

VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMMMMM	MMMMMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMMMMM	MMMMMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMMMMM	MMMMMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSSSSSSSS	LLL	III	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSS	LLL	III	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSS	LLL	III	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	BBBBBBBBBBBB	

```
LL      IIIIII  BBBB BBBB  FFFFFFFF  IIIIII  DDDDDDDD  NN  NN  AAAAAA  MM  MM
LL      IIIIII  BBBB BBBB  FFFFFFFF  IIIIII  DDDDDDDD  NN  NN  AAAAAA  MM  MM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MMMM  MMMM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MMMM  MMMM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MM  MM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MM  MM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MM  MM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MM  MM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MM  MM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MM  MM
LL      II      BB      FF      II      DD      DD  NN  NN  AA      AA  MM  MM
LLLLLLLL  IIIIII  BBBB BBBB  FF      IIIIII  DDDDDDDD  NN  NN  AA      AA  MM  MM
LLLLLLLL  IIIIII  BBBB BBBB  FF      IIIIII  DDDDDDDD  NN  NN  AA      AA  MM  MM
```

```
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
LLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLL  IIIIII  SSSSSSSS
```

```
0001 0 %TITLE 'LIB$FID_TO_NAME - Convert device and file ID to file specification'
0002 0 MODULE LIB$FID_TO_NAME ( ! Convert device and file ID to file spec
0003 0 IDENT = 'V04-000' ) = ! File: LIBFIDNAM.B32 Edit: V03-004
0004 1 BEGIN
0005 1
0006 1 *****
0007 1 *
0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0010 1 * ALL RIGHTS RESERVED.
0011 1 *
0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0017 1 * TRANSFERRED.
0018 1 *
0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0021 1 * CORPORATION.
0022 1 *
0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0025 1 *
0026 1 *
0027 1 *****
0028 1
0029 1 ++
0030 1 FACILITY: General Utility Library
0031 1
0032 1 ABSTRACT:
0033 1
0034 1 This routine converts a device and file identification to a file
0035 1 specification.
0036 1
0037 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0038 1
0039 1 AUTHOR: Martin L. Jack, CREATION DATE: 23-Dec-1981
0040 1
0041 1 MODIFIED BY:
0042 1
0043 1 V03-004 SOP0004 J. R. Sopka 19 September 1983
0044 1 Add test for NULL DIRECTORY to handle directory strings returned
0045 1 by the ACP for ODS-1 formatted disks.
0046 1
0047 1 V03-003 SOP0003 J. R. Sopka 29 August 1983
0048 1 Invoke SY$FILESCAN via $FILESCAN macro in STARLET. This fixes
0049 1 problem of 'insufficient arguments' caused if optional arguments
0050 1 are not supplied.
0051 1
0052 1 V03-002 SOP0002 J. R. Sopka 10 August 1983
0053 1 Re-work to reduce stack usage to be callable in Kernel mode.
0054 1 Return partial spec if any available.
0055 1
0056 1 V03-001 SOP0001 J. R. Sopka 7 June 1983
0057 1 Complete rework to access new ACP attribute which returns an
```

```

: 58      0058 1 | entire file spec in a single call.
: 59      0059 1 |
: 60      0060 1 | 1-004 - Correct two paths on which the routine could return without releasing
: 61      0061 1 | channel and event flag. MLJ 22-Mar-1982
: 62      0062 1 | 1-003 - Add ACP_STATUS parameter to return error that occurred in a file
: 63      0063 1 | operation. Print [?] whenever a lookup error occurs on a directory.
: 64      0064 1 | MLJ 15-Mar-1982
: 65      0065 1 | 1-002 - Print [] rather than [?] if zero backlink at top level. Add
: 66      0066 1 | DIRECTORY_ID parameter. Use device name if logical volume name not
: 67      0067 1 | available. Check ACP type; avoid [?] for Structure Level 1.
: 68      0068 1 | MLJ 15-Jan-1982
: 69      0069 1 | 1-001 - Original from LIBACP.B32. MLJ 23-Dec-1981
: 70      0070 1 | --

```

```
.. 72      0071 1 %SBTTL 'Declarations'
.. 73      0072 1
.. 74      0073 1 : TABLE OF CONTENTS:
.. 75      0074 1
.. 76      0075 1 FORWARD ROUTINE
.. 77      0076 1     LIB$FID_TO_NAME;           ! Convert device and file ID to file specification
.. 78      0077 1
.. 79      0078 1
.. 80      0079 1 : INCLUDE FILES:
.. 81      0080 1
.. 82      0081 1 LIBRARY 'SYS$LIBRARY:LIB';      ! System symbols
.. 83      0082 1
.. 84      0083 1
.. 85      0084 1 : SWITCHES:
.. 86      0085 1
.. 87      0086 1 SWITCHES
.. 88      0087 1     ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
.. 89      0088 1
.. 90      0089 1 : LINKAGES:
.. 91      0090 1
.. 92      0091 1 LINKAGE
.. 93      0092 1     LINKAGE_JSB_3_6 = JSB(REGISTER=0,REGISTER=1,REGISTER=2):
.. 94      0093 1     NOPRESERVE(3,4,5,6),
.. 95      0094 1
.. 96      0095 1     LINKAGE_JSB_2_2 = JSB(REGISTER=0;REGISTER=1,REGISTER=2);
.. 97      0096 1
.. 98      0097 1 : MACROS:
.. 99      0098 1
100      0099 1 MACRO MASTER_FILE_DIRECTORY ( DIRECTORY_STRING_LENGTH,
101      0100 1     DIRECTORY_STRING_ADDRESS )
102      0101 1     =
103      0102 1     ( DIRECTORY_STRING_LENGTH EQL %CHARCOUNT( '[000000]' )
104      0103 1       AND
105      0104 1       CH$EQL( DIRECTORY_STRING_LENGTH, DIRECTORY_STRING_ADDRESS,
106      0105 1         %CHARCOUNT( '[000000]' ), CH$PTR( UPLIT( '[000000]' ) ) ) )
107      0106 1     );
108      0107 1
109      0108 1 MACRO NULL_DIRECTORY ( DIRECTORY_STRING_LENGTH,
110      0109 1     DIRECTORY_STRING_ADDRESS )
111      0110 1     =
112      0111 1     ( DIRECTORY_STRING_LENGTH EQL %CHARCOUNT( '[' )
113      0112 1       AND
114      0113 1       CH$EQL( DIRECTORY_STRING_LENGTH, DIRECTORY_STRING_ADDRESS,
115      0114 1         %CHARCOUNT( '[' ), CH$PTR( UPLIT( '[' ) ) ) )
116      0115 1     );
117      0116 1
118      0117 1
119      0118 1 :
120      0119 1 : EQUATED SYMBOLS:
121      0120 1
122      0121 1 LITERAL
123      0122 1
124      0123 1     ! size of the local buffer for constructing the string to return.
125      0124 1     ! this will be truncated to the maximum RMS file specifier length for return.
126      0125 1     ! 16 is the maximum length of a logical volume name.
127      0126 1     BUF_SIZE = NAM$C_MAXRSS + 16;
128      0127 1
```

```

: 129      0128 1 |
: 130      0129 1 | FIELDS:
: 131      0130 1 |
: 132      0131 1 |      NONE
: 133      0132 1 |
: 134      0133 1 | PSECTS:
: 135      0134 1 |
: 136      0135 1 |      PSECT
: 137      0136 1 |      CODE = _LIB$CODE (READ, NOWRITE, EXECUTE, SHARE, PIC, ADDRESSING_MODE (WORD_RELATIVE)),
: 138      0137 1 |      PLIT = _LIB$CODE (READ, NOWRITE, EXECUTE, SHARE, PIC, ADDRESSING_MODE (WORD_RELATIVE)),
: 139      0138 1 |      OWN = _LIB$DATA (READ, WRITE, NOEXECUTE, NOSHARE, PIC, ADDRESSING_MODE (LONG_RELATIVE)),
: 140      0139 1 |      GLOBAL = _LIB$DATA (READ, WRITE, NOEXECUTE, NOSHARE, PIC, ADDRESSING_MODE (LONG_RELATIVE)) ;
: 141      0140 1 |
: 142      0141 1 | OWN STORAGE:
: 143      0142 1 |
: 144      0143 1 |      NONE
```

```

: 146      0144 1 %SBTTL 'External References'
: 147      0145 1
: 148      0146 1 EXTERNAL REFERENCES:
: 149      0147 1
: 150      0148 1 EXTERNAL ROUTINE
: 151      0149 1     LIB$GET_EF,           ! Allocate an event flag
: 152      0150 1     LIB$FREE_EF,          ! Deallocate an event flag
: 153      0151 1     LIB$ANALYZE_SDESC_R2: LINKAGE_JSB_2_2, ! Analyze descriptor
: 154      0152 1     LIB$SCOPY_R_DX6:      LINKAGE_JSB_3_6; ! String copy
: 155      0153 1
: 156      0154 1 EXTERNAL LITERAL          ! Completion status codes
: 157      0155 1     LIB$_INVARG,           ! Invalid argument
: 158      0156 1     LIB$_STRTRU,          ! String truncated
: 159      0157 1     LIB$_INVFILSPE;       ! Invalid file specification

```

```

161 0158 1 %SBTTL 'LIB$FID TO NAME - Convert device and file ID to file specification'
162 0159 1 GLOBAL ROUTINE LIB$FID_TO_NAME ( ! Convert device and file ID to file specification
163 0160 1     DEVICE_NAME, ! Device name string descriptor
164 0161 1     FILE_ID: REF BLOCK[,BYTE], ! File identification
165 0162 1     FILE_SPEC, ! File specification string descriptor
166 0163 1     FILE_SPEC_LENGTH, ! True length of FILE_SPEC returned
167 0164 1     DIRECTORY_ID: REF BLOCK[,BYTE], ! Directory identification
168 0165 1     ACP_STATUS ! ACP status return returned
169 0166 1 ) =
170 0167 1
171 0168 1
172 0169 1 *
173 0170 1 FUNCTIONAL DESCRIPTION:
174 0171 1     This routine converts a disk device name and file identification to a
175 0172 1     file specification by requesting the ACP file spec attribute.
176 0173 1     This routine will not operate well on Structure Level 1 disks, unless
177 0174 1     the DIRECTORY_ID parameter is used.
178 0175 1
179 0176 1 CALLING SEQUENCE:
180 0177 1     ret_status.wlc.v = LIB$FID_TO_NAME ( device-name.rt.dx,
181 0178 1     file-id.rwu.r,
182 0179 1     file-spec.wt.dx,
183 0180 1     [file-spec-length.wwu.r],
184 0181 1     [directory-id.rwu.r],
185 0182 1     [acp-status.wlu.r] )
186 0183 1
187 0184 1 INPUT PARAMETERS:
188 0185 1
189 0186 1     DEVICE_NAME
190 0187 1     Address of a descriptor for a device name.
191 0188 1     Typically this string is obtained from the
192 0189 1     NAMST_DVI field of a VAX-11 RMS name block.
193 0190 1     The device name must reference a disk device.
194 0191 1     The string must contain no more than 64
195 0192 1     characters.
196 0193 1
197 0194 1     FILE_ID
198 0195 1     Address of an array of three words that
199 0196 1     specifies a file identification. Typically
200 0197 1     this array is obtained from the NAMSW_FID field
201 0198 1     of a VAX-11 RMS name block. The $FIDDEF macro
202 0199 1     defines the structure of this parameter.
203 0200 1
204 0201 1     DIRECTORY_ID
205 0202 1     Address of an array of three words that
206 0203 1     specifies a directory file identification.
207 0204 1     Typically this array is obtained from the
208 0205 1     NAMSW_DID field of a VAX-11 RMS name block.
209 0206 1     The $FIDDEF macro defines the structure of this
210 0207 1     parameter. This is an optional parameter.
211 0208 1
212 0209 1 OUTPUT PARAMETERS:
213 0210 1
214 0211 1     FILE_SPEC
215 0212 1     Address of a descriptor for a string that
216 0213 1     receives the file specification. This is an
217 0214 1     output parameter.
218 0215 1
219 0216 1     FILE_SPEC_LENGTH
220 0217 1     Address of a word to receive the number of
221 0218 1     characters written into file-spec, not

```

```

: 218      0215 1 |
: 219      0216 1 |
: 220      0217 1 |
: 221      0218 1 |
: 222      0219 1 |
: 223      0220 1 |
: 224      0221 1 |
: 225      0222 1 |
: 226      0223 1 |
: 227      0224 1 |
: 228      0225 1 |
: 229      0226 1 |
: 230      0227 1 |
: 231      0228 1 |
: 232      0229 1 |
: 233      0230 1 |
: 234      0231 1 |
: 235      0232 1 |
: 236      0233 1 |
: 237      0234 1 |
: 238      0235 1 |
: 239      0236 1 |
: 240      0237 1 |
: 241      0238 1 |
: 242      0239 1 |
: 243      0240 1 |
: 244      0241 1 |
: 245      0242 1 |
: 246      0243 1 |
: 247      0244 1 |
: 248      0245 1 |
: 249      0246 1 |
: 250      0247 1 |
: 251      0248 1 |
: 252      0249 1 |
: 253      0250 1 |
: 254      0251 1 |
: 255      0252 1 |
: 256      0253 1 |
: 257      0254 1 |
: 258      0255 1 |
: 259      0256 1 |
: 260      0257 1 |
: 261      0258 1 |

```

counting padding in the case of a fixed-length string. If the output string is truncated to the size specified in the file-spec string, file-spec-length is set to this size. Therefore, file-spec-length can always be used by the calling program to access a valid substring of file-spec. This is an optional output parameter, passed by reference.

ACP_STATUS Address of a longword that receives the status resulting from traversing the back links. This is an optional output parameter.

The output parameters are guaranteed to be stored only if the routine value is true.

IMPLICIT INPUTS:
NONE

IMPLICIT OUTPUTS:
NONE

COMPLETION STATUS:

SS\$_NORMAL Normal successful completion

LIB\$_STRTRU Output string truncated (qualified success)

LIB\$_INVARG Required argument omitted, or device-name longer than 64 characters

LIB\$_INVFILSPE Device-name does not reference a disk

LIB\$ANALYZE_SDESC errors

\$ASSIGN errors

\$QIO errors

\$DASSGN errors

SIDE EFFECTS:
NONE

--

```

263 0259 2 BEGIN
264 0260
265 0261 2 LOCAL
266 0262 2 ACP_STATUS_VALUE: ! Value for ACP_STATUS
267 0263 2 UNSIGNED LONG,
268 0264 2 ACP_TYPE: ! ACP type attribute returned by GETDVI
269 0265 2 UNSIGNED LONG
270 0266 2 VOLATILE,
271 0267 2 ATR: ! QIO attribute list
272 0268 2 BLOCKVECTOR[ 2, ATR$S_ATRDEF, BYTE ],
273 0269 2 BUFFER: ! Buffer to construct return string
274 0270 2 VECTOR[ BUF_SIZE, BYTE ]
275 0271 2 VOLATILE,
276 0272 2 BUF_PTR: ! Pointer to BUFFER
277 0273 2 REF VECTOR[ , BYTE ],
278 0274 2 BUF_LENGTH: ! Length of return string
279 0275 2 UNSIGNED WORD
280 0276 2 INITIAL ( 0 ),
281 0277 2 CHANNEL: ! Channel number
282 0278 2 UNSIGNED WORD,
283 0279 2 DESC: ! Descriptor for various arguments
284 0280 2 VECTOR[ 2, LONG ],
285 0281 2 EFN: ! Event flag number
286 0282 2 UNSIGNED WORD,
287 0283 2 FIB:
288 0284 2 BLOCK[ FIB$C_LENGTH, BYTE ],
289 0285 2 FILE_ITM: ! Item list for $FILESCAN
290 0286 2 BLOCKVECTOR[ 4, FSCN$S_ITEM_LEN, BYTE ],
291 0287 2 FINAL_STATUS: ! Status return
292 0288 2 UNSIGNED LONG,
293 0289 2 IOSB: ! I/O status block
294 0290 2 VECTOR[ 4, WORD ];
295 0291
296 0292
297 0293 2 BUILTIN
298 0294 2 ACTUALCOUNT, ! Return number of arguments
299 0295 2 LOCC, ! 'LOCate Character' instruction
300 0296 2 NULLPARAMETER; ! Test if parameter specified
301 0297
302 0298 2 LABEL
303 0299 2 PROCESS; ! Block containing QIO processing to ACP

```

```
.. 305      0300      2      !+
.. 306      0301      2      !- Ensure that the required parameters are present.
.. 307      0302      2      !-
.. 308      0303      2      IF      ACTUALCOUNT() LSSU 3      THEN RETURN LIB$_INVARG;
.. 309      0304      2      !+
.. 310      0305      2      !- Analyze the input device name descriptor and set up the local descriptor.
.. 311      0306      2      !-
.. 312      0307      2      !- BEGIN ! block to use output registers
.. 313      0308      2      REGISTER      R1 = 1,      R2 = 2;
.. 314      0309      2      FINAL_STATUS = LIB$ANALYZE_SD$DESC_R2(.DEVICE NAME; R1, R2);
.. 315      0310      2      IF      NOT .FINAL_STATUS      THEN RETURN .FINAL_STATUS;
.. 316      0311      2      IF      NOT .FINAL_STATUS      THEN RETURN .FINAL_STATUS;
.. 317      0312      2      IF      R1 GTRU LOG$C_NAMLENGTH      THEN RETURN LIB$_INVARG;
.. 318      0313      2      DESC[0] = .R1;
.. 319      0314      2      DESC[1] = .R2;
.. 320      0315      2      END; ! block to use output registers
.. 321      0316      2      !+
.. 322      0317      2      !- Assign a channel to the device.
.. 323      0318      2      !-
.. 324      0319      2      FINAL_STATUS = $ASSIGN( DEVNAM = DESC,      CHAN = CHANNEL );
.. 325      0320      2      IF      NOT .FINAL_STATUS      THEN RETURN .FINAL_STATUS;
.. 326      0321      2      !+
.. 327      0322      2      !- Allocate an event flag.
.. 328      0323      2      !-
.. 329      0324      2      !-
.. 330      0325      2      FINAL_STATUS = LIB$GET_EF(EFN);
.. 331      0326      2      IF      NOT .FINAL_STATUS      THEN
.. 332      0327      2      THEN
.. 333      0328      2      BEGIN
.. 334      0329      2      $DASSGN(CHAN=.CHANNEL);
.. 335      0330      2      RETURN .FINAL_STATUS;
.. 336      0331      2      END;
.. 337      0332      2
```

```

339      0333 2 !+ - + - + - + - + - + - + - + - + - + - + - + -
340      0334 2 ! Beginning of block containing processing on the opened channel. Leave this
341      0335 2 ! block with FINAL_STATUS containing the status to be returned to the caller.
342      0336 2 !-
343      0337 3 PROCESS: BEGIN
344      0338 3
345      0339 3 ACP_STATUS_VALUE = SS$_NORMAL;
346      0340 3
347      0341 3 !+
348      0342 3 ! Attempt to obtain the logical volume name for the device.
349      0343 3 !-
350      0344 4 BEGIN LOCAL DEV_ITM : $ITMLST_DECL( ITEMS=2 );
351      0345 4 $ITMLST_INIT( ITMLST= DEV_ITM,
352      0346 4 ( ITMCD= DVI$_LOGVOLNAM, BUFADR= BUFFER,
353      0347 4 ( BUFLEN= LOG$C_NAMLENGTH, RETLEN= BUF_LENGTH ),
354      0348 4 ( ITMCD= DVI$_ACPTYPE, BUFADR= ACP_TYPE,
355      0349 4 ( BUFLEN= 4, RETLEN= 0 )
356      0350 4 );
357      0351 4
358      0352 4 ACP_TYPE = 0;
359      0353 4
360      0354 4 FINAL_STATUS = $GETDVI( CHAN=.CHANNEL, EFN=.EFN, ITMLST=DEV_ITM );
361      0355 4 IF NOT .FINAL_STATUS THEN LEAVE PROCESS;
362      0356 4 FINAL_STATUS = $WAITFR( EFN=.EFN );
363      0357 4 IF NOT .FINAL_STATUS THEN LEAVE PROCESS;
364      0358 3 END;
365      0359 3
366      0360 3 !+
367      0361 3 ! Set the buffer pointer.
368      0362 3 !-
369      0363 3 BUF_PTR = BUFFER + .BUF_LENGTH;
370      0364 3
371      0365 3 !+
372      0366 3 ! If a logical volume name was returned.
373      0367 3 !-
374      0368 3 IF .BUF_LENGTH NEQ 0
375      0369 3 THEN
376      0370 3 !+
377      0371 3 ! If there is no trailing colon on the logical volume name, add one.
378      0372 3 !-
379      0373 3 IF .BUF_PTR[-1] NEQ %C ':'
380      0374 3 THEN
381      0375 4 BEGIN
382      0376 4 BUF_PTR[0] = %C ':' ;
383      0377 4 BUF_PTR = .BUF_PTR + 1;
384      0378 4 BUF_LENGTH = .BUF_LENGTH + 1;
385      0379 4 END;
386      0380 3 ;

```

```
388 0381 3 !+
389 0382 3 !- Check the ACP type code to ensure that the device is a disk to which we can
390 0383 3 !- issue the necessary ACP functions.
391 0384 3 !-
392 0385 3 IF .ACP_TYPE NEQ DVI$C_ACP_F11V1
393 0386 3 AND
394 0387 3 .ACP_TYPE NEQ DVI$C_ACP_F11V2
395 0388 3 THEN
396 0389 3 BEGIN
397 0390 3 FINAL_STATUS = LIB$_INVFILSPE;
398 0391 3 LEAVE_PROCESS;
399 0392 3 END;
400 0393 3 !+
401 0394 3 !- Set up the FIB for the QIO to get file spec from the ACP.
402 0395 3 !-
403 0396 3 !-
404 0397 3 CH$FILL( 0, FIB$C_LENGTH, FIB );
405 0398 3 IF
406 0399 3 ! The DIRECTORY_ID parameter was specified
407 0400 3 NOT NULLPARAMETER( 5 )
408 0401 3 THEN
409 0402 3 ! We get the file spec for the specified directory id first.
410 0403 3 BEGIN
411 0404 3 FIB[ FIB$W_FID_NUM ] = .DIRECTORY_ID[ FID$W_NUM ];
412 0405 3 FIB[ FIB$W_FID_SEQ ] = .DIRECTORY_ID[ FID$W_SEQ ];
413 0406 3 FIB[ FIB$W_FID_RVN ] = .DIRECTORY_ID[ FID$W_RVN ];
414 0407 3 END
415 0408 3 ELSE
416 0409 3 ! We can get the file spec for the specified file id right away.
417 0410 3 BEGIN
418 0411 3 FIB[ FIB$W_FID_NUM ] = .FILE_ID[ FID$W_NUM ];
419 0412 3 FIB[ FIB$W_FID_SEQ ] = .FILE_ID[ FID$W_SEQ ];
420 0413 3 FIB[ FIB$W_FID_RVN ] = .FILE_ID[ FID$W_RVN ];
421 0414 3 END
422 0415 3 ;
423 0416 3
424 0417 3 DESC[0] = FIB$C_LENGTH;
425 0418 3 DESC[1] = FIB;
426 0419 3
427 0420 3 !+
428 0421 3 !- QIO to get full file specification attribute.
429 0422 3 !-
430 0423 3 CH$FILL( 0, 2*ATR$S_ATRDEF, ATR );
431 0424 3 ATR[ 0, ATR$W_TYPE ] = ATR$C_FILE_SPEC;
432 0425 3 ATR[ 0, ATR$W_SIZE ] = BUF_SIZE -- .BUF_LENGTH;
433 0426 3 ATR[ 0, ATR$W_ADDR ] = .BUF_PTR;
434 0427 3
435 P 0428 3 FINAL_STATUS = $QIOW( FUNC=IOS_ACCESS,
436 P 0429 3 CHAN=.CHANNEL,
437 P 0430 3 IOSB=IOSB,
438 P 0431 3 P1=DESC,
439 0432 3 P5=ATR );
440 0433 3 IF .FINAL_STATUS THEN FINAL_STATUS = .IOSB[0];
441 0434 3 ACP_STATUS_VALUE = .FINAL_STATUS;
442 0435 3 IF NOT .FINAL_STATUS THEN LEAVE_PROCESS;
```

```

444 0436 3 !+
445 0437 3 ! If a logical volume name was available,
446 0438 3 ! then discard the char count and the device name returned by QIO,
447 0439 3 ! else just discard the char count of the string returned by the QIO.
448 0440 3 !-
449 0441 4 BEGIN ! Block to bind FILE_SPEC_LEN
450 0442 4 BIND FILE_SPEC_LEN = .BUF_PTR : UNSIGNED WORD;
451 0443 4
452 0444 4 IF .BUF_LENGTH NEQ 0
453 0445 4 THEN
454 0446 5 BEGIN
455 0447 5 !+
456 0448 5 ! Locate the device portion of the file spec returned by the QIO.
457 0449 5 ! QIO returns the file specification as a counted ASCII string.
458 0450 5 !-
459 0451 5 DESC[0] = .FILE_SPEC_LEN;
460 0452 5 DESC[1] = FILE_SPEC_LEN + 2;
461 0453 5
462 0454 5 CH$FILL( 0, 2*FSCN$S_ITEM_LEN, FILE_ITM );
463 0455 5 FILE_ITM[ 0, FSCN$S_ITEM_CODE ] = FSCN$S_DEVICE; ! zero item terminates list
464 0456 5
465 0457 5 FINAL_STATUS = $FILESKAN( SRCSTR = DESC, VALUELST = FILE_ITM );
466 0458 5 IF NOT .FINAL_STATUS THEN LEAVE PROCESS;
467 0459 5
468 0460 5 !+
469 0461 5 ! Skip the device portion, and append the remainder of the file spec
470 0462 5 ! to the logical volume name.
471 0463 5 !-
472 0464 5 BUF_PTR = CH$MOVE( .FILE_SPEC_LEN - .FILE_ITM[ 0, FSCN$S_LENGTH ],
473 0465 5 .FILE_ITM[ 0, FSCN$S_ADDR ] + .FILE_ITM[ 0, FSCN$S_LENGTH ],
474 0466 5 .BUF_PTR );
475 0467 5 END
476 0468 4 ELSE
477 0469 4 !+
478 0470 4 ! Discard the character count portion of the string returned by the QIO.
479 0471 4 !-
480 0472 4 BUF_PTR = CH$MOVE( .FILE_SPEC_LEN, FILE_SPEC_LEN + 2, .BUF_PTR )
481 0473 4 ;
482 0474 4
483 0475 4 !+
484 0476 4 ! Set the buffer length count.
485 0477 4 !-
486 0478 4 BUF_LENGTH = .BUF_PTR - BUFFER;
487 0479 4
488 0480 3 END; ! Block to bind FILE_SPEC_LEN

```

```
490 0481 3 !+
491 0482 3 ! If the DIRECTORY_ID parameter was specified,
492 0483 3 ! then convert directory file spec to directory format
493 0484 3 ! and append file spec for FILE_ID to it.
494 0485 3 !-
495 0486 3 IF NOT NULLPARAMETER( 5 )
496 0487 3 THEN
497 0488 4 BEGIN LOCAL PTR : REF VECTOR[ BYTE ],
498 0489 4 CLOSE_BRACKET : UNSIGNED BYTE;
499 0490 4
500 0491 4 !+
501 0492 4 ! Extract interesting pieces of the file spec currently in BUFFER.
502 0493 4 !-
503 0494 5 BEGIN
504 0495 5 DESC[0] = .BUF_LENGTH;
505 0496 5 DESC[1] = BUFFER;
506 0497 5
507 0498 5 CH$FILL( 0, 4*FSCN$$ITEM_LEN, FILE_ITM );
508 0499 5
509 0500 5 FILE_ITM[ 0, FSCN$$ITEM_CODE ] = FSCN$_DIRECTORY;
510 0501 5 FILE_ITM[ 1, FSCN$$ITEM_CODE ] = FSCN$_NAME;
511 0502 5 FILE_ITM[ 2, FSCN$$ITEM_CODE ] = FSCN$_TYPE; ! zero item terminates list
512 0503 5
513 0504 5 FINAL_STATUS = $FILESCAN( SRCSTR = DESC, VALUELST = FILE_ITM );
514 0505 5 IF NOT .FINAL_STATUS THEN LEAVE PROCESS;
515 0506 4 END;
516 0507 4
517 0508 4 !+
518 0509 4 ! Save the close bracket character used in directory specification
519 0510 4 !-
520 0511 5 PTR = (.FILE_ITM[ 0, FSCN$_ADDR ] +
521 0512 4 .FILE_ITM[ 0, FSCN$_LENGTH ] - 1 );
522 0513 4 CLOSE_BRACKET = .PTR[0];
523 0514 4
524 0515 4
525 0516 4 IF
526 0517 4 ! directory is a root directory
527 P 0518 4 MASTER_FILE_DIRECTORY( .FILE_ITM[ 0, FSCN$_LENGTH ],
528 0519 5 .FILE_ITM[ 0, FSCN$_ADDR ] )
529 0520 4 OR
530 0521 4 ! a null directory specification has been returned
531 P 0522 4 NULL_DIRECTORY( .FILE_ITM[ 0, FSCN$_LENGTH ],
532 0523 5 .FILE_ITM[ 0, FSCN$_ADDR ] )
533 0524 4 THEN
534 0525 4 !+
535 0526 4 ! Place directory filename as root directory.
536 0527 4 !-
537 0528 4 PTR = CH$MOVE( .FILE_ITM[ 1, FSCN$_LENGTH ],
538 0529 4 .FILE_ITM[ 1, FSCN$_ADDR ],
539 0530 4 .FILE_ITM[ 0, FSCN$_ADDR ] + 1 )
540 0531 4 ELSE
541 0532 4 !+
542 0533 4 ! Place directory filename into directory format as sub-directory.
543 0534 4 ! Here we assume that the filename immediately follows the closing
544 0535 4 ! bracket of the directory spec in the file spec returned by the ACP
545 0536 4 !-
546 0537 5 BEGIN
```

```

547 0538 5 PTR[0] = %C' ';
548 0539 5 PTR = .FILE_ITMC 1, FSCNSL_ADDR ] + .FILE_ITMC 1, FSCNSW_LENGTH ];
549 0540 5 END
550 0541 4 ;
551 0542 4 ;
552 0543 4 ;
553 0544 4 ;
554 0545 4 ;
555 0546 4 ;
556 0547 4 ;
557 0548 4 ;
558 0549 4 ;
559 0550 4 ;
560 0551 4 ;
561 0552 4 ;
562 0553 4 ;
563 0554 4 ;
564 0555 4 ;
565 0556 4 ;
566 0557 4 ;
567 0558 4 ;
568 0559 4 ;
569 0560 4 ;
570 0561 4 ;
571 0562 4 ;
572 0563 4 ;
573 0564 4 ;
574 0565 4 ;
575 0566 5 ;
576 0567 6 ;
577 0568 4 ;
578 0569 4 ;
579 0570 4 ;
580 0571 4 ;
581 0572 4 ;
582 0573 4 ;
583 0574 4 ;
584 0575 4 ;
585 0576 4 ;
586 0577 4 ;
587 0578 4 ;
588 0579 4 ;
589 0580 4 ;
590 0581 4 ;
591 0582 4 ;
592 0583 4 ;
593 0584 4 ;
594 0585 4 ;
595 0586 4 ;
596 0587 5 ;
597 0588 5 ;
598 0589 5 ;
599 0590 5 ;
600 0591 5 ;
601 0592 4 ;
602 0593 4 ;
603 0594 3 ;

;
+
- Close the directory portion of the file spec with saved close bracket
PTR[0] = .CLOSE_BRACKET;

+
- Set length of current buffer contents.
BUF_PTR = .PTR + 1;
BUF_LENGTH = .BUF_PTR - BUFFER;

+
- Set up the FIB for the QIO to get filename from the ACP.
CHSFILL( 0, FIB$C_LENGTH, FIB );
FIB[ FIB$W_FID_NUM ] = .FILE_ID[ FID$W_NUM ];
FIB[ FIB$W_FID_SEQ ] = .FILE_ID[ FID$W_SEQ ];
FIB[ FIB$W_FID_RVN ] = .FILE_ID[ FID$W_RVN ];

+
- QIO to get ASCII file name attribute.
ATR[ 0, ATR$W_TYPE ] = ATR$C_ASCNAME;
ATR[ 0, ATR$W_SIZE ] = (IF (BUF_SIZE - .BUF_LENGTH) LSSU ATR$S_ASCNAME
THEN (BUF_SIZE - .BUF_LENGTH)
ELSE ATR$S_ASCNAME );
ATR[ 0, ATR$L_ADDR ] = .BUF_PTR;

DESC[0] = FIB$C_LENGTH;
DESC[1] = FIB;

FINAL_STATUS = $QIOW( FUNC=IOS$ ACCESS,
CHAN=.CHANNEL,
IOSB=IOSB,
P1=DESC,
P5=ATR );
IF .FINAL_STATUS THEN FINAL_STATUS = .IOSB[0];
ACP_STATUS_VALUE = .FINAL_STATUS;
IF NOT .FINAL_STATUS THEN LEAVE PROCESS;

+
- Scan for trailing spaces and remove these.
- QIO returns the filename with blank fill out to the specified length.
BEGIN ! block to use output registers
REGISTER R0 = 0;
LOCC( %REF(%C' '), %REF(BUF_SIZE - .BUF_LENGTH), .BUF_PTR; R0 );
BUF_LENGTH = BUF_SIZE - .R0;
BUF_PTR = BUFFER + .BUF_LENGTH;
END; ! block to use output registers

END; ! Processing for the DIRECTORY_ID parameter.

```

```
0604      0595     2 END;          ! of block PROCESS  
0605      0596     +  
0606      0597     2 End of block containing processing on the opened channel.  
0607      0598     2 FINAL_STATUS contains the most severe status encountered during processing.  
0608      0599     2 If an error has been encountered  
0609      0600     2 then we will return it rather than any error status we may obtain during  
0610      0601     2 the following wrap-up.  
0611      0602     2 + - + - + - + - + - + - + - + - + - + - + - + - + - + - +
```

```
.. 613 0603 2 !+
614 0604 2 !- Deassign the channel and deallocate the event flag.
615 0605 2 !-
616 0606 2 BEGIN
617 0607 2 LOCAL STATUS_1: UNSIGNED LONG,
618 0608 2 STATUS_2: UNSIGNED LONG;
619 0609 2 STATUS_1 = $DASSGN( CHAN=.CHANNEL );
620 0610 2 STATUS_2 = LIB$FREE_EF( EFN );
621 0611 2 IF .FINAL_STATUS THEN FINAL_STATUS = .STATUS_1;
622 0612 2 IF .FINAL_STATUS THEN FINAL_STATUS = .STATUS_2;
623 0613 2 END;
624 0614 2 !+
625 0615 2 !- Return the ACP status code if requested.
626 0616 2 !-
627 0617 2 IF NOT NULLPARAMETER(6)
628 0618 2 THEN
629 0619 2 .ACP_STATUS = .ACP_STATUS_VALUE;
630 0620 2 !+
631 0621 2 !- If we have even a partial file spec, return it.
632 0622 2 !-
633 0623 2 IF .BUF_LENGTH NEQ 0
634 0624 2 THEN
635 0625 2 BEGIN
636 0626 2 REGISTER R1 = 1;
637 0627 2 LOCAL STATUS: UNSIGNED LONG;
638 0628 2 !+
639 0629 2 !- Do not return a file spec longer than the maximum RMS can handle.
640 0630 2 !-
641 0631 2 IF .BUF_LENGTH GTRU NAM$C_MAXRSS
642 0632 2 THEN
643 0633 2 BEGIN
644 0634 2 BUF_LENGTH = NAM$C_MAXRSS;
645 0635 2 IF .FINAL_STATUS THEN FINAL_STATUS = LIB$_STRTRU;
646 0636 2 END;
647 0637 2 !+
648 0638 2 !- Copy the file specification to its destination,
649 0639 2 !- and determine if truncation occurred.
650 0640 2 !-
651 0641 2 STATUS = LIB$SCOPY_R_DX6(.BUF_LENGTH, BUFFER, .FILE_SPEC);
652 0642 2 IF .FINAL_STATUS THEN FINAL_STATUS = .STATUS;
653 0643 2 LIB$ANALYZE_SDESC R2(.FILE_SPEC; RT);
654 0644 2 IF .R1 LSSU .BUF_LENGTH
655 0645 2 THEN
656 0646 2 BEGIN
657 0647 2 BUF_LENGTH = .R1;
658 0648 2 IF .FINAL_STATUS THEN FINAL_STATUS = LIB$_STRTRU;
659 0649 2 END;
660 0650 2 END;
661 0651 2 !+
662 0652 2 !- Return the length of the file specification if requested.
663 0653 2 !-
664 0654 2 IF NOT NULLPARAMETER(4)
665 0655 2 THEN
666 0656 2
667 0657 2
668 0658 2
669 0659 2
```

```

: 670      0660      2      (.FILE_SPEC_LENGTH)<0,16> = .BUF_LENGTH;
: 671      0661
: 672      0662      2      !+
: 673      0663      2      !- Return the vanilla success status unless an error has occurred.
: 674      0664      2      !-
: 675      0665      3      RETURN (IF      .FINAL_STATUS
: 676      0666      3      THEN      $$$_NORMAL
: 677      0667      2      ELSE      .FINAL_STATUS);
: 678      0668      2
: 679      0669      1      END;

```

										.TITLE	LIB\$FID_TO_NAME LIB\$FID_TO_NAME - Convert device and file ID to	
										.IDENT	\V04-000\	
										.PSECT	_LIB\$CODE,NOWRT, SHR, PIC,2	
5D	30	30	30	30	30	30	5B	00000	P.AAA:	.ASCII	\[000000]\	
				00	00	5D	5B	00008	P.AAB:	.ASCII	\[\<0>\<0>	
										.EXTRN	LIB\$GET_EF, LIB\$FREE_EF	
										.EXTRN	LIB\$ANALYZE_SDESC_R2	
										.EXTRN	LIB\$SCOPY_R_DX6	
										.EXTRN	LIB\$_INVARG, LIB\$STRTRU	
										.EXTRN	LIB\$_INVFILE_SPE, SYSS\$ASSIGN	
										.EXTRN	SYSS\$DASSGN, SYSS\$GETDVI	
										.EXTRN	SYSS\$WAITFR, SYSS\$QIOW	
										.EXTRN	SYSS\$FILESCAN	
										.ENTRY	LIB\$FID_TO_NAME, Save R2,R3,R4,R5,R6,R7,R8,-; 0159	
											R9,R10,R11	
										MOVAB	SYSS\$DASSGN, R11	
										MOVAB	-444(SP), SP	
										CLRW	BUF_LENGTH	
										CMPB	(AP), #3	
										BLSSU	1\$	
										MOVL	DEVICE_NAME, R0	
										JSB	LIB\$ANALYZE_SDESC_R2	
										MOVL	R0, FINAL_STATUS	
										BLBC	FINAL_STATUS, 3\$	
										CMPL	R1, #64	
										BLEQU	2\$	
										MOVL	#LIB\$_INVARG, R0	
										RET		
										MOVQ	R1, DESC	
										CLRQ	-(SP)	
										PUSHAB	CHANNEL	
										PUSHAB	DESC	
										CALLS	#4, SYSS\$ASSIGN	
										MOVL	R0, FINAL_STATUS	
										BLBC	FINAL_STATUS, 3\$	
										PUSHAB	EFN	
										CALLS	#1, LIB\$GET_EF	
										MOVL	R0, FINAL_STATUS	
										BLBS	FINAL_STATUS, 4\$	
										MOVZWL	CHANNEL, -(SP)	

		6B		01	FB	00065	CALLS	#1	SYSSDASSGN		
				0329	31	00068	BRW	29\$			0331
		5A		01	D0	0006B	4\$:				0339
		50	0C	AE	9E	0006E	MOVAB	#1, ACP_STATUS VALUE			0350
		80	002C0040	8F	D0	00072	MOVAB	DEV_ITM, \$\$ITMBLKPTR			
		80	0098	CE	9E	00079	MOVAB	#2883648, (\$\$ITMBLKPTR)+			
		80	04	AE	9E	0007E	MOVAB	BUFFER, (\$\$ITMBLKPTR)+			
		80	00420004	8F	D0	00082	MOVAB	BUF_LENGTH, (\$\$ITMBLKPTR)+			
		80		AD	9E	00089	MOVAB	#4325380, (\$\$ITMBLKPTR)+			
				80	7C	0008D	CLRQ	ACP_TYPE, (\$\$ITMBLKPTR)+			
					AD	D4	0008F	CLRL	ACP_TYPE		0352
					7E	7C	00092	CLRQ	-(SP)		0354
					7E	7C	00094	CLRQ	-(SP)		
				1C	AE	9F	00096	PUSHAB	DEV_ITM		
					7E	D4	00099	CLRL	-(SP)		
		7E			AE	3C	0009B	MOVZWL	CHANNEL, -(SP)		
		7E			AE	3C	0009F	MOVZWL	EFN, -(SP)		
		00			08	FB	000A3	CALLS	#8, SYSSGETDVI		
00000000G		57			50	D0	000AA	MOVAB	RO, FINAL_STATUS		
		41			57	E9	000AD	BLBC	FINAL_STATUS, 6\$		0355
		7E			AE	3C	000B0	MOVZWL	EFN, -(SP)		0356
00000000G		00			01	FB	000B4	CALLS	#1, SYSSWAITFR		
		57			50	D0	000BB	MOVAB	RO, FINAL_STATUS		
		30			57	E9	000BE	BLBC	FINAL_STATUS, 6\$		0357
		50			CE	9E	000C1	MOVAB	BUFFER, RO		0363
		58			AE	3C	000C6	MOVZWL	BUF_LENGTH, BUF_PTR		
		58			50	C0	000CA	ADDL2	RO, BUF_PTR		
					AE	B5	000CD	TSTW	BUF_LENGTH		0368
					0C	13	000D0	BEQL	5\$		
		3A			A8	91	000D2	CMPB	-1(BUF_PTR), #58		0373
					06	13	000D6	BEQL	5\$		
		88			3A	90	000D8	MOVAB	#58, (BUF_PTR)+		0376
					AE	B6	000DB	INCW	BUF_LENGTH		0378
		01			AD	D1	000DE	5\$:	ACP_TYPE, #1		0385
					10	13	000E2	BEQL	7\$		
		02			AD	D1	000E4	CMPB	ACP_TYPE, #2		0387
					0A	13	000E8	BEQL	7\$		
		57	00000000G		8F	D0	000EA	MOVAB	#LIB\$_INVFILSPE, FINAL_STATUS		0390
					0208	31	000F1	6\$:			0391
		6E			00	2C	000F4	7\$:			0397
					AE		000FB	MOVCS	#0, (SP), #0, #64, FIB		
0040	8F				50						
		05			6C	91	000FD	CMPB	(AP), #5		0400
					0B	1F	00100	BLSSU	8\$		
					14	AC	D5	00102	TSTL	20(AP)	
					06	13	00105	BEQL	8\$		
		50			AC	D0	00107	MOVAB	DIRECTORY_ID, RO		0404
					04	11	0010B	BRB	9\$		
		50			AC	D0	0010D	8\$:	FILE_ID, RO		0411
		54			60	D0	00111	9\$:	(RO), FIB+4		
		58			A0	B0	00115	MOVW	4(RO), FIB+8		0413
		0090			8F	9A	0011A	MOVZBL	#64, DESC		0417
		0094			AE	9E	00120	MOVAB	FIB, DESC+4		0418
10					00	2C	00126	MOVCS	#0, (SP), #0, #16, ATR		0423
					EC	AD	0012B				
					2E	B0	0012D	MOVW	#46, ATR+2		0424
		EE	AD		AE	A3	00131	SUBW3	BUF_LENGTH, #271, ATR		0425
		EC	AD		58	D0	00139	MOVL	BUF_PTR, ATR+4		0426
		F0	AD								

			7E	D4	0013D	CLRL	-(SP)	0432
		EC	AD	9F	0013F	PUSHAB	ATR	
			7E	7C	00142	CLRL	-(SP)	
			7E	D4	00144	CLRL	-(SP)	
		00A4	CE	9F	00146	PUSHAB	DESC	
			7E	7C	0014A	CLRL	-(SP)	
		48	AE	9F	0014C	PUSHAB	IOSB	
			32	DD	0014F	PUSHL	#50	
		7E	28	AE	3C	MOVZWL	CHANNEL, -(SP)	
			7E	D4	00155	CLRL	-(SP)	
	00000000G	00	0C	FB	00157	CALLS	#12, SYSSQIOW	
		57	50	D0	0015E	MOVL	R0, FINAL_STATUS	
		04	57	E9	00161	BLBC	FINAL_STATUS, 10\$	0433
		57	28	AE	3C	MOVZWL	IOSB, FINAL_STATUS	
		5A	57	D0	00168	MOVL	FINAL_STATUS, ACP_STATUS_VALUE	0434
		83	57	E9	0016B	BLBC	FINAL_STATUS, 6\$	0435
		56	58	D0	0016E	MOVL	BUF_PTR, R6	0442
			04	AE	B5	TSTW	BUF_LENGTH	0444
			44	13	00174	BEQL	11\$	
	0090	CE	66	3C	00176	MOVZWL	(R6), DESC	0451
	0094	CE	02	A6	9E	MOVAB	2(R6), DESC+4	0452
10	00	6E	00	2C	00181	MOVCS	#0, (SP), #0, #16, FILE_ITM	0454
			30	AE	00186			
	32	AE	03	B0	00188	MOVW	#3, FILE_ITM+2	0455
			7E	D4	0018C	CLRL	-(SP)	0457
			34	AE	9F	PUSHAB	FILE_ITM	
			0098	CE	9F	PUSHAB	DESC	
	00000000G	00	03	FB	00195	CALLS	#3, SYSSFILESCAN	
		57	50	D0	0019C	MOVL	R0, FINAL_STATUS	
		2F	57	E9	0019F	BLBC	FINAL_STATUS, 13\$	0458
		53	66	3C	001A2	MOVZWL	(R6), R3	0464
		50	30	AE	3C	MOVZWL	FILE_ITM, R0	
		53	50	C2	001A9	SUBL2	R0, R3	
		50	30	AE	3C	MOVZWL	FILE_ITM, R0	0465
		50	34	AE	C0	ADDL2	FILE_ITM+4, R0	
68		60	53	28	001B4	MOVCS	R3, (R0), (BUF_PTR)	0466
			05	11	001B8	BRB	12\$	
68	02	A6	66	28	001BA	MOVCS	(R6), 2(R6), (BUF_PTR)	0472
		58	53	D0	001BF	MOVL	R3, BUF_PTR	
		50	0098	CE	9E	MOVAB	BUFFER, R0	0478
04	AE	58	50	A3	001C7	SUBW3	R0, BUF_PTR, BUF_LENGTH	
		05	6C	91	001CC	CMPB	(AP), #5	0486
			03	1E	001CF	BGEQU	14\$	
			0128	31	001D1	BRW	22\$	
			14	AC	D5	TSTL	20(AP)	
			F8	13	001D7	BEQL	13\$	
	0090	CE	04	AE	3C	MOVZWL	BUF_LENGTH, DESC	0495
	0094	CE	0098	CE	9E	MOVAB	BUFFER, DESC+4	0496
		6E	00	2C	001E6	MOVCS	#0, (SP), #0, #32, FILE_ITM	0498
			30	AE	001EB			
	32	AE	05	B0	001ED	MOVW	#5, FILE_ITM+2	0500
	3A	AE	06	B0	001F1	MOVW	#6, FILE_ITM+10	0501
	42	AE	07	B0	001F5	MOVW	#7, FILE_ITM+18	0502
			7E	D4	001F9	CLRL	-(SP)	0504
			34	AE	9F	PUSHAB	FILE_ITM	
			0098	CE	9F	PUSHAB	DESC	
	00000000G	00	03	FB	00202	CALLS	#3, SYSSFILESCAN	

08	00	64	FDC8	57	50	DO	00209	MOVL	R0, FINAL STATUS	0505
				57	E9	0020C		BLBC	FINAL STATUS, 13\$	0511
				54	AE	DO	0020F	MOVL	FILE_ITM+4, R4	0512
				55	AE	3C	00213	MOVZWL	FILE_ITM, R5	0511
				56	FF	A544	9E 00217	MOVAB	-1(R5)[R4], PTR	0513
				59	66	90	0021C	MOVB	(PTR), CLOSE_BRACKET	0519
				08	55	B1	0021F	CMPW	R5, #8	
					0A	12	00222	BNEQ	15\$	
					55	2D	00224	CMPC5	R5, (R4), #0, #8, P.AAA	
					CF		00229			
					0F	13	0022C	BEQL	16\$	
					55	B1	0022E	CMPW	R5, #2	0523
					16	12	00231	BNEQ	17\$	
					55	2D	00233	CMPC5	R5, (R4), #0, #2, P.AAB	
					CF		00238			
					0C	12	0023B	BNEQ	17\$	
					38	AE	28 0023D	MOV3	FILE_ITM+8, @FILE_ITM+12, 1(R4)	0530
					53	DO	00244	MOVL	R3, PTR	
					0B	11	00247	BRB	18\$	0528
					2E	90	00249	MOVB	#46, (PTR)	0538
					38	AE	3C 0024C	MOVZWL	FILE_ITM+8, PTR	0539
					3C	AE	C0 00250	ADDL2	FILE_ITM+12, PTR	
					59	90	00254	MOVB	CLOSE_BRACKET, (PTR)	0546
					01	A6	9E 00257	MOVAB	1(R6), BUF_PTR	0551
					50	CE	9E 0025B	MOVAB	BUFFER, R0	0552
					58	A3	00260	SUBW3	R0, BUF_PTR, BUF_LENGTH	
					6E	00	2C 00265	MOV3	#0, (SP), #0, #64, FIB	0557
					50	AE	0026C			
					50	AC	DO 0026E	MOVL	FILE_ID, R0	0558
					54	AE	60 DO 00272	MOVL	(R0), FIB+4	
					58	AE	04 A0 B0 00276	MOVW	4(R0), FIB+8	0560
					EE	AD	10 B0 0027B	MOVW	#16, ATR+2	0565
					52	AE	3C 0027F	MOVZWL	BUF_LENGTH, R2	0566
					52	C2	9E 00283	MOVAB	-27T(R2), R2	
					50	CE	00288	MNEGL	R2, R0	
					8F	50	D1 0028B	CMPL	R0, #86	
						05	1E 00292	BGEQU	19\$	
						52	CE 00294	MNEGL	R2, R0	0567
						04	11 00297	BRB	20\$	
						8F	9A 00299	MOVZBL	#86, R0	0566
						50	B0 0029D	MOVW	R0, ATR	
						58	DO 002A1	MOVL	BUF_PTR, ATR+4	0569
						8F	9A 002A5	MOVZBL	#64, DESC	0571
						AE	9E 002AB	MOVAB	FIB, DESC+4	0572
						7E	D4 002B1	CLRL	-(SP)	0578
						AD	9F 002B3	PUSHAB	ATR	
						7E	7C 002B6	CLRQ	-(SP)	
						7E	D4 002B8	CLRL	-(SP)	
						CE	9F 002BA	PUSHAB	DESC	
						7E	7C 002BE	CLRQ	-(SP)	
						AE	9F 002C0	PUSHAB	IOSB	
						32	DD 002C3	PUSHL	#50	
						AE	3C 002C5	MOVZWL	CHANNEL, -(SP)	
						7E	D4 002C9	CLRL	-(SP)	
						0C	FB 002CB	CALLS	#12, SYSSQIOW	
						50	DO 002D2	MOVL	R0, FINAL STATUS	
						57	E9 002D5	BLBC	FINAL STATUS, 21\$	0579

	57	28	AE	3C	002D8	MOVZWL	IOSB, FINAL STATUS	:	0580
	5A		57	D0	002DC	21\$: MOVL	FINAL STATUS, ACP_STATUS_VALUE	:	0581
	1A		57	E9	002DF	BLBC	FINAL STATUS, 22\$	:	0589
	50		52	CE	002E2	MNEGL	R2, R0	:	
04	68	010F	50	20	3A	002E5	LOCC	#32, R0, (BUF_PTR)	:
AE			50	A3	002E9	SUBW3	R0, #271, BUF_LENGTH	:	0590
	8F		50	9E	002F0	MOVAB	BUFFER, R0	:	0591
	50	0098	58	CE	002F5	MOVZWL	BUF_LENGTH, BUF_PTR	:	
	58	04	50	3C	002F9	ADDL2	R0, BUF_PTR	:	
	7E		6E	3C	002FC	22\$: MOVZWL	CHANNEL, -(SP)	:	0609
	6B		01	FB	002FF	CALLS	#1, SYS\$DASSGN	:	
	52		50	D0	00302	MOVL	R0, STATUS_1	:	
		08	AE	9F	00305	PUSHAB	EFN	:	0610
00000000G	00		01	FB	00308	CALLS	#1, LIB\$FREE_EF	:	0611
	09		57	E9	0030F	BLBC	FINAL STATUS, 23\$	:	0612
	57		52	D0	00312	MOVL	STATUS_1, FINAL STATUS	:	
	03		57	E9	00315	BLBC	FINAL STATUS, 23\$	:	0618
	57		50	D0	00318	MOVZWL	STATUS_2, FINAL STATUS	:	
	06		6C	91	0031B	23\$: CMPB	(AP), #6	:	
		18	09	1F	0031E	BLSSU	24\$	:	
			AC	D5	00320	TSTL	24(AP)	:	
	18	BC	04	13	00323	BEQL	24\$	:	
			5A	D0	00325	MOVL	ACP_STATUS_VALUE, @ACP_STATUS	:	0620
		04	AE	B5	00329	24\$: TSTW	BUF_LENGTH	:	0625
			50	13	0032C	BEQL	27\$	:	
00FF	8F	04	AE	B1	0032E	CMPW	BUF_LENGTH, #255	:	0633
			0F	1B	00334	BLEQU	25\$	:	
04	AE	FF	8F	9B	00336	MOVZBW	#255, BUF_LENGTH	:	0636
	07		57	E9	0033B	BLBC	FINAL STATUS, 25\$	:	0637
	57	00000000G	8F	D0	0033E	MOVL	#LIB\$STRTRU, FINAL STATUS	:	
	51	0098	CE	9E	00345	25\$: MOVAB	BUFFER, R1	:	0644
	52	0C	AC	D0	0034A	MOVL	FILE_SPEC, R2	:	
	50	04	AE	3C	0034E	MOVZWL	BUF_LENGTH, R0	:	
		00000000G	00	16	00352	JSB	LIB\$SCOPY R DX6	:	
	03		57	E9	00358	BLBC	FINAL STATUS, 26\$	:	0645
	57		50	D0	0035B	MOVL	STATUS, FINAL STATUS	:	
	50	0C	AC	D0	0035E	26\$: MOVL	FILE_SPEC, R0	:	0646
		00000000G	00	16	00362	JSB	LIB\$ANALYZE SDESC R2	:	
51	04	AE	00	ED	00368	CMPZV	R0, #16, BUF_LENGTH, R1	:	0647
			0E	1B	0036E	BLEQU	27\$	:	
	04		51	B0	00370	MOVW	R1, BUF_LENGTH	:	0650
	07		57	E9	00374	BLBC	FINAL STATUS, 27\$	:	0651
	57	00000000G	8F	D0	00377	MOVL	#LIB\$STRTRU, FINAL STATUS	:	
	04		6C	91	0037E	27\$: CMPB	(AP), #4	:	0658
		10	0A	1F	00381	BLSSU	28\$	:	
			AC	D5	00383	TSTL	16(AP)	:	
	10	BC	05	13	00386	BEQL	28\$	:	
		04	AE	B0	00388	MOVW	BUF_LENGTH, @FILE_SPEC_LENGTH	:	0660
	04		57	E9	0038D	28\$: BLBC	FINAL STATUS, 29\$	:	0665
	50		01	D0	00390	MOVL	#1, R0	:	
			04	00393	RET			:	
	50		57	D0	00394	29\$: MOVL	FINAL STATUS, R0	:	0667
			04	00397	RET			:	0669

; Routine Size: 920 bytes, Routine Base: _LIB\$CODE + 000C

LIB\$FID_TO_NAME LIB\$FID_TO_NAME - Convert device and file ID to 16-Sep-1984 02:26:30 VAX-11 Bliss-32 V4.0-742
V04-000 LIB\$FID_TO_NAME - Convert device and file ID to 14-Sep-1984 13:34:27 [VMSLIB.SRC]LIBFIDNAM.B32;1

Page 22
(12)

: 681 0670 1 END ! End of module LIB\$FID_TO_NAME
: 682 0671 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
_LIB\$CODE	932	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	49	0	1000	00:01.9

COMMAND QUALIFIERS

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

: Size: 920 code + 12 data bytes
: Run Time: 00:21.1
: Elapsed Time: 00:23.1
: Lines/CPU Min: 1910
: Lexemes/CPU-Min: 19680
: Memory Used: 305 pages
: Compilation Complete

0436 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY