, including privileged system services  
45 0045 1     and internal exec routines.  
46 0046 1  
47 0047 1 --  
48 0048 1  
49 0049 1  
50 0050 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 29-Sep-1977 16:58  
51 0051 1  
52 0052 1 MODIFIED BY:  
53 0053 1  
54 0054 1     V03-018 HH0044      Hai Huang      09-Aug-1984  
55 0055 1     Correctly parse /CACHE options.  
56 0056 1  
57 0057 1     V03-017 HH0041      Hai Huang      24-Jul-1984
```

58	0058	1	Remove REQUIRE 'LIBDS:[VMSLIB.OBJ]MOUNTMSG.B32'.
59	0059	1	
60	0060	1	V03-016 DAS0003 David Solomon 09-Jul-1984
61	0061	1	Add support for /NOREBUILD.
62	0062	1	
63	0063	1	V03-015 HH0028 Hai Huang 27-Jun-1984
64	0064	1	Make several qualifiers negatable (/CLUSTER, /GROUP,
65	0065	1	/SYSTEM).
66	0066	1	
67	0067	1	V03-014 HH0004 Hai Huang 09-Mar-1984
68	0068	1	Add cluster-wide mount support.
69	0069	1	
70	0070	1	V03-013 WMC0001 Wayne Cardoza 16-Jan-1984
71	0071	1	Disable all journaling qualifiers.
72	0072	1	
73	0073	1	V03-012 MCN0141 Maria del C. Nasr 27-Dec-1983
74	0074	1	Add VALCNVERR message, and eliminate PARSE_ERROR routine
75	0075	1	since it is not needed with new CLI interface.
76	0076	1	
77	0077	1	V03-011 DAS0002 David Solomon 09-Dec-1983
78	0078	1	Fix symbol name that was too long.
79	0079	1	
80	0080	1	V03-010 DAS0001 David Solomon 29-Nov-1983
81	0081	1	Add support for specifying maximum journal record size
82	0082	1	with a new keyword, /JOURNAL=(RECORD_SIZE=n).
83	0083	1	
84	0084	1	V03-009 MCN0138 Maria del C. Nasr 21-Nov-1983
85	0085	1	Turn off NEWJOURNAL when /NOJOURNAL is specified.
86	0086	1	
87	0087	1	V03-008 MCN0137 Maria del C. Nasr 12-Jul-1983
88	0088	1	Change to new CLI interface.
89	0089	1	
90	0090	1	V03-007 LMP0140 L. Mark Pilant 22-Aug-1983
91	0091	1	Add support for alphanumeric UICs.
92	0092	1	
93	0093	1	V03-006 MMD0188 Meg Dumont, 7-Jul-1983 10:00
94	0094	1	Make the default for AVL/AVR the same from the DCL call
95	0095	1	and from the system service call.
96	0096	1	
97	0097	1	V03-005 MMD0116 Meg Dumont, 29-Mar-1983 0:40
98	0098	1	Add support for AVL, AVR and new VMS prot on tape
99	0099	1	
100	0100	1	V03-004 STJ49203 Steven T. Jeffreys, 08-Feb-1982
101	0101	1	Set MNTSV_OVR_SETID if /OVERRIDE=SETID was specified.
102	0102	1	
103	0103	1	V03-003 STJ0318 Steven T. Jeffreys, 15-Aug-1982
104	0104	1	Added support for the journaling qualifiers.
105	0105	1	
106	0106	1	V03-002 STJ0303 Steven T. Jeffreys, 18-May-1982
107	0107	1	Replace the obsolete /UNLOCK qualifier with the /UNLOAD
108	0108	1	qualifier.
109	0109	1	
110	0110	1	V03-001 STJ0239 Steven T. Jeffreys, 17-Mar-1982
111	0111	1	Relax the parsing restrictions on the device name
112	0112	1	as specified in the /PROCESSOR=NAME:<device name>
113	0113	1	qualifier. Specifically, if no ":" is specified in
114	0114	1	the device name, put one there.

```
115 0115 1
116 0116 1
117 0117 1
118 0118 1
119 0119 1
120 0120 1
121 0121 1
122 0122 1
123 0123 1
124 0124 1
125 0125 1
126 0126 1
127 0127 1
128 0128 1
129 0129 1
130 0130 1
131 0131 1
132 0132 1
133 0133 1
134 0134 1
135 0135 1
136 0136 1
137 0137 1
138 0138 1
139 0139 1
140 0140 1
141 0141 1
142 0142 1
143 0143 1
144 0144 1
145 0145 1
146 0146 1
147 0147 1
148 0148 1
149 0149 1
150 0150 1
151 0151 1
152 0152 1
153 0153 1
154 0154 1
155 0155 1
156 0156 1
157 0688 1
158 0820 1
159 0821 1

V02-016 STJ0226 Steven T. Jeffreys, 17-Feb-1982
Do not set the inhibit bit in the final status code.
This effectively undoes edit #14.

V02-015 STJ0213 Steven T. Jeffreys, 11-Feb-1982
Add support for the /COMMENT qualifier.

V02-014 STJ0201 Steven T. Jeffreys, 04-Feb-1982
Set the inhibit bit in the final status code.

V02-013 STJ0187 Steven T. Jeffreys, 25-Jan-1982
Changed MNT$V_MOUNTVER to MNT$V_NOMNTVER.

V02-012 STJ0172 Steven T. Jeffreys, 08-Jan-1982
Changed $MOUNT interface to use *new* item list format.

V02-011 STJ0162 Steven T. Jeffreys, 04-Jan-1982
Added support for the /OVERRIDE=LOCK, /NOCACHE, /MOUNTVER,
and /MESSAGE qualifiers.

V02-010 STJ0153 Steven T. Jeffreys, 02-Jan-1981
Extensive rewrite to support the $MOUNT system service.

V02-009 STJ0147 Steven T. Jeffreys, 01-Dec-1981
Fixed TPARSE table for /PROCESSOR= option.

V02-008 STJ0137 Steven T. Jeffreys, 02-Nov-1981
Convert the command line parser to a separate image,
which will parse the command line and then call the
$MOUNT system service to complete the mount.

V02-007 STJ0036 Steven T. Jeffreys, 11-May-1981
Added support for /ASSIST qualifier.

V02-006 ACG0167 Andrew C. Goldstein, 18-Apr-1980 13:38
Previous revision history moved to MOUNT.REV

**

LIBRARY 'SYSS$LIBRARY:LIB.L32';
REQUIRE 'SRC$:MOUDEF.B32';
REQUIRE 'LIBD$:[VMSLIB.OBJ]INITMSG.REQ';
LIBRARY 'SYSS$LIBRARY:CLIMAC.L32';
LIBRARY 'SYSS$LIBRARY:TPAMAC.L32';
```

```
161 0822 1
162 0823 1 FORWARD ROUTINE
163 0824 1
164 0825 1     CACHE_ACT      : NOVALUE,
165 0826 1     DATACHECK_ACT : NOVALUE,
166 0827 1     DENSITY_ACT   : NOVALUE,
167 0828 1     GET_DEVICE    : NOVALUE,
168 0829 1     GET_LABEL     : NOVALUE,
169 0830 1     GET_LOG_NAME  : NOVALUE,
170 0831 1     INITIALIZE_ACT : NOVALUE,
171 0832 1     JOURNAL_ACT   : NOVALUE,
172 0833 1     OVERRIDE_ACT  : NOVALUE,
173 0834 1     OWNER_UIC_ACT  : NOVALUE,
174 0835 1     PARSE_QUALIFIER : NOVALUE,
175 0836 1     PROCESSOR_ACT  : NOVALUE,
176 0837 1     PROTECTION_ACT : NOVALUE,
177 0838 1     MAIN_HANDLER,
178 0839 1     BUILD_LIST      : NOVALUE;
179 0840 1
180 0841 1
181 0842 1
182 0843 1
183 0844 1
184 0845 1
185 0846 1
186 0847 1
187 0848 1 OWN
188 0849 1     DEVICE_COUNT,      : number of devices specified
189 0850 1     LABEL_COUNT,    : number of volume labels specified
190 0851 1     DEVICE_STRING   : VECTOR [DEVMAX*2], : descriptors of device name strings
191 0852 1     LABEL_STRING    : VECTOR [LABMAX*2], : descriptors of volume label strings
192 0853 1     LOG_NAME        : BBLOCK [DSC$C_S_BLN], : descriptor of logical name string
193 0854 1     MOUNT_OPTIONS    : BITVECTOR [64], : option flags
194 0855 1     MOUNT_FLAGS      : BBLOCK [4], : mount option flags for service
195 0856 1
196 0857 1     ! Value of qualifiers
197 0858 1
198 0859 1     ACCESS,              : value of /ACCESSED qualifier
199 0860 1     ACP_STRING          : BBLOCK [DSC$C_S_BLN], : descriptor of ACP device or name string
200 0861 1     BLOCKSZ,           : value of /BLOCKSIZE qualifier
201 0862 1     EXT_CACHE,         : space to allocate for extent cache
202 0863 1     FID_CACHE,         : space to allocate for file ID cache
203 0864 1     QUO_CACHE,        : space to allocate for quota cache
204 0865 1     COMMENT_STRING    : BBLOCK [DSC$C_S_BLN], : descriptor of /COMMENT string
205 0866 1     DENSITY,            : value of /DENSITY qualifier
206 0867 1     EXTENSION,         : value of /EXTENSION qualifier
207 0868 1     JRNL_QUOTA,         : value of /JOURNAL=QUOTA keyword
208 0869 1     JRNL_EXTEND,       : value of /JOURNAL=EXTEND keyword
209 0870 1     JRNL_SIZE,         : value of /JOURNAL=SIZE keyword
210 0871 1     JRNL_RECORD_SIZE,  : value of /JOURNAL=RECORD SIZE keyword
211 0872 1     OWNER_UIC,         : value of /OWNER UIC qualifier
212 0873 1     PROTECTION,       : value of /PROTECTION qualifier
213 0874 1     RECORDSZ,         : value of /RECORDSZ qualifier
214 0875 1     STRUCT_NAME      : BBLOCK [DSC$C_S_BLN], : descriptor of volume set name
215 0876 1     (value of /BIND qualifier)
216 0877 1     WINDOW,           : value of /WINDOWS qualifier
217 0878 1
```



```
: 218      0879 1      CLI_DESC      : BBLOCK [DSC$C_S_BLN], ! CLI work descriptor
: 219      0880 1      EXT_LIMIT     : INITIAL (-1), ! limit of disk free space to cache
: 220      0881 1      TPARSE_BLOCK  : BBLOCK [TPASK_LENGTH0]
: 221      0882 1      : INITIAL (TPASK_COUNT0, TPASK_BLANKS OR TPASK_ABBREV),
: 222      0883 1      UIC,
: 223      0884 1      ZERO;          ! variable whose value is 0
: 224      0885 1
: 225      0886 1      LITERAL
: 226      0887 1          ITEM_SIZE = 12,
: 227      0888 1          NUMBER_OF_ITEMS = 18,
: 228      0889 1          ITEM_LIST_SIZE = ((ITEM_SIZE * DEVMAX) * 2) + (NUMBER_OF_ITEMS * ITEM_SIZE) + 4;
: 229      0890 1
: 230      0891 1      ! Descriptors for qualifiers names, used while parsing command line.
: 231      0892 1      !
: 232      0893 1      BIND
: 233      0894 1          ACCESSED_DESC = $DESCRIPTOR('ACCESSED'),
: 234      0895 1          ASSIST_DESC   = $DESCRIPTOR('ASSIST'),
: 235      0896 1          AUTOMATIC_DESC = $DESCRIPTOR('AUTOMATIC'),
: 236      0897 1          BIND_DESC    = $DESCRIPTOR('BIND'),
: 237      0898 1          BLOCK_DESC   = $DESCRIPTOR('BLOCKSIZE'),
: 238      0899 1          CACHE_DESC   = $DESCRIPTOR('CACHE'),
: 239      0900 1          CLUSTER_DESC  = $DESCRIPTOR('CLUSTER'),
: 240      0901 1          COMMENT_DESC  = $DESCRIPTOR('COMMENT'),
: 241      0902 1          DATA_DESC    = $DESCRIPTOR('DATA CHECK'),
: 242      0903 1          DENSITY_DESC  = $DESCRIPTOR('DENSITY'),
: 243      0904 1          EXTENSION_DESC = $DESCRIPTOR('EXTENSION'),
: 244      0905 1          FOREIGN_DESC  = $DESCRIPTOR('FOREIGN'),
: 245      0906 1          GROUP_DESC    = $DESCRIPTOR('GROUP'),
: 246      0907 1          HDR3_DESC     = $DESCRIPTOR('HDR3'),
: 247      0908 1          INITIALIZE_DESC = $DESCRIPTOR('INITIALIZE'),
: 248      0909 1          JOURNAL_DESC  = $DESCRIPTOR('JOURNAL'),
: 249      0910 1          LABEL_DESC    = $DESCRIPTOR('LABEL'),
: 250      0911 1          MESSAGE_DESC  = $DESCRIPTOR('MESSAGE'),
: 251      0912 1          MOUNT_VER_DESC = $DESCRIPTOR('MOUNT VERIFICATION'),
: 252      0913 1          NOLABEL_DESC  = $DESCRIPTOR('NOLABEL'),
: 253      0914 1          OVERRIDE_DESC = $DESCRIPTOR('OVERRIDE'),
: 254      0915 1          OWNER_DESC    = $DESCRIPTOR('OWNER UIC'),
: 255      0916 1          PROCESSOR_DESC = $DESCRIPTOR('PROCESSOR'),
: 256      0917 1          PROTECTION_DESC = $DESCRIPTOR('PROTECTION'),
: 257      0918 1          QUOTA_DESC    = $DESCRIPTOR('QUOTA'),
: 258      0919 1          REBUILD_DESC  = $DESCRIPTOR('REBUILD'),
: 259      0920 1          RECORD_DESC   = $DESCRIPTOR('RECORDSIZE'),
: 260      0921 1          SHARE_DESC    = $DESCRIPTOR('SHARE'),
: 261      0922 1          SYSTEM_DESC   = $DESCRIPTOR('SYSTEM'),
: 262      0923 1          UNLOAD_DESC    = $DESCRIPTOR('UNLOAD'),
: 263      0924 1          WINDOW_DESC   = $DESCRIPTOR('WINDOWS'),
: 264      0925 1          WRITE_DESC    = $DESCRIPTOR('WRITE');
: 265      0926 1
: 266      0927 1      ! CLI parsing routines
: 267      0928 1      !
: 268      0929 1      EXTERNAL ROUTINE
: 269      0930 1          LIB$CVT DTB,
: 270      0931 1          STR$COPY DX,
: 271      0932 1          CLISGET VALUE,
: 272      0933 1          CLISPRESENT;
: 273      0934 1
: 274      0935 1      EXTERNAL LITERAL
```

! retrieves qualifiers value
! determines if qualifier appears in
! command

MOUNTIMG
V04-000

D S
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31

VAX-11 Bliss-32 V4.0-742
[MOUNT.SRC]MOUNTIMG.B32;1

Page 6
(2)

:	275	0936	1	CLIS-ABSENT
:	276	0937	1	CLIS-DEFAULTED,
:	277	0938	1	CLIS-NEGATED,
:	278	0939	1	CLIS-PRESENT;
:	279	0940	1	

MOU
V04


```
281 0941 1 GLOBAL ROUTINE PARSE_COMMAND =
282 0942 1
283 0943 1 ++
284 0944 1
285 0945 1 FUNCTIONAL DESCRIPTION:
286 0946 1
287 0947 1 This routine parses the MOUNT command line by calling the CLI
288 0948 1 result parse routines, and leaves the results in the global data
289 0949 1 area.
290 0950 1
291 0951 1
292 0952 1 CALLING SEQUENCE:
293 0953 1 MOUNT_PARSE
294 0954 1
295 0955 1 INPUT PARAMETERS:
296 0956 1
297 0957 1 IMPLICIT INPUTS:
298 0958 1 NONE
299 0959 1
300 0960 1 OUTPUT PARAMETERS:
301 0961 1 NONE
302 0962 1
303 0963 1 IMPLICIT OUTPUTS:
304 0964 1 parser impure area on preceding pages
305 0965 1
306 0966 1 ROUTINE VALUE:
307 0967 1 NONE
308 0968 1
309 0969 1 SIDE EFFECTS:
310 0970 1 NONE
311 0971 1
312 0972 1 --
313 0973 1
314 0974 2 BEGIN
315 0975 2
316 0976 2 LOCAL
317 0977 2 ITEM_LIST : BBLOCK [ITEM_LIST_SIZE], ! Storage for item list
318 0978 2 END_OF_LIST, ! Pointer to end of item list
319 0979 2 STATUS;
320 0980 2
321 0981 2 ! Enable the main condition handler. The handler will ensure that
322 0982 2 the return status will have the MOUNT facility code.
323 0983 2
324 0984 2
325 0985 2 ENABLE MAIN_HANDLER;
326 0986 2
327 0987 2 ! Initialize list for system service.
328 0988 2
329 0989 2 END_OF_LIST = ITEM_LIST;
330 0990 2
331 0991 2 ! Initialize result parsing.
332 0992 2
333 0993 2 ZERO = 0;
334 0994 2 MOUNT_OPTIONS = MOUNT_OPTIONS+4 = 0;
335 0995 2 MOUNT_OPTIONS[OPT_MESSAGE] = 1;
336 0996 2 MOUNT_OPTIONS[OPT_NOSHARE] = 1;
337 0997 2 MOUNT_OPTIONS[OPT_NOLABEL] = 1;
```

```
0998 2 MOUNT_OPTIONS[OPT_NOQUOTA] = 1;
0999 2 MOUNT_OPTIONS[OPT_NOHDR3] = 1;
1000 2 MOUNT_OPTIONS[OPT_NOUNLOAD] = 1;
1001 2
1002 2 ! Initialize CLI descriptor
1003 2
1004 2 CH$FILL ( 0, DSC$C_S_BLN, CLI_DESC );
1005 2 CLI_DESC [DSC$B_CLASS] = DSC$R_CLASS_D;
1006 2
1007 2 PARSE_QUALIFIER ();
1008 2
1009 2 ! Get device names
1010 2
1011 2 GET_DEVICE ();
1012 2
1013 2 ! Get volume labels
1014 2
1015 2 GET_LABEL ();
1016 2
1017 2 ! Get logical name
1018 2
1019 2 GET_LOG_NAME ();
1020 2
1021 2 ! If no label given, construct null label string
1022 2
1023 2
1024 2 IF NOT .MOUNT_OPTIONS [OPT_LABEL]
1025 2 THEN
1026 2 BEGIN
1027 2 LABEL_STRING [0] = 0;
1028 2 LABEL_STRING [1] = LABEL_STRING [1];
1029 2 END;
1030 2
1031 2 ! Create a counted list of the addresses of all device names descriptors
1032 2
1033 2 INCR J FROM 0 TO (.DEVICE_COUNT - 1)
1034 2 DO
1035 2 BUILD_LIST ( MNT$ DEVNAM,
1036 2 .DEVICE_STRING [.J*2],
1037 2 .DEVICE_STRING [(J*2)+1],
1038 2 END_OF_LIST );
1039 2
1040 2 ! Create a counted list of the addresses of all volume name descriptors
1041 2
1042 2 INCR J FROM 0 TO (.LABEL_COUNT - 1)
1043 2 DO
1044 2 BUILD_LIST ( MNT$ VOLNAM,
1045 2 .LABEL_STRING [.J*2],
1046 2 .LABEL_STRING [(J*2)+1],
1047 2 END_OF_LIST );
1048 2
1049 2 ! Set up the parameter addresses for all specified parameters
1050 2
1051 2 ! Process the LOGNAM parameter
1052 2
1053 2
1054 2 IF .MOUNT_OPTIONS [OPT_LOG_NAME]
```

```
395 1055 2 THEN
396 1056     BUILD_LIST ( MNT$_LOGNAM,
397 1057                 .LOG_NAME [DSC$W_LENGTH],
398 1058                 .LOG_NAME [DSC$A_POINTER],
399 1059                 END_OF_LIST );
400 1060
401 1061 2 ! Process the /ACCESSED qualifier
402 1062
403 1063 2 IF .MOUNT_OPTIONS [OPT_ACCESSED]
404 1064 2 THEN
405 1065 2     BUILD_LIST ( MNT$_ACCESSED, 4, ACCESS, END_OF_LIST );
406 1066
407 1067 2 ! Process the /BIND qualifier
408 1068
409 1069 2 IF .MOUNT_OPTIONS [OPT_BIND]
410 1070 2 THEN
411 1071 2     BUILD_LIST ( MNT$_VOLSET, .STRUCT_NAME[DSC$W_LENGTH],
412 1072 2                 .STRUCT_NAME[DSC$A_POINTER], END_OF_LIST );
413 1073
414 1074 2 ! Process the /BLOCKSIZE qualifier
415 1075
416 1076 2 IF .MOUNT_OPTIONS [OPT_BLOCKSIZE]
417 1077 2 THEN
418 1078 2     BUILD_LIST ( MNT$_BLOCKSIZE, 4, BLOCKSZ, END_OF_LIST );
419 1079
420 1080 2 ! Process the /CACHE=([NO]EXTENT) qualifier
421 1081
422 1082 2 IF .EXT_CACHE GTR 0
423 1083 2 THEN
424 1084 2     BEGIN
425 1085 2         BUILD_LIST (MNT$_EXTENT, 4, EXT_CACHE, END_OF_LIST);
426 1086 2     END;
427 1087 2 IF .MOUNT_OPTIONS [OPT_NOEXT_C]
428 1088 2 THEN
429 1089 2     BUILD_LIST (MNT$_EXTENT, 4, ZERO, END_OF_LIST);
430 1090
431 1091 2 ! Process the /CACHE=([NO]FILE_ID) qualifier
432 1092
433 1093 2 IF .MOUNT_OPTIONS [OPT_NOFID_C]
434 1094 2 THEN
435 1095 2     FID_CACHE= 1;
436 1096 2 IF .FID_CACHE GTR 0
437 1097 2 THEN
438 1098 2     BUILD_LIST (MNT$_FILEID, 4, FID_CACHE, END_OF_LIST);
439 1099
440 1100 2 ! Process the /CACHE=(LIMIT) qualifier
441 1101
442 1102 2 IF .EXT_LIMIT GTR -1
443 1103 2 THEN
444 1104 2     BUILD_LIST (MNT$_LIMIT, 4, EXT_LIMIT, END_OF_LIST);
445 1105
446 1106 2 ! Process the /CACHE=([NO]QUOTA) qualifier
447 1107
448 1108 2 IF .MOUNT_OPTIONS [OPT_NOQUO_C]
449 1109 2 THEN
450 1110 2     BUILD_LIST (MNT$_QUOTA, 4, ZERO, END_OF_LIST);
451 1111 2 IF .QUO_CACHE GTR 0
```

```
.. 452 1112 2 THEN
.. 453 1113 2 BUILD_LIST (MNT$_QUOTA, 4, QUO_CACHE, END_OF_LIST);
.. 454 1114 2
.. 455 1115 2 ! Process the /COMMENT qualifier
.. 456 1116 2
.. 457 1117 2 IF .MOUNT_OPTIONS [OPT_COMMENT]
.. 458 1118 2 THEN
.. 459 1119 2 BUILD_LIST ( MNT$ COMMENT, COMMENT_STRING[DSC$W_LENGTH],
.. 460 1120 2 .COMMENT_STRING[DSC$A_POINTER], END_OF_LIST );
.. 461 1121 2
.. 462 1122 2 ! Process the /DENSITY qualifier
.. 463 1123 2
.. 464 1124 2 IF .MOUNT_OPTIONS [OPT_DENSITY]
.. 465 1125 2 THEN
.. 466 1126 2 BUILD_LIST (MNT$_DENSITY, 4, DENSITY, END_OF_LIST);
.. 467 1127 2
.. 468 1128 2 ! Process the /EXTENSION qualifier
.. 469 1129 2
.. 470 1130 2 IF .MOUNT_OPTIONS [OPT_EXTENSION]
.. 471 1131 2 THEN
.. 472 1132 2 BUILD_LIST ( MNT$_EXTENSION, 4, EXTENSION, END_OF_LIST );
.. 473 1133 2
.. 474 1134 2 ! Process the /JOURNAL qualifier options
.. 475 1135 2
.. 476 1136 2 IF .JRNL_SIZE NEQ 0
.. 477 1137 2 THEN
.. 478 1138 2 BUILD_LIST (MNT$_JRNL_SIZE, 4, JRNL_SIZE, END_OF_LIST);
.. 479 1139 2
.. 480 1140 2 IF .JRNL_RECORD_SIZE NEQ 0
.. 481 1141 2 THEN
.. 482 1142 2 BUILD_LIST (MNT$_JRNL_RECORD_SIZE, 4, JRNL_RECORD_SIZE, END_OF_LIST);
.. 483 1143 2
.. 484 1144 2 IF .JRNL_EXTEND NEQ 0
.. 485 1145 2 THEN
.. 486 1146 2 BUILD_LIST (MNT$_JRNL_EXTEND, 4, JRNL_EXTEND, END_OF_LIST);
.. 487 1147 2
.. 488 1148 2 IF .JRNL_QUOTA NEQ 0
.. 489 1149 2 THEN
.. 490 1150 2 BUILD_LIST (MNT$_JRNL_QUOTA, 4, JRNL_QUOTA, END_OF_LIST);
.. 491 1151 2
.. 492 1152 2
.. 493 1153 2 ! Process the /OWNER_UIC qualifier
.. 494 1154 2
.. 495 1155 2 IF .MOUNT_OPTIONS [OPT_OWNER_UIC]
.. 496 1156 2 THEN
.. 497 1157 2 BUILD_LIST (MNT$_OWNER, 4, OWNER_UIC, END_OF_LIST);
.. 498 1158 2
.. 499 1159 2 ! Process the /PROCESSOR qualifier
.. 500 1160 2
.. 501 1161 2 IF .MOUNT_OPTIONS [OPT_UNIQUEACP]
.. 502 1162 2 OR .MOUNT_OPTIONS [OPT_SAMEACP]
.. 503 1163 2 OR .MOUNT_OPTIONS [OPT_FILEACP]
.. 504 1164 2 THEN
.. 505 1165 2 BUILD_LIST ( MNT$ PROCESSOR, .ACP_STRING [DSC$W_LENGTH],
.. 506 1166 2 .ACP_STRING [DSC$A_POINTER], END_OF_LIST);
.. 507 1167 2
.. 508 1168 2 ! Process the /PROTECTION qualifer
```

```
509 1169 2 !
510 1170 2 IF .MOUNT_OPTIONS [OPT_PROTECTION]
511 1171 2 THEN
512 1172 2 BUILD_LIST (MNTSV_VPROT, 4, PROTECTION, END_OF_LIST);
513 1173 2
514 1174 2 ! Process the /RECORDIZE qualifier
515 1175 2
516 1176 2 IF .MOUNT_OPTIONS [OPT_RECORDSZ]
517 1177 2 THEN
518 1178 2 BUILD_LIST ( MNTSV_RECORDSZ, 4, RECORDSZ, END_OF_LIST );
519 1179 2
520 1180 2 ! Process the /WINDOW qualifier
521 1181 2
522 1182 2 IF .MOUNT_OPTIONS [OPT_WINDOW]
523 1183 2 THEN
524 1184 2 BUILD_LIST ( MNTSV_WINDOW, 4, WINDOW, END_OF_LIST );
525 1185 2
526 1186 2
527 1187 2 ! Set the MOUNT flags according to their counterparts in MOUNT_OPTIONS
528 1188 2
529 1189 2
530 1190 2 MOUNT_FLAGS [MNTSV_CLUSTER] = .MOUNT_OPTIONS [OPT_CLUSTER];
531 1191 2 MOUNT_FLAGS [MNTSV_FOREIGN] = .MOUNT_OPTIONS [OPT_FOREIGN] OR .MOUNT_OPTIONS [OPT_NOLABEL];
532 1192 2 MOUNT_FLAGS [MNTSV_GROUP] = .MOUNT_OPTIONS [OPT_GROUP];
533 1193 2 MOUNT_FLAGS [MNTSV_INIT_ALL] = .MOUNT_OPTIONS [OPT_INIT_ALL];
534 1194 2 MOUNT_FLAGS [MNTSV_INIT_CONT] = .MOUNT_OPTIONS [OPT_INIT_CONT];
535 1195 2 MOUNT_FLAGS [MNTSV_MESSAGE] = .MOUNT_OPTIONS [OPT_MESSAGE];
536 1196 2 MOUNT_FLAGS [MNTSV_NEWJRN] = .MOUNT_OPTIONS [OPT_NEWJRN];
537 1197 2 MOUNT_FLAGS [MNTSV_NOASSIST] = NOT .MOUNT_OPTIONS [OPT_ASSIST];
538 1198 2 MOUNT_FLAGS [MNTSV_NOAUTO] = .MOUNT_OPTIONS [OPT_NOAUTO];
539 1199 2 MOUNT_FLAGS [MNTSV_NOCACHE] = .MOUNT_OPTIONS [OPT_NOCACHE];
540 1200 2 MOUNT_FLAGS [MNTSV_NODISKQ] = .MOUNT_OPTIONS [OPT_NOQUOTA];
541 1201 2 MOUNT_FLAGS [MNTSV_NOHDR3] = .MOUNT_OPTIONS [OPT_NOHDR3];
542 1202 2 MOUNT_FLAGS [MNTSV_NOJRN] = .MOUNT_OPTIONS [OPT_NOJRN];
543 1203 2 MOUNT_FLAGS [MNTSV_NOMNTVER] = NOT .MOUNT_OPTIONS [OPT_MOUNTVER];
544 1204 2 MOUNT_FLAGS [MNTSV_NOUNLOAD] = .MOUNT_OPTIONS [OPT_NOUNLOAD];
545 1205 2 MOUNT_FLAGS [MNTSV_NOWRITE] = NOT .MOUNT_OPTIONS [OPT_WRITE];
546 1206 2 MOUNT_FLAGS [MNTSV_OVR_ACCESS] = .MOUNT_OPTIONS [OPT_OVR_ACC];
547 1207 2 MOUNT_FLAGS [MNTSV_OVR_EXP] = .MOUNT_OPTIONS [OPT_OVR_EXP];
548 1208 2 MOUNT_FLAGS [MNTSV_OVR_IDENT] = .MOUNT_OPTIONS [OPT_OVR_ID];
549 1209 2 MOUNT_FLAGS [MNTSV_OVR_LOCK] = .MOUNT_OPTIONS [OPT_OVR_LOCK];
550 1210 2 MOUNT_FLAGS [MNTSV_OVR_SETID] = .MOUNT_OPTIONS [OPT_OVR_SETID];
551 1211 2 MOUNT_FLAGS [MNTSV_OVR_VOLO] = .MOUNT_OPTIONS [OPT_OVR_VOLO];
552 1212 2 MOUNT_FLAGS [MNTSV_READCHECK] = .MOUNT_OPTIONS [OPT_READCHECK];
553 1213 2 MOUNT_FLAGS [MNTSV_SHARE] = .MOUNT_OPTIONS [OPT_SHARE];
554 1214 2 MOUNT_FLAGS [MNTSV_SYSTEM] = .MOUNT_OPTIONS [OPT_SYSTEM];
555 1215 2 MOUNT_FLAGS [MNTSV_WRITECHECK] = .MOUNT_OPTIONS [OPT_WRITECHECK];
556 1216 2 MOUNT_FLAGS [MNTSV_WRITEHRU] = .MOUNT_OPTIONS [OPT_WRITEHRU];
557 1217 2 MOUNT_FLAGS [MNTSV_NOREBUILD] = .MOUNT_OPTIONS [OPT_NOREBUILD];
558 1218 2
559 1219 2 ! Build an item list entry for mount flags, then terminate the item list
560 1220 2 ! with a zero value.
561 1221 2
562 1222 2 BUILD_LIST ( MNTSV_FLAGS, 4, MOUNT_FLAGS, END_OF_LIST );
563 1223 2 .END_OF_LIST = 0;
564 1224 2
565 1225 2 ! Now that all the parameters have been parsed, call the $MOUNT system service.
```

```

: 566      1226 2 ! Note the informational messages may be issued from mount via a $PUTMSG and
: 567      1227 2 ! a status value from the call will be returned as well.
: 568      1228 2
: 569      1229 2 STATUS = $MOUNT (ITMLST = ITEM_LIST);           ! Mount the volume(s)
: 570      1230 2
: 571      1231 3 RETURN (.STATUS)                               ! Return status of $MOUNT call
: 572      1232 3
: 573      1233 1 END;                                           ! end of routine PARSE_COMMAND

```

```

.TITLE MOUNTIMG
.IDENT \V04-000\
.PSECT $SPLITS,NOWRT,NOEXE,2

```

```

      44 45 53 53 45 43 43 41 00000 P.AAB: .ASCII \ACCESSED\
      00000008 00008 P.AAA: .LONG 8
      00000000 0000C P.AAA: .ADDRESS P.AAB
      54 53 49 53 53 41 00010 P.AAD: .ASCII \ASSIST\
      00016 P.AAD: .BLKB 2
      00000006 00018 P.AAC: .LONG 6
      00000000 0001C P.AAC: .ADDRESS P.AAD
      43 49 54 41 4D 4F 54 55 41 00020 P.AAF: .ASCII \AUTOMATIC\
      00029 P.AAF: .BLKB 3
      00000009 0002C P.AAE: .LONG 9
      00000000 00030 P.AAE: .ADDRESS P.AAF
      44 4E 49 42 00034 P.AAH: .ASCII \BIND\
      00000004 00038 P.AAG: .LONG 4
      00000000 0003C P.AAG: .ADDRESS P.AAH
      45 5A 49 53 4B 43 4F 4C 42 00040 P.AAJ: .ASCII \BLOCKSIZE\
      00049 P.AAJ: .BLKB 3
      00000009 0004C P.AAJ: .LONG 9
      00000000 00050 P.AAJ: .ADDRESS P.AAJ
      45 4B 43 41 43 00054 P.AAL: .ASCII \CACHE\
      00059 P.AAL: .BLKB 3
      00000005 0005C P.AAK: .LONG 5
      00000000 00060 P.AAK: .ADDRESS P.AAL
      52 45 54 53 55 4C 43 00064 P.AAN: .ASCII \CLUSTER\
      0006B P.AAN: .BLKB 1
      00000007 0006C P.AAM: .LONG 7
      00000000 00070 P.AAM: .ADDRESS P.AAN
      54 4E 45 4D 4D 4F 43 00074 P.AAP: .ASCII \COMMENT\
      0007B P.AAP: .BLKB 1
      00000007 0007C P.AAO: .LONG 7
      00000000 00080 P.AAO: .ADDRESS P.AAP
      4B 43 45 4B 43 5F 41 54 41 44 00084 P.AAR: .ASCII \DATA_CHECK\
      0008E P.AAR: .BLKB 2
      0000000A 00090 P.AAQ: .LONG 10
      00000000 00094 P.AAQ: .ADDRESS P.AAR
      59 54 49 53 4E 45 44 00098 P.AAT: .ASCII \DENSITY\
      0009F P.AAT: .BLKB 1
      00000007 000A0 P.AAS: .LONG 7
      00000000 000A4 P.AAS: .ADDRESS P.AAT
      4E 4F 49 53 4E 45 54 5B 45 000A8 P.AAV: .ASCII \EXTENSION\
      000B1 P.AAV: .BLKB 3
      00000009 000B4 P.AAU: .LONG 9
      00000000 000B8 P.AAU: .ADDRESS P.AAV

```


4E	47	49	45	52	4F	46	000BC	P.AAX:	.ASCII	\FOREIGN\	:								
							000C3		.BLKB	1	:								
						00000007	000C4	P.AAW:	.LONG	7	:								
						00000000	000C8		.ADDRESS	P.AAX	:								
		50	55	4F	52	47	000CC	P.AAZ:	.ASCII	\GROUP\	:								
							000D1		.BLKB	3	:								
						00000005	000D4	P.AAY:	.LONG	5	:								
						00000000	000D8		.ADDRESS	P.AAZ	:								
			33	52	44	48	000DC	P.ABB:	.ASCII	\HDR3\	:								
						00000004	000E0	P.ABA:	.LONG	4	:								
						00000000	000E4		.ADDRESS	P.ABB	:								
45	5A	49	4C	41	49	54	49	4E	49	000E8	P.ABD:	.ASCII	\INITIALIZE\	:					
										000F2		.BLKB	2	:					
						0000000A	000F4	P.ABC:	.LONG	10	:								
						00000000	000F8		.ADDRESS	P.ABD	:								
		4C	41	4E	52	55	4F	4A		000FC	P.ABF:	.ASCII	\JOURNAL\	:					
										00103		.BLKB	1	:					
						00000007	00104	P.ABE:	.LONG	7	:								
						00000000	00108		.ADDRESS	P.ABF	:								
			4C	45	42	41	4C			0010C	P.ABH:	.ASCII	\LABEL\	:					
										00111		.BLKB	3	:					
						00000005	00114	P.ABG:	.LONG	5	:								
						00000000	00118		.ADDRESS	P.ABH	:								
		45	47	41	53	53	45	4D		0011C	P.ABJ:	.ASCII	\MESSAGE\	:					
										00123		.BLKB	1	:					
						00000007	00124	P.ABI:	.LONG	7	:								
						00000000	00128		.ADDRESS	P.ABJ	:								
54	41	43	49	46	49	52	45	56	5F	54	4E	55	4F	4D	0012C	P.ABL:	.ASCII	\MOUNT_VERIFICATION\	:
												4E	4F	45	0013B				:
															0013E				:
						00000012	00140	P.ABK:	.BLKB	2	:								
						00000000	00144		.LONG	18	:								
									.ADDRESS	P.ABL	:								
		4C	45	42	41	4C	4F	4E		00148	P.ABN:	.ASCII	\NOLABEL\	:					
										0014F		.BLKB	1	:					
						00000007	00150	P.ABM:	.LONG	7	:								
						00000000	00154		.ADDRESS	P.ABN	:								
		45	44	49	52	52	45	56	4F	00158	P.ABP:	.ASCII	\OVERRIDE\	:					
										00160	P.ABO:	.LONG	8	:					
						00000008	00164		.ADDRESS	P.ABP	:								
		43	49	55	5F	52	45	4E	57	4F	00168	P.ABR:	.ASCII	\OWNER_UIC\	:				
											00171		.BLKB	3	:				
						00000009	00174	P.ABQ:	.LONG	9	:								
						00000000	00178		.ADDRESS	P.ABR	:								
		52	4F	53	53	45	43	4F	52	50	0017C	P.ABT:	.ASCII	\PROCESSOR\	:				
											00185		.BLKB	3	:				
						00000009	00188	P.ABS:	.LONG	9	:								
						00000000	0018C		.ADDRESS	P.ABT	:								
4E	4F	49	54	43	45	54	4F	52	50		00190	P.ABV:	.ASCII	\PROTECTION\	:				
											0019A		.BLKB	2	:				
						0000000A	0019C	P.ABU:	.LONG	10	:								
						00000000	001A0		.ADDRESS	P.ABV	:								
				41	54	4F	55	51		001A4	P.ABX:	.ASCII	\QUOTA\	:					
										001A9		.BLKB	3	:					
						00000005	001AC	P.ABW:	.LONG	5	:								
						00000000	001B0		.ADDRESS	P.ABX	:								
		44	4C	49	55	42	45	52		001B4	P.ABZ:	.ASCII	\REBUILD\	:					
										001BB		.BLKB	1	:					

L 5
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31VAX-11 Bliss-32 V4.0-742
[MOUNT.SRC]MOUNTIMG.B32;1Page 14
(3)

								00000007	001BC	P.ABY:	.LONG	7		
								00000000	001C0		.ADDRESS	P.ABZ		
45	5A	49	53	44	52	4F	43	45	52	001C4	P.ACB	.ASCII	\RECORDSIZE\	
										001CE		.BLKB	2	
								0000000A	001D0	P.ACA:	.LONG	10		
								00000000	001D4		.ADDRESS	P.ACB		
					45	52	41	48	53	001D8	P.ACD:	.ASCII	\SHARE\	
										001DD		.BLKB	3	
								00000005	001E0	P.ACC:	.LONG	5		
								00000000	001E4		.ADDRESS	P.ACD		
				4D	45	54	53	59	53	001E8	P.ACF:	.ASCII	\SYSTEM\	
										001EE		.BLKB	2	
								00000006	001F0	P.ACE:	.LONG	6		
								00000000	001F4		.ADDRESS	P.ACF		
				44	41	4F	4C	4E	55	001F8	P.ACH:	.ASCII	\UNLOAD\	
										001FE		.BLKB	2	
								00000006	00200	P.ACG:	.LONG	6		
								00000000	00204		.ADDRESS	P.ACH		
				53	57	4F	44	4E	49	57	00208	P.ACJ:	.ASCII	\WINDOWS\
										0020F		.BLKB	1	
								00000007	00210	P.ACI:	.LONG	7		
								00000000	00214		.ADDRESS	P.ACJ		
										00218	P.ACL:	.ASCII	\WRITE\	
										0021D		.BLKB	3	
								00000005	00220	P.ACK:	.LONG	5		
								00000000	00224		.ADDRESS	P.ACL		

.PSECT \$OWNS,NOEXE,2

00000	DEVICE_COUNT:	.BLKB	4
00004	LABEL_COUNT:	.BLKB	4
00008	DEVICE_STRING:	.BLKB	128
00088	LABEL_STRING:	.BLKB	128
00108	LOG_NAME:	.BLKB	8
00110	MOUNT_OPTIONS:	.BLKB	8
00118	MOUNT_FLAGS:	.BLKB	4
0011C	ACCESS:	.BLKB	4
00120	ACP_STRING:	.BLKB	8
00128	BLOCKSZ:	.BLKB	4
0012C	EXT_CACHE:	.BLKB	4
00130	FID_CACHE:	.BLKB	4
00134	QUO_CACHE:	.BLKB	4
00138	COMMENT_STRING:	.BLKB	8
00140	DENSITY:	.BLKB	4
00144	EXTENSION:		

		00148	JRNL_QUOTA:	.BLKB	4
		0014C	JRNL_EXTEND:	.BLKB	4
		00150	JRNL_SIZE:	.BLKB	4
		00154	JRNL_RECORD_SIZE:	.BLKB	4
		00158	OWNER_UIC:	.BLKB	4
		0015C	PROTECTION:	.BLKB	4
		00160	RECORDSZ:	.BLKB	4
		00164	STRUCT_NAME:	.BLKB	8
		0016C	WINDOW:	.BLKB	4
		00170	CLI_DESC:	.BLKB	8
	FFFFFFF	00178	EXT_LIMIT:	.LONG	-1
00000003	00000008	0017C	TPARSE_BLOCK:	.LONG	8 3
		00184		.BLKB	28
		001A0	UIC:	.BLKB	4
		001A4	ZERO:	.BLKB	4

ACCESSED_DESC=	P.AAA
ASSIST_DESC=	P.AAC
AUTOMATIC_DESC=	P.AAE
BIND_DESC=	P.AAG
BLOCK_DESC=	P.AAI
CACHE_DESC=	P.AAK
CLUSTER_DESC=	P.AAM
COMMENT_DESC=	P.AAO
DATA_DESC=	P.AAQ
DENSITY_DESC=	P.AAS
EXTENSION_DESC=	P.AAU
FOREIGN_DESC=	P.AAW
GROUP_DESC=	P.AAY
HDR3_DESC=	P.ABA
INITIALIZE_DESC=	P.ABC
JOURNAL_DESC=	P.ABE
LABEL_DESC=	P.ABG
MESSAGE_DESC=	P.ABI
MOUNT_VER_DESC=	P.ABK
NOLABEL_DESC=	P.ABM
OVERRIDE_DESC=	P.ABO
OWNER_DESC=	P.ABQ
PROCESSOR_DESC=	P.ABS
PROTECTION_DESC=	P.ABU
QUOTA_DESC=	P.ABW
REBUILD_DESC=	P.ABY
RECORD_DESC=	P.ACA
SHARE_DESC=	P.ACC
SYSTEM_DESC=	P.ACE

:
:

UNLOAD_DESC= P.ACG
WINDOW_DESC= P.ACI
WRITE_DESC= P.ACK
.EXTRN LIB\$CVT_DTB, STR\$COPY_DX
.EXTRN CLIS\$GET-VALUE, CLIS\$PRESENT
.EXTRN CLIS\$ABSENT, CLIS\$DEFAULTED
.EXTRN CLIS\$NEGATED, CLIS\$PRESENT
.EXTRN SYSS\$MOUNT

.PSECT \$CODE\$,NOWRT,2

				00FC 00000	.ENTRY	PARSE_COMMAND, Save R2,R3,R4,R5,R6,R7	0941
	57	0000V	CF	9E 00002	MOVAB	BUILD_LIST, R7	
	56	0000V	CF	9E 00007	MOVAB	MOUNT_FLAGS, R6	
	5E	FDA4	CE	9E 0000C	MOVAB	-604(SP), SP	
	6D	0391	CF	DE 00011	MOVAL	30\$, (FP)	0974
			5E	DD 00016	PUSHL	SP	0989
		008C	C6	D4 00018	CLRL	ZERO	0993
		F8	A6	7C 0001C	CLRQ	MOUNT_OPTIONS	0994
	FE	A6	08	88 0001F	BISB2	#8, MOUNT_OPTIONS+6	0995
	F8	A6	1010	8F A8 00023	BISW2	#4112, MOUNT_OPTIONS	0997
	FD	A6	14	88 00029	BISB2	#20, MOUNT_OPTIONS+5	0999
	F9	A6	04	88 0002D	BISB2	#4, MOUNT_OPTIONS+1	1000
08	00	6E	00	2C 00031	MOVCS	#0, (SP), #0, #8, CLI_DESC	1004
		58	A6	00036			
	5B	A6	02	90 00038	MOVAB	#2, CLI_DESC+3	1005
	0000V	CF	00	FB 0003C	CALLS	#0, PARSE_QUALIFIER	1007
	0000V	CF	00	FB 00041	CALLS	#0, GET_DEVICE	1011
	0000V	CF	00	FB 00046	CALLS	#0, GET_LABEL	1015
	0000V	CF	00	FB 0004B	CALLS	#0, GET_LOG_NAME	1019
		FB	A6	95 00050	TSTB	MOUNT_OPTIONS+3	1024
			0B	19 00053	BLSS	1\$	
	FF74	C6	FF70	C6 D4 00055	CLRL	LABEL_STRING	1027
		53	FF74	C6 9E 00059	MOVAB	LABEL_STRING+4, LABEL_STRING+4	1028
		52	FEEB	C6 D0 00060	MOVL	DEVICE_COUNT, R3	1033
			01	CE 00065	MNEGL	#1, J	
			19	11 00068	BRB	3\$	
			5E	DD 0006A	PUSHL	SP	1035
50		52	01	78 0006C	ASHL	#1, J, R0	1037
			FEF4	C640 DD 00070	PUSHL	DEVICE_STRING+4[R0]	
50		52	01	78 00075	ASHL	#1, J, R0	1036
			FEF0	C640 DD 00079	PUSHL	DEVICE_STRING[R0]	
			01	DD 0007E	PUSHL	#1	1035
		67	04	FB 00080	CALLS	#4, BUILD_LIST	
E3		52	53	F2 00083	AOBLSS	R3, J, 2\$	
		53	FEEC	C6 D0 00087	MOVL	LABEL_COUNT, R3	1042
		52	01	CE 0008C	MNEGL	#1, J	
			19	11 0008F	BRB	5\$	
			5E	DD 00091	PUSHL	SP	1044
50		52	01	78 00093	ASHL	#1, J, R0	1046
			FF74	C640 DD 00097	PUSHL	LABEL_STRING+4[R0]	
50		52	01	78 0009C	ASHL	#1, J, R0	1045
			FF70	C640 DD 000A0	PUSHL	LABEL_STRING[R0]	
			02	DD 000A5	PUSHL	#2	1044
		67	04	FB 000A7	CALLS	#4, BUILD_LIST	
E3		52	53	F2 000AA	AOBLSS	R3, J, 4\$	
OE	FB	A6	05	E1 000AE	BBC	#5, MOUNT_OPTIONS+3, 6\$	1054

			SE	DD	000B3	PUSHL	SP	1056
		F4	A6	DD	000B5	PUSHL	LOG_NAME+4	1058
	7E	FO	A6	3C	000B8	MOVZWL	LOG_NAME, -(SP)	1057
			03	DD	000BC	PUSHL	#3	1056
	67		04	FB	000BE	CALLS	#4, BUILD_LIST	
OC	FB	A6	01	E1	000C1	BBC	#1, MOUNT_OPTIONS+3, 7\$	1063
			SE	DD	000C6	PUSHL	SP	1065
		04	A6	9F	000C8	PUSHAB	ACCESS	
			04	DD	000CB	PUSHL	#4	
			05	DD	000CD	PUSHL	#5	
	67		04	FB	000CF	CALLS	#4, BUILD_LIST	
	OE	FD	A6	E9	000D2	BLBC	MOUNT_OPTIONS+5, 8\$	1069
			SE	DD	000D6	PUSHL	SP	1071
		50	A6	DD	000D8	PUSHL	STRUCT_NAME+4	1072
	7E	4C	A6	3C	000DB	MOVZWL	STRUCT_NAME, -(SP)	1071
			07	DD	000DF	PUSHL	#7	
	67		04	FB	000E1	CALLS	#4, BUILD_LIST	
	OC	FA	A6	E9	000E4	BLBC	MOUNT_OPTIONS+2, 9\$	1076
			SE	DD	000E8	PUSHL	SP	1078
		10	A6	9F	000EA	PUSHAB	BLOCKSZ	
			04	DD	000ED	PUSHL	#4	
			08	DD	000EF	PUSHL	#8	
	67		04	FB	000F1	CALLS	#4, BUILD_LIST	
		14	A6	D5	000F4	TSTL	EXT_CACHE	1082
			OC	15	000F7	BLEQ	10\$	
			SE	DD	000F9	PUSHL	SP	1085
		14	A6	9F	000FB	PUSHAB	EXT_CACHE	
			04	DD	000FE	PUSHL	#4	
			0A	DD	00100	PUSHL	#10	
	67		04	FB	00102	CALLS	#4, BUILD_LIST	
		FD	A6	95	00105	TSTB	MOUNT_OPTIONS+5	1087
			OD	18	00108	BGEQ	11\$	
			SE	DD	0010A	PUSHL	SP	1089
		008C	C6	9F	0010C	PUSHAB	ZERO	
			04	DD	00110	PUSHL	#4	
			0A	DD	00112	PUSHL	#10	
	67		04	FB	00114	CALLS	#4, BUILD_LIST	
	04	FE	A6	E9	00117	BLBC	MOUNT_OPTIONS+6, 12\$	1093
18	A6		01	D0	0011B	MOVL	#1, FID_CACHE	1095
		18	A6	D5	0011F	TSTL	FID_CACHE	1096
			OC	15	00122	BLEQ	13\$	
			SE	DD	00124	PUSHL	SP	1098
		18	A6	9F	00126	PUSHAB	FID_CACHE	
			04	DD	00129	PUSHL	#4	
			0B	DD	0012B	PUSHL	#11	
	67		04	FB	0012D	CALLS	#4, BUILD_LIST	
		60	A6	D5	00130	TSTL	EXT_LIMIT	1102
			OC	19	00133	BLSS	14\$	
			SE	DD	00135	PUSHL	SP	1104
		60	A6	9F	00137	PUSHAB	EXT_LIMIT	
			04	DD	0013A	PUSHL	#4	
			OC	DD	0013C	PUSHL	#12	
			04	FB	0013E	CALLS	#4, BUILD_LIST	
OD	FE	67	01	E1	00141	BBC	#1, MOUNT_OPTIONS+6, 15\$	1108
	A6		SE	DD	00146	PUSHL	SP	1110
		008C	C6	9F	00148	PUSHAB	ZERO	
			04	DD	0014C	PUSHL	#4	

			OF DD 0014E	PUSHL #15	
	67		04 FB 00150	CALLS #4, BUILD_LIST	
		1C	A6 D5 00153 15\$:	TSTL QUO_CACHE	1111
			OC 15 00156	BLEQ 16\$	
			SE DD 00158	PUSHL SP	1113
		1C	A6 9F 0015A	PUSHAB QUO_CACHE	
			04 DD 0015D	PUSHL #4	
			OF DD 0015F	PUSHL #15	
	67		04 FB 00161	CALLS #4, BUILD_LIST	
OE	F8	A6	03 E1 00164 16\$:	BBC #3, MOUNT_OPTIONS, 17\$	1117
			SE DD 00169	PUSHL SP	1119
	24		A6 DD 0016B	PUSHL COMMENT_STRING+4	1120
	7E	20	A6 3C 0016E	MOVZWL COMMENT_STRING, -(SP)	1119
			14 DD 00172	PUSHL #20	
	67		04 FB 00174	CALLS #4, BUILD_LIST	
	OC	F8	A6 E9 00177 17\$:	BLBC MOUNT_OPTIONS, 18\$	1124
			SE DD 0017B	PUSHL SP	1126
		28	A6 9F 0017D	PUSHAB DENSITY	
			04 DD 00180	PUSHL #4	
			09 DD 00182	PUSHL #9	
	67		04 FB 00184	CALLS #4, BUILD_LIST	
		FA	A6 95 00187 18\$:	TSTB MOUNT_OPTIONS+2	1130
			OC 18 0018A	BGEQ 19\$	
			SE DD 0018C	PUSHL SP	1132
		2C	A6 9F 0018E	PUSHAB EXTENSION	
			04 DD 00191	PUSHL #4	
			12 DD 00193	PUSHL #18	
	67		04 FB 00195	CALLS #4, BUILD_LIST	
		38	A6 D5 00198 19\$:	TSTL JRNL_SIZE	1136
			OC 13 0019B	BEQL 20\$	
			SE DD 0019D	PUSHL SP	1138
		38	A6 9F 0019F	PUSHAB JRNL_SIZE	
			04 DD 001A2	PUSHL #4	
			15 DD 001A4	PUSHL #21	
	67		04 FB 001A6	CALLS #4, BUILD_LIST	
		3C	A6 D5 001A9 20\$:	TSTL JRNL_RECORD_SIZE	1140
			OC 13 001AC	BEQL 21\$	
			SE DD 001AE	PUSHL SP	1142
		3C	A6 9F 001B0	PUSHAB JRNL_RECORD_SIZE	
			04 DD 001B3	PUSHL #4	
			18 DD 001B5	PUSHL #24	
	67		04 FB 001B7	CALLS #4, BUILD_LIST	
		34	A6 D5 001BA 21\$:	TSTL JRNL_EXTEND	1144
			OC 13 001BD	BEQL 22\$	
			SE DD 001BF	PUSHL SP	1146
		34	A6 9F 001C1	PUSHAB JRNL_EXTEND	
			04 DD 001C4	PUSHL #4	
			16 DD 001C6	PUSHL #22	
	67		04 FB 001C8	CALLS #4, BUILD_LIST	
		30	A6 D5 001CB 22\$:	TSTL JRNL_QUOTA	1148
			OC 13 001CE	BEQL 23\$	
			SE DD 001D0	PUSHL SP	1150
		30	A6 9F 001D2	PUSHAB JRNL_QUOTA	
			04 DD 001D5	PUSHL #4	
			17 DD 001D7	PUSHL #23	
			04 FB 001D9	CALLS #4, BUILD_LIST	
OC	FA	67	02 E1 001DC 23\$:	BBC #2, MOUNT_OPTIONS+2, 24\$	1155
	A6				

					40	5E DD 001E1	PUSHL SP	1157
						A6 9F 001E3	PUSHAB OWNER_UIC	
						04 DD 001E6	PUSHL #4	
						0D DD 001E8	PUSHL #13	
						04 FB 001EA	CALLS #4, BUILD_LIST	
						02 E0 001ED	BBS #2, MOUNT_OPTIONS+3, 25\$	1161
						03 E0 001F2	BBS #3, MOUNT_OPTIONS+3, 25\$	1162
						04 E1 001F7	BBC #4, MOUNT_OPTIONS+3, 26\$	1163
						5E DD 001FC	PUSHL SP	1165
						A6 DD 001FE	PUSHL ACP_STRING+4	1166
						08 A6 3C 00201	MOVZWL ACP_STRING, -(SP)	1165
						06 DD 00205	PUSHL #6	
						04 FB 00207	CALLS #4, BUILD_LIST	
						01 E1 0020A	BBC #1, MOUNT_OPTIONS+2, 27\$	1170
						5E DD 0020F	PUSHL SP	1172
						44 A6 9F 00211	PUSHAB PROTECTION	
						04 DD 00214	PUSHL #4	
						0E DD 00216	PUSHL #14	
						04 FB 00218	CALLS #4, BUILD_LIST	
						05 E1 0021B	BBC #5, MOUNT_OPTIONS+4, 28\$	1176
						5E DD 00220	PUSHL SP	1178
						48 A6 9F 00222	PUSHAB RECORDSZ	
						04 DD 00225	PUSHL #4	
						10 DD 00227	PUSHL #16	
						04 FB 00229	CALLS #4, BUILD_LIST	
						FB A6 E9 0022C	BLBC MOUNT_OPTIONS+3, 29\$	1182
						5E DD 00230	PUSHL SP	1184
						54 A6 9F 00232	PUSHAB WINDOW	
						04 DD 00235	PUSHL #4	
						11 DD 00237	PUSHL #17	
						04 FB 00239	CALLS #4, BUILD_LIST	
						06 EF 0023C	EXTZV #6, #1, MOUNT_OPTIONS+7, R0	1190
						50 FO 00242	INSV R0, #4, #1, MOUNT_FLAGS+3	
						03 EF 00248	EXTZV #3, #1, MOUNT_OPTIONS+1, R0	1191
						04 EF 0024E	EXTZV #4, #1, MOUNT_OPTIONS+1, R1	
						51 8E 00254	BISB2 R1, R0	
						50 FO 00257	INSV R0, #0, #1, MOUNT_FLAGS	
						07 EF 0025C	EXTZV #7, #1, MOUNT_OPTIONS, R0	1192
						50 FO 00262	INSV R0, #1, #1, MOUNT_FLAGS	
						02 EF 00267	EXTZV #2, #1, MOUNT_OPTIONS+7, R0	1193
						50 FO 0026D	INSV R0, #0, #1, MOUNT_FLAGS+3	
						03 EF 00273	EXTZV #3, #1, MOUNT_OPTIONS+7, R0	1194
						50 FO 00279	INSV R0, #1, #1, MOUNT_FLAGS+3	
						03 EF 0027F	EXTZV #3, #1, MOUNT_OPTIONS+6, R0	1195
						50 FO 00285	INSV R0, #5, #1, MOUNT_FLAGS+1	
						FF A6 FO 00288	INSV MOUNT_OPTIONS+7, #6, #1, MOUNT_FLAGS+2	1196
						02 EF 00292	EXTZV #2, #1, MOUNT_OPTIONS+6, R0	1197
						50 D2 00298	MCOML R0, R0	
						50 FO 0029B	INSV R0, #2, #1, MOUNT_FLAGS	
						01 EF 002A0	EXTZV #1, #1, MOUNT_OPTIONS+7, R0	1198
						50 FO 002A6	INSV R0, #7, #1, MOUNT_FLAGS+2	
						04 EF 002AC	EXTZV #4, #1, MOUNT_OPTIONS+6, R0	1199
						50 FO 002B2	INSV R0, #1, #1, MOUNT_FLAGS+2	
						02 EF 002B8	EXTZV #2, #1, MOUNT_OPTIONS+5, R0	1200
						50 FO 002BE	INSV R0, #3, #1, MOUNT_FLAGS	
						04 EF 002C3	EXTZV #4, #1, MOUNT_OPTIONS+5, R0	1201
						50 FO 002C9	INSV R0, #4, #1, MOUNT_FLAGS	

03	50	FF	A6	67	01			
	A6		01	04				
	50	F9	A6	01				
	51	F9	A6	01				
				50				
	66		01	00				
	50	F8	A6	01				
	66		01	01				
03	50	FF	A6	01	00			
	A6		01	01				
03	50	FF	A6	01	01			
	A6		01	01				
01	50	FE	A6	01	01			
	A6		01	05				
02	50		01	06				
		FE	A6	01				
	66		01	50				
	50		01	02				
02	50	FF	A6	01				
	A6		01	07				
	50	FE	A6	01				
02	50		01	01				
	A6	FD	A6	01				
	66		01	03				
	50	FD	A6	01				
	66		01	04				

MOUNTIMG
V04-000

E 6
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31

VAX-11 Bliss-32 V4.0-742
[MOUNT.SRC]MOUNTIMG.B32;1

Page 20
(3)

02	50	FE	A6	01	07	EF	002CE	EXTZV	#7, #1, MOUNT_OPTIONS+6, R0	1202
	A6		01	05	50	F0	002D4	INSV	R0, #5, #1, MOUNT_FLAGS+2	
	50	FE	A6	C1	06	EF	002DA	EXTZV	#6, #1, MOUNT_OPTIONS+6, R0	1203
02	A6		01	50	50	D2	002E0	MCOML	R0, R0	
	50	F9	A6	03	50	F0	002E3	INSV	R0, #3, #1, MOUNT_FLAGS+2	
02	A6		01	01	02	EF	002E9	EXTZV	#2, #1, MOUNT_OPTIONS+1, R0	1204
	50	F9	A6	04	50	F0	002EF	INSV	R0, #4, #1, MOUNT_FLAGS+2	
				01	01	EF	002F5	EXTZV	#1, #1, MOUNT_OPTIONS+1, R0	1205
	66		01	50	50	D2	002FB	MCOML	R0, R0	
	50	FC	A6	06	50	F0	002FE	INSV	R0, #6, #1, MOUNT_FLAGS	
	66		01	01	06	EF	00303	EXTZV	#6, #1, MOUNT_OPTIONS+4, R0	1206
	50	FA	A6	07	50	F0	00309	INSV	R0, #7, #1, MOUNT_FLAGS	
01	A6		01	01	04	EF	0030E	EXTZV	#4, #1, MOUNT_OPTIONS+2, R0	1207
	50	FA	A6	00	50	F0	00314	INSV	R0, #0, #1, MOUNT_FLAGS+1	
01	A6		01	01	06	EF	0031A	EXTZV	#6, #1, MOUNT_OPTIONS+2, R0	1208
	50	FE	A6	01	50	F0	00320	INSV	R0, #1, #1, MOUNT_FLAGS+1	
02	A6		01	01	05	EF	00326	EXTZV	#5, #1, MOUNT_OPTIONS+6, R0	1209
	50	FA	A6	02	50	F0	0032C	INSV	R0, #2, #1, MOUNT_FLAGS+2	
01	A6		01	01	05	EF	00332	EXTZV	#5, #1, MOUNT_OPTIONS+2, R0	1210
	50	FF	A6	02	50	F0	00338	INSV	R0, #2, #1, MOUNT_FLAGS+1	
03	A6		01	01	04	EF	0033E	EXTZV	#4, #1, MOUNT_OPTIONS+7, R0	1211
	50	FC	A6	02	50	F0	00344	INSV	R0, #2, #1, MOUNT_FLAGS+3	
01	A6		01	01	03	EF	0034A	EXTZV	#3, #1, MOUNT_OPTIONS+4, R0	1212
	50	F8	A6	03	50	F0	00350	INSV	R0, #3, #1, MOUNT_FLAGS+1	
01	A6		01	01	06	EF	00356	EXTZV	#6, #1, MOUNT_OPTIONS, R0	1213
01	A6		01	04	50	F0	0035C	INSV	R0, #4, #1, MOUNT_FLAGS+1	
	50	FC	A6	06	A6	F0	00362	INSV	MOUNT_OPTIONS+1, #6, #1, MOUNT_FLAGS+1	1214
01	A6		01	01	04	EF	00369	EXTZV	#4, #1, MOUNT_OPTIONS+4, R0	1215
	50	FD	A6	07	50	F0	0036F	INSV	R0, #7, #1, MOUNT_FLAGS+1	
02	A6		01	01	05	EF	00375	EXTZV	#6, #1, MOUNT_OPTIONS+5, R0	1216
	50	FF	A6	00	50	F0	0037B	INSV	R0, #0, #1, MOUNT_FLAGS+2	
03	A6		01	01	07	EF	00381	EXTZV	#7, #1, MOUNT_OPTIONS+7, R0	1217
				05	50	F0	00387	INSV	R0, #5, #1, MOUNT_FLAGS+3	
				4040	8F	BB	0038D	PUSHR	#^M<R6,SP>	1222
					04	DD	00391	PUSHL	#4	
				67	04	DD	00393	PUSHL	#4	
					04	FB	00395	CALLS	#4, BUILD LIST	
				00	BF	D4	00398	CLRL	@END OF LIST	1223
				04	AE	9F	0039B	PUSHAB	ITEM LIST	1229
	00000000G		00	01	FB	0039E	CALLS	#1, SYSSMOUNT		
					04	003A5	RET			1233
					0000	003A6	30\$:	.WORD	Save nothing	0974
					7E	D4	003A8	CLRL	-(SP)	
					5E	DD	003AA	PUSHL	SP	
	0000V		7E	04	AC	7D	003AC	MOVQ	4(AP), -(SP)	
			CF		03	FB	003B0	CALLS	#3, MAIN_HANDLER	
					04	003B5	RET			

; Routine Size: 950 bytes, Routine Base: \$CODE\$ + 0000


```
: 575 1234 1 ROUTINE PARSE_QUALIFIER : NOVALUE =
: 576 1235 1
: 577 1236 1 ++
: 578 1237 1
: 579 1238 1 FUNCTIONAL DESCRIPTION:
: 580 1239 1
: 581 1240 1 This routine parses the qualifiers of the MOUNT command line by
: 582 1241 1 calling the CLI result parse routines.
: 583 1242 1
: 584 1243 1 CALLING SEQUENCE:
: 585 1244 1 PARSE_QUALIFIER ( )
: 586 1245 1
: 587 1246 1 INPUT PARAMETERS:
: 588 1247 1 NONE
: 589 1248 1
: 590 1249 1 IMPLICIT INPUTS:
: 591 1250 1 NONE
: 592 1251 1
: 593 1252 1 OUTPUT PARAMETERS:
: 594 1253 1 NONE
: 595 1254 1
: 596 1255 1 IMPLICIT OUTPUTS:
: 597 1256 1 MOUNT_OPTIONS BITS SET
: 598 1257 1
: 599 1258 1 ROUTINE VALUE:
: 600 1259 1 NONE
: 601 1260 1
: 602 1261 1 SIDE EFFECTS:
: 603 1262 1 NONE
: 604 1263 1
: 605 1264 1 --
: 606 1265 1
: 607 1266 2 BEGIN
: 608 1267 2
: 609 1268 2
: 610 1269 2 ! First, parse the qualifiers that do not have values, and cannot be negated.
: 611 1270 2
: 612 1271 2 ! /FOREIGN qualifier
: 613 1272 2
: 614 1273 2
: 615 1274 2 IF CLISPRESNT ( FOREIGN_DESC )
: 616 1275 2 THEN
: 617 1276 2 MOUNT_OPTIONS [OPT_FOREIGN] = 1
: 618 1277 2 ELSE
: 619 1278 2 MOUNT_OPTIONS [OPT_FOREIGN] = 0;
: 620 1279 2
: 621 1280 2
: 622 1281 2 ! /LABEL qualifier
: 623 1282 2
: 624 1283 2 IF CLISPRESNT ( LABEL_DESC )
: 625 1284 2 THEN
: 626 1285 2 BEGIN
: 627 1286 2 MOUNT_OPTIONS [OPT_LABEL] = 1;
: 628 1287 2 MOUNT_OPTIONS [OPT_NOLABEL] = 0;
: 629 1288 2 END;
: 630 1289 2
: 631 1290 2 ! /NOLABEL qualifier
```

```

: 632      1291 2  !
: 633      1292 2  IF CLISPRESNT ( NOLABEL_DESC )
: 634      1293 2  THEN
: 635      1294 2  BEGIN
: 636      1295 2  JUNTOPTIONS [OPT_NOLABEL] = 1;
: 637      1296 2  JUNTOPTIONS [OPT_LABEL] = 0;
: 638      1297 2  END;
: 639      1298 2
: 640      1299 2

```



```
642 1300 2
643 1301 2 ! Now, parse those qualifiers that do not require a value, and can be
644 1302 2 ! negated
645 1303 2
646 1304 2 ! /ASSIST qualifier
647 1305 2
648 1306 2 SELECTONE CLISP$PRESENT ( ASSIST_DESC ) OF
649 1307 2 SET
650 1308 2     [CLISP$PRESENT,
651 1309 2     CLISP$DEFAULTED] : MOUNT_OPTIONS [OPT_ASSIST] = 1;
652 1310 2     [CLISP$NEGATED] : MOUNT_OPTIONS [OPT_ASSIST] = 0;
653 1311 2 TES;
654 1312 2
655 1313 2 ! /AUTOMATIC qualifier
656 1314 2
657 1315 2 SELECTONE CLISP$PRESENT ( AUTOMATIC_DESC ) OF
658 1316 2 SET
659 1317 2     [CLISP$PRESENT,
660 1318 2     CLISP$DEFAULTED] : MOUNT_OPTIONS [OPT_NOAUTO] = 0;
661 1319 2     [CLISP$NEGATED] : MOUNT_OPTIONS [OPT_NOAUTO] = 1;
662 1320 2 TES;
663 1321 2
664 1322 2 ! /CLUSTER qualifier (default is /NOCLUSTER)
665 1323 2
666 1324 2 SELECTONE CLISP$PRESENT ( CLUSTER_DESC ) OF
667 1325 2 SET
668 1326 2     [CLISP$PRESENT] : MOUNT_OPTIONS [OPT_CLUSTER] = 1;
669 1327 2     [CLISP$DEFAULTED,
670 1328 2     CLISP$ABSENT,
671 1329 2     CLISP$NEGATED] : MOUNT_OPTIONS [OPT_CLUSTER] = 0;
672 1330 2 TES;
673 1331 2
674 1332 2
675 1333 2 ! /GROUP qualifier
676 1334 2
677 1335 2 SELECTONE CLISP$PRESENT ( GROUP_DESC ) OF
678 1336 2 SET
679 1337 2     [CLISP$PRESENT] : MOUNT_OPTIONS [OPT_GROUP] = 1;
680 1338 2     [CLISP$DEFAULTED,
681 1339 2     CLISP$ABSENT,
682 1340 2     CLISP$NEGATED] : MOUNT_OPTIONS [OPT_GROUP] = 0;
683 1341 2 TES;
684 1342 2
685 1343 2
686 1344 2 ! /HDR3 qualifier
687 1345 2
688 1346 2 SELECTONE CLISP$PRESENT ( HDR3_DESC ) OF
689 1347 2 SET
690 1348 2     [CLISP$PRESENT,
691 1349 2     CLISP$DEFAULTED] : MOUNT_OPTIONS [OPT_NOHDR3] = 0;
692 1350 2     [CLISP$NEGATED] : MOUNT_OPTIONS [OPT_NOHDR3] = 1;
693 1351 2 TES;
694 1352 2
695 1353 2 ! /MESSAGE qualifier
696 1354 2
697 1355 2 SELECTONE CLISP$PRESENT ( MESSAGE_DESC ) OF
698 1356 2
```

```

699 1357 2 SET
700 1358 2 [CLIS_PRESENT,
701 1359 2 [CLIS_DEFAULTED] : MOUNT_OPTIONS [OPT_MESSAGE] = 1;
702 1360 2 [CLIS_NEGATED] : MOUNT_OPTIONS [OPT_MESSAGE] = 0;
703 1361 2 TES;
704 1362 2
705 1363 2 ! /MOUNT_VERIFICATION qualifier
706 1364 2
707 1365 2 SELECTONE CLISPRESENT ( MOUNT_VER_DESC ) OF
708 1366 2 SET
709 1367 2 [CLIS_PRESENT,
710 1368 2 [CLIS_DEFAULTED] : MOUNT_OPTIONS [OPT_MOUNTVER] = 1;
711 1369 2 [CLIS_NEGATED] : MOUNT_OPTIONS [OPT_MOUNTVER] = 0;
712 1370 2 TES;
713 1371 2
714 1372 2 ! /QUOTA qualifier
715 1373 2
716 1374 2 SELECTONE CLISPRESENT ( QUOTA_DESC ) OF
717 1375 2 SET
718 1376 2 [CLIS_PRESENT,
719 1377 2 [CLIS_DEFAULTED] : MOUNT_OPTIONS [OPT_NOQUOTA] = 0;
720 1378 2 [CLIS_NEGATED] : MOUNT_OPTIONS [OPT_NOQUOTA] = 1;
721 1379 2 TES;
722 1380 2
723 1381 2 ! /SHARE qualifier (default is NOSHARE)
724 1382 2
725 1383 2 SELECTONE CLISPRESENT ( SHARE_DESC ) OF
726 1384 2 SET
727 1385 2 [CLIS_PRESENT] : BEGIN
728 1386 2 MOUNT_OPTIONS [OPT_SHARE] = 1;
729 1387 2 MOUNT_OPTIONS [OPT_NOSHARE] = 0;
730 1388 2 END;
731 1389 2 [CLIS_DEFAULTED,
732 1390 2 [CLIS_NEGATED] : MOUNT_OPTIONS [OPT_NOSHARE] = 1;
733 1391 2 TES;
734 1392 2
735 1393 2 ! /SYSTEM qualifier
736 1394 2
737 1395 2
738 1396 2 SELECTONE CLISPRESENT ( SYSTEM_DESC ) OF
739 1397 2 SET
740 1398 2 [CLIS_PRESENT] : MOUNT_OPTIONS [OPT_SYSTEM] = 1;
741 1399 2 [CLIS_DEFAULTED,
742 1400 2 [CLIS_ABSENT,
743 1401 2 [CLIS_NEGATED] : MOUNT_OPTIONS [OPT_SYSTEM] = 0;
744 1402 2 TES;
745 1403 2
746 1404 2 ! /UNLOAD qualifier
747 1405 2
748 1406 2
749 1407 2 SELECTONE CLISPRESENT ( UNLOAD_DESC ) OF
750 1408 2 SET
751 1409 2 [CLIS_PRESENT,
752 1410 2 [CLIS_DEFAULTED] : MOUNT_OPTIONS [OPT_NOUNLOAD] = 0;
753 1411 2 [CLIS_NEGATED] : MOUNT_OPTIONS [OPT_NOUNLOAD] = 1;
754 1412 2 TES;
755 1413 2
```

```

: 756      1414 2  ! /WRITE qualifier
: 757      1415 2  !
: 758      1416 2  SELECTONE CLISPRESNT ( WRITE_DESC ) OF
: 759      1417 2  SET
: 760      1418 2      [CLIS_PRESENT
: 761      1419 2      CLIS_DEFAULTED] : MOUNT_OPTIONS [OPT_WRITE] = 1;
: 762      1420 2      [CLIS_NEGATED] : MOUNT_OPTIONS [OPT_WRITE] = 0;
: 763      1421 2  TES;
: 764      1422 2  !
: 765      1423 2  ! /[NO]REBUILD qualifier
: 766      1424 2  !
: 767      1425 2  SELECTONE CLISPRESNT ( REBUILD_DESC ) OF
: 768      1426 2  SET
: 769      1427 2      [CLIS_PRESENT
: 770      1428 2      CLIS_DEFAULTED] : MOUNT_OPTIONS [OPT_NOREBUILD] = 0;
: 771      1429 2      [CLIS_NEGATED] : MOUNT_OPTIONS [OPT_NOREBUILD] = 1;
: 772      1430 2  TES;
```

```

774 1431 2
775 1432 3
776 1433 3
777 1434 3
778 1435 3
779 1436 3
780 1437 3
781 1438 3
782 1439 3
783 1440 4
784 1441 4
785 1442 4
786 1443 3
787 1444 3
788 1445 3
789 1446 3
790 1447 3
791 1448 3
792 1449 3
793 1450 3
794 1451 3
795 1452 3
796 1453 3
797 1454 3
798 1455 3
799 1456 3
800 1457 2
801 1458 2
802 1459 2
803 1460 3
804 1461 3
805 1462 3
806 1463 3
807 1464 3
808 1465 4
809 1466 4
810 1467 4
811 1468 3
812 1469 3
813 1470 3
814 1471 3
815 1472 3
816 1473 3
817 1474 3
818 1475 3
819 1476 3
820 1477 3
821 1478 3
822 1479 3
823 1480 3
824 1481 3
825 1482 3
826 1483 3
827 1484 3
828 1485 3
829 1486 3
830 1487 3

: 774 1431 2
: 775 1432 3
: 776 1433 3
: 777 1434 3
: 778 1435 3
: 779 1436 3
: 780 1437 3
: 781 1438 3
: 782 1439 3
: 783 1440 4
: 784 1441 4
: 785 1442 4
: 786 1443 3
: 787 1444 3
: 788 1445 3
: 789 1446 3
: 790 1447 3
: 791 1448 3
: 792 1449 3
: 793 1450 3
: 794 1451 3
: 795 1452 3
: 796 1453 3
: 797 1454 3
: 798 1455 3
: 799 1456 3
: 800 1457 2
: 801 1458 2
: 802 1459 2
: 803 1460 3
: 804 1461 3
: 805 1462 3
: 806 1463 3
: 807 1464 3
: 808 1465 4
: 809 1466 4
: 810 1467 4
: 811 1468 3
: 812 1469 3
: 813 1470 3
: 814 1471 3
: 815 1472 3
: 816 1473 3
: 817 1474 3
: 818 1475 3
: 819 1476 3
: 820 1477 3
: 821 1478 3
: 822 1479 3
: 823 1480 3
: 824 1481 3
: 825 1482 3
: 826 1483 3
: 827 1484 3
: 828 1485 3
: 829 1486 3
: 830 1487 3

: Finally, parse the qualifiers that might have values, or require values
: /ACCESSED qualifier
: IF ( MOUNT_OPTIONS [OPT_ACCESSED] = CLISPRESNT (ACCESSED_DESC) )
: THEN
: BEGIN
:   CLISGET_VALUE ( ACCESSED_DESC, CLI_DESC );
:   IF NOT T LIB$CVT_DTB ( .CLI_DESC [DSC$W_LENGTH],
:     .CLI_DESC [DSC$A_POINTER],
:     ACCESS ) )
:   THEN
:     ERR_EXIT (MOUN$_VALCNVERR);
:   END;
: /BIND qualifier
: IF ( MOUNT_OPTIONS [OPT_BIND] = CLISPRESNT (BIND_DESC) )
: THEN
: BEGIN
:   CLISGET_VALUE ( BIND_DESC, CLI_DESC );
:   CH$FILL ( 0, DSC$S_BLN, STRUCT_NAME );
:   STRUCT_NAME [DSC$B_DTYPE] = DSC$R_DTYPE_T;
:   STRUCT_NAME [DSC$B_CLASS] = DSC$K_CLASS_D;
:   STR$COPY_DX ( STRUCT_NAME, CLI_DESC );
: END;
: /BLOCKSIZE qualifier
: IF ( MOUNT_OPTIONS [OPT_BLOCK] = CLISPRESNT (BLOCK_DESC) )
: THEN
: BEGIN
:   CLISGET_VALUE ( BLOCK_DESC, CLI_DESC );
:   IF NOT T LIB$CVT_DTB ( .CLI_DESC [DSC$W_LENGTH],
:     .CLI_DESC [DSC$A_POINTER],
:     BLOCKSZ ) )
:   THEN
:     ERR_EXIT (MOUN$_VALCNVERR);
:   IF .BLOCKSZ GTRU 65534
:   THEN
:     ERR_EXIT (MOUN$_SZTOOBIG);
:   MOUNT_OPTIONS [OPT_BLOCKSIZE] = 1;
: END;
: /CACHE qualifier. If the /NOCACHE qualifier was explicit, then inhibit
: all options.
: SELECTONE CLISPRESNT (CACHE_DESC) OF
: SET
:   [CLIS_PRESENT] : BEGIN
:     MOUNT_OPTIONS [OPT_CACHE] = 1;
:     CACHE_ACT ();
:     END;
:   [CLIS_NEGATED] : BEGIN
:     MOUNT_OPTIONS [OPT_NOCACHE] = 1;
```

```

831      1488      3      MOUNT_OPTIONS [OPT_WTHRU] = 1;
832      1489      3      MOUNT_OPTIONS [OPT_NOEXT_C] = 1;
833      1490      3      MOUNT_OPTIONS [OPT_NOFID_C] = 1;
834      1491      3      MOUNT_OPTIONS [OPT_NOQUO_C] = 1;
835      1492      3      END;
836      1493      3      TES;
837      1494      3      ! /COMMENT qualifier
838      1495      3      !
839      1496      3      IF ( MOUNT_OPTIONS [OPT_COMMENT] = CL$PRESENT (COMMENT_DESC) )
840      1497      3      THEN
841      1498      2      BEGIN
842      1499      3      CL$GET_VALUE ( COMMENT_DESC, CLI_DESC );
843      1500      3      CH$FILL ( 0, DSC$S_BLN, COMMENT_STRING );
844      1501      3      COMMENT_STRING [DSC$B_DTYPE] = DSC$K_DTYPE_T;
845      1502      3      COMMENT_STRING [DSC$B_CLASS] = DSC$K_CLASS_D;
846      1503      3      STR$COPY_DX ( COMMENT_STRING, CLI_DESC );
847      1504      3      END;
848      1505      2
849      1506      2      ! /DATA_CHECK qualifier (value not required)
850      1507      2      !
851      1508      2      IF CL$PRESENT (DATA_DESC)
852      1509      2      THEN
853      1510      2      DATACHECK_ACT ();
854      1511      2
855      1512      2      ! /DENSITY qualifier
856      1513      2      !
857      1514      2      IF ( MOUNT_OPTIONS [OPT_DENSITY] = CL$PRESENT (DENSITY_DESC) )
858      1515      3      THEN
859      1516      2      DENSITY_ACT ();
860      1517      2
861      1518      2      ! /EXTENSION qualifier
862      1519      2      !
863      1520      2      IF ( MOUNT_OPTIONS [OPT_EXTENSION] = CL$PRESENT (EXTENSION_DESC) )
864      1521      3      THEN
865      1522      2      BEGIN
866      1523      3      CL$GET_VALUE ( EXTENSION_DESC, CLI_DESC );
867      1524      3      IF NOT ? LIB$CVT_DTB ( .CLI_DESC [DSC$W_LENGTH],
868      1525      4      .CLI_DESC [DSC$A_POINTER],
869      1526      4      EXTENSION ) )
870      1527      4      THEN
871      1528      3      ERR_EXIT (MOUN$VALCNVERR);
872      1529      3      END;
873      1530      2
874      1531      2      ! /INITIALIZE qualifier
875      1532      2      !
876      1533      2      IF CL$PRESENT ( INITIALIZE_DESC )
877      1534      2      THEN
878      1535      2      INITIALIZE_ACT ();
879      1536      2
880      1537      2      ! /JOURNAL qualifier (value not required)
881      1538      2      !
882      1539      2      !**JNL** SELECTONE CL$PRESENT (JOURNAL_DESC) OF
883      1540      2      !**JNL** SET
884      1541      2      !**JNL** [CL$PRESENT] : JOURNAL_ACT ();
885      1542      2      !**JNL** [CL$NEGATED] : BEGIN
886      1543      2
887      1544      2
```

```

888 1545 2  **JNL**      MOUNT_OPTIONS [OPT_NOJRNL] = 1;
889 1546 2  **JNL**      MOUNT_OPTIONS [OPT_NEWJRNL] = 0;
890 1547 2  **JNL**      JRNL_SIZE = 0;
891 1548 2  **JNL**      JRNL_EXTEND = 0;
892 1549 2  **JNL**      JRNL_QUOTA = 0;
893 1550 2  **JNL**      JRNL_RECORD_SIZE = 0;
894 1551 2  **JNL**      END;
895 1552 2  **JNL** TES;
896 1553 2
897 1554 2  ! /OVERRIDE qualifier
898 1555 2
899 1556 2  IF CLISPRESNT (OVERRIDE_DESC)
900 1557 2  THEN
901 1558 2      OVERRIDE_ACT ();
902 1559 2
903 1560 2  ! /OWNER_UIC qualifier
904 1561 2
905 1562 2  IF ( MOUNT_OPTIONS [OPT_OWNER_UIC] = CLISPRESNT (OWNER_DESC) )
906 1563 2  THEN
907 1564 2      OWNER_UIC_ACT ();
908 1565 2
909 1566 2  ! /PROCESSOR qualifier
910 1567 2
911 1568 2  IF CLISPRESNT (PROCESSOR_DESC)
912 1569 2  THEN
913 1570 2      PROCESSOR_ACT ();
914 1571 2
915 1572 2  ! /PROTECTION qualifier
916 1573 2
917 1574 2  IF ( MOUNT_OPTIONS [OPT_PROTECTION] = CLISPRESNT (PROTECTION_DESC) )
918 1575 2  THEN
919 1576 2      PROTECTION_ACT ();
920 1577 2
921 1578 2  ! /RECORDSIZE qualifier
922 1579 2
923 1580 3  IF ( MOUNT_OPTIONS [OPT_RECORDSZ] = CLISPRESNT (RECORD_DESC) )
924 1581 2  THEN
925 1582 3      BEGIN
926 1583 3          CLISGET_VALUE ( RECORD_DESC, CLI_DESC );
927 1584 4          IF NOT 7 LIB$CVT_DTB ( - CLI_DESC [DSC$W_LENGTH],
928 1585 4              . CLI_DESC [DSC$A_POINTER],
929 1586 4              RECORDSZ ) )
930 1587 3      THEN
931 1588 3          ERR_EXIT (MOUN$_VALCNVERR);
932 1589 3
933 1590 3      IF .RECORDSZ GTRU 65534
934 1591 3      THEN
935 1592 3          ERR_EXIT (MOUN$_SZTOOBIG);
936 1593 2      END;
937 1594 2
938 1595 2  ! /WINDOWS qualifier
939 1596 2
940 1597 2  IF ( MOUNT_OPTIONS [OPT_WINDOW] = CLISPRESNT (WINDOW_DESC) )
941 1598 2  THEN
942 1599 2      BEGIN
943 1600 2          CLISGET_VALUE ( WINDOW_DESC, CLI_DESC );
944 1601 2
```



```
: 945      1602  4      IF NOT ( LIB$CVT_DTB ( .CLI_DESC [DSC$W_LENGTH],  
: 946      1603      4      .CLI_DESC [DSC$A_POINTER],  
: 947      1604      4      WINDOW ) )  
: 948      1605      3      THEN  
: 949      1606      3      ERR_EXIT (MOUN$_VALCNVERR);  
: 950      1607      2      END;  
: 951      1608      2  
: 952      1609      1  END;  
                                ! of PARSE_QUALIFIER routine
```

```
OFFC 00000 PARSE_QUALIFIER:  
      5B 00000000G 8F D0 00002      .WORD      Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11      : 1234  
      5A 00000000G 8F D0 00009      MOVL      #CLIS_DEFAULTED, R11  
      59 00000000G 8F D0 00010      MOVL      #CLIS_NEGATED, R10  
      58      0000' CF 9E 00017      MOVL      #CLIS_PRESENT, R9  
      57 00000000G 00 9E 0001C      MOVAB     ACCESSED_DESC, R8  
      56      0000' CF 9E 00023      MOVAB     CLISPRESENT, R7  
      55      00BC  C8 9F 00028      MOVAB     MOUNT_OPTIONS, R6  
      67      01 FB 0002C      PUSHAB    FOREIGN_DESC  
      06      50 E9 0002F      CALLS      #1, CLISPRESENT  
01  A6      08 88 00032      BLBC      R0, 1$      : 1274  
      04 11 00036      BISB2     #8, MOUNT_OPTIONS+1  
01  A6      08 8A 00038 1$:      BRB      2$      : 1276  
      010C  C8 9F 0003C 2$:      BICB2     #8, MOUNT_OPTIONS+1  
      67      01 FB 00040      PUSHAB    LABEL_DESC      : 1278  
      09      50 E9 00043      CALLS      #1, CLISPRESENT      : 1283  
      03  A6      80 8F 88 00046      BLBC      R0, 3$  
01  A6      10 8A 00048      BISB2     #128, MOUNT_OPTIONS+3      : 1286  
      0148  C8 9F 0004F 3$:      BICB2     #16, MOUNT_OPTIONS+1  
      67      01 FB 00053      PUSHAB    NOLABEL_DESC  
      09      50 E9 00056      CALLS      #1, CLISPRESENT  
01  A6      10 88 00059      BLBC      R0, 4$      : 1292  
03  A6      80 8F 8A 0005D      BISB2     #16, MOUNT_OPTIONS+1      : 1295  
      10  A8 9F 00062 4$:      BICB2     #128, MOUNT_OPTIONS+3  
      67      01 FB 00065      PUSHAB    ASSIST_DESC  
      59      50 D1 00068      CALLS      #1, CLISPRESENT  
      5B      05 13 0006B      CMPL      R0, R9  
      06  A6      06 12 00070      BEQL      5$      : 1308  
      5A      04 88 00072 5$:      CMPL      R0, R11  
      09      50 D1 00076      BNEQ      6$      : 1309  
      06  A6      04 12 00078 6$:      BRB      7$      : 1310  
      24  A6      04 8A 0007D      CMPL      R0, R10  
      67      01 FB 00084 7$:      BNEQ      7$      : 1315  
      59      50 D1 00087      BICB2     #4, MOUNT_OPTIONS+6  
      5B      05 13 0008A      PUSHAB    AUTOMATIC_DESC  
      07  A6      06 12 0008F      CALLS      #1, CLISPRESENT  
      5A      09 11 00095      CMPL      R0, R9  
      50  D1 00097 8$:      BEQL      8$      : 1317  
      02 8A 00091 8$:      CMPL      R0, R11  
      09 11 00095 9$:      BNEQ      9$      : 1318  
      50  D1 00097 9$:      BRB      10$      : 1319  
      5A      50  D1 00097 9$:      CMPL      R0, R10
```

07	A6		04	12	0009A	BNEQ	10\$		
			02	88	0009C	BISB2	#2, MOUNT_OPTIONS+7		
		64	A8	9F	000A0	PUSHAB	CLUSTER_DESC	1325	
	67		01	FB	000A3	CALLS	#1, CLISPPRESENT		
	59		50	D1	000A6	CMPL	R0, R9	1327	
			07	12	000A9	BNEQ	11\$		
	A6	40	8F	88	000AB	BISB2	#64, MOUNT_OPTIONS+7		
			18	11	000B0	BRB	13\$		
	5B		50	D1	000B2	CMPL	R0, R11	1328	
			0E	13	000B5	BEQL	12\$		
00000000G	8F		50	D1	000B7	CMPL	R0, #CLIS_ABSENT		
			05	13	000BE	BEQL	12\$		
	5A		50	D1	000C0	CMPL	R0, R10		
			05	12	000C3	BNEQ	13\$		
	A6	40	8F	8A	000C5	BICB2	#64, MOUNT_OPTIONS+7	1330	
		00CC	C8	9F	000CA	PUSHAB	GROUP_DESC	1336	
	67		01	FB	000CE	CALLS	#1, CLISPPRESENT		
	59		50	D1	000D1	CMPL	R0, R9	1338	
			06	12	000D4	BNEQ	14\$		
	66	80	8F	88	000D6	BISB2	#128, MOUNT_OPTIONS		
			17	11	000DA	BRB	16\$		
	5B		50	D1	000DC	CMPL	R0, R11	1339	
			0E	13	000DF	BEQL	15\$		
00000000G	8F		50	D1	000E1	CMPL	R0, #CLIS_ABSENT		
			05	13	000E8	BEQL	15\$		
	5A		50	D1	000EA	CMPL	R0, R10		
			04	12	000ED	BNEQ	16\$		
	66	80	8F	8A	000EF	BICB2	#128, MOUNT_OPTIONS	1341	
		00D8	C8	9F	000F3	PUSHAB	HDR3_DESC	1347	
	67		01	FB	000F7	CALLS	#1, CLISPPRESENT		
	59		50	D1	000FA	CMPL	R0, R9	1349	
			05	13	000FD	BEQL	16\$		
	5B		50	D1	000FF	CMPL	R0, R11		
			06	12	00102	BNEQ	18\$		
	A6		10	8A	00104	BICB2	#16, MOUNT_OPTIONS+5	1350	
			09	11	00108	BRB	19\$		
	5A		50	D1	0010A	CMPL	R0, R10	1351	
			04	12	0010D	BNEQ	19\$		
	A6	011C	10	88	0010F	BISB2	#16, MOUNT_OPTIONS+5		
			C8	9F	00113	PUSHAB	MESSAGE_DESC	1356	
	67		01	FB	00117	CALLS	#1, CLISPPRESENT		
	59		50	D1	0011A	CMPL	R0, R9	1358	
			05	13	0011D	BEQL	20\$		
	5B		50	D1	0011F	CMPL	R0, R11		
			06	12	00122	BNEQ	21\$		
	A6		08	88	00124	BISB2	#8, MOUNT_OPTIONS+6	1359	
			09	11	00128	BRB	22\$		
	5A		50	D1	0012A	CMPL	R0, R10	1360	
			04	12	0012D	BNEQ	22\$		
	A6	0138	08	8A	0012F	BICB2	#8, MOUNT_OPTIONS+6		
			C8	9F	00133	PUSHAB	MOUNT_VER_DESC	1365	
	67		01	FB	00137	CALLS	#1, CLISPPRESENT		
	59		50	D1	0013A	CMPL	R0, R9	1367	
			05	13	0013D	BEQL	23\$		
	5B		50	D1	0013F	CMPL	R0, R11		
			07	12	00142	BNEQ	24\$		
	A6	40	8F	88	00144	BISB2	#64, MOUNT_OPTIONS+6	1368	

	5A		0A 11 00149	BRB	25\$		
			50 D1 0014B	CMPL	R0, R10		1369
06	A6	40	05 12 0014E	BNEQ	25\$		
		01A4	8F 8A 00150	BICB2	#64, MOUNT_OPTIONS+6		
	67		C8 9F 00155	PUSHAB	QUOTA_DESC		1374
	59		01 FB 00159	CALLS	#1, CLISPRESNT		
			50 D1 0015C	CMPL	R0, R9		1376
	5B		05 13 0015F	BEQL	26\$		
			50 D1 00161	CMPL	R0, R11		
05	A6		06 12 00164	BNEQ	27\$		
			04 8A 00166	BICB2	#4, MOUNT_OPTIONS+5		1377
	5A		09 11 0016A	BRB	28\$		
			50 D1 0016C	CMPL	R0, R10		1378
05	A6		04 12 0016F	BNEQ	28\$		
		01D8	04 88 00171	BISB2	#4, MOUNT_OPTIONS+5		
	67		C8 9F 00175	PUSHAB	SHARE_DESC		1383
	59		01 FB 00179	CALLS	#1, CLISPRESNT		
			50 D1 0017C	CMPL	R0, R9		1385
			09 12 0017F	BNEQ	29\$		
	66	40	8F 88 00181	BISB2	#64, MOUNT_OPTIONS		1386
	66		10 8A 00185	BICB2	#16, MOUNT_OPTIONS		1387
			0D 11 00188	BRB	31\$		1383
	5B		50 D1 0018A	CMPL	R0, R11		1389
			05 13 0018D	BEQL	30\$		
	5A		50 D1 0018F	CMPL	R0, R10		
			03 12 00192	BNEQ	31\$		
	66		10 88 00194	BISB2	#16, MOUNT_OPTIONS		1390
		01E8	C8 9F 00197	PUSHAB	SYSTEM_DESC		1396
	67		01 FB 0019B	CALLS	#1, CLISPRESNT		
	59		50 D1 0019E	CMPL	R0, R9		1398
			06 12 001A1	BNEQ	32\$		
01	A6		01 88 001A3	BISB2	#1, MOUNT_OPTIONS+1		
			17 11 001A7	BRB	34\$		
	5B		50 D1 001A9	CMPL	R0, R11		1399
			0E 13 001AC	BEQL	33\$		
00000000G	8F		50 D1 001AE	CMPL	R0, #CLIS_ABSENT		
			05 13 001B5	BEQL	33\$		
	5A		50 D1 001B7	CMPL	R0, R10		
			04 12 001BA	BNEQ	34\$		
01	A6		01 8A 001BC	BICB2	#1, MOUNT_OPTIONS+1		1401
		01F8	C8 9F 001C0	PUSHAB	UNLOAD_DESC		1407
	67		01 FB 001C4	CALLS	#1, CLISPRESNT		
	59		50 D1 001C7	CMPL	R0, R9		1409
			05 13 001CA	BEQL	35\$		
	5B		50 D1 001CC	CMPL	R0, R11		
			06 12 001CF	BNEQ	36\$		
01	A6		04 8A 001D1	BICB2	#4, MOUNT_OPTIONS+1		1410
			09 11 001D5	BRB	37\$		
	5A		50 D1 001D7	CMPL	R0, R10		1411
			04 12 001DA	BNEQ	37\$		
01	A6		04 88 001DC	BISB2	#4, MOUNT_OPTIONS+1		
		0218	C8 9F 001E0	PUSHAB	WRITE_DESC		1416
	67		01 FB 001E4	CALLS	#1, CLISPRESNT		
	59		50 D1 001E7	CMPL	R0, R9		1418
			05 13 001EA	BEQL	38\$		
	5B		50 D1 001EC	CMPL	R0, R11		
			06 12 001EF	BNEQ	39\$		

	01	A6		02	88	001F1	38\$:	BISB2	#2, MOUNT_OPTIONS+1	1419
		5A		09	11	001F5		BRB	40\$	
				50	D1	001F7	39\$:	CMPL	R0, R10	1420
	01	A6		04	12	001FA		BNEQ	40\$	
			01B4	02	8A	001FC		BICB2	#2, MOUNT_OPTIONS+1	
		67		C8	9F	00200	40\$:	PUSHAB	REBUILD_DESC	1425
		59		01	FB	00204		CALLS	#1, CLISPRESNT	
				50	D1	00207		CMPL	R0, R9	1427
		5B		05	13	0020A		BEQL	41\$	
				50	D1	0020C		CMPL	R0, R11	
				07	12	0020F		BNEQ	42\$	
	07	A6	80	8F	8A	00211	41\$:	BICB2	#128, MOUNT_OPTIONS+7	1428
		5A		0A	11	00216		BRB	43\$	
				50	D1	00218	42\$:	CMPL	R0, R10	1429
				05	12	0021B		BNEQ	43\$	
	07	A6	80	8F	88	0021D		BISB2	#128, MOUNT_OPTIONS+7	
				58	DD	00222	43\$:	PUSHL	R8	1436
		67		01	FB	00224		CALLS	#1, CLISPRESNT	
03	A6	01		50	F0	00227		INSV	R0, #1, #1, MOUNT_OPTIONS+3	
		2D		50	E9	0022D		BLBC	R0, 44\$	
			60	A6	9F	00230		PUSHAB	CLI_DESC	1439
				58	DD	00233		PUSHL	R8	
		00000000G	00	02	FB	00235		CALLS	#2, CLISGET_VALUE	
			0C	A6	9F	0023C		PUSHAB	ACCESS	1440
			64	A6	DD	0023F		PUSHL	CLI_DESC+4	1441
		7E	60	A6	3C	00242		MOVZWL	CLI_DESC, -(SP)	1440
		00000000G	00	03	FB	00246		CALLS	#3, LIB\$CVT_DTB	
		0D		50	E8	0024D		BLBS	R0, 44\$	
			0072805C	8F	DD	00250		PUSHL	#7503964	1444
		00000000G	00	01	FB	00256		CALLS	#1, LIB\$STOP	
			30	A8	9F	0025D	44\$:	PUSHAB	BIND_DESC	1449
		67		01	FB	00260		CALLS	#1, CLISPRESNT	
05	A6	01		50	F0	00263		INSV	R0, #0, #1, MOUNT_OPTIONS+5	
		00		50	E9	00269		BLBC	R0, 45\$	
		27		60	A6	9F	0026C	PUSHAB	CLI_DESC	1452
			30	A8	9F	0026F		PUSHAB	BIND_DESC	
		00000000G	00	02	FB	00272		CALLS	#2, CLISGET_VALUE	
08		00		00	2C	00279		MOVCS	#0, (SP), #0, #8, STRUCT_NAME	1453
		6E		54	A6	0027E				
	56	A6	020E	8F	B0	00280		MOVW	#526, STRUCT_NAME+2	1454
			60	A6	9F	00286		PUSHAB	CLI_DESC	1456
			54	A6	9F	00289		PUSHAB	STRUCT_NAME	
		00000000G	00	02	FB	0028C		CALLS	#2, STR\$COPY_DX	
			44	A8	9F	00293	45\$:	PUSHAB	BLOCK_DESC	1461
		67		01	FB	00296		CALLS	#1, CLISPRESNT	
01	A6	01		50	F0	00299		INSV	R0, #7, #1, MOUNT_OPTIONS+1	
		07		50	E9	0029F		BLBC	R0, 48\$	
		49		60	A6	9F	002A2	PUSHAB	CLI_DESC	1464
			44	A8	9F	002A5		PUSHAB	BLOCK_DESC	
		00000000G	00	02	FB	002A8		CALLS	#2, CLISGET_VALUE	
			18	A6	9F	002AF		PUSHAB	BLOCKSZ	1465
			64	A6	DD	002B2		PUSHL	CLI_DESC+4	1466
		7E	60	A6	3C	002B5		MOVZWL	CLI_DESC, -(SP)	1465
		00000000G	00	03	FB	002B9		CALLS	#3, LIB\$CVT_DTB	
		0D		50	E8	002C0		BLBS	R0, 46\$	
			0072805C	8F	DD	002C3		PUSHL	#7503964	1469
		00000000G	00	01	FB	002C9		CALLS	#1, LIB\$STOP	

		0000FFFE	8F	18	A6	D1	002D0	46\$:	CMPL	BLOCKSZ, #65534	1471
					0D	1B	002D8		BLEQU	47\$	1473
		00000000G	00	0072817C	8F	DD	002DA		PUSHL	#7504252	1474
		02	A6		01	FB	002E0		CALLS	#1, LIB\$STOP	1480
				54	01	88	002E7	47\$:	BISB2	#1, MOUNT_OPTIONS+2	1482
			67		A8	9F	002EB	48\$:	PUSHAB	CACHE_DESC	1483
			59		01	FB	002EE		CALLS	#1, CLISPRESNT	1484
					50	D1	002F1		CMPL	R0, R9	1486
		05	A6		0B	12	002F4		BNEQ	49\$	1491
		0000V	CF		20	88	002F6		BISB2	#32, MOUNT_OPTIONS+5	1497
					00	FB	002FA		CALLS	#0, CACHE_ACT	1500
			5A		0B	11	002FF		BRB	50\$	1501
					50	D1	00301	49\$:	CMPL	R0, R10	1502
		05	A6	13C0	06	12	00304		BNEQ	50\$	1504
				74	8F	A8	00306		BISW2	#5056, MOUNT_OPTIONS+5	1509
66			67		A8	9F	0030C	50\$:	PUSHAB	COMMENT_DESC	1511
	01		03		01	FB	0030F		CALLS	#1, CLISPRESNT	1515
			27		50	F0	00312		INSV	R0, #3, #1, MOUNT_OPTIONS	1517
				60	50	F9	00317		BLBC	R0, 51\$	1521
				74	A6	9F	0031A		PUSHAB	CLI_DESC	1524
		00000000G	00		A8	9F	0031D		PUSHAB	COMMENT_DESC	1525
08	00		6E		02	FB	00320		CALLS	#2, CLISGET_VALUE	1526
				28	00	2C	00327		MOVCS	#0, (SP), #0, #8, COMMENT_STRING	1529
		2A	A6	020E	A6		0032C				1535
				60	8F	B0	0032E		MOVW	#526, COMMENT_STRING+2	1537
				28	A6	9F	00334		PUSHAB	CLI_DESC	1540
		00000000G	00		A6	9F	00337		PUSHAB	COMMENT_STRING	1544
				0088	02	FB	0033A		CALLS	#2, STR\$COPY_DX	1548
			67		C8	9F	00341	51\$:	PUSHAB	DATA_DESC	1551
			05		01	FB	00345		CALLS	#1, CLISPRESNT	1555
		0000V	CF		50	E9	00348		BLBC	R0, 52\$	1559
				0098	00	FB	0034B		CALLS	#0, DATACHECK_ACT	1563
			67		C8	9F	00350	52\$:	PUSHAB	DENSITY_DESC	1567
66	01		00		01	FB	00354		CALLS	#1, CLISPRESNT	1571
			05		50	F0	00357		INSV	R0, #0, #1, MOUNT_OPTIONS	1575
		0000V	CF		50	E9	0035C		BLBC	R0, 53\$	1579
				00AC	00	FB	0035F		CALLS	#0, DENSITY_ACT	1583
			67		C8	9F	00364	53\$:	PUSHAB	EXTENSION_DESC	1587
02	A6		07		01	FB	00368		CALLS	#1, CLISPRESNT	1591
			2F		50	F0	0036B		INSV	R0, #7, #1, MOUNT_OPTIONS+2	1595
				60	50	E9	00371		BLBC	R0, 54\$	1599
				00AC	A6	9F	00374		PUSHAB	CLI_DESC	1603
		00000000G	00		C8	9F	00377		PUSHAB	EXTENSION_DESC	1607
				34	02	FB	0037B		CALLS	#2, CLISGET_VALUE	1611
				64	A6	9F	00382		PUSHAB	EXTENSION	1615
			7E		A6	DD	00385		PUSHL	CLI_DESC+4	1619
		00000000G	00		A6	3C	00388		MOVZWL	CLI_DESC, -(SP)	1623
			0D		03	FB	0038C		CALLS	#3, LIB\$CVT_DTB	1627
				0072805C	50	E8	00393		BLBS	R0, 54\$	1631
		00000000G	00		8F	DD	00396		PUSHL	#7503964	1635
				00EC	01	FB	0039C		CALLS	#1, LIB\$STOP	1639
			67		C8	9F	003A3	54\$:	PUSHAB	INITIALIZE_DESC	1643
			05		01	FB	003A7		CALLS	#1, CLISPRESNT	1647
		0000V	CF		50	E9	003AA		BLBC	R0, 55\$	1651
				0158	00	FB	003AD		CALLS	#0, INITIALIZE_ACT	1655
			67		C8	9F	003B2	55\$:	PUSHAB	OVERRIDE_DESC	1659
					01	FB	003B6		CALLS	#1, CLISPRESNT	1663

			05		50	E9 003B9	BLBC	R0, 56\$		
			CF		00	FB 003BC	CALLS	#0, OVERRIDE_ACT	1558	
				016C	C8	9F 003C1	PUSHAB	OWNER_DESC	1562	
02	A6	01	67		01	FB 003C5	CALLS	#1, CLISPRESNT		
			02		50	FO 003C8	INSV	R0, #2, #1, MOUNT_OPTIONS+2		
			05		50	E9 003CE	BLBC	R0, 57\$		
			CF		00	FB 003D1	CALLS	#0, OWNER_UIC_ACT	1564	
				0180	C8	9F 003D6	PUSHAB	PROCESSOR_DESC	1568	
			67		01	FB 003DA	CALLS	#1, CLISPRESNT		
			05		50	E9 003DD	BLBC	R0, 58\$		
			CF		00	FB 003E0	CALLS	#0, PROCESSOR_ACT	1570	
				0194	C8	9F 003E5	PUSHAB	PROTECTION_DESC	1574	
02	A6	01	67		01	FB 003E9	CALLS	#1, CLISPRESNT		
			01		50	FO 003EC	INSV	R0, #1, #1, MOUNT_OPTIONS+2		
			05		50	E9 003F2	BLBC	R0, 59\$		
			CF		00	FB 003F5	CALLS	#0, PROTECTION_ACT	1576	
				01C8	C8	9F 003FA	PUSHAB	RECORD_DE_L	1580	
04	A6	01	67		01	FB 003FE	CALLS	#1, CLISPRESNT		
			05		50	FO 00401	INSV	R0, #5, #1, MOUNT_OPTIONS+4		
			46		50	E9 00407	BLBC	R0, 61\$		
				60	A6	9F 0040A	PUSHAB	CLI_DESC	1583	
				01C8	C8	9F 0040D	PUSHAB	RECORD_DESC		
			00		02	FB 00411	CALLS	#2, CLISGET_VALUE	1584	
				50	A6	9F 00418	PUSHAB	RECORDSZ	1585	
				64	A6	DD 0041B	PUSHL	CLI_DESC+4	1584	
			7E		60	A6	3C 0041E	MOVZWL	CLI_DESC, -(SP)	
			00		03	FB 00422	CALLS	#3, LIB\$CVT_DTB		
			0D		50	E8 00429	BLBS	R0, 60\$		
				0072805C	8F	DD 0042C	PUSHL	#7503964	1588	
			00		01	FB 00432	CALLS	#1, LIB\$STOP		
			0000FFFE		50	A6	D1 00439	CMPL	RECORDSZ, #65534	1590
					0D	1B 00441	BLEQU	61\$		
				0072817C	8F	DD 00443	PUSHL	#7504252	1592	
			00		01	FB 00449	CALLS	#1, LIB\$STOP		
				0208	C8	9F 00450	PUSHAB	WINDOW_DESC	1597	
03	A6	01	67		01	FB 00454	CALLS	#1, CLISPRESNT		
			00		50	FO 00457	INSV	R0, #0, #1, MOUNT_OPTIONS+3		
			2F		50	E9 0045D	BLBC	R0, 62\$		
				60	A6	9F 00460	PUSHAB	CLI_DESC	1600	
				0208	C8	9F 00463	PUSHAB	WINDOW_DESC		
			00		02	FB 00467	CALLS	#2, CLISGET_VALUE		
				5C	A6	9F 0046E	PUSHAB	WINDOW	1602	
				64	A6	DD 00471	PUSHL	CLI_DESC+4	1603	
			7E		60	A6	3C 00474	MOVZWL	CLI_DESC, -(SP)	1602
			00		03	FB 00478	CALLS	#3, LIB\$CVT_DTB		
			0D		50	E8 0047F	BLBS	R0, 62\$		
				0072805C	8F	DD 00482	PUSHL	#7503964	1606	
			00		01	FB 00488	CALLS	#1, LIB\$STOP		
					04	0048F	RET		1609	

; Routine Size: 1168 bytes. Routine Base: \$CODE\$ + 03B6

```

: 954      1610 1 ROUTINE BUILD_LIST (ITEM_CODE, ITEM_LENGTH, ITEM_ADDRESS, LIST_PTR) : NOVALUE =
: 955      1611 1
: 956      1612 1
: 957      1613 1 ++
: 958      1614 1 Functional description:
: 959      1615 1 This routine will build an item list entry from the input parameters.
: 960      1616 1
: 961      1617 1 Input:
: 962      1618 1
: 963      1619 1     ITEM_ADDRESS      : Address of item
: 964      1620 1     ITEM_CODE       : Item code value
: 965      1621 1     ITEM_LENGTH    : Length of item (in bytes)
: 966      1622 1     LIST_PTR      : Address of a pointer to the end of the list
: 967      1623 1
: 968      1624 1
: 969      1625 1 Implicit Input:
: 970      1626 1
: 971      1627 1 The list is assumed to be long enough.
: 972      1628 1
: 973      1629 1 Output:
: 974      1630 1
: 975      1631 1     LIST              : points to new end of list
: 976      1632 1
: 977      1633 1 Implicit output:
: 978      1634 1
: 979      1635 1 None.
: 980      1636 1
: 981      1637 1 Side effects:
: 982      1638 1
: 983      1639 1 None.
: 984      1640 1
: 985      1641 1 Routine value:
: 986      1642 1
: 987      1643 1 None.
: 988      1644 1 --
: 989      1645 1
: 990      1646 2 BEGIN                                ! Start of BUILD_ENTRY
: 991      1647 2
: 992      1648 2 LOCAL
: 993      1649 2     LIST              : REF BBLOCK;
: 994      1650 2
: 995      1651 2 MACRO
: 996      1652 2     LENGTH          = 0, 0, 16, 0%;
: 997      1653 2     CODE           = 2, 0, 16, 0%;
: 998      1654 2     ADDRESS        = 4, 0, 32, 0%;
: 999      1655 2     UNUSED         = 8, 0, 32, 0%;
1000      1656 2
1001      1657 2 LIST = ..LIST_PTR;                    ! Get address of start of entry
1002      1658 2 LIST [LENGTH] = ..ITEM_LENGTH;        ! Set the item length
1003      1659 2 LIST [CODE] = ..ITEM_CODE;             ! Set the item code
1004      1660 2 LIST [ADDRESS] = ..ITEM_ADDRESS;        ! Set the item address
1005      1661 2 LIST [UNUSED] = 0;                      ! Clear the unused portion
1006      1662 2 ..LIST_PTR = ..LIST + ITEM_SIZE;      ! Set new end of list
1007      1663 2
1008      1664 1 END;                                ! End of BUILD_ENTRY
```

H 7
16-Sep-1984 01:06:29 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:45:31 [MOUNT.SRC]MOUNTING.B32;1

Page 36
(7)

		0000 00000 BUILD_LIST:				
	50	10	BC	D0	00002	.WORD Save nothing : 1610
	60	08	AC	B0	00006	MOVW @LIST_PTR, LIST : 1657
	A0	04	AC	B0	0000A	MOVW ITEM_LENGTH, (LIST) : 1658
02	A0	0C	AC	D0	0000F	MOVW ITEM_CODE, 2(LIST) : 1659
04	A0	08	AO	D4	00014	MOVL ITEM_ADDRESS, 4(LIST) : 1660
						CLRL 8(LIST) : 1661
10	BC	0C	AO	9E	00017	MOVAB 12(R0), @LIST_PTR : 1662
				04	0001C	RET : 1664

```
; Routine Size: 29 bytes,    Routine Base: $CODES + 0846
```

[illegible]


```
1010 1665 1 ROUTINE MAIN_HANDLER (SIGNAL, MECHANISM) =
1011 1666 1
1012 1667 1 ++
1013 1668 1
1014 1669 1 FUNCTIONAL DESCRIPTION:
1015 1670 1
1016 1671 1 This routine is the main level condition handler for the MOUNT
1017 1672 1 utility. It undoes anything that MOUNT has done so far and returns
1018 1673 1 the condition code as status to MOUNT's caller (i.e., the CLI).
1019 1674 1
1020 1675 1
1021 1676 1 CALLING SEQUENCE:
1022 1677 1 MAIN_HANDLER (ARG1, ARG2)
1023 1678 1
1024 1679 1 INPUT PARAMETERS:
1025 1680 1 ARG1: address of signal array
1026 1681 1 ARG2: address of mechanism array
1027 1682 1
1028 1683 1 IMPLICIT INPUTS:
1029 1684 1 NONE
1030 1685 1
1031 1686 1 OUTPUT PARAMETERS:
1032 1687 1 NONE
1033 1688 1
1034 1689 1 IMPLICIT OUTPUTS:
1035 1690 1 NONE
1036 1691 1
1037 1692 1 ROUTINE VALUE:
1038 1693 1 NONE
1039 1694 1
1040 1695 1 SIDE EFFECTS:
1041 1696 1 stack unwound, control passed to CLI
1042 1697 1
1043 1698 1 --
1044 1699 1
1045 1700 2 BEGIN
1046 1701 2
1047 1702 2 MAP
1048 1703 2 SIGNAL : REF BBLOCK, ! signal array
1049 1704 2 MECHANISM : REF BBLOCK; ! mechanism array
1050 1705 2
1051 1706 2
1052 1707 2 ! Force the facility code to be mount and resignal the
1053 1708 2 ! error to be printed by the catch all handler.
1054 1709 2
1055 1710 2
1056 1711 2 IF .BBLOCK [SIGNAL[CHFSL_SIG_NAME], STSSV_FAC_NO] EQL 0
1057 1712 2 OR .BBLOCK [SIGNAL[CHFSL_SIG_NAME], STSSV_FAC_NO] EQL INITS_FACILITY
1058 1713 2 THEN BBLOCK [SIGNAL[CHFSL_SIG_NAME], STSSV_FAC_NO] = MOUNS_FACILITY;
1059 1714 2
1060 1715 2 RETURN SSS_RESIGNAL;
1061 1716 2
1062 1717 1 END; ! end of routine MAIN_HANDLER
```

MOUNTING
V04-000

J 7
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31

VAX-11 Bliss-32 V4.0-742
[MOUNT.SRC]MOUNTING.B32;1

Page 38
(8)

				0000 00000 MAIN_HANDLER:								
				50	04	AC	D0	00002	.WORD	Save nothing	:	1665
			OFFF	8F	06	AO	B3	00006	MOVL	SIGNAL, R0	:	1711
00000075	8F	06	AO			OC	13	0000C	BITW	6(R0), #4095	:	
				OC		00	ED	0000E	BEQL	1\$	:	1712
						0A	12	00018	CMPZV	#0, #12, 6(R0), #117	:	
06	AO		OC	00	00000072	8F	F0	0001A	BNEQ	2\$	:	1713
				50	0918	8F	3C	00024	INSV	#114, #0, #12, 6(R0)	:	1715
						04	00029		MOVZWL	#2328, R0	:	1717
									RET		:	

; Routine Size: 42 bytes, Routine Base: \$CODE\$ + 0863


```
1064 1718 1 1+
1065 1719 1 1
1066 1720 1 1 Parameter and qualifier action routines. Each routine is named corresponding
1067 1721 1 1 to its associated parameter or qualifier. Each routine does whatever
1068 1722 1 1 conversion is necessary and stores the parameter or qualifier value in
1069 1723 1 1 the appropriate location in the output area.
1070 1724 1 1
1071 1725 1 1 -
1072 1726 1 1
1073 1727 1 1
1074 1728 1 1 ROUTINE CACHE_ACT : NOVALUE =
1075 1729 2 2 BEGIN
1076 1730 2 2
1077 1731 2 2 EXTERNAL
1078 1732 2 2     CACHE_STB      : VECTOR [0],    ! state table address
1079 1733 2 2     CACHE_KTB      : VECTOR [0];    ! keyword table address
1080 1734 2 2
1081 1735 2 2 EXTERNAL ROUTINE
1082 1736 2 2     LIB$TPARSE;
1083 1737 2 2
1084 1738 2 2
1085 1739 2 2     Initialize work area.
1086 1740 2 2
1087 1741 2 2
1088 1742 2 2     EXT_CACHE = -1;                ! Set value for EXTENT not seen
1089 1743 2 2     FID_CACHE = -1;                ! Set value for FILE_ID not seen
1090 1744 2 2     QUO_CACHE = -1;                ! Set value for QUOTA not seen
1091 1745 2 2
1092 1746 2 2     Parse the cache control options and set appropriate flags.
1093 1747 2 2
1094 1748 2 2
1095 1749 2 2 WHILE CLISGET_VALUE ( CACHE_DESC, CLI_DESC ) DO
1096 1750 2 2 BEGIN
1097 1751 2 2     TPARSE_BLOCK[TPASL_STRINGCNT] = .CLI_DESC[DSCSW_LENGTH];
1098 1752 2 2     TPARSE_BLOCK[TPASL_STRINGPTR] = .CLI_DESC[DSCSA_POINTER];
1099 1753 2 2     IF NOT LIB$TPARSE (TPARSE_BLOCK, CACHE_STB, CACHE_KTB)
1100 1754 2 2     THEN
1101 1755 2 2         ERR_EXIT (MOUN$_BADCACHE);
1102 1756 2 2 END;
1103 1757 2 2
1104 1758 2 2
1105 1759 2 2     Check to see if caching should be turned off:
1106 1760 2 2
1107 1761 2 2     /CACHE=EXTENT:0      disables extent caching
1108 1762 2 2     /CACHE=FILE_ID:1    disables FID caching
1109 1763 2 2     /CACHE=QUOTA:0      disables quota caching
1110 1764 2 2
1111 1765 2 2
1112 1766 2 2 IF .EXT_CACHE EQL 0      ! /CACHE=EXTENT:0
1113 1767 2 2 THEN
1114 1768 2 2     MOUNT_OPTIONS [OPT_NOEXT_C] = 1;
1115 1769 2 2
1116 1770 2 2 IF .FID_CACHE EQL 1      ! /CACHE=FILE_ID:1
1117 1771 2 2 THEN
1118 1772 2 2     MOUNT_OPTIONS [OPT_NOFID_C] = 1;
1119 1773 2 2
1120 1774 2 2 IF .QUO_CACHE EQL 0      ! /CACHE=QUOTA:0
```

```
: 1121      1775  2 THEN
: 1122      1776  2      MOUNT_OPTIONS [OPT_NOQUO_C] = 1;
: 1123      1777  2
: 1124      1778  2
: 1125      1779  1 END;
```

! end of routine CACHE_ACT

				.EXTRN	CACHE_STB	CACHE_KTB	
				.EXTRN	LIB\$TPARSE		
0004 00000 CACHE_ACT:							
	52	0000'	CF 9E 00002	.WORD	Save R2		: 1728
	62		01 CE 00007	MOVAB	EXT_CACHE, R2		
04	A2		01 CE 0000A	MNEGL	#1, EXT_CACHE		: 1742
08	A2		01 CE 0000E	MNEGL	#1, FID_CACHE		: 1743
		44	A2 9F 00012	MNEGL	#1, QUO_CACHE		: 1744
		0000'	CF 9F 00015	PUSHAB	CLI_DESC		: 1749
00000000G	00		02 FB 00019	PUSHAB	CACHE_DESC		
	32		50 E9 00020	CALLS	#2, CLISGET_VALUE		
58	A2	44	A2 3C 00023	BLBC	R0, 2\$		: 1751
5C	A2	48	A2 D0 00028	MOVZWL	CLI_DESC, TPARSE_BLOCK+8		: 1752
		00000000G	00 9F 0002D	MOVL	CLI_DESC+4, TPARSE_BLOCK+12		: 1753
		00000000G	00 9F 00033	PUSHAB	CACHE_KTB		
		50	A2 9F 00039	PUSHAB	CACHE_STB		
00000000G	00		03 FB 0003C	PUSHAB	TPARSE_BLOCK		
	CC		50 E8 00043	CALLS	#3, LIB\$TPARSE		
		007281C4	8F DD 00046	BLBS	R0, 1\$		
00000000G	00		01 FB 0004C	PUSHL	#7504324		: 1755
			BD 11 00053	CALLS	#1, LIB\$STOP		
			62 D5 00055	BRB	1\$		: 1749
			05 12 00057	TSTL	EXT_CACHE		: 1766
E9	A2	80	8F 88 00059	BNEQ	3\$		
	01	04	A2 D1 0005E	BISB2	#128, MOUNT_OPTIONS+5		: 1768
			04 12 00062	CMPL	FID_CACHE, #1		: 1770
EA	A2		01 88 00064	BNEQ	4\$		
		08	A2 D5 00068	BISB2	#1, MOUNT_OPTIONS+6		: 1772
			04 12 0006B	TSTL	QUO_CACHE		: 1774
EA	A2		02 88 0006D	BNEQ	5\$		
			04 00071	BISB2	#2, MOUNT_OPTIONS+6		: 1776
				RET			: 1779

: Routine Size: 114 bytes, Routine Base: \$CODE\$ + 088D

```
1127 1780 1 ROUTINE DATACHECK_ACT : NOVALUE =
1128 1781 2 BEGIN
1129 1782 2
1130 1783 2 EXTERNAL
1131 1784 2     DATACHECK_STB : VECTOR [0]; ! state table address
1132 1785 2     DATACHECK_KTB : VECTOR [0]; ! keyword table address
1133 1786 2
1134 1787 2 EXTERNAL ROUTINE
1135 1788 2     LIB$TPARSE;
1136 1789 2
1137 1790 2 LOCAL
1138 1791 2     VALUE_FOUND; ! set when value present
1139 1792 2
1140 1793 2 ! Parse the DATACHECK options string.
1141 1794 2 !
1142 1795 2
1143 1796 2 VALUE_FOUND = 0;
1144 1797 2
1145 1798 2 WHILE CLISGET_VALUE (DATA_DESC, CLI_DESC) DO
1146 1799 2 BEGIN
1147 1800 2     TPARSE_BLOCK[TPASL_STRINGCNT] = .CLI_DESC[DSCSW_LENGTH];
1148 1801 2     TPARSE_BLOCK[TPASL_STRINGPTR] = .CLI_DESC[DSCSA_POINTER];
1149 1802 2     IF NOT LIB$TPARSE (TPARSE_BLOCK, DATACHECK_STB, DATACHECK_KTB)
1150 1803 2     THEN
1151 1804 2         ERR_EXIT (MOUN$_BADDATCHK);
1152 1805 2     VALUE_FOUND = 1;
1153 1806 2
1154 1807 2 END;
1155 1808 2 ! If the qualifier /DATA_CHECK was specified with no value, then
1156 1809 2 ! WRITE data check is the default. Set the corresponding bit.
1157 1810 2 !
1158 1811 2 IF .VALUE_FOUND EQL 0
1159 1812 2 THEN
1160 1813 2     MOUNT_OPTIONS [OPT_WRITECHECK] = 1;
1161 1814 2
1162 1815 2
1163 1816 1 END; ! end of routine DATACHECK_ACT
```

.EXTRN DATACHECK_STB, DATACHECK_KTB

000C 00000 DATACHECK ACT:						
	53	0000'	CF 9E 00002	WORD	Save R2,R3	1780
			52 D4 00007	MOVAB	CLI_DESC, R3	
			53 DD 00009	CLRL	VALUE_FOUND	1796
				PUSHL	R3	1798
		0000'	CF 9F 0000B	PUSHAB	DATA_DESC	
00000000G	00		02 FB 0000F	CALLS	#2, CLISGET_VALUE	
	34		50 E9 00016	BLBC	R0, 3\$	
14	A3		63 3C 00019	MOVZWL	CLI_DESC, TPARSE_BLOCK+8	1800
18	A3	04	A3 D0 0001D	MOVL	CLI_DESC+4, TPARSE_BLOCK+12	1801
		00000000G	00 9F 00022	PUSHAB	DATACHECK_KTB	1802
		00000000G	00 9F 00028	PUSHAB	DATACHECK_STB	
		0C	A3 9F 0002E	PUSHAB	TPARSE_BLOCK	
00000000G	00		03 FB 00031	CALLS	#3, LIB\$TPARSE	
	0D		50 E8 00038	BLBS	R0, 2\$	

MOUNTIMG
V04-000

N 7
16-Sep-1984 01:06:29 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:45:31 [MOUNT.SRC]MOUNTIMG.B32;1

Page 42
(10)

00000000G	00	0072800C	8F	DD	00038	PUSHL	#7503884	:	1804	
	52		01	FB	00041	CALLS	#1, LIB\$STOP	:		
			01	D0	00048	2\$:	MOVL	#1, VALUE_FOUND	:	1805
			BC	11	00048	BRB	1\$	:	1798	
			52	D5	0004D	3\$:	TSTL	VALUE_FOUND	:	1811
			04	12	0004F	BNEQ	4\$	:		
A4	A3		10	88	00051	BISB2	#16, MOUNT_OPTIONS+4	:	1813	
			04	00055	4\$:	RET		:	1816	

; Routine Size: 86 bytes, Routine Base: \$CODE\$ + 08FF

```
1165 1817 1 ROUTINE DENSITY_ACT : NOVALUE =
1166 1818 1
1167 1819 2 BEGIN
1168 1820 2
1169 1821 2 CLISGET_VALUE ( DENSITY_DESC, CLI_DESC );
1170 1822 2
1171 1823 3 IF NOT ( LIB$CVT_DTB ( .CLI_DESC [DSC$W_LENGTH],
1172 1824 3 .CLI_DESC [DSC$A_POINTER],
1173 1825 3 DENSITY ) )
1174 1826 2 THEN
1175 1827 2 ERR_EXIT (MOUN$_BADDENS);
1176 1828 2
1177 1829 2 SELECTONE .DENSITY OF
1178 1830 2 SET
1179 1831 2
1180 1832 2 [800] : MOUNT_OPTIONS [OPT_DENS_800] = 1;
1181 1833 2 [1600] : MOUNT_OPTIONS [OPT_DENS_1600] = 1;
1182 1834 2 [6250] : 1;
1183 1835 2 [OTHERWISE] : ERR_EXIT (MOUN$_BADDENS);
1184 1836 2
1185 1837 2 TES;
1186 1838 2
1187 1839 1 END;
```

000C 00000 DENSITY_ACT:						
	53	00000000G	00	9E 00002	.WORD Save R2,R3	1817
	52	0000'	CF	9E 00009	MOVAB LIB\$STOP, R3	
			52	DD 0000E	MOVAB CLI_DESC, R2	
		0000'	CF	9F 00010	PUSHL R2	1821
00000000G	00		02	FB 00014	PUSHAB DENSITY_DESC	
		D0	A2	9F 0001B	CALLS #2, CLISGET_VALUE	1823
		04	A2	DD 0001E	PUSHAB DENSITY	1824
	7E		62	3C 00021	PUSHL CLI_DESC+4	
00000000G	00		03	FB 00024	MOVZWL CLI_DESC, -(SP)	1823
	09		50	E8 0002B	CALLS #3, LIB\$CVT_DTB	
		00728014	8F	DD 0002E	BLBS R0, 1\$	
	63		01	FB 00034	PUSHL #7503892	1827
	50	D0	A2	D0 00037	CALLS #1, LIB\$STOP	
00000320	8F		50	D1 0003B	MOVCL DENSITY, R0	1829
			05	12 00042	CMPL R0, #800	1832
	A0	A2	02	88 00044	BNEQ 2\$	
				04 00048	BISB2 #2, MOUNT_OPTIONS	
				04 00048	RET	
00000640	8F		50	D1 00049	CMPL R0, #1600	1833
			05	12 00050	BNEQ 3\$	
	A5	A2	08	88 00052	BISB2 #8, MOUNT_OPTIONS+5	
				04 00056	RET	
0000186A	8F		50	D1 00057	CMPL R0, #6250	1834
			09	13 C005E	BEQL 4\$	
		00728014	8F	DD 00060	PUSHL #7503892	1835
	63		01	FB 00066	CALLS #1, LIB\$STOP	
			04	00069	RET	1839

MOUNTIMG
V04-000

⁸
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31

VAX-11 Bliss-32 V4.0-742
[MOUNT.SRC]MOUNTIMG.B32;1

Page 44
(11)

; Routine Size: 106 bytes, Routine Base: \$CODE\$ + 0955

; 1188 1840 1

MO
VO

.


```
1190 1841 1 ROUTINE GET_DEVICE : NOVALUE =
1191 1842 1
1192 1843 2 BEGIN
1193 1844 2
1194 1845 2 DEVICE_COUNT = 0;
1195 1846 2
1196 1847 2 WHILE CLI$GET_VALUE ( $DESCRIPTOR('DEVICES'), CLI_DESC )
1197 1848 2 DO
1198 1849 2 BEGIN
1199 1850 2
1200 1851 2 BIND
1201 1852 2     DEVICE_DESC = DEVICE_STRING [ .DEVICE_COUNT * 2 ] : $BBLOCK;
1202 1853 2
1203 1854 2 IF .DEVICE_COUNT GEQ DEVMAX
1204 1855 2 THEN
1205 1856 2     ERR_EXIT ( MOUN$MAXDEV );
1206 1857 2
1207 1858 2 CH$FILL ( 0, DSC$S_BLN, DEVICE_DESC );
1208 1859 2 DEVICE_DESC [ DSC$B_DTYPE ] = DSC$R_DTYPE_T;
1209 1860 2 DEVICE_DESC [ DSC$B_CLASS ] = DSC$K_CLASS_D;
1210 1861 2 STR$COPY DX ( DEVICE_DESC, CLI_DESC );
1211 1862 2 DEVICE_COUNT = .DEVICE_COUNT + 1;
1212 1863 2 END;
1213 1864 1 END;
```

! of routine GET_DEVICE

.PSECT \$SPLITS, NOWRT, NOEXE, 2

```
53 45 43 49 56 45 44 00228 P.ACN: .ASCII \DEVICES\
                                0022F .BLKB 1
                                00000007, 00230 P.ACM: .LONG 7
                                00000000, 00234 .ADDRESS P.ACN
```

.PSECT \$CODE\$, NOWRT, 2

```
00FC 00000 GET_DEVICE:
57 0000' CF 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7
                                MOVAB DEVICE_COUNT, R7
                                CLRL DEVICE_COUNT
                                0170 C7 9F 00009 1$: PUSHAB CLI_DESC
                                0000' CF 9F 0000D PUSHAB P.ACM
                                00000000G 00 02 FB 00011 CALLS #2, CLI$GET_VALUE
                                38 50 E9 00018 BLBC R0, 3$
50 67 01 78 0001B ASHL #1, DEVICE_COUNT, R0
56 08 A740 DE 0001F MOVAL DEVICE_STRING[R0], R6
10 67 D1 00024 CMPL DEVICE_COUNT, #16
                                0D 19 00027 BLSS 2$
                                00728084 8F DD 00029 PUSHL #7504004
                                00 01 FB 0002F CALLS #1, LIB$STOP
08 00 00000000G 00 00 2C 00036 2$: MOVCS #0, (SP), #0, #8, (R6)
                                66 0003B
                                02 A6 020E 8F B0 0003C MOVW #526, 2(R6)
                                0170 C7 9F 00042 PUSHAB CLI_DESC
                                56 DD 00046 PUSHL R6
```

MOUNTING
V04-000

E 8
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31

VAX-11 Bliss-32 V4.0-742
[MOUNT.SRC]MOUNTING.B32;1

Page 46
(12)

00000000G 00

02 FB 00048
67 D6 0004F
86 11 00051
04 00053 3\$:

CALLS #2, STR\$COPY-DX
INCL DEVICE_COUNT-
BRB 1\$
RET

: 1862
: 1847
: 1864

; Routine Size: 84 bytes, Routine Base: \$CODE\$ + 09BF

MO
VO


```
1215 1865 1 ROUTINE GET_LABEL : NOVALUE =
1216 1866 1
1217 1867 2 BEGIN
1218 1868 2
1219 1869 2 LABEL_COUNT = 0;
1220 1870 2
1221 1871 2 WHILE CLI$GET_VALUE ( $DESCRIPTOR('VOLUMES'), CLI_DESC )
1222 1872 2 DO
1223 1873 2 BEGIN
1224 1874 2
1225 1875 2 BIND
1226 1876 2 LABEL_DESC = LABEL_STRING [ LABEL_COUNT * 2 ] : $BLOCK;
1227 1877 2
1228 1878 2 IF LABEL_COUNT GEQ LABMAX
1229 1879 2 THEN
1230 1880 2 ERR_EXIT ( MOUN$MAXLAB );
1231 1881 2
1232 1882 2 CH$FILL ( 0, DSC$S_BLN, LABEL_DESC );
1233 1883 2 LABEL_DESC [DSC$B_DTYPE] = DSC$R_DTYPE_T;
1234 1884 2 LABEL_DESC [DSC$B_CLASS] = DSC$R_CLASS_D;
1235 1885 2 STR$COPY DX ( LABEL_DESC, CLI_DESC );
1236 1886 2 LABEL_COUNT = LABEL_COUNT + 1;
1237 1887 2 END;
1238 1888 1 END; ! of routine GET_LABEL
```

.PSECT \$SPLITS, NOWRT, NOEXE, 2

```
53 45 4D 55 4C 4F 56 00238 P.ACP: .ASCII \VOLUMES\
0023F .BLKB 1
00000007 00240 P.ACO: .LONG 7
00000000 00244 .ADDRESS P.ACP
```

.PSECT \$CODE\$, NOWRT, 2

```
00FC 00000 GET_LABEL:
57 0000' CF 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7
67 D4 00007 MOVAB LABEL_COUNT, R7
016C C7 9F 00009 1$: CLRL LABEL_COUNT
0000' CF 9F 0000D PUSHAB CLI_DESC
00000000G 00 02 FB 00011 PUSHAB P.ACO
39 50 E9 00018 CALLS #2, CLI$GET_VALUE
50 67 01 78 0001B BLBC R0, 3$
56 0084 C740 DE 0001F ASHL #1, LABEL_COUNT, R0
10 67 D1 00025 MOVAL LABEL_STRING[R0], R6
00 0D 19 00028 CMPL LABEL_COUNT, #16
0072808C 8F DD 0002A BLSS 2$
00000000G 00 01 FB 00030 PUSHL #7504012
08 00 00 2C 00037 2$: CALLS #1, LIB$STOP
6E 66 0003C MOVCS #0, (SP), #0, #8, (R6)
02 A6 020E 8F B0 0003D MOVW #526, 2(R6)
016C C7 9F 00043 PUSHAB CLI_DESC
56 DD 00047 PUSHL R6
```

MOUNTIMG
V04-000

6 8
16-Sep-1984 01:06:29 VAX-11 B11ss-32 V4.0-742
14-Sep-1984 12:43:31 [MOUNT.SRC]MOUNTIMG.B32;1

Page 48
(13)

00000000G 00

02	FB	00049	CALLS	#2, STR\$COPY_DX
67	D6	00050	INCL	LABEL_COUNT
B5	11	00052	BRB	1\$
04	00054	3\$:	RET	

: 1886
: 1871
: 1888

; Routine Size: 85 bytes, Routine Base: \$CODE\$ + 0A13

```

1240 1889 1 ROUTINE GET_LOG_NAME: NOVALUE =
1241 1890 1
1242 1891 2 BEGIN
1243 1892 2
1244 1893 2 LOCAL
1245 1894 2 P; ! string scan pointer
1246 1895 2
1247 1896 2 ! Copy the logical name descriptor
1248 1897 2
1249 1898 2
1250 1899 2 IF CLISGET_VALUE ( $DESCRIPTOR('LOGNAMES'), CLI_DESC )
1251 1900 2 THEN
1252 1901 2 BEGIN
1253 1902 2 MOUNT OPTIONS [OPT_LOG_NAME] = 1;
1254 1903 2 CH$FILL ( 0, DSC$S_BCN, LOG_NAME );
1255 1904 2 LOG_NAME [DSC$B_DTYPE] = DSC$R_DTYPE_T;
1256 1905 2 LOG_NAME [DSC$B_CLASS] = DSC$K_CLASS_D;
1257 1906 2 STR$COPY_DX ( LOG_NAME, CLI_DESC );
1258 1907 2
1259 1908 2 ! If logical name is greater than maximum size, return error.
1260 1909 2
1261 1910 2
1262 1911 2 IF .LOG_NAME [DSC$W_LENGTH] GTR (LOG$C_NAMLENGTH - 1)
1263 1912 2 THEN
1264 1913 2 ERR_EXIT ( MOUN$_LOGNAME );
1265 1914 2
1266 1915 2 ! Scan for a trailing of embedded colon. If found, use string preceding
1267 1916 2 the colon.
1268 1917 2
1269 1918 2 P = CH$FIND_CH ( .LOG_NAME [DSC$W_LENGTH], .LOG_NAME [DSC$A_POINTER], ':' );
1270 1919 2
1271 1920 2 IF NOT CH$FAIL (.P)
1272 1921 2 THEN
1273 1922 2 LOG_NAME [DSC$W_LENGTH] = .P - .LOG_NAME [DSC$A_POINTER];
1274 1923 2
1275 1924 2 END;
1276 1925 1 END; ! end of routine LOG_NAME_ACT

```

```

53 45 4D 41 4E 47 4F 4C 00248 P.ACR: .ASCII \LOGNAME\
00000008, 00250 P.ACQ: .LONG 8
00000000, 00254 .ADDRESS P.ACR

```

				.PSECT	\$CODE\$,NOWRT,2
007C 00000 GET_LOG_NAME:					
				.WORD	Save R2,R3,R4,R5,F.6
56	0000'	CF 9E 00002		MOVAB	LOG_NAME, R6
	68	A6 9F 00007		PUSHAB	CLI_DESC
	0000'	CF 9F 0000A		PUSHAB	P.ACQ
00000000G	00	02 FB 0000E		CALLS	#2, CLI\$GET_VALUE
	40	50 E9 00015		BLBC	R0, 3\$

1889
1899

MOUNTING
V04-000

16-Sep-1984 01:06:29 VAX-11 BLISS-32 V4.0-742
14-Sep-1984 12:45:31 [MOUNT.SRC]MOUNTING.B32;1

Page 50
(14)

08	00	0B	A6	20	88	00018	BISB2	#32, MOUNT_OPTIONS+3	1902
		6E		00	2C	0001C	MOVCS	#0, (SP), #0, #8, LOG_NAME	1903
				66		00021			
		02	A6	8F	B0	00022	MOVW	#526, LOG_NAME+2	1904
			020E	A6	9F	00028	PUSHAB	CLI_DESC	1906
			68	56	DD	0002B	PUSHL	R6	
	00000000G	00		02	FB	0002D	CALLS	#2, STR\$COPY_DX	
		3F		66	B1	00034	CMPL	LOG_NAME, #63	1911
				0D	1B	00037	BLEQU	1\$	
			0072807C	8F	DD	00039	PUSHL	#7503996	1913
	00000000G	00		01	FB	0003F	CALLS	#1, LIB\$STOP	
04	B6	66		3A	3A	00046	LOCC	#58, LOG_NAME, @LOG_NAME+4	1918
				02	12	0004B	BNEQ	2\$	
				51	D4	0004D	CLRL	R1	
				51	D5	0004F	TSTL	P	1920
				05	13	00051	BEQL	3\$	
	66	51	04	A6	A3	00053	SUBW3	LOG_NAME+4, P, LOG_NAME	1922
				04	00	00053	RET		1925

; Routine Size: 89 bytes, Routine Base: \$CODE\$ + 0A68

```
: 1278      1926 1 ROUTINE INITIALIZE_ACT : NOVALUE =
: 1279      1927 1
: 1280      1928 2 BEGIN
: 1281      1929 2
: 1282      1930 2 EXTERNAL
: 1283      1931 2     INITIALIZE_STB : VECTOR [0],      ! state table address
: 1284      1932 2     INITIALIZE_KTB : VECTOR [0];    ! keyword table address
: 1285      1933 2
: 1286      1934 2 EXTERNAL ROUTINE
: 1287      1935 2     LIB$TPARSE;
: 1288      1936 2
: 1289      1937 2 ! Parse the INITIALIZE string and set appropriate flags.
: 1290      1938 2 !
: 1291      1939 2
: 1292      1940 2 WHILE CLISGET_VALUE ( INITIALIZE_DESC, CLI_DESC ) DO
: 1293      1941 2 BEGIN
: 1294      1942 2     TPARSE_BLOCK[TPASL_STRINGCNT] = .CLI_DESC[DSCSW_LENGTH];
: 1295      1943 2     TPARSE_BLOCK[TPASL_STRINGPTR] = .CLI_DESC[DSCSA_POINTER];
: 1296      1944 2     IF NOT LIB$TPARSE (TPARSE_BLOCK, INITIALIZE_STB, INITIALIZE_KTB)
: 1297      1945 2     THEN
: 1298      1946 2         ERR_EXIT (MOUN$BADINIT);
: 1299      1947 2     END;
: 1300      1948 2
: 1301      1949 1 END;
```

.EXTRN INITIALIZE_STB, INITIALIZE_KTB

0004 00000 INITIALIZE ACT:						
	52	0000'	CF 9E 00002	.WORD	Save R2	1926
			52 DD 00007	MOVAB	CLI_DESC, R2	
		0000'	CF 9F 00009	PUSHL	R2	1940
00000000G	00		02 FB 0000D	PUSHAB	INITIALIZE_DESC	
	31		50 E9 00014	CALLS	#2, CLISGET_VALUE	
14 A2			62 3C 00017	BLBC	R0, 2\$	1942
18 A2	04		A2 D0 0001B	MOVZWL	CLI_DESC, TPARSE_BLOCK+8	
		00000000G	00 9F 00020	MOVL	CLI_DESC+4, TPARSE_BLOCK+12	1943
		00000000G	00 9F 00026	PUSHAB	INITIALIZE_KTB	1944
		0C	A2 9F 0002C	PUSHAB	INITIALIZE_STB	
00000000G	00		03 FB 0002F	PUSHAB	TPARSE_BLOCK	
	CE		50 E8 00036	CALLS	#3, LIB\$TPARSE	
		00728224	8F DD 00039	BLBS	R0, 1\$	
00000000G	00		01 FB 0003F	PUSHL	#7504420	1946
			BF 11 00046	CALLS	#1, LIB\$STOP	
			04 00048	BRB	1\$	1940
				RET		1949

; Routine Size: 73 bytes, Routine Base: \$CODE\$ + 0AC1

```
1303 1950 1 ROUTINE JOURNAL_ACT : NOVALUE =
1304 1951 2 BEGIN
1305 1952 3
1306 1953 4 LITERAL
1307 1954 5     MOUNT$K_DEF_JRNL_RECORD_SIZE = 600; ! Default value for max record size
1308 1955 6
1309 1956 7 EXTERNAL
1310 1957 8     JOURNAL_STB      : VECTOR [0];    ! state table address
1311 1958 9     JOURNAL_KTB      : VECTOR [0];    ! keyword table address
1312 1959 10
1313 1960 11 EXTERNAL ROUTINE
1314 1961 12     LIB$TPARSE;
1315 1962 13
1316 1963 14 ! Parse the journal control options and set appropriate flags.
1317 1964 15
1318 1965 16 MOUNT_OPTIONS [OPT_NOJRNL] = 0;
1319 1966 17
1320 1967 18 WHILE CL$GET_VALUE ( JOURNAL_DESC, CLI_DESC ) DO
1321 1968 19 BEGIN
1322 1969 20     TPARSE_BLOCK[TPASL_STRINGCNT]      = .CLI_DESC[DSCSW_LENGTH];
1323 1970 21     TPARSE_BLOCK[TPASL_STRINGPTR]      = .CLI_DESC[DSCSA_POINTER];
1324 1971 22     IF NOT LIB$TPARSE (TPARSE_BLOCK, JOURNAL_STB, JOURNAL_KTB)
1325 1972 23     THEN
1326 1973 24         ERR_EXIT (MOUN$_BADJRNL);
1327 1974 25
1328 1975 26 END;
1329 1976 27
1330 1977 28 ! If this is a MOUNT/JOURNAL=NEWFILE, then make sure RECORD_SIZE has a value.
1331 1978 29 ! Otherwise, ensure that no values were specified for journal creation
1332 1979 30 ! keywords.
1333 1980 31
1334 1981 32 IF .MOUNT_OPTIONS [OPT_NEWJRNL]
1335 1982 33 THEN
1336 1983 34 BEGIN
1337 1984 35     IF .JRNL_RECORD_SIZE EQL 0
1338 1985 36     THEN
1339 1986 37         JRNL_RECORD_SIZE = MOUNT$K_DEF_JRNL_RECORD_SIZE
1340 1987 38     END
1341 1988 39 ELSE IF ((.JRNL_SIZE NEQ 0) OR (.JRNL_RECORD_SIZE NEQ 0) OR (.JRNL_EXTEND NEQ 0)
1342 1989 40     OR (.JRNL_QUOTA NEQ 0))
1343 1990 41 THEN
1344 1991 42     ERR_EXIT (MOUN$_BADJRNL);
1345 1992 43
1345 1992 44 ! end of routine JOURNAL_ACT
```

.EXTRN JOURNAL_STB, JOURNAL_KTB

```
000C 00000 JOURNAL_ACT:
      53 00000000G 00 9E 00002 .WORD Save R2,R3
      52 0000'   CF 9E 00009 MOVAB LIB$STOP, R3
      C2 A2 80 8F 8A 0000E MOVAB JRNL_RECORD_SIZE, R2
      1C A2 9F 00013 1$: BICB2 #128, MOUNT_OPTIONS+6
      0000'   CF 9F 00016 PUSHAB CLI_DESC
      00000000G 00 02 FB 0001A PUSHAB JOURNAL_DESC
      2E 50 E9 00021 CALLS #2, CL$GET_VALUE
      BLBC R0, 2$
```

1950
1965
1967

30	A2	1C	A2	3C	00024	MOVZWL	CLI-DESC, TPARSE_BLOCK+8	1969
34	A2	20	A2	D0	00029	MOVL	CLI-DESC+4, TPARSE_BLOCK+12	1970
		00000000G	00	9F	0002E	PUSHAB	JOURNAL_KTB	1971
		00000000G	00	9F	00034	PUSHAB	JOURNAL-STB	
		28	A2	9F	0003A	PUSHAB	TPARSE_BLOCK	
00000000G	00		03	FB	0003D	CALLS	#3, LIB\$TPARSE	
	CC		50	E8	00044	BLBS	R0, 1\$	
		00728214	8F	DD	00047	PUSHL	#7504404	1973
	63		01	FB	0004D	CALLS	#1, LIB\$STOP	
			C1	11	00050	BRB	1\$	1967
	0A	C3	A2	E9	00052	BLBC	MOUNT_OPTIONS+7, 3\$	1980
			62	D5	00056	TSTL	JRNL_RECORD_SIZE	1983
			22	12	00058	BNEQ	5\$	
	62	0258	8F	3C	0005A	MOVZWL	#600, JRNL_RECORD_SIZE	1985
				04	0005F	RET		1982
		FC	A2	D5	00060	TSTL	JRNL_SIZE	1987
			0E	12	00063	BNEQ	4\$	
			62	D5	00065	TSTL	JRNL_RECORD_SIZE	
			0A	12	00067	BNEQ	4\$	
		F8	A2	D5	00069	TSTL	JRNL_EXTEND	
			05	12	0006C	BNEQ	4\$	
		F4	A2	D5	0006E	TSTL	JRNL_QUOTA	1988
			09	13	00071	BEQL	5\$	
		00728214	8F	DD	00073	PUSHL	#7504404	1990
	63		01	FB	00079	CALLS	#1, LIB\$STOP	
			04	0007C	5\$:	RET		1992

; Routine Size: 125 bytes, Routine Base: \$CODE\$ + 0B0A

```
1347 1993 1 ROUTINE OVERRIDE_ACT : NOVALUE =
1348 1994 2 BEGIN
1349 1995 2
1350 1996 2 EXTERNAL
1351 1997 2     OVERRIDE_STB      : VECTOR [0],      ! state table address
1352 1998 2     OVERRIDE_KTB      : VECTOR [0];    ! keyword table address
1353 1999 2
1354 2000 2 EXTERNAL ROUTINE
1355 2001 2     LIB$TPARSE;
1356 2002 2
1357 2003 2 ! Parse the OVERRIDE string and set appropriate flags.
1358 2004 2 !
1359 2005 2
1360 2006 2 WHILE CLISGET_VALUE ( OVERRIDE_DESC, CLI_DESC ) DO
1361 2007 3 BEGIN
1362 2008 3     TPARSE_BLOCK[TPASL_STRINGCNT] = .CLI_DESC[DSCSW_LENGTH];
1363 2009 3     TPARSE_BLOCK[TPASL_STRINGPTR] = .CLI_DESC[DSCSA_POINTER];
1364 2010 3     IF NOT LIB$TPARSE (TPARSE_BLOCK, OVERRIDE_STB, OVERRIDE_KTB)
1365 2011 3     THEN
1366 2012 3         ERR_EXIT (MOUN$_BADOVR);
1367 2013 2 END;
1368 2014 2
1369 2015 1 END;                                     ! end of routine OVERRIDE_ACT
```

```
                                .EXTRN  OVERRIDE_STB, OVERRIDE_KTB
                                0004 00000 OVERRIDE_ACT:
                                .WORD    Save R2
                                52      0000' CF 9E 00002 MOVAB  CLI_DESC, R2
                                52      0000' DD 00007 1$:  PUSHL  R2
                                00000000G 00      0000' CF 9F 00009 PUSHAB OVERRIDE_DESC
                                14      31      02 FB 0000D CALLS  #2, CLISGET_VALUE
                                18      A2      50 E9 00014 BLBC  R0, 2$
                                04      A2      62 3C 00017 MOVZWL CLI_DESC, TPARSE_BLOCK+8
                                00000000G 00      00 9F 0001B MOVL  CLI_DESC+4, TPARSE_BLOCK+12
                                00000000G 00      00 9F 00020 PUSHAB OVERRIDE_KTB
                                0C      A2      00 9F 00026 PUSHAB OVERRIDE_STB
                                00000000G 00      03 FB 0002C PUSHAB TPARSE_BLOCK
                                CE      50      03 FB 0002F CALLS  #3, LIB$TPARSE
                                0072816C 8F DD 00036 BLBS  R0, 1$
                                00000000G 00      01 FB 0003F PUSHL  #7504236
                                01      BF      11 00046 CALLS  #1, LIB$STOP
                                04      04 00048 2$:  BRB   1$
                                RET
```

; Routine Size: 73 bytes, Routine Base: \$CODE\$ + 0987

1993
2006
2008
2009
2010
2012
2006
2015


```
: 1371      2016 1 ROUTINE OWNER_UIC_ACT : NOVALUE =
: 1372      2017 2 BEGIN
: 1373      2018 2
: 1374      2019 2 EXTERNAL
: 1375      2020      UIC_STB      : VECTOR [0],      ! state table address
: 1376      2021      UIC_KTB      : VECTOR [0];      ! keyword table address
: 1377      2022 2
: 1378      2023 2 EXTERNAL ROUTINE
: 1379      2024 2     LIB$PARSE;
: 1380      2025 2
: 1381      2026 2 ! Parse the UIC string and store it in the owner UIC longword.
: 1382      2027 2 !
: 1383      2028 2
: 1384      2029 2 WHILE CLISGET_VALUE ( OWNER_DESC, CLI_DESC ) DO
: 1385      2030 2 BEGIN
: 1386      2031 3     TPARSE_BLOCK[TPASL_STRINGCNT] = .CLI_DESC[DSCSW_LENGTH];
: 1387      2032 3     TPARSE_BLOCK[TPASL_STRINGPTR] = .CLI_DESC[DSCSA_POINTER];
: 1388      2033 3     IF NOT LIB$PARSE (TPARSE_BLOCK, UIC_STB, UIC_KTB)
: 1389      2034 3     THEN
: 1390      2035 3         ERR_EXIT (MOUN$_BADUIC);
: 1391      2036 2 END;
: 1392      2037 2
: 1393      2038 2 OWNER_UIC = .UIC;
: 1394      2039 2
: 1395      2040 1 END;                                     ! end of routine OWNER_UIC_ACT
```

```
                                .EXTRN  UIC_STB, UIC_KTB
                                0004 0000 OWNER_UIC_ACT:
                                .WORD    Save R2
                                52      0000' CF 9E 00002 1$: MOVAB  CLI_DESC, R2
                                52      0000' CF 9F 00009 1$: PUSHL  R2
                                00000000G 00      02 FB 0000D 1$: PUSHAB OWNER_DESC
                                14      31      50 E9 00014 1$: CALLS  #2, CLISGET_VALUE
                                18      A2      62 3C 00017 1$: BLBC   R0, 2$
                                04      A2      62 3C 00017 1$: MOVZWL CLI_DESC, TPARSE_BLOCK+8
                                00000000G 00      00 9F 00020 1$: MOVL   CLI_DESC+4, TPARSE_BLOCK+12
                                00000000G 00      00 9F 00026 1$: PUSHAB UIC_KTB
                                0C      A2      03 FB 0002F 1$: PUSHAB UIC_STB
                                00000000G 00      03 FB 0002F 1$: PUSHAB TPARSE_BLOCK
                                CE      50      8F DD 00039 1$: CALLS  #3, LIB$PARSE
                                00728024 8F DD 00039 1$: BLBS   R0, 1$
                                00000000G 00      01 FB 0003F 1$: PUSHL  #7503908
                                E8      A2      30 A2 D0 00048 2$: CALLS  #1, LIB$STOP
                                BF      11 00046 2$: BRB    1$
                                04      04 0004D 2$: MOVL   UIC, OWNER_UIC
                                RET
```

; Routine Size: 78 bytes. Routine Base: \$CODE\$ + 0BD0

```
: 1397      2041 1
: 1398      2042 1 ROUTINE PROCESSOR_ACT : NOVALUE =
: 1399      2043 2 BEGIN
: 1400      2044 2
: 1401      2045 2 EXTERNAL
: 1402      2046 2     PROCESSOR_STB   : VECTOR [0],   ! state table address
: 1403      2047 2     PROCESSOR_KTB  : VECTOR [0];   ! keyword table address
: 1404      2048 2
: 1405      2049 2 EXTERNAL ROUTINE
: 1406      2050 2     LIB$PARSE;
: 1407      2051 2
: 1408      2052 2 ! Parse the PROCESSOR switch options (leaving values and bits set).
: 1409      2053 2 !
: 1410      2054 2
: 1411      2055 2 CL$GET VALUE ( PROCESSOR_DESC, CLI_DESC );
: 1412      2056 2 TPARSE_BLOCK[TPASL_STRINGCNT] = .CLI_DESC[DSCSW_LENGTH];
: 1413      2057 2 TPARSE_BLOCK[TPASL_STRINGPTR] = .CLI_DESC[DSCSA_POINTER];
: 1414      2058 2
: 1415      2059 2 IF NOT LIB$PARSE (TPARSE_BLOCK, PROCESSOR_STB, PROCESSOR_KTB)
: 1416      2060 2 THEN
: 1417      2061 2     ERR_EXIT (MOUN$_BADACP);
: 1418      2062 2
: 1419      2063 1 END;                                ! end of routine PROCESSOR_ACT
```

```
                                .EXTRN  PROCESSOR_STB, PROCESSOR_KTB
                                0004 0000 PROCESSOR_ACT:
                                .WORD    Save R2
                                52      0000' CF 9E 00002 MOVAB CLI_DESC, R2
                                52      0000' DD 00007 PUSHL R2
                                00000000G 00 0000' CF 9F 00009 PUSHAB PROCESSOR_DESC
                                14      A2      02 FB 0000D CALLS #2, CL$GET VALUE
                                18      A2      62 3C 00014 MOVZWL CLI_DESC, TPARSE_BLOCK+8
                                00000000G 00 04      A2 D0 00018 MOVL CLI_DESC+4, TPARSE_BLOCK+12
                                00000000G 00 9F 0001D PUSHAB PROCESSOR_KTB
                                0C      A2      9F 00023 PUSHAB PROCESSOR_STB
                                00000000G 00 0C      9F 00029 PUSHAB TPARSE_BLOCK
                                0D      03 FB 0002C CALLS #3, LIB$PARSE
                                0072815C 50 E8 00033 BLBS R0, 1$
                                00000000G 00 8F DD 00036 PUSHL #7504220
                                01 FB 0003C CALLS #1, LIB$STOP
                                04 00043 1$: RET
```

; Routine Size: 68 bytes, Routine Base: \$CODE\$ + 0C1E

```
: 1421 2064 1 ROUTINE PROTECTION_ACT : NOVALUE =
: 1422 2065 1
: 1423 2066 2 BEGIN
: 1424 2067 2
: 1425 2068 2 EXTERNAL
: 1426 2069 2 PROTECTION_STB : VECTOR [0], ! state table address
: 1427 2070 2 PROTECTION_KTB : VECTOR [0]; ! keyword table address
: 1428 2071 2
: 1429 2072 2 EXTERNAL ROUTINE
: 1430 2073 2 LIB$TPARSE;
: 1431 2074 2
: 1432 2075 2 ! Parse the PROTECTION qualifier string storing the binary protection.
: 1433 2076 2 ! Complement thereafter, since the parser produces the complement.
: 1434 2077 2
: 1435 2078 2
: 1436 2079 2 WHILE CLISGET_VALUE ( PROTECTION_DESC, CLI_DESC ) DO
: 1437 2080 2 BEGIN
: 1438 2081 2 TPARSE_BLOCK[TPASL_STRINGCNT] = .CLI_DESC[DSCSW_LENGTH];
: 1439 2082 2 TPARSE_BLOCK[TPASL_STRINGPTR] = .CLI_DESC[DSCSA_POINTER];
: 1440 2083 2 IF NOT LIB$TPARSE (TPARSE_BLOCK, PROTECTION_STB, PROTECTION_KTB)
: 1441 2084 2 THEN
: 1442 2085 2 ERR_EXIT (MOUN$BADPRO);
: 1443 2086 2 END;
: 1444 2087 2
: 1445 2088 2 PROTECTION <0, 16> = NOT .PROTECTION <0, 16>;
: 1446 2089 2
: 1447 2090 1 END; ! end of routine PROTECTION_ACT
```

```
.EXTRN PROTECTION_STB, PROTECTION_KTB

0004 00000 PROTECTION_ACT:
      52      0000' CF 9E 00002 .WORD Save R2
      52      DD 00007 1$: MOVAB CLI_DESC, R2
      0000' CF 9F 00009 PUSHAB R2
      00000000G 00 02 FB 00000 PUSHAB PROTECTION_DESC
      31 50 E9 00014 CALLS #2, CLISGET_VALUE
      14 A2 62 3C 00017 BLBC R0, 2$
      18 A2 04 A2 D0 0001B MOVZWL CLI_DESC, TPARSE_BLOCK+8
      00000000G 00 00 9F 00020 MOVL CLI_DESC+4, TPARSE_BLOCK+12
      00000000G 00 00 9F 00026 PUSHAB PROTECTION_KTB
      0C A2 9F 0002C PUSHAB PROTECTION_STB
      00000000G 00 03 FB 0002F PUSHAB TPARSE_BLOCK
      CE 50 E8 00036 CALLS #3, LIB$TPARSE
      0072801C 8F DD 00039 GLBS R0, 1$
      00000000G 00 01 FB 0003F PUSHAB #7503900
      EC A2 EC A2 B2 00048 2$: CALLS #1, LIB$STOP
      04 0004D BRB 1$
      RET MCOMW PROTECTION, PROTECTION
      RET
```

; Routine Size: 78 bytes, Routine Base: \$CODE\$ + 0C62

```

: 1449      2091 1
: 1450      2092 1
: 1451      2093 1
: 1452      2094 1
: 1453      2095 1
: 1454      2096 1
: 1455      2097 1
: 1456      2098 1
: 1457      2099 1
: 1458      2100 1
: 1459      2101 1
: 1460      2102 1
: 1461      2103 2
: 1462      2104 2
: 1463      2105 2
: 1464      2106 2
: 1465      2107 2
: 1466      2108 1

```

TPARSE action routines for the following TPARSE tables.

Clear the 'NEW JOURNAL FILE' option bit. (We just saw NONEWFILE.)

ROUTINE CLEAR_NEWJRNL =

BEGIN

MOUNT_OPTIONS [OPT_NEWJRNL] = 0;

RETURN 1;

END;

```

0000 00000 CLEAR_NEWJRNL:
0000' CF 01 8A 00002 .WORD Save nothing
50 01 D0 00007 BICB2 #1, MOUNT_OPTIONS+7
04 0000A MOVL #1, R0
RET

```

```

: 2102
: 2105
: 2106
: 2108

```

; Routine Size: 11 bytes, Routine Base: \$CODE\$ + 0CB0

; 1467 2109 1

```
1469 2110 1 !
1470 2111 1 ! Store ACP string (either device name or file name).
1471 2112 1
1472 2113 1 ROUTINE GET_ACP_NAME =
1473 2114 1
1474 2115 2 BEGIN
1475 2116 2
1476 2117 2 LOCAL
1477 2118 2 ACP_DESC : BBLOCK [DSC$C_S_BLN];
1478 2119 2
1479 2120 2 TPARSE_ARGS (CONTEXT);
1480 2121 2
1481 2122 2 IF .CONTEXT[TPASL_TOKENCNT] GTR 20
1482 2123 2 THEN ERR_EXIT (MOONS_ACPNAME);
1483 2124 2
1484 2125 2 ! Initialize local descriptor and load values
1485 2126 2
1486 2127 2 CH$FILL ( 0, DSC$C_S_BLN, ACP_DESC);
1487 2128 2 ACP_DESC [DSC$B_DTYPE] = DSC$K_CLASS_D;
1488 2129 2 ACP_DESC [DSC$W_LENGTH] = .CONTEXT [TPASL_TOKENCNT];
1489 2130 2 ACP_DESC [DSC$A_POINTER] = .CONTEXT [TPASL_TOKENPTR];
1490 2131 2
1491 2132 2 ! Now, move values to the text descriptor. We need to use a temporary
1492 2133 2 ! ACP descriptor, because the CLI_DESC contains the keyword 'SAME:'
1493 2134 2 ! when that option is used.
1494 2135 2
1495 2136 2 CH$FILL ( 0, DSC$C_S_BLN, ACP_STRING );
1496 2137 2 ACP_STRING [DSC$B_DTYPE] = DSC$K_DTYPE_T;
1497 2138 2 ACP_STRING [DSC$B_CLASS] = DSC$K_CLASS_D;
1498 2139 2 STR$COPY_DX ( ACP_STRING, ACP_DESC );
1499 2140 2 RETURN 1;
1500 2141 2
1501 2142 1 END;

! end of routine GET_ACP_NAME
```

```
003C 00000 GET_ACP_NAME:
                                .WORD      Save R2,R3,R4,R5
                                SUBL2      #8, SP
                                CMPL      16(CONTEXT), #20
                                BLEQ      1$
                                PUSHL     #7504196
                                CALLS     #1, LIB$STOP
                                MOVCS     #0, (SP), #0, #8, ACP_DESC
                                MOVB      #2, ACP_DESC+2
                                MOVW      16(CONTEXT), ACP_DESC
                                MOVL      20(CONTEXT), ACP_DESC+4
                                MOVCS     #0, (SP), #0, #8, ACP_STRING
                                MOVW      #526, ACP_STRING+2
                                PUSHL     SP
                                PUSHAB   ACP_STRING
                                CALLS     #2, STR$COPY_DX
                                MOVL      #1, R0

08      00      00000000G      00      00728144      08      C2      00002      .WORD
                                14      10      AC      D1      00005      SUBL2
                                00      00      0D      15      00009      CMPL
                                00      00      8F      DD      0000B      BLEQ
                                00      00      01      FB      00011      PUSHL
                                00      2C      00018      1$:      CALLS
                                6E      00      0001D      MOVCS
                                02      AE      0001E      MOVB
                                6E      10      AC      B0      00022      MOVW
                                04      AE      14      AC      D0      00026      MOVL
                                00      00      00      2C      0002B      MOVCS
                                0000'      CF      00030      MOVW
                                0000'      CF      020E      8F      B0      00033      MOVW
                                0000'      CF      0003A      5E      DD      0003A      PUSHL
                                0000'      CF      0003C      9F      9F      0003C      PUSHAB
                                00000000G      00      02      FB      00040      CALLS
                                50      01      D0      00047      MOVL
```

MOUNTIMG
V04-000

F 9
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31

VAX-11 Bliss.32 V4.0-742
[MOUNT.SRC]MOUNTIMG.B32;1

Page 60
(22)

MO
VO

04 0004A

RET

; 2142

; Routine Size: 75 bytes, Routine Base: \$CODES + 0CBB


```
1503 2143 1 |
1504 2144 1 | Store ACP string as specified by the :SAME option.
1505 2145 1 | Append a ':' to the device name.
1506 2146 1 |
1507 2147 1 ROUTINE GET_SAME_ACP =
1508 2148 2 BEGIN
1509 2149 2
1510 2150 2 LOCAL
1511 2151 2 ACP_DESC : BBLOCK [DSC$C_S_BLN],
1512 2152 2 SAME_ACP : VECTOR [21,BYTE];
1513 2153 2
1514 2154 2 TPARSE_ARGS (CONTEXT);
1515 2155 2
1516 2156 2 IF .CONTEXT[TPASL_TOKENCNT] GTR 20
1517 2157 2 THEN ERR_EXIT (MOONS_ACPNAME);
1518 2158 2
1519 2159 2 | Add the colon (:) to the device name.
1520 2160 2
1521 2161 2 CHSMOVE (.CONTEXT[TPASL_TOKENCNT], .CONTEXT[TPASL_TOKENPTR], SAME_ACP);
1522 2162 2 SAME_ACP [.CONTEXT [TPASL_TOKENCNT]] = %ASCII ':';
1523 2163 2
1524 2164 2 | Initialize local descriptor and load values. The size of the device
1525 2165 2 | name has increased by 1, because of the colon that was added.
1526 2166 2
1527 2167 2 CHSFILL ( 0, DSC$C_S_BLN, ACP_DESC);
1528 2168 2 ACP_DESC [DSC$B_DTYPE] = DSC$K_CLASS_D;
1529 2169 2 ACP_DESC [DSC$W_LENGTH] = .CONTEXT [TPASL_TOKENCNT] + 1;
1530 2170 2 ACP_DESC [DSC$A_POINTER] = SAME_ACP;
1531 2171 2
1532 2172 2 | Now, move values to the text descriptor.
1533 2173 2
1534 2174 2 CHSFILL ( 0, DSC$C_S_BLN, ACP_STRING );
1535 2175 2 ACP_STRING [DSC$B_DTYPE] = DSC$K_DTYPE_T;
1536 2176 2 ACP_STRING [DSC$B_CLASS] = DSC$K_CLASS_D;
1537 2177 2 STRSCOPY_DX ( ACP_STRING, ACP_DESC );
1538 2178 2 RETURN 1;
1539 2179 2
1540 2180 1 END;
```

! end of routine GET_SAME_ACP

				003C 00000 GET_SAME_ACP:				
				20	C2 00002	WORD	Save R2,R3,R4,R5	: 2147
		5E		AC	D1 00005	SUBL2	#32, SP	
		14	10	OD	15 00009	CMPL	16(CONTEXT), #20	: 2156
				8F	DD 0000B	BLEQ	1\$	
			00728144	01	FB 00011	PUSHL	#7504196	: 2157
		00000000G	00	AC	28 00018	CALLS	#1, 'IB\$STOP	
	6E	14	BC	6E	9E 0001F	MOVCL	16(CONTEXT), @20(CONTEXT), SAME_ACP	: 2161
		50	10	3A	90 00021	MOVAB	SAME_ACP, R0	: 2162
		10 BC40		0C	2C 00026	MOVB	#58, @16(CONTEXT)[R0]	
08	00	6E		AE	0002B	MOVCL	#0, (SP). #0, #8, ACP_DESC	: 2167
			18	02	90 0002D	MOVB	#2, ACP_DESC+2	: 2168
	18	AE	10	01	A1 00031	ADDW3	#1, 16(CONTEXT), ACP_DESC	: 2169

MOUNTIMG
V04-000

H 9
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31

VAX-11 Bliss-32 V4.0-742
[MOUNT.SRC]MOUNTIMG.B32;1

Page 62
(23)

08	0C	1C	AE	6E	9E	00037
			6E	0C	2C	0003B
		0000'	CF	CF		00040
			020E	8F	B0	00043
			18	AE	9F	0004A
		0000'	CF	9F	0004D	
	00000000G		00	02	FB	00051
			50	01	D0	00058
				04	0005B	

MOVAB	SAME_ACP, ACP_DESC+4
MOVCS	#0, TSP), #0, #8, ACP_STRING
MOVW	#526, ACP_STRING+2
PUSHAB	ACP_DESC
PUSHAB	ACP_STRING
CALLS	#2, STR\$COPY_DX
MOVL	#1, R0
RET	

:	2170
:	2174
:	2175
:	2177
:	
:	2178
:	2180

; Routine Size: 92 bytes, Routine Base: \$CODE\$ + 0D06


```
1542 2181 1 1+
1543 2182 1
1544 2183 1 TPARSE state tables to parse the various qualifier value strings.
1545 2184 1
1546 2185 1 -
1547 2186 1
1548 2187 1
1549 2188 1 Parse /CACHE options (EXTENT=n, LIMIT=n, FILE_ID=n, QUOTA=n, NOEXTENT,
1550 2189 1 NOFILE_ID, NOQUOTA, and WRITETHROUGH).
1551 2190 1
1552 2191 1 $INIT_STATE (CACHE_STB, CACHE_KTB);
1553 2192 1
1554 P 2193 1 $STATE (NEXT_CACHE,
1555 P 2194 1 ('EXTENT', CACHE_EXT,, 1^(OPT_CACHE-32), MOUNT_OPTIONS+4),
1556 P 2195 1 ('FILE_ID', CACHE_FID,, 1^(OPT_CACHE-32), MOUNT_OPTIONS+4),
1557 P 2196 1 ('LIMIT', LIMIT_EXT),
1558 P 2197 1 ('NOEXTENT', 1^(OPT_NOEXT_C-32), MOUNT_OPTIONS+4),
1559 P 2198 1 ('NOFILE_ID', 1^(OPT_NOFID_C-32), MOUNT_OPTIONS+4),
1560 P 2199 1 ('NOQUOTA', 1^(OPT_NOQUO_C-32), MOUNT_OPTIONS+4),
1561 P 2200 1 ('NOWRITETHROUGH'),
1562 P 2201 1 ('QUOTA', CACHE_QUO,, 1^(OPT_CACHE-32), MOUNT_OPTIONS+4),
1563 P 2202 1 ('WRITETHROUGH', 1^(OPT_WTHRU-32), MOUNT_OPTIONS+4)
1564 2203 1 );
1565 2204 1
1566 P 2205 1 $STATE (END_CACHE,
1567 P 2206 1 ('NEXT_CACHE',
1568 P 2207 1 (TPAS_EOS, TPAS_EXIT)
1569 2208 1 );
1570 2209 1
1571 P 2210 1 $STATE (CACHE_EXT,
1572 P 2211 1 (''),
1573 P 2212 1 ('='),
1574 2213 1 );
1575 2214 1
1576 P 2215 1 $STATE (
1577 P 2216 1 (TPAS_DECIMAL, END_CACHE,,, EXT_CACHE)
1578 2217 1 );
1579 2218 1
1580 2219 1
1581 P 2220 1 $STATE (CACHE_FID,
1582 P 2221 1 (''),
1583 P 2222 1 ('='),
1584 2223 1 );
1585 2224 1
1586 P 2225 1 $STATE (
1587 P 2226 1 (TPAS_DECIMAL, END_CACHE,,, FID_CACHE)
1588 2227 1 );
1589 2228 1
1590 2229 1
1591 P 2230 1 $STATE (CACHE_QUO,
1592 P 2231 1 (''),
1593 P 2232 1 ('='),
1594 2233 1 );
1595 2234 1
1596 P 2235 1 $STATE (
1597 P 2236 1 (TPAS_DECIMAL, END_CACHE,,, QUO_CACHE)
1598 2237 1 );
```

```
1599      2238 1
1600      P 2239 1 $STATE (LIMIT_EXT,
1601      P 2240 1      ('='),
1602      P 2241 1      ('='),
1603      2242 1      );
1604      2243 1
1605      P 2244 1 $STATE (
1606      P 2245 1      (TPAS_DECIMAL, END_CACHE,,, EXT_LIMIT)
1607      2246 1      );
1608      2247 1
1609      2248 1
1610      2249 1 Parse /DATA_CHECK options, of the form [READ][WRITE]. Default is write.
1611      2250 1
1612      2251 1 $INIT_STATE (DATACHECK_STB, DATACHECK_KTB);
1613      2252 1
1614      P 2253 1 $STATE (
1615      P 2254 1      (TPAS_EOS, TPAS_EXIT,, 1^(OPT_WRITECHECK-32), MOUNT_OPTIONS+4),
1616      P 2255 1      (TPAS_LAMBDA)
1617      2256 1      );
1618      2257 1
1619      P 2258 1 $STATE (CHECKOPT,
1620      P 2259 1      ('READ',,, 1^(OPT_READCHECK-32), MOUNT_OPTIONS+4),
1621      P 2260 1      ('WRITE',,, 1^(OPT_WRITECHECK-32), MOUNT_OPTIONS+4)
1622      2261 1      );
1623      2262 1
1624      P 2263 1 $STATE (
1625      P 2264 1      ('CHECKOPT),
1626      P 2265 1      (TPAS_EOS, TPAS_EXIT)
1627      2266 1      );
1628      2267 1
1629      2268 1
1630      2269 1 Parse INITIALIZE options (ALL, CONTINUATION)
1631      2270 1
1632      2271 1
1633      2272 1 $INIT_STATE (INITIALIZE_STB, INITIALIZE_KTB);
1634      2273 1
1635      P 2274 1 $STATE (NEXTINI,
1636      P 2275 1      ('ALL',,, 1^(OPT_INIT_ALL-32), MOUNT_OPTIONS+4),
1637      P 2276 1      ('CONTINUATION',,, 1^(OPT_INIT_CONT-32), MOUNT_OPTIONS+4)
1638      2277 1      );
1639      2278 1
1640      P 2279 1 $STATE (
1641      P 2280 1      ('NEXTINI),
1642      P 2281 1      (TPAS_EOS, TPAS_EXIT)
1643      2282 1      );
1644      2283 1
1645      2284 1 Parse JOURNAL options ([NO]NEWFILE, SIZE=n, EXTENSION=n, QUOTA=n, RECORD_SIZE=n)
1646      2285 1
1647      2286 1 $INIT_STATE (JOURNAL_STB, JOURNAL_KTB);
1648      2287 1
1649      P 2288 1 $STATE (NEXT_JOURNAL,
1650      P 2289 1      ('NEWFILE',,, 1^(OPT_NEWJRN-32), MOUNT_OPTIONS+4),
1651      P 2290 1      ('NONNEWFILE',,, CLEAR_NEWJRN),
1652      P 2291 1      ('SIZE', JOURNAL_SIZE),
1653      P 2292 1      ('RECORD_SIZE', JOURNAL_RECORD_SIZE),
1654      P 2293 1      ('EXTENSION', JOURNAL_EXTEND),
1655      P 2294 1      ('QUOTA', JOURNAL_QUOTA),
```

```
: 1656      P 2295 1      (TPAS_EOS,      TPAS_EXIT)
: 1657      P 2296 1      );
: 1658      P 2297 1
: 1659      P 2298 1 $STATE (END_JOURNAL,
: 1660      P 2299 1      (
: 1661      P 2300 1      (TPAS_EOS,      NEXT_JOURNAL),
: 1662      P 2301 1      (TPAS_EXIT)
: 1663      P 2302 1      );
: 1664      P 2303 1 $STATE (JOURNAL_SIZE,
: 1665      P 2304 1      (':'),
: 1666      P 2305 1      ('='),
: 1667      P 2306 1      );
: 1668      P 2307 1
: 1669      P 2308 1 $STATE (
: 1670      P 2309 1      (TPAS_DECIMAL, END_JOURNAL,,, JRNL_SIZE)
: 1671      P 2310 1      );
: 1672      P 2311 1
: 1673      P 2312 1 $STATE (JOURNAL_RECORD_SIZE,
: 1674      P 2313 1      (':'),
: 1675      P 2314 1      ('='),
: 1676      P 2315 1      );
: 1677      P 2316 1
: 1678      P 2317 1 $STATE (
: 1679      P 2318 1      (TPAS_DECIMAL, END_JOURNAL,,, JRNL_RECORD_SIZE)
: 1680      P 2319 1      );
: 1681      P 2320 1
: 1682      P 2321 1 $STATE (JOURNAL_EXTEND,
: 1683      P 2322 1      (':'),
: 1684      P 2323 1      ('='),
: 1685      P 2324 1      );
: 1686      P 2325 1
: 1687      P 2326 1 $STATE (
: 1688      P 2327 1      (TPAS_DECIMAL, END_JOURNAL,,, JRNL_EXTEND)
: 1689      P 2328 1      );
: 1690      P 2329 1
: 1691      P 2330 1 $STATE (JOURNAL_QUOTA,
: 1692      P 2331 1      (':'),
: 1693      P 2332 1      ('='),
: 1694      P 2333 1      );
: 1695      P 2334 1
: 1696      P 2335 1 $STATE (
: 1697      P 2336 1      (TPAS_DECIMAL, END_JOURNAL,,, JRNL_QUOTA)
: 1698      P 2337 1      );
: 1699      P 2338 1
: 1700      P 2339 1
: 1701      P 2340 1      Parse /OVERRIDE options (ACCESSIBILITY, EXPIRATION, SETIDENTIFICATION,
: 1702      P 2341 1      IDENTIFICATION, OWNER_IDENTIFIER).
: 1703      P 2342 1
: 1704      P 2343 1 $INIT_STATE (OVERRIDE_STB, OVERRIDE_KTB);
: 1705      P 2344 1
: 1706      P 2345 1 $STATE (NEXTOVR,
: 1707      P 2346 1      ('ACCESSIBILITY', 1^(OPT_OVR_ACC-32), MOUNT_OPTIONS+4),
: 1708      P 2347 1      ('EXPIRATION', 1^(OPT_OVR_EXP, MOUNT_OPTIONS),
: 1709      P 2348 1      ('SETIDENTIFICATION', 1^(OPT_OVR_SETID, MOUNT_OPTIONS),
: 1710      P 2349 1      ('LOCK', 1^(OPT_OVR_LOCK-32), MOUNT_OPTIONS+4),
: 1711      P 2350 1      ('IDENTIFICATION', 1^(OPT_OVR_ID, MOUNT_OPTIONS),
: 1712      P 2351 1      ('OWNER_IDENTIFIER', 1^(OPT_OVR_VOLO-32), MOUNT_OPTIONS+4)
```

```
: 1713      2352 1      );
: 1714      2353 1
: 1715      P 2354 1 $STATE (
: 1716      P 2355 1      ('NEXTOVR)
: 1717      P 2356 1      (TPAS_EOS, TPAS_EXIT)
: 1718      2357 1      );
: 1719      2358 1
: 1720      2359 1
: 1721      2360 1      |
: 1722      2361 1      | Parse /OWNER_UIC string and store binary value.
: 1723      2362 1
: 1724      2363 1 $INIT_STATE (UIC_STB, UIC_KTB);
: 1725      2364 1
: 1726      P 2365 1 $STATE (
: 1727      P 2366 1      (TPAS_IDENT,...,UIC)
: 1728      2367 1      );
: 1729      2368 1
: 1730      P 2369 1 $STATE (
: 1731      P 2370 1      (TPAS_EOS, TPAS_EXIT)
: 1732      2371 1      );
: 1733      2372 1
: 1734      2373 1      |
: 1735      2374 1      | Parse PROCESSOR options, set bits and store name.
: 1736      2375 1
: 1737      2376 1 $INIT_STATE (PROCESSOR_STB, PROCESSOR_KTB);
: 1738      2377 1
: 1739      P 2378 1 $STATE (
: 1740      P 2379 1      ('UNIQUE',,, GET_ACP_NAME, 1^OPT_UNIQUEACP, MOUNT_OPTIONS),
: 1741      P 2380 1      ('SAME', SAMEPROC,, 1^OPT_SAMEACP, MOUNT_OPTIONS),
: 1742      P 2381 1      (('FILENAME)',, GET_ACP_NAME, 1^OPT_FILEACP, MOUNT_OPTIONS)
: 1743      2382 1      );
: 1744      2383 1
: 1745      P 2384 1 $STATE (ENDPROC,
: 1746      P 2385 1      (TPAS_EOS, TPAS_EXIT)
: 1747      2386 1      );
: 1748      2387 1
: 1749      P 2388 1 $STATE (SAMEPROC,
: 1750      P 2389 1      (':'),
: 1751      P 2390 1      ('='),
: 1752      2391 1      );
: 1753      2392 1
: 1754      P 2393 1 $STATE (
: 1755      P 2394 1      (('DEVICENAME)',, GET_ACP_NAME),
: 1756      P 2395 1      (TPAS_SYMBOL,, GET_SAME_ACP)
: 1757      2396 1      );
: 1758      2397 1
: 1759      P 2398 1 $STATE (
: 1760      P 2399 1      (TPAS_LAMBDA, TPAS_EXIT)
: 1761      2400 1      );
: 1762      2401 1
: 1763      P 2402 1 $STATE (FILENAME,
: 1764      P 2403 1      (TPAS_SYMBOL, FILENAME),
: 1765      P 2404 1      (':', FILENAME),
: 1766      P 2405 1      (':', FILENAME),
: 1767      P 2406 1      (TPAS_LAMBDA, TPAS_EXIT)
: 1768      2407 1      );
: 1769      2408 1
```

```
: 1770 P 2409 1 $STATE (DEVICENAME,  
: 1771 P 2410 1 (TPAS_SYMBOL)  
: 1772 P 2411 1 );  
: 1773 P 2412 1  
: 1774 P 2413 1 $STATE (  
: 1775 P 2414 1 (':')  
: 1776 P 2415 1 );  
: 1777 P 2416 1  
: 1778 P 2417 1 $STATE (  
: 1779 P 2418 1 (TPAS_EOS, TPAS_EXIT)  
: 1780 P 2419 1 );  
: 1781 P 2420 1  
: 1782 P 2421 1  
: 1783 P 2422 1 ; Parse /PROTECTION string '(SYSTEM:RWED,OWNER:RWED,GROUP:RWED,WORLD:RWED)'  
: 1784 P 2423 1  
: 1785 P 2424 1 $INIT_STATE (PROTECTION_STB, PROTECTION_KTB);  
: 1786 P 2425 1  
: 1787 P 2426 1 $STATE (NEXTPRO,  
: 1788 P 2427 1 ('SYSTEM', SYPR,, XX'000F0000', PROTECTION),  
: 1789 P 2428 1 ('OWNER', OWPR,, XX'00F00000', PROTECTION),  
: 1790 P 2429 1 ('GROUP', GRPR,, XX'0F000000', PROTECTION),  
: 1791 P 2430 1 ('WORLD', WOPR,, XX'F0000000', PROTECTION)  
: 1792 P 2431 1 );  
: 1793 P 2432 1  
: 1794 P 2433 1 $STATE (SYPR,  
: 1795 P 2434 1 (':'),  
: 1796 P 2435 1 ('='),  
: 1797 P 2436 1 (TPAS_LAMBDA, ENDPRO)  
: 1798 P 2437 1 );  
: 1799 P 2438 1  
: 1800 P 2439 1 $STATE (SYPRO,  
: 1801 P 2440 1 ('R', SYPRO,, XX'0001', PROTECTION),  
: 1802 P 2441 1 ('W', SYPRO,, XX'0002', PROTECTION),  
: 1803 P 2442 1 ('E', SYPRO,, XX'0004', PROTECTION),  
: 1804 P 2443 1 ('P', SYPRO,, XX'0004', PROTECTION),  
: 1805 P 2444 1 ('D', SYPRO,, XX'0008', PROTECTION),  
: 1806 P 2445 1 ('L', SYPRO,, XX'0008', PROTECTION),  
: 1807 P 2446 1 (TPAS_LAMBDA, ENDPRO)  
: 1808 P 2447 1 );  
: 1809 P 2448 1  
: 1810 P 2449 1 $STATE (OWPR,  
: 1811 P 2450 1 (':'),  
: 1812 P 2451 1 ('='),  
: 1813 P 2452 1 (TPAS_LAMBDA, ENDPRO)  
: 1814 P 2453 1 );  
: 1815 P 2454 1  
: 1816 P 2455 1 $STATE (OWPRO,  
: 1817 P 2456 1 ('R', OWPRO,, XX'0010', PROTECTION),  
: 1818 P 2457 1 ('W', OWPRO,, XX'0020', PROTECTION),  
: 1819 P 2458 1 ('E', OWPRO,, XX'0040', PROTECTION),  
: 1820 P 2459 1 ('P', OWPRO,, XX'0040', PROTECTION),  
: 1821 P 2460 1 ('D', OWPRO,, XX'0080', PROTECTION),  
: 1822 P 2461 1 ('L', OWPRO,, XX'0080', PROTECTION),  
: 1823 P 2462 1 (TPAS_LAMBDA, ENDPRO)  
: 1824 P 2463 1 );  
: 1825 P 2464 1  
: 1826 P 2465 1 $STATE (GRPR,
```

```
: 1827 P 2466 1 (':'),
: 1828 P 2467 1 ('='),
: 1829 P 2468 1 (TPAS_LAMBDA, ENDPRO)
: 1830 P 2469 1 );
: 1831 P 2470 1
: 1832 P 2471 1 $STATE (GRPRO,
: 1833 P 2472 1 ('R', GRPRO, XX'0100', PROTECTION),
: 1834 P 2473 1 ('W', GRPRO, XX'0200', PROTECTION),
: 1835 P 2474 1 ('E', GRPRO, XX'0400', PROTECTION),
: 1836 P 2475 1 ('P', GRPRO, XX'0400', PROTECTION),
: 1837 P 2476 1 ('D', GRPRO, XX'0800', PROTECTION),
: 1838 P 2477 1 ('L', GRPRO, XX'0800', PROTECTION),
: 1839 P 2478 1 (TPAS_LAMBDA, ENDPRO)
: 1840 P 2479 1 );
: 1841 P 2480 1
: 1842 P 2481 1 $STATE (WOPRO,
: 1843 P 2482 1 (':'),
: 1844 P 2483 1 ('='),
: 1845 P 2484 1 (TPAS_LAMBDA, ENDPRO)
: 1846 P 2485 1 );
: 1847 P 2486 1
: 1848 P 2487 1 $STATE (WOPRO,
: 1849 P 2488 1 ('R', WOPRO, XX'1000', PROTECTION),
: 1850 P 2489 1 ('W', WOPRO, XX'2000', PROTECTION),
: 1851 P 2490 1 ('E', WOPRO, XX'4000', PROTECTION),
: 1852 P 2491 1 ('P', WOPRO, XX'4000', PROTECTION),
: 1853 P 2492 1 ('D', WOPRO, XX'8000', PROTECTION),
: 1854 P 2493 1 ('L', WOPRO, XX'8000', PROTECTION),
: 1855 P 2494 1 (TPAS_LAMBDA, ENDPRO)
: 1856 P 2495 1 );
: 1857 P 2496 1
: 1858 P 2497 1 $STATE (ENDPRO,
: 1859 P 2498 1 ('', NEXTPRO),
: 1860 P 2499 1 (TPAS_EOS, TPAS_EXIT)
: 1861 P 2500 1 );
: 1862 P 2501 1
: 1863 P 2502 1 END
: 1864 P 2503 0 ELUDOM
```

```
.PSECT _LIB$KEY1$,NOWRT, SHR, PIC,1
00000 ;TPASKEYSTO
54 4E 45 54 58 45 00000 U.2: .BLKB 0
;TPASKEYST
FF 00006 U.4: .ASCII \EXTENT\
00007 ;TPASKEYSTO -1
;TPASKEYSTO
44 49 5F 45 4C 49 46 00007 U.10: .BLKB 0
;TPASKEYST
FF 0000E U.12: .ASCII \FILE_ID\
0000F ;TPASKEYSTO -1
;TPASKEYSTO
54 49 4D 49 4C 0000F U.18: .BLKB 0
;TPASKEYST
U.20: .ASCII \LIMIT\
```



```

B 10
16-Sep-1984 01:06:29      VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:45:31      [MOUNT.SRC]MOUNTIMG.B32;1

```

Page 69
(24)

MOL
V04

[illegible]

Offset	Hex	ASCII	Comment
00071	U.127:	TPASKEYSTO	0
00071	U.129:	TPASKEYST	0
00078	FF	TPASKEYSTO	-1
00079	U.133:	TPASKEYST	0
00079	U.135:	TPASKEYST	0
00082	FF	TPASKEYSTO	-1
00083	U.138:	TPASKEYST	0
00083	U.140:	TPASKEYST	0
00087	FF	TPASKEYSTO	-1
00088	U.144:	TPASKEYST	0
00088	U.146:	TPASKEYST	0
00093	FF	TPASKEYSTO	-1
00094	U.150:	TPASKEYST	0
00094	U.152:	TPASKEYST	0
0009D	FF	TPASKEYSTO	-1
0009E	U.156:	TPASKEYST	0
0009E	U.158:	TPASKEYST	0
000A3	FF	TPASKEYSTO	-1
000A4	FF	TPASKEYSTO	-1
000A5	U.164:	TPASKEYSTO	-1
000A5	U.190:	TPASKEYST	0
000A5	U.192:	TPASKEYST	0
000B2	FF	TPASKEYSTO	-1
000B3	U.196:	TPASKEYST	0
000B3	U.198:	TPASKEYST	0
000BD	FF	TPASKEYSTO	-1
000BE	U.202:	TPASKEYST	0
000BE	U.204:	TPASKEYST	0
000CD	4E	TPASKEYSTO	-1
000CF	FF	TPASKEYSTO	-1
000D0	U.208:	TPASKEYST	0
000D0	U.210:	TPASKEYST	0
000D4	FF	TPASKEYSTO	-1
000D5	U.214:	TPASKEYST	0
000D5	U.216:	TPASKEYST	0


```

      FF 000E3      .BYTE      -1
      000E4      ;TPASKEYSTO
45  49  46  49  54  4E  45  44  49  5F  52  45  4E  57  4F  000E4      U.220:      .BLKB      0
      000E4      ;TPASKEYST
      U.222:      .ASCII      \OWNER_IDENTIFIERN
      52 000F3      .BYTE      -1
      FF 000F4      ;TPASKEYFILL
      FF 000F5      U.226:      .BYTE      -1
      000F6      ;TPASKEYSTO
      U.237:      .BLKB      0
      45  55  51  49  4E  55  000F6      ;TPASKEYST
      U.239:      .ASCII      \UNIQUE\
      FF 000FC      .BYTE      -1
      000FD      ;TPASKEYSTO
      45  4D  41  53  000FD      U.244:      .BLKB      0
      000FD      ;TPASKEYST
      FF 00101      U.246:      .ASCII      \SAME\
      FF 00102      .BYTE      -1
      00103      ;TPASKEYFILL
      U.258:      .BYTE      -1
      00103      ;TPASKEYSTO
      4D  45  54  53  59  53  00103      U.284:      .BLKB      0
      00103      ;TPASKEYST
      U.286:      .ASCII      \SYSTEM\
      FF 00109      .BYTE      -1
      0010A      ;TPASKEYSTO
      52  45  4E  57  4F  0010A      U.292:      .BLKB      0
      0010A      ;TPASKEYST
      FF 0010F      U.294:      .ASCII      \OWNER\
      00110      .BYTE      -1
      00110      ;TPASKEYSTO
      50  55  4F  52  47  00110      U.300:      .BLKB      0
      00110      ;TPASKEYST
      FF 00115      U.302:      .ASCII      \GROUP\
      00116      .BYTE      -1
      00116      ;TPASKEYSTO
      44  4C  52  4F  57  00116      U.308:      .BLKB      0
      00116      ;TPASKEYST
      FF 0011B      U.310:      .ASCII      \WORLD\
      FF 0011C      .BYTE      -1
      0011C      ;TPASKEYFILL
      U.316:      .BYTE      -1

      .PSECT      _LIB$STATES,NOWRT,  SHR,  PIC,1

      00000      CACHE_STB::
      .BLKB      0
      00000      NEXT_CACHE:
      .BLKB      0
      7100      00000      ;TPASTYPE
      U.5:      .WORD      28928
      00000000* 00002      ;TPASADDR
      U.6:      .LONG      <<<MOUNT_OPTIONS+4>-U.6>-4>
      00002000      00006      ;TPASMASK
      U.7:      .LONG      8192
      0000* 0000A      ;TPASTARGET
```

7101	0000C	U.9: .WORD	<<U.8-U.9>-2>	:
		:TPASTYPE		:
00000000*	0000E	U.13: .WORD	28929	:
		:TPASADDR		:
00002000	00012	U.14: .LONG	<<<MOUNT_OPTIONS+4>-U.14>-4>	:
		:TPASMASK		:
0000*	00016	U.15: .LONG	8192	:
		:TPASTARGET		:
1102	00018	U.17: .WORD	<<U.16-U.17>-2>	:
		:TPASTYPE		:
0000*	0001A	U.21: .WORD	4354	:
		:TPASTARGET		:
6103	0001C	U.23: .WORD	<<U.22-U.23>-2>	:
		:TPASTYPE		:
00000000*	0001E	U.27: .WORD	24835	:
		:TPASADDR		:
00008000	00022	U.28: .LONG	<<<MOUNT_OPTIONS+4>-U.28>-4>	:
		:TPASMASK		:
6104	00026	U.29: .LONG	32768	:
		:TPASTYPE		:
00000000*	00028	U.33: .WORD	24836	:
		:TPASADDR		:
00C10000	0002C	U.34: .LONG	<<<MOUNT_OPTIONS+4>-U.34>-4>	:
		:TPASMASK		:
6105	00030	U.35: .LONG	65536	:
		:TPASTYPE		:
00000000*	00032	U.39: .WORD	24837	:
		:TPASADDR		:
00020000	00036	U.40: .LONG	<<<MOUNT_OPTIONS+4>-U.40>-4>	:
		:TPASMASK		:
0106	0003A	U.41: .LONG	131072	:
		:TPASTYPE		:
7107	0003C	U.45: .WORD	262	:
		:TPASTYPE		:
00000000*	0003E	U.49: .WORD	28935	:
		:TPASADDR		:
00002000	00042	U.50: .LONG	<<<MOUNT_OPTIONS+4>-U.50>-4>	:
		:TPASMASK		:
0000*	00046	U.51: .LONG	8192	:
		:TPASTARGET		:
6508	00048	U.53: .WORD	<<U.52-U.53>-2>	:
		:TPASTYPE		:
00000000*	0004A	U.57: .WORD	25864	:
		:TPASADDR		:
00004000	0004E	U.58: .LONG	<<<MOUNT_OPTIONS+4>-U.58>-4>	:
		:TPASMASK		:
	00052	U.59: .LONG	16384	:
		END_CACHE:		:
102C	00052	.BLKB	0	:
		:TPASTYPE		:
0900*	00054	U.61: .WORD	4140	:
		:TPASTARGET		:
15F7	00056	U.62: .WORD	<<NEXT_CACHE-U.62>-2>	:
		:TPASTYPE		:
FFFF	00058	U.63: .WORD	5623	:
		:TPASTARGET		:
		U.64: .WORD	-1	:

	0005A	:CACHE_EXT	
	U.8:	.BLKB	0
003A	0005A	:TPASTYPE	
	U.65:	.WORD	58
043D	0005C	:TPASTYPE	
	U.66:	.WORD	1085
55F3	0005E	:TPASTYPE	
	U.67:	.WORD	22003
00000000*	00060	:TPASADDR	
	U.68:	.LONG	<<EXT_CACHE-U.68>-4>
0000*	00064	:TPASTARGET	
	U.69:	.WORD	<<END_CACHE-U.69>-2>
	00066	:CACHE_FID	
	U.16:	.BLKB	0
003A	00066	:TPASTYPE	
	U.70:	.WORD	58
043D	00068	:TPASTYPE	
	U.71:	.WORD	1085
55F3	0006A	:TPASTYPE	
	U.72:	.WORD	22003
00000000*	0006C	:TPASADDR	
	U.73:	.LONG	<<FID_CACHE-U.73>-4>
0000*	00070	:TPASTARGET	
	U.74:	.WORD	<<END_CACHE-U.74>-2>
	00072	:CACHE_QUO	
	U.52:	.BLKB	0
003A	00072	:TPASTYPE	
	U.75:	.WORD	58
043D	00074	:TPASTYPE	
	U.76:	.WORD	1085
55F3	00076	:TPASTYPE	
	U.77:	.WORD	22003
00000000*	00078	:TPASADDR	
	U.78:	.LONG	<<QUO_CACHE-U.78>-4>
0000*	0007C	:TPASTARGET	
	U.79:	.WORD	<<END_CACHE-U.79>-2>
	0007E	:LIMIT_EXT	
	U.22:	.BLKB	0
003A	0007E	:TPASTYPE	
	U.80:	.WORD	58
043D	00080	:TPASTYPE	
	U.81:	.WORD	1085
55F3	00082	:TPASTYPE	
	U.82:	.WORD	22003
00000000*	00084	:TPASADDR	
	U.83:	.LONG	<<EXT_LIMIT-U.83>-4>
0000*	00088	:TPASTARGET	
	U.84:	.WORD	<<END_CACHE-U.84>-2>
	0008A	:BLKB	2
	0008C	:DATACHECK_STB:	
	U.84:	.BLKB	0
71F7	0008C	:TPASTYPE	
	U.86:	.WORD	29175
00000000*	0008E	:TPASADDR	
	U.87:	.LONG	<<<MOUNT_OPTIONS+4>-U.87>-4>
00000010	00092	:TPASMASK	
	U.88:	.LONG	16

FFFF	00096	:TPASTARGET			
		U.89:	WORD	-1	:
05F6	00098	:TPASTYPE			:
		U.90:	WORD	1526	:
	0009A	CHECKOPT:			:
		BLKB		0	:
6100	0009A	:TPASTYPE			:
		U.94:	WORD	24832	:
00000000*	0009C	:TPASADDR			:
		U.95:	LONG	<<<MOUNT_OPTIONS+4>-U.95>-4>	:
00000008	000A0	:TPASMASK			:
		U.96:	LONG	8	:
6501	000A4	:TPASTYPE			:
		U.100:	WORD	25857	:
00000000*	000A6	:TPASADDR			:
		U.101:	LONG	<<<MOUNT_OPTIONS+4>-U.101>-4>	:
00000010	000AA	:TPASMASK			:
		U.102:	LONG	16	:
102C	000AE	:TPASTYPE			:
		U.104:	WORD	4140	:
0000*	000B0	:TPASTARGET			:
		U.105:	WORD	<<CHECKOPT-U.105>-2>	:
15F7	000B2	:TPASTYPE			:
		U.106:	WORD	5623	:
FFFF	000B4	:TPASTARGET			:
		U.107:	WORD	-1	:
	000B6	BLKB		2	:
	000B8	INITIALIZE_STB:			:
		BLKB		0	:
	000B8	NEXTINI:	BLKB	0	:
6100	000B8	:TPASTYPE			:
		U.112:	WORD	24832	:
00000000*	000BA	:TPASADDR			:
		U.113:	LONG	<<<MOUNT_OPTIONS+4>-U.113>-4>	:
04000000	000BE	:TPASMASK			:
		U.114:	LONG	67108864	:
6501	000C2	:TPASTYPE			:
		U.118:	WORD	25857	:
00000000*	000C4	:TPASADDR			:
		U.119:	LONG	<<<MOUNT_OPTIONS+4>-U.119>-4>	:
08000000	000C8	:TPASMASK			:
		U.120:	LONG	134217728	:
102C	000CC	:TPASTYPE			:
		U.122:	WORD	4140	:
0000*	000CE	:TPASTARGET			:
		U.123:	WORD	<<NEXTINI-U.123>-2>	:
15F7	000D0	:TPASTYPE			:
		U.124:	WORD	5623	:
FFFF	000D2	:TPASTARGET			:
		U.125:	WORD	-1	:
	000D4	JOURNAL_STB:			:
		BLKB		0	:
	000D4	NEXT_JOURNAL:			:
		BLKB		0	:
6100	000D4	:TPASTYPE			:
		U.130:	WORD	24832	:
00000000*	000D6	:TPASADDR			:

01000000	000DA	U.131: .LONG	<<MOUNT_OPTIONS+4>-U.131>-4>	:
		:TPASMASK		:
8101	000DE	U.132: .LONG	16777216	:
		:TPASTYPE		:
00000000*	000E0	U.136: .WORD	-32511	:
		:TPASACTION		:
1102	000E4	U.137: .LONG	<<CLEAR_NEWJRNLU.137>-4>	:
		:TPASTYPE		:
0000*	000E6	U.141: .WORD	4354	:
		:TPASTARGET		:
1103	000E8	U.143: .WORD	<<U.142-U.143>-2>	:
		:TPASTYPE		:
0000*	000EA	U.147: .WORD	4355	:
		:TPASTARGET		:
1104	000EC	U.149: .WORD	<<U.148-U.149>-2>	:
		:TPASTYPE		:
0000*	000EE	U.153: .WORD	4356	:
		:TPASTARGET		:
1105	000F0	U.155: .WORD	<<U.154-U.155>-2>	:
		:TPASTYPE		:
0000*	000F2	U.159: .WORD	4357	:
		:TPASTARGET		:
15F7	000F4	U.161: .WORD	<<U.160-U.161>-2>	:
		:TPASTYPE		:
FFFF	000F6	U.162: .WORD	5623	:
		:TPASTARGET		:
	000F8	U.163: .WORD	-1	:
		END_JOURNAL:		:
102C	000F8	.BLKB	0	:
		:TPASTYPE		:
0000*	000FA	U.165: .WORD	4140	:
		:TPASTARGET		:
15F7	000FC	U.166: .WORD	<<NEXT_JOURNAL-U.166>-2>	:
		:TPASTYPE		:
FFFF	000FE	U.167: .WORD	5623	:
		:TPASTARGET		:
	00100	U.168: .WORD	-1	:
		:JOURNAL_SIZE		:
003A	00100	U.142: .BLKB	0	:
		:TPASTYPE		:
043D	00102	U.169: .WORD	58	:
		:TPASTYPE		:
55F3	00104	U.170: .WORD	1085	:
		:TPASTYPE		:
00000000*	00106	U.171: .WORD	22003	:
		:TPASADDR		:
0000*	0010A	U.172: .LONG	<<JRNLU_SIZE-U.172>-4>	:
		:TPASTARGET		:
	0010C	U.173: .WORD	<<END_JOURNAL-U.173>-2>	:
		:JOURNAL_RECORD_SIZE		:
003A	0010C	U.148: .BLKB	0	:
		:TPASTYPE		:
043D	0010E	U.174: .WORD	58	:
		:TPASTYPE		:
55F3	00110	U.175: .WORD	1085	:
		:TPASTYPE		:
		U.176: .WORD	22003	:

00000000*	00112	:TPASADDR			
		U.177:	LONG	<<JRNL_RECORD_SIZE-U.177>-4>	:
0000*	00116	:TPASTARGET			:
		U.178:	WORD	<<END_JOURNAL-U.178>-2>	:
	00118	:JOURNAL_EXTEND			:
		U.154:	BLKB	0	:
003A	00118	:TPASTYPE			:
		U.179:	WORD	58	:
043D	0011A	:TPASTYPE			:
		U.180:	WORD	1085	:
55F3	0011C	:TPASTYPE			:
		U.181:	WORD	22003	:
00000000*	0011E	:TPASADDR			:
		U.182:	LONG	<<JRNL_EXTEND-U.182>-4>	:
0000*	00122	:TPASTARGET			:
		U.183:	WORD	<<END_JOURNAL-U.183>-2>	:
	00124	:JOURNAL_QUOTA			:
		U.160:	BLKB	0	:
003A	00124	:TPASTYPE			:
		U.184:	WORD	58	:
043D	00126	:TPASTYPE			:
		U.185:	WORD	1085	:
55F3	00128	:TPASTYPE			:
		U.186:	WORD	22003	:
00000000*	0012A	:TPASADDR			:
		U.187:	LONG	<<JRNL_QUOTA-U.187>-4>	:
0000*	0012E	:TPASTARGET			:
		U.188:	WORD	<<END_JOURNAL-U.188>-2>	:
	00130	OVERWRITE_STB:			:
		U.189:	BLKB	0	:
	00130	NEXTOVR:	BLKB	0	:
6100	00130	:TPASTYPE			:
		U.193:	WORD	24832	:
00000000*	00132	:TPASADDR			:
		U.194:	LONG	<<<MOUNT_OPTIONS+4>-U.194>-4>	:
00000040	00136	:TPASMASK			:
		U.195:	LONG	64	:
6101	0013A	:TPASTYPE			:
		U.199:	WORD	24833	:
00000000*	0013C	:TPASADDR			:
		U.200:	LONG	<<MOUNT_OPTIONS-U.200>-4>	:
00100000	00140	:TPASMASK			:
		U.201:	LONG	1048576	:
6102	00144	:TPASTYPE			:
		U.205:	WORD	24834	:
00000000*	00146	:TPASADDR			:
		U.206:	LONG	<<MOUNT_OPTIONS-U.206>-4>	:
00200000	0014A	:TPASMASK			:
		U.207:	LONG	2097152	:
6103	0014E	:TPASTYPE			:
		U.211:	WORD	24835	:
00000000*	00150	:TPASADDR			:
		U.212:	LONG	<<<MOUNT_OPTIONS+4>-U.212>-4>	:
00200000	00154	:TPASMASK			:
		U.213:	LONG	2097152	:
6104	00158	:TPASTYPE			:
		U.217:	WORD	24836	:

00000000*	0015A	:TPASADDR			
		U.218:	LONG	<<MOUNT_OPTIONS-U.218>-4>	:
00400000	0015E	:TPASMASK			:
		U.219:	LONG	4194304	:
6505	00162	:TPASTYPE			:
		U.223:	WORD	25861	:
00000000*	00164	:TPASADDR			:
		U.224:	LONG	<<<MOUNT_OPTIONS+4>-U.224>-4>	:
10000000	00168	:TPASMASK			:
		U.225:	LONG	268435456	:
102C	0016C	:TPASTYPE			:
		U.227:	WORD	4140	:
0000*	0016E	:TPASTARGET			:
		U.228:	WORD	<<NEXTTOVR-U.228>-2>	:
15F7	00170	:TPASTYPE			:
		U.229:	WORD	5623	:
FFFF	00172	:TPASTARGET			:
		U.230:	WORD	-1	:
	00174	UIC_STB::			:
			BLKB	0	:
45EC	00174	:TPASTYPE			:
		U.232:	WORD	17900	:
00000000*	00176	:TPASADDR			:
		U.233:	LONG	<<UIC-U.233>-4>	:
15F7	0017A	:TPASTYPE			:
		U.234:	WORD	5623	:
FFFF	0017C	:TPASTARGET			:
		U.235:	WORD	-1	:
	0017E		BLKB	2	:
	00180	PROCESSOR_STB::			:
			BLKB	0	:
E100	00180	:TPASTYPE			:
		U.240:	WORD	-7936	:
00000000*	00182	:TPASACTION			:
		U.241:	LONG	<<GET_ACP_NAME-U.241>-4>	:
00000000*	00186	:TPASADDR			:
		U.242:	LONG	<<MOUNT_OPTIONS-U.242>-4>	:
04000000	0018A	:TPASMASK			:
		U.243:	LONG	67108864	:
7101	0018E	:TPASTYPE			:
		U.247:	WORD	28929	:
00000000*	00190	:TPASADDR			:
		U.248:	LONG	<<MOUNT_OPTIONS-U.248>-4>	:
08000000	00194	:TPASMASK			:
		U.249:	LONG	134217728	:
0000*	00198	:TPASTARGET			:
		U.251:	WORD	<<U.250-U.251>-2>	:
EDF8	0019A	:TPASTYPE			:
		U.252:	WORD	-4616	:
0000*	0019C	:TPASSUBEXP			:
		U.254:	WORD	<<U.253-U.254>-2>	:
00000000*	0019E	:TPASACTION			:
		U.255:	LONG	<<GET_ACP_NAME-U.255>-4>	:
00000000*	001A2	:TPASADDR			:
		U.256:	LONG	<<MOUNT_OPTIONS-U.256>-4>	:
10000000	001A6	:TPASMASK			:
		U.257:	LONG	268435456	:

	001AA	ENDPROC: .BLKB	0
15F7	001AA	:TPASTYPE	
		U.259: .WORD	5623
FFFF	001AC	:TPASTARGET	
		U.260: .WORD	-1
	001AE	:SAMEPROC	
		U.250: .BLKB	0
003A	001AE	:TPASTYPE	
		U.261: .WORD	58
043D	001B0	:TPASTYPE	
		U.262: .WORD	1085
89F8	001B2	:TPASTYPE	
		U.263: .WORD	-30216
0000*	001B4	:TPASUBEXP	
		U.265: .WORD	<<U.264-U.265>-2>
00000000*	001B6	:TPASACTION	
		U.266: .LONG	<<GET_ACP_NAME-U.266>-4>
85F1	001BA	:TPASTYPE	
		U.267: .WORD	-31247
00000000*	001BC	:TPASACTION	
		U.268: .LONG	<<GET_SAME_ACP-U.268>-4>
15F6	001C0	:TPASTYPE	
		U.269: .WORD	5622
FFFF	001C2	:TPASTARGET	
		U.270: .WORD	-1
	001C4	:FILENAME	
		U.253: .BLKB	0
11F1	001C4	:TPASTYPE	
		U.271: .WORD	4593
0000*	001C6	:TPASTARGET	
		U.272: .WORD	<<U.253-U.272>-2>
102E	001C8	:TPASTYPE	
		U.273: .WORD	4142
0000*	001CA	:TPASTARGET	
		U.274: .WORD	<<U.253-U.274>-2>
103B	001CC	:TPASTYPE	
		U.275: .WORD	4155
0000*	001CE	:TPASTARGET	
		U.276: .WORD	<<U.253-U.276>-2>
15F6	001D0	:TPASTYPE	
		U.277: .WORD	5622
FFFF	001D2	:TPASTARGET	
		U.278: .WORD	-1
	001D4	:DEVICENAME	
		U.264: .BLKB	0
05F1	001D4	:TPASTYPE	
		U.279: .WORD	1521
043A	001D6	:TPASTYPE	
		U.280: .WORD	1082
15F7	001D8	:TPASTYPE	
		U.281: .WORD	5623
FFFF	001DA	:TPASTARGET	
		U.282: .WORD	-1
	001DC	PROTECTION STB: .	
		.BLKB	0
	001DC	NEXTPRO: .BLKB	0
7100	001DC	:TPASTYPE	

00000000*	001DE	U.287: .WORD	28928	:
		;TPASADDR		:
000F0000	001E2	U.288: .LONG	<<PROTECTION-U.288>-4>	:
		;TPASMASK		:
0000*	001E6	U.289: .LONG	983040	:
		;TPASTARGET		:
7101	001E8	U.291: .WORD	<<U.290-U.291>-2>	:
		;TPASTYPE		:
		U.295: .WORD	28929	:
00000000*	001EA	;TPASADDR		:
		U.296: .LONG	<<PROTECTION-U.296>-4>	:
00F00000	001EE	;TPASMASK		:
		U.297: .LONG	15728640	:
0000*	001F2	;TPASTARGET		:
		U.299: .WORD	<<U.298-U.299>-2>	:
7102	001F4	;TPASTYPE		:
		U.303: .WORD	28930	:
00000000*	001F6	;TPASADDR		:
		U.304: .LONG	<<PROTECTION-U.304>-4>	:
0F000000	001FA	;TPASMASK		:
		U.305: .LONG	251658240	:
0000*	001FE	;TPASTARGET		:
		U.307: .WORD	<<U.306-U.307>-2>	:
7503	00200	;TPASTYPE		:
		U.311: .WORD	29955	:
00000000*	00202	;TPASADDR		:
		U.312: .LONG	<<PROTECTION-U.312>-4>	:
F0000000	00206	;TPASMASK		:
		U.313: .LONG	-268435456	:
0000*	0020A	;TPASTARGET		:
		U.315: .WORD	<<U.314-U.315>-2>	:
	0020C	;SYPR		:
		U.290: .BLKB	0	:
003A	0020C	;TPASTYPE		:
		U.317: .WORD	58	:
003D	0020E	;TPASTYPE		:
		U.318: .WORD	61	:
15F6	00210	;TPASTYPE		:
		U.319: .WORD	5622	:
0000*	00212	;TPASTARGET		:
		U.321: .WORD	<<U.320-U.321>-2>	:
	00214	SYPRO: .BLKB	0	:
7052	00214	;TPASTYPE		:
		U.322: .WORD	28754	:
00000000*	00216	;TPASADDR		:
		U.323: .LONG	<<PROTECTION-U.323>-4>	:
00000001	0021A	;TPASMASK		:
		U.324: .LONG	1	:
0000*	0021E	;TPASTARGET		:
		U.325: .WORD	<<SYPRO-U.325>-2>	:
7057	00220	;TPASTYPE		:
		U.326: .WORD	28759	:
00000000*	00222	;TPASADDR		:
		U.327: .LONG	<<PROTECTION-U.327>-4>	:
00000002	00226	;TPASMASK		:
		U.328: .LONG	2	:
0000*	0022A	;TPASTARGET		:

7045	0022C	U.329: .WORD	<<SYPRO-U.329>-2>
		;TPASTYPE	
00000000*	0022E	U.330: .WORD	28741
		;TPASADDR	
00000004	00232	U.331: .LONG	<<PROTECTION-U.331>-4>
		;TPASMASK	
0000*	00236	U.332: .LONG	4
		;TPASTARGET	
7050	00238	U.333: .WORD	<<SYPRO-U.333>-2>
		;TPASTYPE	
00000000*	0023A	U.334: .WORD	28752
		;TPASADDR	
00000004	0023E	U.335: .LONG	<<PROTECTION-U.335>-4>
		;TPASMASK	
0000*	00242	U.336: .LONG	4
		;TPASTARGET	
7044	00244	U.337: .WORD	<<SYPRO-U.337>-2>
		;TPASTYPE	
00000000*	00246	U.338: .WORD	28740
		;TPASADDR	
00000008	0024A	U.339: .LONG	<<PROTECTION-U.339>-4>
		;TPASMASK	
0000*	0024E	U.340: .LONG	8
		;TPASTARGET	
704C	00250	U.341: .WORD	<<SYPRO-U.341>-2>
		;TPASTYPE	
00000000*	00252	U.342: .WORD	28748
		;TPASADDR	
00000008	00256	U.343: .LONG	<<PROTECTION-U.343>-4>
		;TPASMASK	
0000*	0025A	U.344: .LONG	8
		;TPASTARGET	
15F6	0025C	U.345: .WORD	<<SYPRO-U.345>-2>
		;TPASTYPE	
0000*	0025E	U.346: .WORD	5622
		;TPASTARGET	
	00260	U.347: .WORD	<<U.320-U.347>-2>
		;OWPR	
003A	00260	U.298: .BLKB	0
		;TPASTYPE	
003D	00262	U.348: .WORD	58
		;TPASTYPE	
15F6	00264	U.349: .WORD	61
		;TPASTYPE	
0000*	00266	U.350: .WORD	5622
		;TPASTARGET	
	00268	U.351: .WORD	<<U.320-U.351>-2>
		;OWPRO: .BLKB	0
7052	00268	U.352: .WORD	28754
		;TPASTYPE	
00000000*	0026A	U.353: .LONG	<<PROTECTION-U.353>-4>
		;TPASADDR	
00000010	0026E	U.354: .LONG	16
		;TPASMASK	
0000*	00272	U.355: .LONG	
		;TPASTARGET	
7057	00274	U.355: .WORD	<<OWPRO-U.355>-2>
		;TPASTYPE	

N 10
16-Sep-1984 01:06:29 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:45:31 [MOUNT.SRC]MOUNTIMG.B32:1

Page 81
(24)

MOL
V04

Address	Offset	Field	Value
00000000	00276	U.356: .WORD	28759
		;TPAS\$ADDR	
00000020	0027A	U.357: .LONG	<<PROTECTION-U.357>-4>
		;TPAS\$MASK	
0000	0027E	U.358: .LONG	32
		;TPAS\$TARGET	
7045	00280	U.359: .WORD	<<OWPRO-U.359>-2>
		;TPAS\$TYPE	
00000000	00282	U.360: .WORD	28741
		;TPAS\$ADDR	
00000040	00286	U.361: .LONG	<<PROTECTION-U.361>-4>
		;TPAS\$MASK	
0000	0028A	U.362: .LONG	64
		;TPAS\$TARGET	
7050	0028C	U.363: .WORD	<<OWPRO-U.363>-2>
		;TPAS\$TYPE	
00000000	0028E	U.364: .WORD	28752
		;TPAS\$ADDR	
00000040	00292	U.365: .LONG	<<PROTECTION-U.365>-4>
		;TPAS\$MASK	
0000	00296	U.366: .LONG	64
		;TPAS\$TARGET	
7044	00298	U.367: .WORD	<<OWPRO-U.367>-2>
		;TPAS\$TYPE	
00000000	0029A	U.368: .WORD	28740
		;TPAS\$ADDR	
00000080	0029E	U.369: .LONG	<<PROTECTION-U.369>-4>
		;TPAS\$MASK	
0000	002A2	U.370: .LONG	128
		;TPAS\$TARGET	
704C	002A4	U.371: .WORD	<<OWPRO-U.371>-2>
		;TPAS\$TYPE	
000000C0	002A6	U.372: .WORD	28748
		;TPAS\$ADDR	
00000080	002AA	U.373: .LONG	<<PROTECTION-U.373>-4>
		;TPAS\$MASK	
0000	002AE	U.374: .LONG	128
		;TPAS\$TARGET	
15F6	002B0	U.375: .WORD	<<OWPRO-U.375>-2>
		;TPAS\$TYPE	
0000	002B2	U.376: .WORD	5622
		;TPAS\$TARGET	
	002B4	U.377: .WORD	<<U.320-U.377>-2>
		;GRPR	
003A	002B4	U.306: .BLKB	0
		;TPAS\$TYPE	
003D	002B6	U.378: .WORD	58
		;TPAS\$TYPE	
15F6	002B8	U.379: .WORD	61
		;TPAS\$TYPE	
0000	002BA	U.380: .WORD	5622
		;TPAS\$TARGET	
	002BC	U.381: .WORD	<<U.320-U.381>-2>
7052	002BC	GRPRO: .BLKB	0
		;TPAS\$TYPE	
00000000	002BE	U.382: .WORD	28754
		;TPAS\$ADDR	

00000100	002C2	U.383: .LONG	<<PROTECTION-U.383>-4>	:
		;TP\$MASK		
0000*	002C6	U.384: .LONG	256	:
		;TP\$TARGET		
7057	002C8	U.385: .WORD	<<GRPRO-U.385>-2>	:
		;TP\$TYPE		
00000000*	002CA	U.386: .WORD	28759	:
		;TP\$ADDR		
00000200	002CE	U.387: .LONG	<<PROTECTION-U.387>-4>	:
		;TP\$MASK		
0000*	002D2	U.388: .LONG	512	:
		;TP\$TARGET		
7045	002D4	U.389: .WORD	<<GRPRO-U.389>-2>	:
		;TP\$TYPE		
00000000*	002D6	U.390: .WORD	28741	:
		;TP\$ADDR		
00000400	002DA	U.391: .LONG	<<PROTECTION-U.391>-4>	:
		;TP\$MASK		
0000*	002DE	U.392: .LONG	1024	:
		;TP\$TARGET		
7050	002E0	U.393: .WORD	<<GRPRO-U.393>-2>	:
		;TP\$TYPE		
00000000*	002E2	U.394: .WORD	28752	:
		;TP\$ADDR		
00000400	002E6	U.395: .LONG	<<PROTECTION-U.395>-4>	:
		;TP\$MASK		
0000*	002EA	U.396: .LONG	1024	:
		;TP\$TARGET		
7044	002EC	U.397: .WORD	<<GRPRO-U.397>-2>	:
		;TP\$TYPE		
00000000*	002EE	U.398: .WORD	28740	:
		;TP\$ADDR		
00000800	002F2	U.399: .LONG	<<PROTECTION-U.399>-4>	:
		;TP\$MASK		
0000*	002F6	U.400: .LONG	2048	:
		;TP\$TARGET		
704C	002F8	U.401: .WORD	<<GRPRO-U.401>-2>	:
		;TP\$TYPE		
00000000*	002FA	U.402: .WORD	28748	:
		;TP\$ADDR		
00000800	002FE	U.403: .LONG	<<PROTECTION-U.403>-4>	:
		;TP\$MASK		
0000*	00302	U.404: .LONG	2048	:
		;TP\$TARGET		
15F6	00304	U.405: .WORD	<<GRPRO-U.405>-2>	:
		;TP\$TYPE		
0000*	00306	U.406: .WORD	5622	:
		;TP\$TARGET		
	00308	U.407: .WORD	<<U.320-U.407>-2>	:
		;WOPR		
003A	00308	U.314: .BLKB	0	:
		;TP\$TYPE		
003D	0030A	U.408: .WORD	58	:
		;TP\$TYPE		
15F6	0030C	U.409: .WORD	61	:
		;TP\$TYPE		
		U.410: .WORD	5622	:

```
0000* 0030E :TPASTARGET
              U.411: .WORD    <<U.320-U.411>-2>
              WOPRO: .BLKB    0
7052 00310 :TPASTYPE
              U.412: .WORD    28754
00000000* 00312 :TPASADDR
              U.413: .LONG    <<PROTECTION-U.413>-4>
00001000 00316 :TPASMASK
              U.414: .LONG    4096
0000* 0031A :TPASTARGET
              U.415: .WORD    <<WOPRO-U.415>-2>
7057 0031C :TPASTYPE
              U.416: .WORD    28759
00000000* 0031E :TPASADDR
              U.417: .LONG    <<PROTECTION-U.417>-4>
00002000 00322 :TPASMASK
              U.418: .LONG    8192
0000* 00326 :TPASTARGET
              U.419: .WORD    <<WOPRO-U.419>-2>
7045 00328 :TPASTYPE
              U.420: .WORD    28741
00000000* 0032A :TPASADDR
              U.421: .LONG    <<PROTECTION-U.421>-4>
00004000 0032E :TPASMASK
              U.422: .LONG    16384
0000* 00332 :TPASTARGET
              U.423: .WORD    <<WOPRO-U.423>-2>
7050 00334 :TPASTYPE
              U.424: .WORD    28752
00000000* 00336 :TPASADDR
              U.425: .LONG    <<PROTECTION-U.425>-4>
00004000 0033A :TPASMASK
              U.426: .LONG    16384
0000* 0033E :TPASTARGET
              U.427: .WORD    <<WOPRO-U.427>-2>
7044 00340 :TPASTYPE
              U.428: .WORD    28740
00000000* 00342 :TPASADDR
              U.429: .LONG    <<PROTECTION-U.429>-4>
00008000 00346 :TPASMASK
              U.430: .LONG    32768
0000* 0034A :TPASTARGET
              U.431: .WORD    <<WOPRO-U.431>-2>
704C 0034C :TPASTYPE
              U.432: .WORD    28748
00000000* 0034E :TPASADDR
              U.433: .LONG    <<PROTECTION-U.433>-4>
00008000 00352 :TPASMASK
              U.434: .LONG    32768
0000* 00356 :TPASTARGET
              U.435: .WORD    <<WOPRO-U.435>-2>
15F6 00358 :TPASTYPE
              U.436: .WORD    5622
0000* 0035A :TPASTARGET
              U.437: .WORD    <<U.320-U.437>-2>
              0035C :ENDPRO
              U.320: .BLKB    0
```

```
102C 0035C :TPASTYPE
          U.438: .WORD 4140
0000* 0035E :TPASTARGET
          U.439: .WORD <<NEXTPRG-U.439>-2>
15F7 00360 :TPASTYPE
          U.440: .WORD 5623
FFFF 00362 :TPASTARGET
          U.441: .WORD -1
          .PSECT _LIB$KEY0$,NOWRT, SHR, PIC,1
00000 CACHE_KTB::
          .BLKB 0
00000 :TPASKEY0
          U.1: .BLKB 0
0000* 00000 :TPASKEY
          U.3: .WORD <U.2-U.1>
0000* 00002 :TPASKEY
          U.11: .WORD <U.10-U.1>
0000* 00004 :TPASKEY
          U.19: .WORD <U.18-U.1>
0000* 00006 :TPASKEY
          U.25: .WORD <U.24-U.1>
0000* 00008 :TPASKEY
          U.31: .WORD <U.30-U.1>
0000* 0000A :TPASKEY
          U.37: .WORD <U.36-U.1>
0000* 0000C :TPASKEY
          U.43: .WORD <U.42-U.1>
0000* 0000E :TPASKEY
          U.47: .WORD <U.46-U.1>
0000* 00010 :TPASKEY
          U.55: .WORD <U.54-U.1>
00012 .BLKB 2
00014 DATACHECK_KTB::
          .BLKB 0
00014 :TPASKEY0
          U.85: .BLKB 0
0000* 00014 :TPASKEY
          U.92: .WORD <U.91-U.85>
0000* 00016 :TPASKEY
          U.98: .WORD <U.97-U.85>
00018 INITIALIZE_KTB::
          .BLKB 0
00018 :TPASKEY0
          U.108: .BLKB 0
0000* 00018 :TPASKEY
          U.110: .WORD <U.109-U.108>
0000* 0001A :TPASKEY
          U.116: .WORD <U.115-U.108>
0001C JOURNAL_KTB::
          .BLKB 0
0001C :TPASKEY0
          U.126: .BLKB 0
0000* 0001C :TPASKEY
          U.128: .WORD <U.127-U.126>
0000* 0001E :TPASKEY
```


0000* 00020 U.134: .WORD <U.133-U.126>
 ;TPASKEY
0000* 00022 U.139: .WORD <U.138-U.126>
 ;TPASKEY
0000* 00024 U.145: .WORD <U.144-U.126>
 ;TPASKEY
0000* 00026 U.151: .WORD <U.150-U.126>
 ;TPASKEY
 U.157: .WORD <U.156-U.126>
00028 OVERRIDE_KTB::
 ;BLKB 0
00028 ;TPASKEY0
 U.189: .BLKB 0
0000* 00028 ;TPASKEY
 U.191: .WORD <U.190-U.189>
0000* 0002A ;TPASKEY
 U.197: .WORD <U.196-U.189>
0000* 0002C ;TPASKEY
 U.203: .WORD <U.202-U.189>
0000* 0002E ;TPASKEY
 U.209: .WORD <U.208-U.189>
0000* 00030 ;TPASKEY
 U.215: .WORD <U.214-U.189>
0000* 00032 ;TPASKEY
 U.221: .WORD <U.220-U.189>
00034 UIC_KTB::
 ;BLKB 0
00034 ;TPASKEY0
 U.231: .BLKB 0
00034 PROCESSOR_KTB::
 ;BLKB 0
00034 ;TPASKEY0
 U.236: .BLKB 0
0000* 00034 ;TPASKEY
 U.238: .WORD <U.237-U.236>
0000* 00036 ;TPASKEY
 U.245: .WORD <U.244-U.236>
00038 PROTECTION_KTB::
 ;BLKB 0
00038 ;TPASKEY0
 U.283: .BLKB 0
0000* 00038 ;TPASKEY
 U.285: .WORD <U.284-U.283>
0000* 0003A ;TPASKEY
 U.293: .WORD <U.292-U.283>
0000* 0003C ;TPASKEY
 U.301: .WORD <U.300-U.283>
0000* 0003E ;TPASKEY
 U.309: .WORD <U.308-U.283>

.EXTRN LIB\$STOP

PSECT SUMMARY

Name Bytes Attributes

MOUNTIMG
V04-000

F 11
16-Sep-1984 01:06:29
14-Sep-1984 12:45:31

VAX-11 Bliss-32 V4.0-742
[MOUNT.SRC]MOUNTIMG.B32;1

Page 86
(24)

```
: SOWNS          424 NOVEC, WRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
: SPLITS         600 NOVEC,NOWRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
: $CODES        3426 NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
: _LIB$KEYOS      64 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(1)
: _LIB$STATES    868 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(1)
: _LIB$KEY1$     285 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(1)
```

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	100	0	1000	00:02.0
_\$255\$DUA28:[SYSLIB]CLIMAC.L32;1	14	0	0	9	00:00.1
_\$255\$DUA28:[SYSLIB]TPAMAC.L32;1	42	29	69	14	00:00.1

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:MOUNTIMG/OBJ=OBJ\$:MOUNTIMG MSRC\$:MOUNTIMG/UPDATE=(ENHS:MOUNTIMG)

```
: Size:          3426 code + 2241 data bytes
: Run Time:      01:52.3
: Elapsed Time:  03:33.8
: Lines/CPU Min: 1337
: Lexemes/CPU-Min: 67996
: Memory Used:  502 pages
: Compilation Complete
```


0245 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

