

[illegible]

```
DDDDDDDD  EEEEEEEEE  AAAAAA  CCCCCCCC  CCCCCCCC  SSSSSSSS
DDDDDDDD  EEEEEEEEE  AAAAAA  CCCCCCCC  CCCCCCCC  SSSSSSSS
DD      DD  EE      AA      AA  CC      CC  CC      SS
DD      DD  EE      AA      AA  CC      CC  CC      SS
DD      DD  EE      AA      AA  CC      CC  CC      SS
DD      DD  EE      AA      AA  CC      CC  CC      SS
DD      DD  EEEEEEE  AA      AA  CC      CC  CC      SSSSSS
DD      DD  EEEEEEE  AA      AA  CC      CC  CC      SSSSSS
DD      DD  EE      AAAAAAAAAA  CC      CC  CC      SS
DD      DD  EE      AAAAAAAAAA  CC      CC  CC      SS
DD      DD  EE      AA      AA  CC      CC  CC      SS
DD      DD  EE      AA      AA  CC      CC  CC      SS
DDDDDDDD  EEEEEEEEE  AA      AA  CCCCCCCC  CCCCCCCC  SSSSSSSS
DDDDDDDD  EEEEEEEEE  AA      AA  CCCCCCCC  CCCCCCCC  SSSSSSSS
```

```
....
....
....
....
```

```
LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```



```
1 0001 0 MODULE DEACCS (
2 0002 0 LANGUAGE (BLISS32),
3 0003 0 IDENT = 'V04-000'
4 0004 0 ) =
5 0005 1 BEGIN
6 0006 1
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 * ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 * TRANSFERRED.
20 0020 1 *
21 0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 * CORPORATION.
24 0024 1 *
25 0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *
29 0029 1 *****
30 0030 1
31 0031 1 ++
32 0032 1
33 0033 1 FACILITY: F11ACP Structure Level 2
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1 This routine implements the DEACCESS function.
38 0038 1
39 0039 1 ENVIRONMENT:
40 0040 1
41 0041 1 STARLET operating system, including privileged system services
42 0042 1 and internal exec routines.
43 0043 1
44 0044 1 --
45 0045 1
46 0046 1
47 0047 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 6-Jan-1977 23:29
48 0048 1
49 0049 1 MODIFIED BY:
50 0050 1
51 0051 1 V03-012 CDS0008 Christian D. Saether 21-Aug-1984
52 0052 1 Changes to handle stale fcbs.
53 0053 1
54 0054 1 V03-011 CDS0007 Christian D. Saether 19-Apr-1984
55 0055 1 Many changes to reflect modified access lock handling.
56 0056 1
57 0057 1 V03-010 CDS0006 Christian D. Saether 29-Dec-1983
```



```
58 0058 1 Use L_NORM linkage and BIND_COMMON macro.
59 0059 1
60 0060 1 V03-009 CDS0005 Christian D. Saether 23-Sep-1983
61 0061 1 Manually merge in ACG0343, ACG59616, STJ3109.
62 0062 1
63 0063 1 V03-008 ACG0343 Andrew C. Goldstein, 19-Jul-1983 16:46
64 0064 1 Inhibit revision date count if NORECORD is specified
65 0065 1
66 0066 1 V03-007 ACG59616 Andrew C. Goldstein, 21-Jun-1983 15:53
67 0067 1 Create common subroutine for revision and expiration dates
68 0068 1
69 0069 1 V03-006 STJ3109 Steven T. Jeffreys, 06-Jun-1983
70 0070 1 Copy FWHM from FCB to file header.
71 0071 1
72 0072 1 V03-005 CDS0004 Christian D. Saether 14-Sep-1983
73 0073 1 Modify SERIAL_FILE interface.
74 0074 1
75 0075 1 V03-004 LMP0149 L. Mark Pilant, 13-Sep-1983 11:26
76 0076 1 Correct a logic problem that caused problems during the
77 0077 1 protection check of a write attribute operation.
78 0078 1
79 0079 1 V03-003 CDS0003 Christian D. Saether 4-May-1983
80 0080 1 Synchronize processing by FID using SERIAL_FILE.
81 0081 1
82 0082 1 V03-002 CDS0002 Christian D. Saether 21-Apr-1983
83 0083 1 Modify truncate access arbitration checks to permit
84 0084 1 cluster operation. Possibly defer truncation or
85 0085 1 perform a deferred truncate operation.
86 0086 1
87 0087 1 V03-001 CDS0001 Christian D. Saether 7-Apr-1983
88 0088 1 Make mark-for-delete checks work in a cluster.
89 0089 1
90 0090 1 V02-006 ACG0258 Andrew C. Goldstein, 26-Jan-1982 16:56
91 0091 1 Fix reference to RVN 1 in expiration date processing
92 0092 1
93 0093 1 V02-005 ACG0230 Andrew C. Goldstein, 23-Dec-1981 23:46
94 0094 1 Add expiration date support
95 0095 1
96 0096 1 V02-004 ACG0247 Andrew C. Goldstein, 23-Dec-1981 20:49
97 0097 1 Update revision count only if written
98 0098 1
99 0099 1 V02-003 ACG0245 Andrew C. Goldstein, 23-Dec-1981 20:48
100 0100 1 Move queueing of spool file to cleanup
101 0101 1
102 0102 1 V02-002 ACG0167 Andrew C. Goldstein, 16-Apr-1980 19:25
103 0103 1 Previous revision history moved to [F11B.SRC]F11B.REV
104 0104 1 **
105 0105 1
106 0106 1
107 0107 1 LIBRARY 'SYSS$LIBRARY:LIB.L32';
108 0108 1 REQUIRE 'SRC$:FCPDEF.B32';
109 1099 1
110 1100 1 FORWARD ROUTINE
111 1101 1 SET REVISION : L_NORM NOVALUE, ! set revision and expiration date
112 1102 1 TRUNC_HANDLER; ! handler for delayed truncate
```



```
1103 1 GLOBAL ROUTINE DEACCESS : L_NORM =
1104 1
1105 1 ++
1106 1
1107 1 FUNCTIONAL DESCRIPTION:
1108 1
1109 1     This routine implements the DEACCESS function.
1110 1     If an attribute list is present, attributes are written.
1111 1
1112 1 CALLING SEQUENCE:
1113 1     DEACCESS ()
1114 1
1115 1 INPUT PARAMETERS:
1116 1     NONE
1117 1
1118 1 IMPLICIT INPUTS:
1119 1     IO_PACKET: I/O packet in process
1120 1     CURRENT_WINDOW: window of file
1121 1     PRIMARY_FCB: FCB of file
1122 1
1123 1 OUTPUT PARAMETERS:
1124 1     NONE
1125 1
1126 1 IMPLICIT OUTPUTS:
1127 1     NONE
1128 1
1129 1 ROUTINE VALUE:
1130 1     NONE
1131 1
1132 1 SIDE EFFECTS:
1133 1     file deaccessed
1134 1     FCB may be deleted
1135 1     header may be modified
1136 1
1137 1 --
1138 1
1139 2 BEGIN
1140 2
1141 2 LABEL
1142 2     DELAY_TRUNC;                ! truncation delay block
1143 2
1144 2 LOCAL
1145 2     DO_EXPIRE,                ! flag indicating expiration to be updated
1146 2     MODIFIED,                ! flag indicating file has been modified
1147 2     K,                        ! local copy of truncate lock count
1148 2     ABD                      : REF BBLOCKVECTOR [ABD$C_LENGTH],
1149 2     FIB                      : REF BBLOCK,      FIB
1150 2     FCB                      : REF BBLOCK,      ! pointer to FCB
1151 2     HEADER                   : REF BBLOCK;      ! file header
1152 2
1153 2 BIND_COMMON;
1154 2
1155 2 EXTERNAL ROUTINE
1156 2     REBLD_PRIM_FCB : L_NORM NOVALUE, ! rebuild primary fcb from header
1157 2     BUILD_EXT_FCBS : L_NORM NOVALUE, ! build extension fcb chain
1158 2     CONV_ACCLOCK  : L_NORM,          ! convert access lock.
1159 2     LOCK_COUNT    : L_NORM,          ! get count of granted locks.
1160 2
1161 2
1162 2
1163 2
1164 2
1165 2
1166 2
1167 2
1168 2
1169 2
1170 2
```



```
171 1160 2 SERIAL_FILE : L_NORM, ! interlock file processing
172 1161 2 TRUNC_CHECKS : L_JSB 2ARGS NOVALUE, ! parameter checks
173 1162 2 GET_FIB : L_NORM, ! get FIB of request
174 1163 2 READ_HEADER : L_NORM, ! read file header
175 1164 2 MARK_DIRTY : L_NORM, ! mark buffer for write-back
176 1165 2 WRITE_ATTRIB : L_NORM, ! write attributes routine
177 1166 2 TRUNCATE : L_NORM, ! truncate file
178 1167 2 UPDATE_FCB : L_NORM, ! update contents of FCB
179 1168 2 CHECKSUM : L_NORM, ! compute file header checksum
180 1169
181 1170 2 ! Set the cleanup flags to cause the deaccess to occur.
182 1171 2 ! Find the buffer descriptor and FIB.
183 1172 2
184 1173 2
185 1174 2 CLEANUP_FLAGS[CLF_ZCHANNEL] = 1;
186 1175 2 CLEANUP_FLAGS[CLF_DEACCESS] = 1;
187 1176 2 CLEANUP_FLAGS[CLF_DELWINDOW] = 1;
188 1177 2
189 1178 2 ! pointer to buffer descriptors
190 1179 2 ABD = .BBLOCK [.IO_PACKET[IRP$L_SVAPTE], AIB$L_DESCRIPTOR];
191 1180 2 FIB = GET_FIB (.ABD);
192 1181 2 FCB = .PRIMARY_FCB;
193 1182 2
194 1183 2 ! Synchronize further file processing.
195 1184 2
196 1185 2
197 1186 2 PRIM_LCKIDX = SERIAL_FILE (FCB [FCB$W_FID]);
198 1187 2
199 1188 2 ! Make sure irrelevant parameters are not present.
200 1189 2
201 1190 2
202 1191 2 IF .FIB[FIB$V_EXTEND]
203 1192 2 THEN ERR_STATUS (SS$_BADPARAM);
204 1193 2
205 1194 2 ! If the access lock is held in NL mode, and this file is cluster
206 1195 2 ! accessible, then set the stale flag to force rebuild of the fcbs
207 1196 2 ! from the header(s).
208 1197 2
209 1198 2
210 1199 2 IF .FCB [FCB$B_ACCLKMODE] EQL LCK$K_NLMODE
211 1200 2 AND .FCB [FCB$L_ACCLKID] NEQ 0
212 1201 2 THEN
213 1202 2 FCB [FCB$V_STALE] = 1;
214 1203 2
215 1204 2 ! Determine if the expiration date is to be updated, and if the file has
216 1205 2 ! actually been modified.
217 1206 2
218 1207 2
219 1208 2 DO_EXPIRE = .CURRENT_WINDOW[WCB$V_EXPIRE]
220 1209 2 AND NOT .FIB[FIB$V_NORECORD]
221 1210 2 AND (.CURRENT_WINDOW[WCB$L_WRITES] NEQ 0
222 1211 2 OR .CURRENT_WINDOW[WCB$L_READS] NEQ 0
223 1212 2 OR .FCB[FCB$L_EFBLK] EQL 0);
224 1213 2 MODIFIED = .CURRENT_WINDOW[WCB$V_WRITE]
225 1214 2 AND NOT .FIB[FIB$V_NORECORD]
226 1215 2 AND (.CURRENT_WINDOW[WCB$L_WRITES] NEQ 0
227 1216 2 OR .IO_PACKET[IRP$W_BCNT] GTRU ABD$C_ATTRIB
```



```
228 1217 2          OR .FIB[FIB$V_TRUNC]);
229 1218 2
230 1219 2 ! If the file is accessed for write, if we must update the expiration
231 1220 2 ! date, or if the file is marked for delete or is marked bad and this
232 1221 2 ! is the last access, read the header.
233 1222 2
234 1223 2
235 1224 2 IF .CURRENT WINDOW[WCBSV_WRITE]
236 1225 2 OR .DO EXPIRE
237 1226 4 OR ((.FCB[FCBSV_MARKDEL]
238 1227 4         OR .FCB[FCBSV_DELAYTRNC]
239 1228 4         OR .FCB[FCBSV_STALE]
240 1229 4         OR .FCB[FCBSV_BADBLK]
241 1230 4         OR .CLEANUP_FLAGS[CLF_SPOOLFILE])
242 1231 3 AND .FCB[FCBSW_REFCNT] EQL 1)
243 1232 2 THEN
244 1233 2 BEGIN
245 1234 2 HEADER = READ_HEADER (0, .FCB);
246 1235 2
247 1236 3 IF .FCB[FCBSV_STALE]
248 1237 3 THEN
249 1238 4 BEGIN
250 1239 4 REBLD_PRIM_FCB (.FCB, .HEADER);
251 1240 4 BUILD_EXT_FCBS (.HEADER);
252 1241 3 END;
253 1242 2 END;
254 1243 2
255 1244 2 ! If this the last deaccess from a file marked for delete, delete the file.
256 1245 2 ! If the file is a spool file, send it to the job controller.
257 1246 2
258 1247 2
259 1248 2 IF .FCB[FCBSW_REFCNT] EQL 1
260 1249 2 THEN
261 1250 2 BEGIN
262 1251 3 IF .FCB[FCBSV_MARKDEL]
263 1252 3 THEN
264 1253 3
265 1254 3 ! Make sure we are the only accessor left in the entire cluster.
266 1255 3
267 1256 3
268 1257 3 IF LOCK_COUNT (.FCB[FCBSL_ACCLKID]) EQL 1
269 1258 3 THEN
270 1259 3 CLEANUP_FLAGS[CLF_DELFILE] = 1;
271 1260 3
272 1261 3 IF .CLEANUP_FLAGS[CLF_SPOOLFILE]
273 1262 3 THEN CLEANUP_FLAGS[CLF_DOSPOOL] = 1;
274 1263 3
275 1264 3 ! If the FCB is marked bad, now set the bad block bit in the file header.
276 1265 3
277 1266 3
278 1267 3 IF .FCB[FCBSV_BADBLK]
279 1268 3 THEN
280 1269 4 BEGIN
281 1270 4 HEADER[FH2$V_BADBLOCK] = 1;
282 1271 4 MARK_DIRTY (.HEADER);
283 1272 3 END;
284 1273 2 END;
```



```
285 1274 2
286 1275 ! Update revision count, date, and expiration date as appropriate.
287 1276 !
288 1277
289 1278 IF .MODIFIED
290 1279 OR .DO EXPIRE
291 1280 THEN SET_REVISION (.HEADER, .MODIFIED);
292 1281
293 1282 ! Do deaccess processing for a write accessed file. If a deaccess lock
294 1283 was requested on the file, set the lock bit. Then process the write
295 1284 attributes, if any. If attributes were written, then clear the
296 1285 lock bit.
297 1286 !
298 1287
299 1288 IF .CURRENT_WINDOW[WCBSV_WRITE]
300 1289 THEN
301 1290 BEGIN
302 1291 MARK_DIRTY (.HEADER);
303 1292
304 1293 ! Update the FHWM in the file header.
305 1294
306 1295 o If the FHWM is not supported in this header, do nothing.
307 1296
308 1297 o If the volume FHWM attribute is disabled, then set the FHWM
309 1298 to the file size + 1. This will protect the contents of the
310 1299 file should it be opened and modified some time in the future
311 1300 when the volume's FHWM attribute is enabled.
312 1301
313 1302 o If the FCB FHWM is 0, and the file header supports FHWM, then
314 1303 set the header's FHWM to the file size + 1. This will likewise
315 1304 protect the file contents.
316 1305
317 1306 o If the FCB FHWM is nonzero, and the file header supports FHWM, and
318 1307 the volume FHWM attribute is enabled, simply copy the FCB FHWM to
319 1308 the file header.
320 1309
321 1310 For FHWM to be supported, the 'highwater' field in the
322 1311 header must be present. All files created on version 4
323 1312 or later systems will have this characteristic.
324 1313
325 1314
326 1315 IF .HEADER[FH2$B_IDOFFSET] GEQU ($BYTEOFFSET(FH2$L_HIGHWATER)+4)/2
327 1316 THEN
328 1317 IF .CURRENT_VCB[VCBSV_NOHIGHWATER]
329 1318 OR .FCB[FCB$L_HIGHWATER] EQL 0
330 1319 THEN
331 1320 HEADER[FH2$L_HIGHWATER] = .FCB[FCB$L_FILESIZE] + 1
332 1321 ELSE
333 1322 HEADER[FH2$L_HIGHWATER] = .FCB[FCB$L_HIGHWATER];
334 1323
335 1324
336 1325 IF .CURRENT_WINDOW[WCBSV_DLOCK]
337 1326 THEN HEADER[FH2$V_LOCKED] = 1;
338 1327
339 1328 IF .IO_PACKET[IRPSW_BCNT] GTR ABD$C_ATTRIB
340 1329 AND .USER_STATUS[0]
341 1330 THEN
```

```
342 1331 4 BEGIN
343 1332 4 WRITE ATTRIB (.HEADER, .ABD, 0);
344 1333 4 HEADER = .FILE_HEADER;
345 1334 4 HEADER[FH2$V_LOCKED] = 0;
346 1335 4 END;
347 1336 4
348 1337 4 ! If a truncate is requested, do it, if the file was write accessed and
349 1338 4 there are not other accessors now, else delay the truncation until
350 1339 4 the last reader deaccess.
351 1340 4
352 1341 4
353 1342 4 IF .FIB[FIB$V_TRUNC]
354 1343 4 AND NOT .FCB[FCB$V_MARKDEL]
355 1344 4 THEN
356 1345 4 BEGIN
357 1346 4 IF .CURRENT_VCB[VCB$V_NOALLOC]
358 1347 4 THEN ERR_EXIT (SS$WRITLCK);
359 1348 4
360 1349 4 IF .FCB[FCB$W_REFCNT] EQL 1
361 1350 4 AND LOCK_COUNT (.FCB[FCB$L_ACCLKID]) EQL 1
362 1351 4 THEN
363 1352 4 BEGIN
364 1353 4
365 1354 4 CHECKSUM (.HEADER);
366 1355 4 TRUNCATE (.FIB, .HEADER, .FIB[FIB$L_EXVBN]);
367 1356 4 CLEANUP_FLAGS[CLF_FIXFCB] = 0;
368 1357 4 UPDATE_FCB (.FILE_HEADER);
369 1358 4 END
370 1359 4 ELSE
371 1360 4 IF .FCB[FCB$W_WCNT] EQL 1 ! 1 is just us.
372 1361 4 AND (.FCB[FCB$V_EXCL] ! must be a NOLOCK somewhere
373 1362 4 OR .FCB[FCB$W_LCNT] NEQ 0 ! must be us
374 1363 4 OR CONV_ACCLOCK (LCK$K_PWMODE, .FCB))
375 1364 4 ! lock will be converted back
376 1365 4 ! in MAKE_DEACCESS
377 1366 4
378 1367 4 ! There are other readers, but no writers, accessing the file, so we will make
379 1368 4 checks to see if the truncation arguments make sense, and if so,
380 1369 4 store appropriate context in the FCB so that the last reader to deaccess
381 1370 4 the file will perform the truncation.
382 1371 4
383 1372 4
384 1373 4 THEN
385 1374 4 BEGIN
386 1375 4 LOCAL
387 1376 4 TRNVBN;
388 1377 4
389 1378 4 TRNVBN = .FIB[FIB$L_EXVBN];
390 1379 4 TRUNC_CHECKS (.FIB, .HEADER);
391 1380 4
392 1381 4 ! lock will be converted when new lock mode is calculated and lock
393 1382 4 converted in MAKE_DEACCESS. Even if it was not converted up to
394 1383 4 PW above (i.e., was already held in either), it will have to be
395 1384 4 lowered because this thread means the last writer on this node is
396 1385 4 going away.
397 1386 4
398 1387 4
```



```

399      FCB [FCB$V_DELAYTRNC] = 1;
400      FCB [FCB$L_TRUNCVBVN] = .TRNVBN;
401      FIB [FIB$L_EXVBVN] = .FCB [FCB$L_FILESIZE] + 1;
402      END
403      ELSE
404      ERR_EXIT (SS$_ACCONFLICT);
405
406      END;          ! of wanted to do a truncation
407
408      END          ! of was write accessed.
409
410      ELSE
411      DELAY_TRUNC:
412      BEGIN          ! not write accessd
413
414      BUILTIN FP;
415
416      LOCAL SAVE_US1;
417
418      IF NOT .FCB [FCB$V_DELAYTRNC]
419      THEN LEAVE DELAY_TRUNC;
420
421      IF .FCB [FCB$W_REFCNT] NEQ 1
422      OR .FCB [FCB$V_MARKDEL]
423      OR .FCB [FCB$L_TRUNCVBVN] EQL 0
424      OR LOCK_COUNT (.FCB [FCB$L_ACCLKID]) NEQ 1
425      THEN LEAVE DELAY_TRUNC;
426
427      SAVE_US1 = .USER STATUS [1];
428      CHECKSUM (.HEADER);
429
430      .FP = TRUNC_HANDLER;
431      TRUNCATE (SECOND_FIB, .HEADER, .FCB [FCB$L_TRUNCVBVN]);
432      .FP = 0;
433
434      USER_STATUS [1] = .SAVE_US1;
435
436      END;
437
438      ! Return failure to let the error cleanup do the actual deaccessing.
439      !
440
441      RETURN 0;
442
443      END;          ! end of routine DEACCESS
```

```

.TITLE DEACCS
.IDENT \V04-000\

.EXTRN REBLD PRIM_FCB, BUILD_EXT_FCBS
.EXTRN CONV_ACCLOCK, LOCK_COUNT
.EXTRN SERIAL_FILE, TRUNC_CHECKS
.EXTRN GET_FIB, READ_HEADER
.EXTRN MARK_DIRTY, WRITE_ATTRIB
.EXTRN TRUNCATE, UPDATE_FCB
.EXTRN CHECKSUM
```

				.PSECT	\$CODES\$,NOWRT,2	
			OBFC 00000	.ENTRY	DEACCESS, Save R2,R3,R4,R5,R6,R7,R8,R9,R11	1103
	5B	0000G	CF 9E 00002	MOVAB	LOCK_COUNT, R11	
	56	80	AA 9E 00007	MOVAB	-128(BASE), R6	1151
	58	0C	AA 9E 0000B	MOVAB	12(BASE), R8	
02	AA	0403	8F A8 0000F	BISW2	#1027, 2(BASE)	1176
	50	90	AA D0 00015	MOVL	-112(BASE), R0	1179
	59	2C	B0 D0 00019	MOVL	@44(R0), ABD	
			59 DD 0001D	PUSHL	ABD	1180
0000G	CF		01 FB 0001F	CALLS	#1, GET_FIB	
	54		50 D0 00024	MOVL	R0, FIB	
	52	08	AA D0 00027	MOVL	8(BASE), FCB	1181
		24	A2 9F 0002B	PUSHAB	36(FCB)	1186
0000G	CF		01 FB 0002E	CALLS	#1, SERIAL_FILE	
18	AA		50 D0 00033	MOVL	R0, 24(BASE)	
		16	A4 95 00037	TSTB	22(FIB)	1191
			06 18 0003A	BGEQ	1\$	
	03		66 E9 0003C	BLBC	(R6), 1\$	1192
	66		14 B0 0003F	MOVW	#20, (R6)	
		0B	A2 95 00042 1\$:	TSTB	11(FCB)	1199
			09 12 00045	BNEQ	2\$	
		48	A2 D5 00047	TSTL	72(FCB)	1200
			04 13 0004A	BEQL	2\$	
23	A2		01 88 0004C	BISB2	#1, 35(FCB)	1202
	50		68 D0 00050 2\$:	MOVL	(R8), R0	1208
	01		07 EF 00053	EXTZV	#7, #1, 11(R0), R5	1209
	01		15 EF 00059	EXTZV	#21, #1, (FIB), R1	
	55		51 CA 0005E	BICL2	R1, R5	
			53 D4 00061	CLRL	R3	1210
		28	A0 D5 00063	TSTL	40(R0)	
			02 13 00066	BEQL	3\$	
			53 D6 00068	INCL	R3	
			51 D4 0006A 3\$:	CLRL	R1	1211
		24	A0 D5 0006C	TSTL	36(R0)	
			02 13 0006F	BEQL	4\$	
			51 D6 00071	INCL	R1	
	51		53 C8 00073 4\$:	BISL2	R3, R1	1212
			50 D4 00076	CLRL	R0	
		3C	A2 D5 00078	TSTL	60(FCB)	
			02 12 0007B	BNEQ	5\$	
			50 D6 0007D	INCL	R0	
	50		51 C8 0007F 5\$:	BISL2	R1, R0	1210
	57		55 D2 00082	MCOML	R5, DO_EXPIRE	
	50		57 CB 00085	BICL3	DO_EXPIRE, R0, DO_EXPIRE	1213
	50		68 D0 00089	MOVL	(R8), R0	1214
	01		01 EF 0008C	EXTZV	#1, #1, 11(R0), R5	
	01		15 EF 00092	EXTZV	#21, #1, (FIB), R1	
	55		51 CA 00097	BICL2	R1, R5	1215
			53 D4 0009A	CLRL	R3	
		28	A0 D5 0009C	TSTL	40(R0)	
			02 13 0009F	BEQL	6\$	
			53 D6 000A1	INCL	R3	
	50	90	AA D0 000A3 6\$:	MOVL	-112(BASE), R0	1216
			51 D4 000A7	CLRL	R1	
	05	32	A0 B1 000A9	CMPW	50(R0), #5	

50	17	A4	51	02	1B	000AD	BLEQU	7\$	:		
			01	51	D6	000AF	INCL	R1	:		
			51	53	C8	000B1	7\$: BISL2	R3, R1	:	1217	
			53	00	EF	000B4	EXTZV	#0, #1, 23(FIB), R0	:		
			51	50	C8	000BA	BISL2	R0, R1	:	1215	
			53	55	D2	000BD	MCOML	R5, MODIFIED	:		
			51	53	CB	000C0	BICL3	MODIFIED, R1, MODIFIED	:	1224	
			50	68	D0	000C4	MOVL	(R8), R0	:		
			A0	01	E0	000C7	BBS	#1, 11(R0), 9\$	:	1225	
			1D	57	E8	000CC	BLBS	DO EXPIRE, 9\$	:	1226	
			A2	01	E0	000CF	BBS	#1, 34(FCB), 8\$	:	1227	
			22	01	E0	000D4	BBS	#1, 35(FCB), 8\$	:	1228	
			A2	23	E8	000D9	BLBS	35(FCB), 8\$	:	1229	
			09	02	E0	000DD	BBS	#2, 34(FCB), 8\$	:	1230	
			A2	6A	95	000E2	TSTB	(BASE)	:		
			01	24	18	000E4	BGEQ	10\$	:	1231	
				A2	B1	000E6	8\$: CMPW	24(FCB), #1	:		
				1E	12	000EA	BNEQ	10\$	:	1234	
				52	DD	000EC	9\$: PUSHL	FCB	:		
				7E	D4	000EE	CLRL	-(SP)	:		
			0000G	02	FB	000F0	CALLS	#2, READ_HEADER	:		
			55	50	D0	000F5	MOVL	R0, HEADER	:		
			OE	23	A2	E9	000F8	BLBC	35(FCB), 10\$	1236	
				24	BB	000FC	PUSHR	#^M<R2,R5>	:	1239	
			0000G	02	FB	000FE	CALLS	#2, REBLD_PRIM_FCB	:		
			CF	55	DD	00103	PUSHL	HEADER	:	1240	
			0000G	01	FB	00105	CALLS	#1, BUILD_EXT_FCBS	:		
			01	18	A2	B1	0010A	10\$: CMPW	:	1248	
				2C	12	0010E	BNEQ	13\$	:		
			OF	22	A2	01	E1	00110	:	1251	
					A2	DD	00115	PUSHL	72(FCB)	1257	
					01	FB	00118	CALLS	#1, LOCK_COUNT		
			6B		50	D1	0011B	CMPL	R0, #1		
			01		04	12	0011E	BNEQ	11\$		
			02	AA	20	88	00120	BISB2	#32, 2(BASE)	1259	
					6A	95	00124	11\$: TSTB	:	1261	
					03	18	00126	BGEQ	12\$		
					04	88	00128	BISB2	#4, (BASE)	1262	
			OC	22	A2	02	E1	0012B	12\$: BBC	:	1267
			35	A5	40	8F	88	00130	BISB2	#64, 53(HEADER)	1270
					55	DD	00135	PUSHL	HEADER	1271	
			0000G	CF	01	FB	00137	CALLS	#1, MARK_DIRTY		
			03	03	53	E8	0013C	13\$: BLBS	:	1278	
			09	09	57	E9	0013F	BLBC	DO EXPIRE, 15\$	1279	
					53	DD	00142	14\$: PUSHL	:	1280	
					55	DD	00144	PUSHL	HEADER		
			0000V	CF	02	FB	00146	CALLS	#2, SET_REVISION		
			50		68	D0	0014B	15\$: MOVL	:	1288	
			OB	A0	01	E0	0014E	BBS	#1, 11(R0), 16\$		
			03		00D6	31	00153	BRW	27\$		
					55	DD	00156	16\$: PUSHL	:	1291	
			0000G	CF	01	FB	00158	CALLS	#1, MARK_DIRTY		
			28		65	91	0015D	CMPB	(HEADER), #40	1315	
					1B	1F	00160	BLSSU	19\$		
			50	98	AA	D0	00162	MOVL	-104(BASE), R0	1317	
			A0	44	04	E0	00166	BBS	#4, 83(R0), 17\$		
			05	53	A2	D5	0016B	TSTL	68(FCB)	1318	

4C	A5	38	A2	08	12	0016E	BNEQ	18\$		
				01	C1	00170	ADDL3	#1, 56(FCB), 76(HEADER)		1320
		4C	A5	05	11	00176	BRB	19\$		
				44	A2	D0 00178	MOVL	68(FCB), 76(HEADER)		1322
		50		68	D0	0017D	MOVL	(R8), R0		1325
	05	14	A0	01	E1	00180	BBC	#1, 20(R0), 20\$		
		34	A5	8F	88	00185	BISB2	#64, 52(HEADER)		1326
			50	AA	D0	0018A	MOVL	-112(BASE), R0		1328
			05	A0	B1	0018E	CMPW	50(R0), #5		
				17	1B	00192	BLEQU	21\$		
		14		66	E9	00194	BLBC	(R6), 21\$		1329
				7E	D4	00197	CLRL	-(SP)		1332
		0000G	CF	0220	8F	BB 00199	PUSHR	#*M<R5,R9>		
			55	03	FB	0019D	CALLS	#3, WRITE_ATTRIB		
		34	A5	04	AA	D0 001A2	MOVL	4(BASE), HEADER		1333
			76	8F	8A	001A6	BICB2	#64, 52(HEADER)		1334
				17	A4	E9 001AB	BLBC	23(FIB), 25\$		1342
	71	22	A2	01	E0	001AF	BBS	#1, 34(FCB), 25\$		1343
			50	AA	D0	001B4	MOVL	-104(BASE), R0		1346
	05	0B	A0	04	E1	001B8	BBC	#4, 11(R0), 22\$		
				025C	8F	BF 001BD	CHMU	#604		1347
					04	001C1	RET			
			01	18	A2	B1 001C2	CMPW	24(FCB), #1		1349
					29	12 001C6	BNEQ	23\$		
				48	A2	DD 001C8	PUSHL	72(FCB)		1350
		6B		01	FB	001CB	CALLS	#1, LOCK_COUNT		
		01		50	D1	001CE	CMPL	R0, #1		
				1E	12	001D1	BNEQ	23\$		
				55	DD	001D3	PUSHL	HEADER		1354
		0000G	CF	01	FB	001D5	CALLS	#1, CHECKSUM		
				1C	A4	DD 001DA	PUSHL	28(FIB)		1355
					30	BB 001DD	PUSHR	#*M<R4,R5>		
		0000G	CF	03	FB	001DF	CALLS	#3, TRUNCATE		
		6A		02	8A	001E4	BICB2	#2, (BASE)		1356
				04	AA	DD 001E7	PUSHL	4(BASE)		1357
		0000G	CF	01	FB	001EA	CALLS	#1, UPDATE_FCB		
				7F	11	001EF	BRB	28\$		1349
			01	1C	A2	B1 001F1	CMPW	28(FCB), #1		1360
					30	12 001F5	BNEQ	26\$		
	11	22	A2	03	E0	001F7	BBS	#3, 34(FCB), 24\$		1361
				1E	A2	B5 001FC	TSTW	30(FCB)		1362
					0C	12 001FF	BNEQ	24\$		
					52	DD 00201	PUSHL	FCB		1363
					04	DD 00203	PUSHL	#4		
		0000G	CF	02	FB	00205	CALLS	#2, CONV_ACCLOCK		
			1A	50	E9	0020A	BLBC	R0, 26\$		
			53	1C	A4	D0 0020D	MOVL	28(FIB), TRNVBN		1378
			50		54	7D 00211	MOVQ	FIB, R0		1379
				0000G	30	00214	BSBW	TRUNC_CHECKS		
		23	A2	02	88	00217	BISB2	#2, 35(FCB)		1388
		50	A2	53	D0	0021B	MOVL	TRNVBN, 80(FCB)		1389
		38	A2	01	C1	0021F	ADDL3	#1, 56(FCB), 28(FIB)		1390
1C	A4			49	11	00225	BRB	28\$		1360
				0800	8F	BF 00227	CHMU	#2048		1393
					04	0022B	RET			
		3F	A2	01	E1	0022C	BBC	#1, 35(FCB), 28\$		1407
			01	18	A2	B1 00231	CMPW	24(FCB), #1		1410

DEACCS
V04-000

B 7
16-Sep-1984 00:12:40
14-Sep-1984 12:30:15

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[F11X.SRC]DEACCS.B32;1

Page 12
(2)

34	22	A2		39	12	00235	BNEQ	28\$		
				01	E0	00237	BBS	#1, 34(FCB), 28\$		1411
			50	A2	D5	0023C	TSTL	80(FCB)		1412
				2F	13	0023F	BEQL	28\$		
			48	A2	DD	00241	PUSHL	72(FCB)		1413
		6B		01	FB	00244	CALLS	#1, LOCK_COUNT		
		01		50	D1	00247	CMPL	R0, #1		
				24	12	0024A	BNEQ	28\$		
		53	04	A6	D0	0024C	MOVL	4(R6), SAVE_US1		1416
				55	DD	00250	PUSHL	HEADER		1417
0000G		CF		01	FB	00252	CALLS	#1, CHECKSUM		
		6D	0000V	CF	9E	00257	MOVAB	TRUNC_HANDLER, (FP)		1419
			50	A2	DD	0025C	PUSHL	80(FCB)		1420
				55	DD	0025F	PUSHL	HEADER		
			0244	CA	9F	00261	PUSHAB	580(BASE)		
0000G		CF		03	FB	00265	CALLS	#3, TRUNCATE		
				6D	D4	0026A	CLRL	(FP)		1421
		04	A6	53	D0	0026C	MOVL	SAVE_US1, 4(R6)		1423
				50	D4	00270	CLRL	R0		1430
				04	00272	28\$:	RET			1432

; Routine Size: 627 bytes, Routine Base: \$CODE\$ + 0000

; 444 1433 1


```
: 446      1434 1 ROUTINE TRUNC_HANDLER (SIGNAL, MECHANISM) =
: 447      1435 1
: 448      1436 1 !++
: 449      1437 1 !--
: 450      1438 1
: 451      1439 1
: 452      1440 2 BEGIN
: 453      1441 2
: 454      1442 2 MAP
: 455      1443 2         SIGNAL          : REF BBLOCK,
: 456      1444 2         MECHANISM       : REF BBLOCK;
: 457      1445 2
: 458      1446 2 IF .SIGNAL [CHF$$_SIG_NAME] EQL SS$_CMODUSER
: 459      1447 2 THEN
: 460      1448 2     $UNWIND (DEPADR = MECHANISM [CHF$_MCH_DEPTH]);
: 461      1449 2
: 462      1450 2 SS$_RESIGNAL
: 463      1451 1 END;
```

				.EXTRN SYSSUNWIND		
				0000 00000 TRUNC_HANDLER:		
				AC	D0 00002	.WORD Save nothing : 1434
				AO	D1 00006	MOVL SIGNAL, R0 : 1446
00000424	50	04		0E	12 0000E	CMPL 4(R0), #1060
				7E	D4 00010	BNEQ 1\$
				08	C1 00012	CLRL -(SP) : 1448
7E 08	AC			02	FB 00017	ADDL3 #8, MECHANISM, -(SP)
00000000G	00			8F	3C 0001E 1\$:	CALLS #2, SYSSUNWIND
	50	0918		04	00023	MOVZWL #2328, R0 : 1451
						RET

; Routine Size: 36 bytes, Routine Base: \$CODE\$ + 0273


```

: 465      1452 1 GLOBAL ROUTINE SET_REVISION (HEADER, MODE) : L_NORM NOVALUE =
: 466      1453 1
: 467      1454 1 ++
: 468      1455 1
: 469      1456 1 FUNCTIONAL DESCRIPTION:
: 470      1457 1
: 471      1458 1     This routine updates the revision count and date, and the
: 472      1459 1     expiration date in the file header as specified.
: 473      1460 1
: 474      1461 1 CALLING SEQUENCE:
: 475      1462 1     SET_REVISION (HEADER, MODE)
: 476      1463 1
: 477      1464 1 INPUT PARAMETERS:
: 478      1465 1     HEADER: address of file header to operate on
: 479      1466 1     MODE: 0 to just update expiration date
: 480      1467 1           1 to set revision and expiration date
: 481      1468 1           3 to do above and clear backup date
: 482      1469 1
: 483      1470 1 IMPLICIT INPUTS:
: 484      1471 1     NONE
: 485      1472 1
: 486      1473 1 OUTPUT PARAMETERS:
: 487      1474 1     NONE
: 488      1475 1
: 489      1476 1 IMPLICIT OUTPUTS:
: 490      1477 1     NONE
: 491      1478 1
: 492      1479 1 ROUTINE VALUE:
: 493      1480 1     NONE
: 494      1481 1
: 495      1482 1 SIDE EFFECTS:
: 496      1483 1     file header modified and marked dirty
: 497      1484 1
: 498      1485 1 --
: 499      1486 1
: 500      1487 2 BEGIN
: 501      1488 2
: 502      1489 2 LABEL
: 503      1490 2     CHECK_EXPIRE:                ! check file expiration date
: 504      1491 2
: 505      1492 2 MAP
: 506      1493 2     HEADER          : REF BBLOCK,    ! file header
: 507      1494 2     MODE            : BITVECTOR;    ! routine mode flags
: 508      1495 2
: 509      1496 2 LOCAL
: 510      1497 2     DAY_TIME        : VECTOR [2],    ! time of day
: 511      1498 2     DAY_TIME2      : VECTOR [2],    ! time of day
: 512      1499 2     UCB            : REF BBLOCK,    ! UCB of RVN 1
: 513      1500 2     PRIMARY_VCB    : REF BBLOCK,    ! VCB of RVN 1
: 514      1501 2     IDENT_AREA     : REF BBLOCK;    ! header ident area
: 515      1502 2
: 516      1503 2 BIND_COMMON;
: 517      1504 2
: 518      1505 2 EXTERNAL ROUTINE
: 519      1506 2     MARK_DIRTY      : L_NORM;        ! mark buffer for write-back
: 520      1507 2
: 521      1508 2
```



```

1509 2 ! Locate the ident area and check that the date fields are present.
1510 2 !
1511 2
1512 2 IDENT AREA = .HEADER + .HEADER[FH2$B_IDOFFSET]*2;
1513 2 IF .HEADER[FH2$B_MPOFFSET] - .HEADER[FH2$B_IDOFFSET] LSSU
1514 2 ($BYTEOFFSET(FI2$Q_EXPDATE) + FI2$S_EXPDATE) / 2
1515 2 THEN RETURN;
1516 2
1517 2 ! Update the expiration date of the file.
1518 2 !
1519 2
1520 2 MARK DIRTY (.HEADER);
1521 2 CHECK EXPIRE: BEGIN
1522 2 PRIMARY_VCB = .CURRENT_VCB;
1523 2 IF .PRIMARY_VCB[VCB$W_RVN] NEQ 0
1524 2 THEN
1525 2 BEGIN
1526 2 UCB = .VECTOR [CURRENT_RVT[RVTS$L_UCBLST], 0];
1527 2 IF .UCB EQL 0
1528 2 THEN LEAVE CHECK EXPIRE;
1529 2 PRIMARY_VCB = .UCB[UCB$L_VCB];
1530 2 END;
1531 2
1532 2 $GETTIM (TIMADR = DAY TIME);
1533 2 IF .(PRIMARY_VCB[VCB$Q_RETAINMAX]+4) NEQ 0
1534 2 THEN
1535 2 BEGIN
1536 2 SUBQ (PRIMARY_VCB[VCB$Q_RETAINMAX], DAY_TIME, DAY_TIME2);
1537 2 IF CMPQ (IDENT_AREA[FI2$Q_EXPDATE], GEQ, DAY_TIME2)
1538 2 THEN LEAVE CHECK EXPIRE;
1539 2 CH$MOVE (8, DAY_TIME2, IDENT_AREA[FI2$Q_EXPDATE]);
1540 2 END;
1541 2 END; ! end of block CHECK_EXPIRE
1542 2
1543 2 ! Increment the revision count of the file if specified.
1544 2 !
1545 2
1546 2 IF .MODE[0]
1547 2 THEN
1548 2 BEGIN
1549 2 IDENT_AREA[FI2$W_REVISION] = .IDENT_AREA[FI2$W_REVISION] + 1;
1550 2 CH$MOVE (8, DAY_TIME, IDENT_AREA[FI2$Q_REVDATE]);
1551 2 END;
1552 2
1553 2 ! Clear the backup date if requested.
1554 2 !
1555 2
1556 2 IF .MODE[1]
1557 2 THEN
1558 2 BEGIN
1559 2 (IDENT_AREA[FI2$Q_BAKDATE]) = 0;
1560 2 (IDENT_AREA[FI2$Q_BAKDATE])+4 = 0;
1561 2 END;
1562 2
1563 1 END; ! end of routine SET_REVISION
```


				.EXTRN		SYSSGETTIM		
				007C	00000	.ENTRY	SET_REVISION, Save R2,R3,R4,R5,R6	1452
5E				10	C2 00002	SUBL2	#16, SP	
51				04	AC D0 00005	MOVL	HEADER, R1	1512
50				61	9A 00009	MOVZBL	(R1), R0	
56				6140	3E 0000C	MOVAV	(R1)[R0], IDENT_AREA	
50				01	A1 9A 00010	MOVZBL	1(R1), R0	1513
52				61	9A 00014	MOVZBL	(R1), R2	
50				52	C2 00017	SUBL2	R2, R0	
17				50	D1 0001A	CMPL	R0, #23	1514
				7D	1F 0001D	BLSSU	9\$	
				51	DD 0001F	PUSHL	R1	1520
	0000G	CF		01	FB 00021	CALLS	#1, MARK DIRTY	
		52		98	AA D0 00026	MOVL	-104(BASE), PRIMARY_VCB	1522
				0E	A2 B5 0002A	TSTW	14(PRIMARY_VCB)	1523
				0E	13 0002D	BEQL	1\$	
		50		9C	AA D0 0002F	MOVL	-100(BASE), R0	1526
		50		44	A0 D0 00033	MOVL	68(R0), UCB	
				4E	13 00037	BEQL	7\$	1527
		52		34	A0 D0 00039	MOVL	52(UCB), PRIMARY_VCB	1529
				08	AE 9F 0003D	PUSHAB	DAY_TIME	1532
	00000000G	00		01	FB 00040	CALLS	#1, SYSSGETTIM	
				78	A2 D5 00047	TSTL	120(PRIMARY_VCB)	1533
				3B	13 0004A	BEQL	7\$	
6E	08	AE		74	A2 C3 0004C	SUBL3	116(PRIMARY_VCB), DAY_TIME, DAY_TIME2	1536
	04	AE		0C	AE D0 00052	MOVL	DAY_TIME, DAY_TIME2	
	04	AE		78	A2 D9 00057	SBWC	120(PRIMARY_VCB), DAY_TIME2	
		50		01	CE 0005C	MNEGL	#1, R0	1537
	04	AE		2A	A6 D1 0005F	CMPL	42(IDENT_AREA), DAY_TIME2	
				0E	19 00064	BLSS	4\$	
				08	14 00066	BGTR	2\$	
		6E		26	A6 D1 00068	CMPL	38(IDENT_AREA), DAY_TIME2	
				04	13 0006C	BEQL	3\$	
				04	1F 0006E	BLSSU	4\$	
				50	D6 00070	INCL	R0	2\$:
				50	D6 00072	INCL	R0	3\$:
02	FFFFFFF	8F		50	CF 00074	CASEL	R0, #-1, #2	4\$:
000B		000B		0006	0007C	.WORD	6\$-5\$,-	5\$:
							7\$-5\$,-	
							7\$-5\$	
26	A6		6E	08	28 00082	MOV3	#8, DAY TIME2, 38(IDENT_AREA)	1539
			09	08	AC E9 00087	BLBC	MODE, 8\$	1546
				14	A6 B6 0008B	INCW	20(IDENT_AREA)	1549
1E	A6	08	AE	08	28 0008E	MOV3	#8, DAY TIME, 30(IDENT_AREA)	1550
	03	08	AC	01	E1 00094	BBC	#1, MODE, 9\$	1556
				2E	A6 7C 00099	CLRQ	46(IDENT_AREA)	1559
					04 0009C	RET		1563

; Routine Size: 157 bytes, Routine Base: \$CODE\$ + 0297

: 577 1564 1
: 578 1565 1 END
: 579 1566 0 ELUDOM

PSECT SUMMARY

```

:
:      Name                Bytes                Attributes
:
: $CODE$                  820 NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
:

```

Library Statistics

```

:
:      File                ----- Symbols ----- Pages Processing
:                        Total   Loaded   Percent   Mapped   Time
:
: _$255$DUA28:[SYSLIB]LIB.L32;1    18619      76      0      1000      00:01.9
:

```

COMMAND QUALIFIERS

```

:
: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:DEACCS/OBJ=OBJ$:DEACCS MSRC$:DEACCS/UPDATE=(ENH$:DEACCS)
:

```

```

: Size:      820 code + 0 data bytes
: Run Time:   00:36.6
: Elapsed Time: 01:34.9
: Lines/CPU Min: 2570
: Lexemes/CPU-Min: 46076
: Memory Used: 343 pages
: Compilation Complete
:

```


0169 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

