

FFFFFFFFFFFFFF	DDDDDDDDDDDD		LLL
FFFFFFFFFFFFFF	DDDDDDDDDDDD		LLL
FFFFFFFFFFFFFF	DDDDDDDDDDDD		LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFFFFFFFFFFFFF	DDD	DDD	LLL
FFFFFFFFFFFFFF	DDD	DDD	LLL
FFFFFFFFFFFFFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDD	DDD	LLL
FFF	DDDDDDDDDDDD	DDD	LLLLLLLLLLLLLLLL
FFF	DDDDDDDDDDDD		LLLLLLLLLLLLLLLL
FFF	DDDDDDDDDDDD		LLLLLLLLLLLLLLLL

```

FFFFFFFFF DDDDDDD LL      GGGGGGG EEEEEEEEE NN      NN
FFFFFFFFF DDDDDDD LL      GGGGGGG EEEEEEEEE NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FFFFFFFFF DD      DD LL      GG          EE          NN      NN
FFFFFFFFF DD      DD LL      GG          EE          NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FF          DD      DD LL      GG          EE          NN      NN
FF          DDDDDDD LLLLLLLLL GGGGGG  EEEEEEEEE NN      NN
FF          DDDDDDD LLLLLLLLL GGGGGG  EEEEEEEEE NN      NN

```

```

LL          IIIII SSSSSSS
LL          IIIII SSSSSSS
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 IIIII SSSSSSS
LLLLLLLLLL IIIII SSSSSSS

```

.....

```

1 0001 0 %TITLE 'VAX-11 FDL Utilities'
2 0002 0 MODULE FDLGEN ( IDENT='V04-000',
3 0003 0 ADDRESSING_MODE ( EXTERNAL = GENERAL ),
4 0004 0 ADDRESSING_MODE ( NONEXTERNAL = GENERAL ),
5 0005 0 OPTLEVEL=3
6 0006 0 ) =
7 0007 0
8 0008 1 BEGIN
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 SUBJECT TO CHANGE WITHOUT NOTICE
24 0024 1 * AND SHOULD NOT BE CONSTRUED 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 NO SUPPLIED BY DIGITAL.
29 0029 1 *
30 0030 1 *
31 0031 1 :*****

```

```

33 0032 1 ++
34 0033 1
35 0034 1 Facility: VAX-11 FDL Utilities
36 0035 1
37 0036 1 Abstract:
38 0037 1 Callable routines
39 0038 1
40 0039 1 Contents:
41 0040 1 FDL$$GEN_SPEC
42 0041 1 GEN_PRIMARY
43 0042 1 CHECK_XAB
44 0043 1 FDL$$CHECK_BLOCK
45 0044 1 FDL$$FORMAT_LINE
46 0045 1 FETCH_FIELD
47 0046 1 FDL$$OUTPUT_LINE
48 0047 1
49 0048 1 Environment:
50 0049 1
51 0050 1 VAX/VMS Operating System
52 0051 1
53 0052 1 --
54 0053 1
55 0054 1
56 0055 1 Author: Ken F Henderson Jr Creation Date 2 Dec 1982
57 0056 1
58 0057 1
59 0058 1 Modified by:
60 0059 1
61 0060 1 V03-016 DAS0001 David Solomon 06 Jul 1984
62 0061 1 Don't generate DATA_FILL or INDEX_FILL secondary's
63 0062 1 for the key primary if $XABALL's don't exist.
64 0063 1
65 0064 1 V03-015 RRB0015 Rowland R. Bradley 29 Feb 1984
66 0065 1 Comment out or remove references
67 0066 1 ACLs and Erase_on_Delete
68 0067 1
69 0068 1 V03-013 KFH0012 Ken Henderson 8 Oct 1983
70 0069 1 Fix generation of bits that have
71 0070 1 inverted sense when set:
72 0071 1 Data Key Comp, Data Rec Comp,
73 0072 1 Index Comp, and Block Span
74 0073 1
75 0074 1 V03-012 KFH0011 Ken Henderson 26 Sep 1983
76 0075 1 Fix generation of format=fixed.
77 0076 1
78 0077 1 V03-011 KFH0010 Ken Henderson 23 Aug 1983
79 0078 1 Fixed calls to GET_VM and FREE_VM.
80 0079 1
81 0080 1 V03-010 KFH0009 Ken Henderson 29 Jul 1983
82 0081 1 Fixed CHECK_XAB and FETCH_FIELD
83 0082 1 Check status of calls to CIB$ and SYSS
84 0083 1 Changed RU JNL bits
85 0084 1 Added DEFERRED_WRITE, ERASE_ON_DELETE
86 0085 1
87 0086 1 V03-009 KFH0008 Ken Henderson 31 Jan 1983
88 0087 1 Enabled XAB$C_INB and XAB$C_BN8
89 0088 1

```

```

: 90
: 91
: 92
: 93
: 94
: 95
: 96
: 97
: 98
: 99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117

```

```

0089 1
0090 1
0091 1
0092 1
0093 1
0094 1
0095 1
0096 1
0097 1
0098 1
0099 1
0100 1
0101 1
0102 1
0103 1
0104 1
0105 1
0106 1
0107 1
0108 1
0109 1
0110 1
0111 1
0112 1
0113 1
0114 1
0115 1
0116 1 !****

```

```

V03-008 KFH0007 Ken Henderson 28 Jan 1983
Only stuff FDL$AB_AREA_BKZ if not
deallocating blocks

V03-007 KFH0006 Ken Henderson 21 Jan 1983
Fixed deallocation of file name
in FDL$$CHECK_BLOCKS

V03-006 KFH0005 Ken Henderson 6 Jan 1983
Fixed deallocation of key names
in CHECK_XAB

V03-005 KFH0004 Ken Henderson 5 Jan 1983
Added alloc/dealloc of FDL$AB_AREA_BZK
to FDL$$CHECK_BLOCKS

V03-004 KFH0003 Ken Henderson 4 Jan 1983
Fixed CHECK_XAB deallocation of
KEYXABs and FDL$$CHECK_BLOCKS
deallocation of NAM blocks

V03-003 KFH0002 Ken Henderson 30 Dec 1982
Fixed broken branches

V03-002 KFH0001 Ken Henderson 15 Dec 1982
Finished IDENT, SYSTEM, POSITION

```

```

119 0117 1
120 0118 1 PSECT
121 0119 1     OWN      = FDL$OWN      (PIC),
122 0120 1     GLOBAL   = FDL$GLOBAL (PIC),
123 0121 1     PLIT     = FDL$PLIT   (SHARE,PIC),
124 0122 1     CODE     = FDL$CODE   (SHARE,PIC);
125 0123 1
126 0124 1 LIBRARY 'SYSS$LIBRARY:STARLET';
127 0125 1 REQUIRE 'SRCS:FDLUTIL';
128 0310 1 REQUIRE 'LIBS:FDLPARDEF';
129 0849 1
130 0850 1 EXTERNAL ROUTINE
131 0851 1     SYSS$FAO,
132 0852 1     SYSS$ASC$IM,
133 0853 1     LIBS$GET_VM,
134 0854 1     LIBS$FREE_VM,
135 0855 1     STR$APPEND,
136 0856 1     FDL$$FREE_VM,
137 0857 1     FDL$$READ_ERROR      : NO'VALUE,
138 0858 1     FDL$$RMS_OPEN_ERROR  : NOVALUE;
139 0859 1
140 0860 1 FORWARD ROUTINE
141 0861 1     GEN_PRIMARY,
142 0862 1     FDL$$FORMAT_LINE,
143 0863 1     FDL$$OUTPUT_LINE,
144 0864 1     FDL$$CHECK_BLOCKS,
145 0865 1     FETCH_FIELD;
146 0866 1
147 0867 1
148 0868 1 EXTERNAL
149 0869 1     FDL$AB_PRI TABLE      :
150 0870 1     BLOCKVECTOR [ FDL$C_PRITAB_SIZE, FDL$C_PRIBLK_SIZE ] FIELD (PRITAB_FIELDS),
151 0871 1
152 0872 1     FDL$AB_SEC TABLE     :
153 0873 1     BLOCKVECTOR [ FDL$C_SECTAB_SIZE, FDL$C_SECBLK_SIZE ] FIELD (SECTAB_FIELDS),
154 0874 1
155 0875 1     FDL$AB_OUT_STRING     : REF DESC_BLK,
156 0876 1     FDL$AB_GENFAB        : REF BLOCK [ ,BYTE ],
157 0877 1     FDL$AB_GENRAB        : REF BLOCK [ ,BYTE ],
158 0878 1     FDL$AB_FDL_RAB       : BLOCK [ ,BYTE ],
159 0879 1     FDL$AB_CTRL          : BLOCK [ ,BYTE ],
160 0880 1     FDL$AB_BLOCK_BLK     : VECTOR [ 4, LONG ],
161 0881 1     FDL$AB_AREA_BLK      : REF VECTOR [ ,BYTE ],
162 0882 1     FDL$GL_INVBCK_PTR,
163 0883 1     FDL$GL_STNUPTR,
164 0884 1     FDL$GL_MAXLINE,
165 0885 1     FDL$GL_SECNUM,
166 0886 1     FDL$GL_PRIMARY,
167 0887 1     FDL$GL_PRINUM,
168 0888 1     FDL$GL_SECONDARY,
169 0889 1     FDL$AB_FDL_STRING    : DESC_BLK,
170 0890 1     FDL$AB_LINE         : DESC_BLK,
171 0891 1     FDL$AB_UPCASED      : DESC_BLK,
172 0892 1     FDL$AB_KEY_TABLE,
173 0893 1     FDL$AB_STATE_TABLE,
174 0894 1     FDL$AB_TPARSE_BLOCK : BLOCK [ ,BYTE ];
175 0895 1

```

```

: 176 0896 1 OWN
: 177 0897 1
: 178 0898 1 TEMP_DESC
: 179 0899 1
: 180 0900 1 FAO_DESC
: 181 0901 1
: 182 0902 1
: 183 0903 1 TIME_BUF
: 184 0904 1
: 185 0905 1
: 186 0906 1 TIME_TEMP
: 187 0907 1 FAO_LENGTH
: 188 0908 1 FAO_PARAM
: 189 0909 1 FAO_PARAM2
: 190 0910 1 FAO_PARAM3
: 191 0911 1 FAO_PARAM4
: 192 0912 1 STRBYTES
: 193 0913 1 OCHAR
: 194 0914 1 XABPRO_PTR
: 195 0915 1 XABRDT_PTR
: 196 0916 1 XABDAT_PTR
: 197 0917 1 XABJNL_PTR
: 198 0918 1 XABALL_PTR
: 199 0919 1 XABKEY_PTR
: 200 0920 1 SAVE_POINTER
: 201 0921 1
: 202 0922 1 PROT_VALUES
: 203 0923 1 %ASCID 'RWED',
: 204 0924 1 %ASCID 'WED',
: 205 0925 1 %ASCID 'RED',
: 206 0926 1 %ASCID 'ED',
: 207 0927 1 %ASCID 'RWD',
: 208 0928 1 %ASCID 'WD',
: 209 0929 1 %ASCID 'RD',
: 210 0930 1 %ASCID 'D',
: 211 0931 1 %ASCID 'RWE',
: 212 0932 1 %ASCID 'WE',
: 213 0933 1 %ASCID 'RE',
: 214 0934 1 %ASCID 'E',
: 215 0935 1 %ASCID 'RW',
: 216 0936 1 %ASCID 'W',
: 217 0937 1 %ASCID 'R',
: 218 0938 1 %ASCID '' );
: 219 0939 1
: 220 0940 1
: 221 0941 1 DEFINE_ERROR_CODES;

```

```

: DESC_BLK
PRESETT [ DSC$B_CLASS ] = DSC$K_CLASS_S,
[ DSC$B_DTYPE ] = DSC$K_DTYPE_T ),
: DESC_BLK
PRESETT [ DSC$B_CLASS ] = DSC$K_CLASS_S,
[ DSC$B_DTYPE ] = DSC$K_DTYPE_T ),
: DESC_BLK
PRESETT [ DSC$B_CLASS ] = DSC$K_CLASS_S,
[ DSC$B_DTYPE ] = DSC$K_DTYPE_T ),
: VECTOR [ 23, BYTE ],
: LONG,
: LONG,
: LONG,
: LONG,
: LONG,
: LONG,
: REF VECTOR [ , BYTE ],
: LONG,
: LONG,
: LONG,
: LONG,
: REF BLOCK [ , BYTE ],
: REF BLOCK [ , BYTE ],
: REF BLOCK [ , BYTE ],
: VECTOR [ 16, LONG ] INITIAL (

```

```

0942 1 %SBTTL 'FDL$$GEN_SPEC'
0943 1 GLOBAL ROUTINE FDL$$GEN_SPEC =
0944 1 ++
0945 1
0946 1 Functional Description:
0947 1
0948 1 This routine xxxxxxxxxxxxxxxx
0949 1
0950 1 Calling Sequence:
0951 1
0952 1 fdl$$gen_spec( fdl_string )
0953 1
0954 1 Input Parameters:
0955 1
0956 1 fdl_desc - descriptor of the fdl file name string (required)
0957 1 file_name - descriptor file name to override the name specified
0958 1 in the fdl file (optional)
0959 1 default_name - descriptor default file name to override the default
0960 1 name specified in the fdl file (optional)
0961 1
0962 1 flags - address of flags longword (optional)
0963 1 FDL$V_SIGNAL signal errors instead of returning
0964 1 FDL$V_FDL_STRING input fdl-spec is a char string
0965 1 FDL$V_SCALLBACK used by EDF
0966 1
0967 1 Implicit Inputs:
0968 1 none
0969 1
0970 1 Output Parameters:
0971 1
0972 1 result_name - descriptor to receive the file name which was created
0973 1 (optional)
0974 1 fid_block - address of a 3 longword used block to receive the fid
0975 1 of the file created (optional)
0976 1
0977 1 stmt-num - address of longword to receive statement
0978 1 number (optional)
0979 1
0980 1 Implicit Outputs:
0981 1 none
0982 1
0983 1 Routine Value:
0984 1
0985 1 success or error code
0986 1
0987 1 Side Effects:
0988 1 none
0989 1
0990 1 --
0991 1
0992 2 BEGIN
0993 2
0994 2 LOCAL
0995 2 BYTES : LONG,
0996 2 LINE : REF BLOCK [ FDL$C_SECBLK_SIZE ] FIELD (SECTAB_FIELDS);
0997 2
0998 2 ! Waste 23 bytes to create a buffer to hold the time

```



```
280 0999 2 !
281 1000 2 BYTES = 23;
282 1001 2 CH$FILL ( 0, .BYTES, TIME_TEMP );
283 1002 2 TIME_BUF [ DSC$A_POINTER ] = TIME_TEMP;
284 1003 2 TIME_BUF [ DSC$W_LENGTH ] = .BYTES;
285 1004 2
286 1005 2 ! See if the RMS control blocks are kosher ( DON'T DEALLOCATE THEM )
287 1006 2 ! It also saves the addresses of any relevant XABs it finds
288 1007 2 ! It also saves the bucketsizes of any AREA XABs it finds
289 1008 2
290 1009 2 FDL$AB_CTRL [ FDL$V_DEALLOC ] = CLEAR;
291 1010 2 FDL$$CHECK_BLOCKS ( .FDL$AB_GENFAB, .FDL$AB_GENRAB );
292 1011 2 FAO_LENGTH = 0;
293 1012 2
294 1013 2 ! Generate the FDL primaries in their proper order
295 1014 2
296 1015 2 GEN_PRIMARY ( FDL$C_IDENT, 0, 0, 0 );
297 1016 2 GEN_PRIMARY ( FDL$C_SYSTEM, FDL$C_SOURCE, FDL$C_SOURCE, 0 );
298 1017 2 GEN_PRIMARY ( FDL$C_FILE, FDL$C_FILE_BEG, FDL$C_FILE_END, .XABPRO_PTR );
299 1018 2 ! GEN_PRIMARY ( FDL$C_ACL, FDL$C_ACL_BEG, FDL$C_ACL_END, .XABACL_PTR );
300 1019 2 GEN_PRIMARY ( FDL$C_DATE, FDL$C_DATE_BEG, FDL$C_DATE_END, .XABDAT_PTR );
301 1020 2 GEN_PRIMARY ( FDL$C_JNL, FDL$C_JOURNAL_BEG, FDL$C_JOURNAL_END, .XABJNL_PTR );
302 1021 2 GEN_PRIMARY ( FDL$C_RECORD, FDL$C_RECORD_BEG, FDL$C_RECORD_END, 0 );
303 1022 2 GEN_PRIMARY ( FDL$C_ACCESS, FDL$C_ACCESS_BEG, FDL$C_ACCESS_END, 0 );
304 1023 2 GEN_PRIMARY ( FDL$C_SHARING, FDL$C_SHARING_BEG, FDL$C_SHARING_END, 0 );
305 1024 2 GEN_PRIMARY ( FDL$C_CONNECT, FDL$C_CONNECT_BEG, FDL$C_CONNECT_END, 0 );
306 1025 2
307 1026 2 ! Cycle through all the AREAs
308 1027 2
309 1028 2 UNTIL .XABALL_PTR EQLU 0
310 1029 2 DO
311 1030 2 BEGIN
312 1031 2
313 1032 2 GEN_PRIMARY ( FDL$C_AREA, FDL$C_AREA_BEG, FDL$C_AREA_END, .XABALL_PTR );
314 1033 2
315 1034 2 DO
316 1035 2
317 1036 2 XABALL_PTR = .XABALL_PTR [ XAB$NXT ]
318 1037 2
319 1038 2 UNTIL (
320 1039 2 ( .XABALL_PTR EQLU 0 )
321 1040 2 OR
322 1041 2 ( .XABALL_PTR [ XAB$B_COD ] EQLU XAB$C_ALL ));
323 1042 2
324 1043 2 END;
325 1044 2
326 1045 2 ! Cycle through all the KEYS
327 1046 2
328 1047 2 UNTIL .XABKEY_PTR EQLU 0
329 1048 2 DO
330 1049 2 BEGIN
331 1050 2
332 1051 2 GEN_PRIMARY ( FDL$C_KEY, FDL$C_KEY_BEG, FDL$C_KEY_END, .XABKEY_PTR );
333 1052 2
334 1053 2 DO
335 1054 2
336 1055 2 XABKEY_PTR = .XABKEY_PTR [ XAB$NXT ]
```

```

: 337
: 338
: 339
: 340
: 341
: 342
: 343
: 344
: 345
: 346
: 347
1056 3
1057 4
1058 5
1059 4
1060 3
1061 3
1062 2
1063 2
1064 2
1065 2
1066 1

```

```

UNTIL (
( .XABKEY_PTR EQLU 0 )
OR
( .XABKEY_PTR [ XAB$B_COD ] EQLU XAB$C_KEY ));
END;
RETURN SSS_NORMAL;
END;

```

```

.TITLE FDLGEN VAX-11 FDL Utilities
.IDENT \V04-000\
.PSECT _FDL$PLIT,NOWRT,NOEXE, SHR, PIC,2

```

```

44 45 57 52 00000 P.AAB: .ASCII \RWED\
      010E0004 00004 P.AAA: .LONG 17694724
      00000000 00008 .ADDRESS P.AAB
00 44 45 57 0000C P.AAD: .ASCII \WED\<0>
      010E0003 00010 P.AAC: .LONG 17694723
      00000000 00014 .ADDRESS P.AAD
00 44 45 52 00018 P.AAF: .ASCII \RED\<0>
      010E0003 0001C P.AAE: .LONG 17694723
      00000000 00020 .ADDRESS P.AAF
00 00 44 45 00024 P.AAH: .ASCII \ED\<0><0>
      010E0002 00028 P.AAG: .LONG 17694722
      00000000 0002C .ADDRESS P.AAH
00 44 57 52 00030 P.AAJ: .ASCII \RWD\<0>
      010E0003 00034 P.AAI: .LONG 17694723
      00000000 00038 .ADDRESS P.AAJ
00 00 44 57 0003C P.AAL: .ASCII \WD\<0><0>
      010E0002 00040 P.AAK: .LONG 17694722
      00000000 00044 .ADDRESS P.AAL
00 00 44 52 00048 P.AAN: .ASCII \RD\<0><0>
      010E0002 0004C P.AAM: .LONG 17694722
      00000000 00050 .ADDRESS P.AAN
00 00 00 44 00054 P.AAP: .ASCII \D\<0><0><0>
      010E0001 00058 P.AAO: .LONG 176 .721
      00000000 0005C .ADDRESS P.AAP
00 45 57 52 00060 P.AAR: .ASCII \RWE\<0>
      010E0003 00064 P.AAQ: .LONG 17694723
      00000000 00068 .ADDRESS P.AAR
00 00 45 57 0006C P.AAT: .ASCII \WE\<0><0>
      010E0002 00070 P.AAS: .LONG 17694722
      00000000 00074 .ADDRESS P.AAT
00 00 45 52 00078 P.AAV: .ASCII \RE\<0><0>
      010E0002 0007C P.AAU: .LONG 17694722
      00000000 00080 .ADDRESS P.AAV
00 00 00 45 00084 P.AAX: .ASCII \E\<0><0><0>
      010E0001 00088 P.AAW: .LONG 17694721
      00000000 0008C .ADDRESS P.AAX
00 00 57 52 00090 P.AAZ: .ASCII \RW\<0><0>
      010E0002 00094 P.AAY: .LONG 17694722
      00000000 00098 .ADDRESS P.AAZ

```

```

00 00 00 57 0009C P.ABB: .ASCII \W\<0><0><0>
      010E0001 000A0 P.ABA: .LONG 17694721
      00000000' 000A4 .ADDRESS P.ABB
00 00 00 52 000AB P.ABD: .ASCII \R\<0><0><0>
      010E0001 000AC P.ABC: .LONG 17694721
      00000000' 000B0 .ADDRESS P.ABD
      000B4 P.ABF: .BLKB 0
      010E0000 000B4 P.ABE: .LONG 17694720
      00000000' 000B8 .ADDRESS P.ABF

```

.PSECT \_FDL\$OWN,NOEXE, PIC,2

```

00# 00000 TEMP_DESC:
      .BYTE 0[2]
01 0E 00002 .BYTE 14, 1
      00004 .BLKB 4
00# 00008 FAO_DESC:
      .BYTE 0[2]
01 0E 0000A .BYTE 14, 1
      0000C .BLKB 4
00# 00010 TIME_BUF:
      .BYTE 0[2]
01 0E 00012 .BYTE 14, 1
      00014 .BLKB 4
      00018 TIME_TEMP:
      .BLKB 23
      0002F .BLKB 1
      00030 FAO_LENGTH:
      .BLKB 4
      00034 FAO_PARAM:
      .BLKB 4
      00038 FAO_PARAM2:
      .BLKB 4
      0003C FAO_PARAM3:
      .BLKB 4
      00040 FAO_PARAM4:
      .BLKB 4
      00044 STRBYTES:
      .BLKB 4
      00048 OCHAR: .BLKB 4
      0004C XABPRO_PTR:
      .BLKB 4
      00050 XABRDT_PTR:
      .BLKB 4
      00054 XABDAT_PTR:
      .BLKB 4
      00058 XABJNL_PTR:
      .BLKB 4
      0005C XABALL_PTR:
      .BLKB 4
      00060 XABKEY_PTR:
      .BLKB 4
      00064 SAVE_POINTER:
      .BLKB 4

```

```

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00068 PROT_VALUES:
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00080 .ADDRESS P.AAA, P.AAC, P.AAE, P.AAG, P.AAI, -
      P.AAK, P.AAM, P.AAO, P.AAO, P.AAS, P.AAU, -

```

00000000' 00000000' 00000000' 00000000' 00098

P.AAW, P.AAY, P.AA, P.ABC, P.ABE ;

```

.EXTRN SYSS$FAO, SYSS$ASC, IM
.EXTRN LIB$GET_VM, LIB$FREE_VM
.EXTRN STR$APPEND, FDL$$FREE_VM
.EXTRN FDL$$READ_ERROR
.EXTRN FDL$$RMS_OPEN_ERROR
.EXTRN FDL$AB_PRI_TABLE
.EXTRN FDL$AB_SEC_TABLE
.EXTRN FDL$AB_OUT_STRING
.EXTRN FDL$AB_GENFAB, FDL$AB_GENRAB
.EXTRN FDL$AB_FDL_RAB, FDL$AB_CTRL
.EXTRN FDL$AB_BLOCK_BLK
.EXTRN FDL$AB_AREA_BLK
.EXTRN FDL$GL_INVBLK_PTR
.EXTRN FDL$GL_STNUMPTR
.EXTRN FDL$GL_MAXLINE, FDL$GL_SECNUM
.EXTRN FDL$GL_PRIMARY, FDL$GL_PRINUM
.EXTRN FDL$GL_SECONDARY
.EXTRN FDL$AB_FDL_STRING
.EXTRN FDL$AB_LINE, FDL$AB_UPCASED
.EXTRN FDL$AB_KEY_TABLE
.EXTRN FDL$AB_STATE_TABLE
.EXTRN FDL$AB_TPARSE_BLOCK
.EXTRN FDL$FACILITY, FDL$FAO_MAX
.EXTRN FDL$ABKW, FDL$ABPRIKW
.EXTRN FDL$CREATE, FDL$CREATED
.EXTRN FDL$CREATEDSTM
.EXTRN FDL$FDLERROR, FDL$ILL_ARG
.EXTRN FDL$INSVIRMEM, FDL$INVBLK
.EXTRN FDL$INVDATIM, FDL$MULPRI
.EXTRN FDL$MULSEC, FDL$NOQUAL
.EXTRN FDL$NULLPRI, FDL$OPENFDL
.EXTRN FDL$OUTORDER, FDL$OPENOUT
.EXTRN FDL$WRITEERR, FDL$READERR
.EXTRN FDL$RFLOC, FDL$TITLE
.EXTRN FDL$SYNTAX, FDL$VALPRI
.EXTRN FDL$UNQUAKW, FDL$UNPRIKW
.EXTRN FDL$UNSECKW, FDL$WARNING

```

.PSECT \_FDL\$CODE, NOWRT, SHR, PIC, 2

```

                    01FC 00000
58 00000000V 00 9E 00002
57 00000000' 00 9E 00009
56 17 D0 00010
6E 00 2C 00013
                    BC A7 00018
                    BC A7 9E 0001A
                    B4 A7 56 B0 0001F
00000000G 00 20 8A 00023
                    00000000G 00 DD 0002A
                    00000000G 00 DC 00030
00000000V 00 02 FB 00036
                    D4 A7 D4 0003D
                    7E 7C 00040
                    7E 09 7D 00042

```

```

.ENTRY FDL$$GEN_SPEC, Save R2,R3,R4,R5,R6,R7,R8 ; 0943
MOVAB GEN_PRIMARY, R8
MOVAB XAB$ALL_PTR, R7
MOVL #23, BYTES ; 1000
MOVCS #0, (SP), #0, BYTES, TIME_TEMP ; 1001
MOVAB TIME_TEMP, TIME_BUF+4 ; 1002
MOVW BYTES, TIME_BUF ; 1003
BICB2 #32, FDL$AB_CTRL+2 ; 1009
PUSHL FDL$AB_GENRAB ; 1010
PUSHL FDL$AB_GENFAB
CALLS #2, FDL$$CHECK_BLOCKS ; 1011
CLRL FAO_LENGTH ; 1015
CLRQ -(SP)
MOVQ #9, -(SP)

```

68		04	FB	00045	CALLS	#4, GEN_PRIMARY	
		7E	D4	00048	CLRL	-(SP)	1016
7E	95	8F	9A	0004A	MOVZBL	#149, -(SP)	
7E	95	8F	9A	0004E	MOVZBL	#149, -(SP)	
		0E	DD	00052	PUSHL	#14	
68		04	FB	00054	CALLS	#4, GEN_PRIMARY	
	F0	A7	DD	00057	PUSHL	XABPRO_PTR	1017
7E	6F	8F	9A	0005A	MOVZBL	#111, =(SP)	
7E	48	8F	9A	0005E	MOVZBL	#72, -(SP)	
		08	DD	00062	PUSHL	#8	
68		04	FB	00064	CALLS	#4, GEN_PRIMARY	
	F8	A7	DD	00067	PUSHL	XABDAT_PTR	1019
7E	47	8F	9A	0006A	MOVZBL	#71, -(SP)	
7E	44	8F	9A	0006E	MOVZBL	#68, -(SP)	
		07	DD	00072	PUSHL	#7	
68		04	FB	00074	CALLS	#4, GEN_PRIMARY	
	FC	A7	DD	00077	PUSHL	XABJNL_PTR	1020
7E	76	8F	9A	0007A	MOVZBL	#118, =(SP)	
7E	70	8F	9A	0007E	MOVZBL	#112, -(SP)	
		0A	DD	00082	PUSHL	#10	
68		04	FB	00084	CALLS	#4, GEN_PRIMARY	
		7E	D4	00087	CLRL	-(SP)	1021
7E	8C	8F	9A	00089	MOVZBL	#140, -(SP)	
7E	88	8F	9A	0008D	MOVZBL	#136, -(SP)	
		0C	DD	00091	PUSHL	#12	
68		04	FB	00093	CALLS	#4, GEN_PRIMARY	
7E		07	7D	00096	MOVQ	#7, -(SP)	1022
		01	DD	00099	PUSHL	#1	
		01	DD	0009B	PUSHL	#1	
68		04	FB	0009D	CALLS	#4, GEN_PRIMARY	
		7E	D4	000A0	CLRL	-(SP)	1023
7E	93	8F	9A	000A2	MOVZBL	#147, -(SP)	
7E	8D	8F	9A	000A6	MOVZBL	#141, -(SP)	
		0D	DD	000AA	PUSHL	#13	
68		04	FB	000AC	CALLS	#4, GEN_PRIMARY	
		7E	D4	000AF	CLRL	-(SP)	1024
7E	43	8F	9A	000B1	MOVZBL	#67, -(SP)	
		23	DD	000B5	PUSHL	#35	
		06	DD	000B7	PUSHL	#6	
68		04	FB	000B9	CALLS	#4, GEN_PRIMARY	
52		67	D0	000BC	MOVL	XABALL_PTR, R2	1028
		52	D5	000BF	TSTL	R2	
		1E	13	000C1	BEQL	3\$	
		52	DD	000C3	PUSHL	R2	1032
		22	DD	000C5	PUSHL	#34	
		1B	DD	000C7	PUSHL	#27	
		05	DD	000C9	PUSHL	#5	
68		04	FB	000CB	CALLS	#4, GEN_PRIMARY	
52		67	D0	000CE	MOVL	XABALL_PTR, R2	1036
67	04	A2	D0	000D1	MOVL	4(R2), XABALL_PTR	
52		67	D0	000D5	MOVL	XABALL_PTR, R2	1039
		E5	13	000D8	BEQL	1\$	
14		62	91	000DA	CMPB	(R2), #20	1041
		F2	12	000DD	BNEQ	2\$	
		DE	11	000DF	BRB	1\$	1028
52	04	A7	D0	000E1	MOVL	XABKEY_PTR, R2	1047
		52	D5	000E5	TSTL	R2	

		25	13	000E7	BEQL	6\$		
		52	DD	000E9	PUSHL	R2		1051
	7E	87	8F	9A 000EB	MOVZBL	#135, -(SP)		
	7E	77	8F	9A 000EF	MOVZBL	#119, -(SP)		
			0B	DD 000F3	PUSHL	#11		
	68		04	FB 000F5	CALLS	#4, GEN PRIMARY		
	52	04	A7	D0 000F8	MOVL	XABKEY_PTR, R2		1055
04	A7	04	A2	D0 000FC	5\$:	MOVL	4(R2), XABKEY_PTR	
	52	04	A7	D0 00101	MOVL	XABKEY_PTR, R2		1058
			DE	13 00105	BEQL	4\$		
	15		62	91 00107	CMPB	(R2), #21		1060
			F0	12 0010A	BNEQ	5\$		
			D7	11 0010C	BRB	4\$		1047
	50		01	D0 0010E	6\$:	MOVL	#1, R0	1064
			04	00111	RET			1066

; Routine Size: 274 bytes, Routine Base: \_FDL\$CODE + 0000

```

349 1067 1 %SBTTL 'GEN PRIMARY'
350 1068 1 ROUTINE GEN_PRIMARY ( WHICH : LONG, BEG : LONG, GEB : LONG, XAB_PTR : LONG ) =
351 1069 1 ++
352 1070 1
353 1071 1 Functional Description:
354 1072 1
355 1073 1 This routine xxxxxxxxxxxxxxxx
356 1074 1
357 1075 1 Calling Sequence:
358 1076 1
359 1077 1 fdl$$gen_line( fdl_string )
360 1078 1
361 1079 1 Input Parameters:
362 1080 1
363 1081 1 fdl_desc - descriptor of the fdl file name string (required)
364 1082 1 file_name - descriptor file name to override the name specified
365 1083 1 in the fdl file (optional)
366 1084 1 default_name - descriptor default file name to override the default
367 1085 1 name specified in the fdl file (optional)
368 1086 1
369 1087 1 flags - address of flags longword (optional)
370 1088 1 FDL$V_SIGNAL signal errors instead of returning
371 1089 1 FDL$V_FDL_STRING input fdl-spec is a char string
372 1090 1 FDL$V_$CALLBACK used by EDF
373 1091 1
374 1092 1 Implicit Inputs:
375 1093 1 none
376 1094 1
377 1095 1 Output Parameters:
378 1096 1
379 1097 1 result_name - descriptor to receive the file name which was created
380 1098 1 (optional)
381 1099 1 fid_block - address of a 3 longword used block to receive the fid
382 1100 1 of the file created (optional)
383 1101 1
384 1102 1 stmt-num - address of longword to recieve statement
385 1103 1 number (optional)
386 1104 1
387 1105 1 Implicit Outputs:
388 1106 1 none
389 1107 1
390 1108 1 Routine Value:
391 1109 1
392 1110 1 success or error code
393 1111 1
394 1112 1 Side Effects:
395 1113 1 none
396 1114 1
397 1115 1 --
398 1116 1
399 1117 2 BEGIN
400 1118 2
401 1119 2 LOCAL
402 1120 2 TEMP_BYTE : BYTE,
403 1121 2 BOFF : LONG,
404 1122 2 XAB : REF BLOCK [ BYTE ],
405 1123 2 PLINE : REF BLOCK [ FDL$C_P$IBLK_SIZE ] FIELD (PRITAB_FIELDS),

```

```

: 406      1124  2      LINE          : REF BLOCK [ FDL$C_SECBLK_SIZE ] FIELD (SECTAB_FIELDS);
: 407      1125  2
: 408      1126  2      ! Setup the xab pointer
: 409      1127  2
: 410      1128  2      FDL$AB_BLOCK_BLK [ FDL$C_XAB ] = .XAB_PTR;
: 411      1129  2      XAB = .XAB_PTR;
: 412      1130  2      FDL$GL_PRIMARY = .WHICH;
: 413      1131  2
: 414      1132  2      ! Point to the record which describe this primary attribute
: 415      1133  2
: 416      1134  2      PLINE = ( FDL$AB_PRI_TABLE + (.FDL$GL_PRIMARY * FDL$C_PRI_BLK_SIZE * 4) );
: 417      1135  2
: 418      1136  2      ! Setup the AREA number or KEY number
: 419      1137  2
: 420      1138  2      IF .PLINE [ FDL$V_NUM_ATTACH ]
: 421      1139  2      THEN
: 422      1140  2          BEGIN
: 423      1141  2
: 424      1142  2          BOFF = .PLINE [ FDL$V_PRI_BOFF ];
: 425      1143  2          FDL$GL_PRINUM = .XAB [ .BOFF,0,8,0 ];
: 426      1144  2
: 427      1145  2          END;
: 428      1146  2
: 429      1147  2      ! The following is a list of all the reasons why not to output this
: 430      1148  2      particular primary attribute
: 431      1149  2
: 432      1150  2      null table entry
: 433      1151  2      no date xab
: 434      1152  2      no journal xab
: 435      1153  2      not doing fullgen and primary = connect or sharing or access
: 436      1154  2
: 437      1155  4      IF ( NOT (
: 438      1156  5      ( .PLINE [ FDL$V_PRI_FAO ] EQLU 0 )
: 439      1157  4      OR
: 440      1158  5      (( .FDL$GL_PRIMARY EQLU FDL$C_DATE ) AND ( .XABDAT_PTR EQLU 0 ))
: 441      1159  4      OR
: 442      1160  5      (( .FDL$GL_PRIMARY EQLU FDL$C_JNL ) AND ( .XABJNL_PTR EQLU 0 ))
: 443      1161  4      OR
: 444      1162  6      ((
: 445      1163  7          ( .FDL$GL_PRIMARY EQLU FDL$C_CONNECT )
: 446      1164  6          OR
: 447      1165  7          ( .FDL$GL_PRIMARY EQLU FDL$C_SHARING )
: 448      1166  6          OR
: 449      1167  7          ( .FDL$GL_PRIMARY EQLU FDL$C_ACCESS )
: 450      1168  5      ) AND NOT .FDL$AB_CTRL [ FDL$V_FOLLGEN ] )
: 451      1169  2      )) THEN
: 452      1170  2          BEGIN
: 453      1171  2
: 454      1172  2          IF .FDL$GL_PRIMARY NEQU FDL$C_IDENT
: 455      1173  2          THEN
: 456      1174  2              ! Output a blank line
: 457      1175  2              FDL$$OUTPUT_LINE ( -1 );
: 458      1176  2
: 459      1177  2          ! Format and output the Primary Attribute Line
: 460      1178  2
: 461      1179  2          TEMP_BYTE = ..PLINE [ FDL$V_PRI_FAO ];
: 462      1180  2

```



```

463      1181      3      FAO_DESC [ DSC$W_LENGTH ] = .TEMP BYTE;
464      1182      3      FAO_DESC [ DSC$A_POINTER ] = .PLINE [ FDL$V_PRI_FAO ] + 1;
465      1183      3
466      1184      3      IF .FDL$GL_PRIMARY EQLU FDL$C_IDENT
467      1185      3      THEN
468      1186      4          BEGIN
469      1187      4
470      1188      4              RET ON ERROR ( SYSSASCTIM ( 0, TIME_BUF, 0, 0 ) );
471      1189      4              TIME_BUF [ DSC$W_LENGTH ] = 20;
472      1190      4
473      1191      3          END;
474      1192      3
475      P 1193      3      RET ON ERROR (
476      1194      3      SYSSFAO ( FAO_DESC, FAO_LENGTH, FDL$AB_LINE, .FDL$GL_PRINUM, TIME_BUF ) );
477      1195      3      FDL$$OUTPUT_LINE ( .FAO_LENGTH );
478      1196      3
479      1197      3      IF .FDL$GL_PRIMARY EQLU FDL$C_IDENT
480      1198      3      THEN
481      1199      3          RETURN S$$_NORMAL;
482      1200      3
483      1201      3      ! Cycle through the secondary attributes
484      1202      3      !
485      1203      3      INCR SEC FROM .BEG TO .GEB
486      1204      3      DO
487      1205      4          BEGIN
488      1206      4
489      1207      4              ! Skip it if this is a null table entry
490      1208      4              ! (which means we don't generate it - only ANALYZE/RMS/FDL does)
491      1209      4              ! Also skip it if the RMS block we need doesn't exist
492      1210      4              !
493      1211      4              LINE = ( FDL$AB_SEC_TABLE + (.SEC * FDL$C_SECBLK_SIZE * 4) );
494      1212      4
495      1213      4              ! FILE REVISION is a special case
496      1214      4              ! We have to bring in the XABRDT if it exists
497      1215      4              ! whereas FILE had used XABPRO previously
498      1216      4
499      1217      4              IF .SEC EQLU FDL$C_REVISN
500      1218      4              THEN
501      1219      4                  FDL$AB_BLOCK_BLK [ FDL$C_XAB ] = .XABRDT_PTR;
502      1220      4
503      1221      4              ! The following is a list of conditions to be satisfied
504      1222      4              ! in order to put this secondary out
505      1223      4              !
506      1224      4              ! table entry is not null
507      1225      4              ! pointe: to relevant RMS control block is not 0
508      1226      4              ! we're doing a fullgen, or the full-only bit is clear
509      1227      4              ! if this is the ifill or dfill secondary to the key primary,
510      1228      4              ! an xaball exists
511      1229      4
512      1230      5              IF (
513      1231      6                  ( .LINE [ FDL$V_SEC_FAO ] NEQU 0 )
514      1232      5              AND
515      1233      6                  ( .FDL$AB_BLOCK_BLK [ .LINE [ FDL$V_BLK_TYPE ] ] NEQU 0 )
516      1234      5              AND
517      1235      6                  ( ( .FDL$AB_CTRL [ FDL$V_FULLGEN ] ) OR ( NOT .LINE [ FDL$V_FULL_ONLY ] ) )
518      1236      5              AND
519      1237      6              NOT (

```

```

: 520 1238 6 ( .FDL$GL_PRIMARY EQLU FDL$C_KEY ) AND
: 521 1239 6 ( ( .SEC EQLU FDL$C_IFILL ) OR ( .SEC EQLU FDL$C_DFILL ) ) AND
: 522 1240 7 ( .XABALL_PTR EQLU 0 )
: 523 1241 5 )
: 524 1242 4 ) THEN
: 525 1243 5 BEGIN
: 526 1244 5
: 527 1245 5 FDL$GL_SECONDARY = .SEC;
: 528 1246 5 FDL$$OUTPUT_LINE ( FDL$$FORMAT_LINE ( ) );
: 529 1247 5
: 530 1248 4 END;
: 531 1249 4
: 532 1250 3 END;
: 533 1251 3
: 534 1252 2 END;
: 535 1253 2 RETURN S$$_NORMAL;
: 536 1254 2
: 537 1255 2
: 538 1256 1 END;

```

		01FC	00000	GEN_PRIMARY:			
				.WORD	Save R2,R3,R4,R5,R6,R7,R8		1068
		58	00000000G	00 9E 00002	MOVAB	FDL\$GL_PRINUM, R8	
		57	00000000C	00 9E 00009	MOVAB	FDL\$AB_BLOCK_BLK+8, R7	
		56	00000000V	00 9E 00010	MOVAB	FDL\$\$OUTPUT_LINE, R6	
		55	00000000G	00 9E 00017	MOVAB	FDL\$GL_PRIMARY, R5	
		54	00000000'	00 9E 0001E	MOVAB	TIME BOF, R4	
		67		10 AC D0 00025	MOVL	XAB_PTR, FDL\$AB_BLOCK_BLK+8	1128
		50		10 AC D0 00029	MOVL	XAB_PTR, XAB	1129
		65		04 AC D0 0002D	MOVL	WHICH, FDL\$GL_PRIMARY	1130
		51		65 D0 00031	MOVL	FDL\$GL_PRIMARY, R1	1134
	08	52	00000000G0041	7E 00034	MOVAQ	FDL\$AB_PRI_TABLE[R1], PLINE	
		A2		01 E1 0003C	BBC	#1, 4(PLINE), 1\$	1138
		53		06 A2 3C 00041	MOVZWL	6(PLINE), BOFF	1142
		68		6340 9A 00045	MOVZBL	(BOFF)[XAB], FDL\$GL_PRINUM	1143
				62 D5 00049	1\$: TSTL	(PLINE)	1156
				2B 13 0004B	BEQL	5\$	
		07		51 D1 0004D	CMPL	R1, #7	1158
				05 12 00050	BNEQ	2\$	
			44	A4 D5 00052	TSTL	XABDAT_PTR	
				21 13 00055	BEQL	5\$	
		0A		51 D1 00057	2\$: CMPL	R1, #10	1160
				05 12 0005A	BNEQ	3\$	
			48	A4 D5 0005C	TSTL	XABJNL_PTR	
				17 13 0005F	BEQL	5\$	
		06		51 D1 00061	3\$: CMPL	R1, #6	1163
				0A 13 00064	BEQL	4\$	
		0D		51 D1 00066	CMPL	R1, #13	1165
				05 13 00069	BEQL	4\$	
		01		51 D1 0006B	CMPL	R1, #1	1167
				0B 12 0006E	BNEQ	6\$	
	03	00000000G	00	04 E0 00070	4\$: BBS	#4, FDL\$AB_CTRL+2, 6\$	1168
				00C3 31 00078	5\$: BRW	16\$	

		09		51	D1	0007B	6\$:	C MPL	R1, #9	1172
		7E		06	13	0007E		BEQL	7\$	1176
		66		01	CE	00080		MNEGL	#1, -(SP)	1180
		50		01	FB	00083		CALLS	#1, FDL\$\$OUTPUT_LINE	1181
		A4	FC	00	B2	90 00086	7\$:	MOVW	@0(PLINE), TEMP_BYTE	1182
		62		50	9B	0008A		MOVZBW	TEMP_BYTE, FAO_DESC	1184
		09		01	C1	0008E		ADDL3	#1, (PLINE), FAO_DESC+4	1188
				65	D1	00093		C MPL	FDL\$GL_PRIMARY, #9	
				13	12	00096		BNEQ	8\$	
				7E	7C	00098		CLRQ	-(SP)	
				54	DD	0009A		PUSHL	R4	
				7E	D4	0009C		CLRL	-(SP)	
	00000000G	00		04	FB	0009E		CALLS	#4, SYSSASCTIM	
		1A		50	E9	000A5		BLBC	STATUS, 9\$	
		64		14	B0	000A8		MOVW	#20, TIME_BUF	1189
				54	DD	000AB	8\$:	PUSHL	R4	1194
				68	DD	000AD		PUSHL	FDL\$GL_PRINUM	
	00000000G	00		00	9F	000AF		PUSHAB	FDL\$AB_LINE	
		20		A4	9F	000B5		PUSHAB	FAO_LENGTH	
		F8		A4	9F	000B8		PUSHAB	FAO_DESC	
	00000000G	00		05	FB	000BB		CALLS	#5, -SYSSFAO	
		7C		50	E9	000C2	9\$:	BLBC	STATUS, 17\$	
				20	A4	DD 000C5		PUSHL	FAO_LENGTH	1195
		66		01	FB	000C8		CALLS	#1, FDL\$\$OUTPUT_LINE	
		09		65	D1	000CB		C MPL	FDL\$GL_PRIMARY, #9	1197
				6E	13	000CE		BEQL	16\$	
52	08	AC		01	C3	000D0		SUBL3	#1, BEG, SEC	1203
				62	11	000D5		BRB	15\$	
50		52		0C	C5	000D7	10\$:	MULL3	#12, SEC, R0	1211
		53		04	9E	000DB		MOVAB	FDL\$AB_SEC_TABLE[R0], LINE	
	00000067	8F		52	D1	000E3		C MPL	SEC, #T03	1217
				04	12	000EA		BNEG	11\$	
		67		40	A4	D0 000EC		MOVL	XABRDT_PTR, FDL\$AB_BLOCK_BLK+8	1219
				63	D5	000F0	11\$:	TSTL	(LINE)	1231
				45	13	000F2		BEQL	15\$	
		50		07	A3	9A 000F4		MOVZBL	7(LINE), R0	1233
				F8	A740	D5 000F8		TSTL	FDL\$AB_BLOCK_BLK[R0]	
				3B	13	000FC		BEQL	15\$	
04	00000000G	00		04	E0	000FE		BBS	#4, FDL\$AB_CTRL+2, 12\$	1235
		2F		04	A3	E8 00106		BLBS	4(LINE), 15\$	
		0B		65	D1	0010A	12\$:	C MPL	FDL\$GL_PRIMARY, #11	1238
				17	12	0010D		BNEQ	14\$	
	0000007F	8F		52	D1	0010F		C MPL	SEC, #127	1239
				09	13	00116		BEQL	13\$	
	00000079	8F		52	D1	00118		C MPL	SEC, #121	
				05	12	0011F		BNEQ	14\$	
				4C	A4	D5 00121	13\$:	TSTL	XABALL_PTR	1240
				13	13	00124		BEQL	15\$	
	00000000G	00		52	D0	00126	14\$:	MOVL	SEC, FDL\$GL_SECONDARY	1245
	00000000V	00		00	FB	0012D		CALLS	#0, FDL\$\$FORMAT_LINE	1246
				50	DD	00134		PUSHL	R0	
		66		01	FB	00136		CALLS	#1, FDL\$\$OUTPUT_LINE	
99		52		0C	AC	F3 00139	15\$:	AOBLEQ	GEB, SEC, 10\$	1203
		50		01	D0	0013E	16\$:	MOVL	#1, R0	1254
				04	00141	17\$:	RET			1256

; Routine Size: 322 bytes, Routine Base: \_FDL\$CODE + 0112

FDLGEN  
V04-000

VAX-11 FDL Utilities  
GEN\_PRIMARY

E 14  
16-Sep-1984 01:41:00  
14-Sep-1984 12:31:18

VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[FDL.SRC]FDLGEN.B32;1 Page 18  
(5)

FD  
VO

```

540 1257 1 %SBTTL 'FDL$$FORMAT LINE'
541 1258 1 GLOBAL ROUTINE FDL$$FORMAT_LINE =
542 1259 1 ++
543 1260 1
544 1261 1 Functional Description:
545 1262 1
546 1263 1     This routine xxxxxxxxxxxxxxxx
547 1264 1
548 1265 1 Calling Sequence:
549 1266 1
550 1267 1     fdl$$gen_line( fdl_string )
551 1268 1
552 1269 1 Input Parameters:
553 1270 1
554 1271 1     fdl_desc      - descriptor of the fdl file name string (required)
555 1272 1     file_name     - descriptor file name to override the name specified
556 1273 1                   in the fdl file (optional)
557 1274 1     default_name  - descriptor default file name to override the default
558 1275 1                   name specified in the fdl file (optional)
559 1276 1
560 1277 1     flags         - address of flags longword (optional)
561 1278 1                   FDL$V_SIGNAL      signal errors instead of returning
562 1279 1                   FDL$V_FDL_STRING  input fdl-spec is a char string
563 1280 1                   FDL$V_$CALLBACK   used by EDF
564 1281 1
565 1282 1 Implicit Inputs:
566 1283 1     none
567 1284 1
568 1285 1 Output Parameters:
569 1286 1
570 1287 1     result_name   - descriptor to receive the file name which was created
571 1288 1                   (optional)
572 1289 1     fid_block     - address of a 3 longword used block to receive the fid
573 1290 1                   of the file created (optional)
574 1291 1
575 1292 1     stmt-num     - address of longword to receive statement
576 1293 1                   number (optional)
577 1294 1
578 1295 1 Implicit Outputs:
579 1296 1     none
580 1297 1
581 1298 1 Routine Value:
582 1299 1
583 1300 1     success or error code
584 1301 1
585 1302 1 Side Effects:
586 1303 1     none
587 1304 1
588 1305 1 --
589 1306 1
590 1307 2 BEGIN
591 1308 2
592 1309 2 LOCAL
593 1310 2     TEMP_BYTE    : BYTE,
594 1311 2     STATUS       : LONG,
595 1312 2     LINE         : REF BLOCK [ FDL$C_SECBLK_SIZE ] FIELD (SECTAB_FIELDS);
596 1313 2

```

```
597 1314 2 ! Get the offset into the table
598 1315 2 ! This is done here (as well as in FDL$$GEN_SPEC) because EDF can
599 1316 2 ! call FDL$$FORMAT_LINE directly
600 1317 2
601 1318 2 LINE = ( FDL$AB_SEC_TABLE + (.FDL$GL_SECONDARY * FDL$C_SECBLK_SIZE * 4) );
602 1319 2
603 1320 2 ! If it's 'PROLOG', skip it unless it's KEY 0
604 1321 2
605 1322 3 IF (
606 1323 4 ( .FDL$GL_SECONDARY EQLU FDL$C_PROL )
607 1324 3 AND
608 1325 4 ( .FDL$GL_PRINUM NEQU 0 )
609 1326 2 ) THEN
610 1327 2 RETURN 0;
611 1328 2
612 1329 2 ! Construct a descriptor: get the length from the ASCII, then the Address
613 1330 2
614 1331 2 TEMP_BYTE = ..LINE [ FDL$V_SEC_FAO ];
615 1332 2 FAO_DESC [ DSC$W_LENGTH ] = .TEMP_BYTE;
616 1333 2 FAO_DESC [ DSC$A_POINTER ] = .LINE [ FDL$V_SEC_FAO ] + 1;
617 1334 2
618 1335 2 ! If it's a 'SEgn_XXX' secondary, loop thru up to 8 times
619 1336 2
620 1337 3 IF (
621 1338 4 ( .FDL$GL_SECONDARY EQLU FDL$C_SEGLEN )
622 1339 3 OR
623 1340 4 ( .FDL$GL_SECONDARY EQLU FDL$C_SEGPOS )
624 1341 2 ) THEN
625 1342 3 BEGIN
626 1343 3
627 1344 3 INCR J FROM 0 TO 7
628 1345 3 DO
629 1346 4 BEGIN
630 1347 4
631 1348 4 LOCAL
632 1349 4 BLK : REF BLOCK [ ,BYTE ],
633 1350 4 FAO_CHAR : REF VECTOR [ ,BYTE ];
634 1351 4
635 1352 4 BLK = .FDL$AB_BLOCK_BLK [ FDL$C_XAB ];
636 1353 4 FDL$GL_SECNUM = .J;
637 1354 4
638 1355 4 ! Get the value for the line
639 1356 4
640 1357 4 FETCH_FIELD ( .LINE );
641 1358 4
642 1359 4 ! Put the formatted text into it
643 1360 4
644 1361 4 RET ON ERROR (
645 1362 4 SYSSFAO ( FAO_DESC,FAO_LENGTH,FDL$AB_LINE, .FAO_PARAM ));
646 1363 4
647 1364 4 ! Stuff the 'n' in the "SEgn_XXX" line
648 1365 4
649 1366 4 FAO_CHAR = .FDL$AB_LINE [ DSC$A_POINTER ];
650 1367 4 FAO_CHAR [ 4 ] = .FDL$GL_SECNUM + '0';
651 1368 4
652 1369 4 ! Look ahead to see if there are any more segments
653 1370 4
```

```

654 1371 5 IF (CASE (.J+1) FROM 1 TO 8 OF
655 1372 5 SET
656 1373 5 [ 1 ] : .BLK [ XAB$B_SIZ1 ];
657 1374 5 [ 2 ] : .BLK [ XAB$B_SIZ2 ];
658 1375 5 [ 3 ] : .BLK [ XAB$B_SIZ3 ];
659 1376 5 [ 4 ] : .BLK [ XAB$B_SIZ4 ];
660 1377 5 [ 5 ] : .BLK [ XAB$B_SIZ5 ];
661 1378 5 [ 6 ] : .BLK [ XAB$B_SIZ6 ];
662 1379 5 [ 7 ] : .BLK [ XAB$B_SIZ7 ];
663 1380 5 [ 8 ] : 0;
664 1381 4 TES) EQLU 0
665 1382 4 THEN
666 1383 4 EXITLOOP
667 1384 4 ELSE
668 1385 4 : If there is more, 'PUT' this one before looping
669 1386 4 :
670 1387 4 FDL$$OUTPUT_LINE ( .FAO_LENGTH );
671 1388 4
672 1389 3 END;
673 1390 3 END
674 1391 2 ELSE
675 1392 2 BEGIN
676 1393 3 : Get the value for the line
677 1394 3 :
678 1395 3 IF FETCH_FIELD ( .LINE )
679 1396 3 THEN
680 1397 3 BEGIN
681 1398 4 : Put the formatted text into it
682 1399 4 :
683 1400 4 RET_ON_ERROR ( SYSS$FAO (
684 P 1401 4 :
685 P P 1402 4 :
686 P P 1403 4 :
687 P P 1404 4 :
688 P P 1405 4 :
689 P P 1406 4 :
690 P 1407 4 :
691 1408 4 :
692 1409 4 :
693 1410 4 IF .LINE [ FDL$V_DATA_TYPE ] EQLU FDL$C_STRING
694 1411 4 THEN
695 1412 5 BEGIN
696 1413 5 LOCAL STATUS;
697 1414 5
698 1415 6 IF NOT ( STATUS = LIB$FREE_VM ( STRBYTES, OCHAR ))
699 1416 5 THEN
700 1417 5 SIGNAL_STOP ( .STATUS );
701 1418 4 END;
702 1419 4
703 1420 4 END
704 1421 3 ELSE
705 1422 3 FAO_LENGTH = 0;
706 1423 3
707 1424 2 END;
708 1425 2
709 1426 2 RETURN .FAO_LENGTH;
710 1427 2

```





	52	33	A2	9A	000C1	9\$:	MOVZBL	51(BLK), R2	:	1377
			0E	11	000C5		BRB	13\$	:	
	52	34	A2	9A	000C7	10\$:	MOVZBL	52(BLK), R2	:	1378
			08	11	000CB		BRB	13\$	:	
	52	35	A2	9A	000CD	11\$:	MOVZBL	53(BLK), R2	:	1379
			02	11	000D1		BRB	13\$	:	
			52	D4	000D3	12\$:	CLRL	R2	:	1371
			4F	13	000D5	13\$:	BEQL	17\$	:	1381
			65	DD	000D7		PUSHL	FAO_LENGTH	:	1387
	00000000V	00	01	FB	000D9		CALLS	#1, FDL\$\$OUTPUT_LINE	:	
89		54	07	F3	000E0		AOBLEQ	#7, J, 3\$	:	1344
			40	11	000E4		BRB	17\$	:	1337
			53	DD	000E6	14\$:	PUSHL	LINE	:	1396
	68		01	FB	000E8		CALLS	#1, FETCH_FIELD	:	
	36		50	E9	000EB		BLBC	R0, 16\$	:	
	7E	0C	A5	7D	00CEE		MOVQ	FAO_PARAM3, -(SP)	:	1408
	7E	04	A5	7D	000F2		MOVQ	FAO_PARAM, -(SP)	:	
		0060	8F	BB	000F6		PUSHR	#*M2R5, R6>	:	
		D8	A5	9F	000FA		PUSHAB	FAO_DESC	:	
	69		07	FB	000FD		CALLS	#7, -SYSS\$FAO	:	
	29		50	E9	00100	15\$:	BLBC	STATUS, 19\$	:	
	07	05	A3	91	00103		CMPB	5(LINE), #7	:	1410
			1D	12	00107		BNEQ	17\$	:	
		18	A5	9F	00109		PUSHAB	OCHAR	:	1415
		14	A5	9F	0010C		PUSHAB	STRBYTES	:	
	00000000G	00	02	FB	0010F		CALLS	#2, LIB\$FREE_VM	:	
		0D	50	E8	00116		BLBS	STATUS, 17\$	:	
			50	DD	00119		PUSHL	STATUS	:	1417
	00000000G	00	01	FB	0011B		CALLS	#1, LIB\$STOP	:	
			02	11	00122		BRB	17\$	:	1396
			65	D4	00124	16\$:	CLRL	FAO_LENGTH	:	1422
		50	65	D0	00126	17\$:	MOVL	FAO_LENGTH, R0	:	1426
				04	00129		RET		:	
			50	D4	0012A	18\$:	CLRL	R0	:	1428
			04	0012C	19\$:	RET		:		

; Routine Size: 301 bytes, Routine Base: \_FDL\$CODE + 0254

```

1429 1 %SBTTL 'FDL$$OUTPUT_LINE'
1430 1 GLOBAL ROUTINE FDL$$OUTPUT_LINE ( OUT_LEN : LONG ) =
1431 1 ++
1432 1
1433 1 Functional Description:
1434 1
1435 1     This routine xxxxxxxxxxxxxxxx
1436 1
1437 1 Calling Sequence:
1438 1
1439 1     fdl$$gen_line( fdl_string )
1440 1
1441 1 input Parameters:
1442 1
1443 1     fdl_desc      - descriptor of the fdl file name string (required)
1444 1     file_name     - descriptor file name to override the name specified
1445 1                   in the fdl file (optional)
1446 1     default_name  - descriptor default file name to override the default
1447 1                   name specified in the fdl file (optional)
1448 1
1449 1     flags         - address of flags longword (optional)
1450 1                   FDL$V_SIGNAL      signal errors instead of returning
1451 1                   FDL$V_FDL_STRING  input fdl-spec is a char string
1452 1                   FDL$V_$CALLBACK   used by EDF
1453 1
1454 1 Implicit Inputs:
1455 1     none
1456 1
1457 1 Output Parameters:
1458 1
1459 1     result_name   - descriptor to receive the file name which was created
1460 1                   (optional)
1461 1     fid_block     - address of a 3 longword used block to receive the fid
1462 1                   of the file created (optional)
1463 1
1464 1     stmt-num     - address of longword to receive statement
1465 1                   number (optional)
1466 1
1467 1 Implicit Outputs:
1468 1     none
1469 1
1470 1 Routine Value:
1471 1
1472 1     success or error code
1473 1
1474 1 Side Effects:
1475 1     none
1476 1
1477 1 --
1478 1
1479 2 BEGIN
1480 2
1481 2 LOCAL
1482 2     TEMP_LEN      : WORD;
1483 2
1484 2     ! Don't bother with nothing
1485 2

```

```

770 1486 2 IF .OUT_LEN EQLU 0
771 1487 2 THEN
772 1488 2 RETURN SSS_NORMAL;
773 1489 2
774 1490 2 ! But less than nothing means skip a line
775 1491 2 !
776 1492 2 IF .OUT_LEN LSS 0
777 1493 2 THEN
778 1494 2 OUT_LEN = 0;
779 1495 2
780 1496 2 IF .FDLSAB_CTRL [ FDLSV_STRING_SPEC ]
781 1497 2 THEN
782 1498 2 BEGIN
783 1499 2 LOCAL
784 1500 2 TEMP : LONG;
785 1501 2
786 1502 2 TEMP_LEN = .FDLSAB_LINE [ DSCSW_LENGTH ];
787 1503 2 TEMP = .FDLSAB_LINE [ DSCSA_POINTER ] + .OUT_LEN;
788 1504 2 .TEMP = ' ';
789 1505 2 OUT_LEN = .OUT_LEN + 1;
790 1506 2 FDL$AB_LINE [ DSCSW_LENGTH ] = .OUT_LEN;
791 1507 2 RET_ON_ERROR( STR$APPEND( .FDLSAB_OUT_STRING, FDL$AB_LINE ) );
792 1508 2 FDL$AB_LINE [ DSCSW_LENGTH ] = .TEMP_LEN;
793 1509 2
794 1510 2 END
795 1511 2 ELSE
796 1512 2 ! Put the new line to the FDL file.
797 1513 2 !
798 1514 2 BEGIN
799 1515 2
800 1516 2 FDL$AB_FDL_RAB [ RAB$W_RSZ ] = .OUT_LEN;
801 1517 2 RET_ON_ERROR( $PUT ( RAB=FDL$AB_FDL_RAB, ERR=FDL$$READ_ERROR ) );
802 1518 2
803 1519 2 END;
804 1520 2
805 1521 2 RETURN SSS_NORMAL;
806 1522 2
807 1523 2 END;

```

					.EXTRN	SYSSPUT	
			000C 00000		.ENTRY	FDL\$\$OUTPUT_LINE, Save R2,R3	: 1430
	53	00000000G	00	9E 00002	MOVAB	FDL\$AB_LINE, R3	
			04	AC D5 00009	TSTL	OUT_LEN	: 1486
				55 13 0000C	BEQL	3\$	
				03 18 0000E	BGEQ	1\$	: 1492
			04	AC D4 00010	CLRL	OUT_LEN	: 1494
2A	00G	00000G	00	04 E1 00013	BBC	#4, FDL\$AB_CTRL+1, 2\$	: 1496
			52	63 B0 0001B	MOVW	FDL\$AB_LINE, TEMP_LEN	: 1502
50		04	A3	04 AC C1 0001E	ADDL3	OUT_LEN, FDL\$AB_LINE+4, TEMP	: 1503
			60	3B D0 00024	MOVL	#59, (TEMP)	: 1504
				04 AC D6 00027	INCL	OUT_LEN	: 1505
			63	04 AC B0 0002A	MOVW	OUT_LEN, FDL\$AB_LINE	: 1506
				53 DD 0002E	PUSHL	R3	: 1507
			00000000G	00 DD 00030	PUSHL	FDL\$AB_OUT_STRING	

00000000G	00		02	FB	00036	CALLS	#2, STR\$APPEND	:	
	26		50	E9	0003D	BLBC	STATUS, 4\$	:	
	63		52	B0	00040	MOVW	TEMP_LEN, FDL\$AB_LINE	:	1508
			1E	11	00043	BRB	3\$	:	1496
00000000G	00	04	AC	B0	00045	2\$: MOVW	OUT_LEN, FDL\$AB_FDL_RAB+34	:	1516
		00000000G	G0	9F	0004D	PUSHAB	FDL\$\$READ_ERROR	:	1517
		00000000G	00	9F	00053	PUSHAB	FDL\$AB_FDL_RAB	:	
00000000G	00		02	FB	00059	CALLS	#2, SY\$SPUT	:	
	03		50	E9	00060	BLBC	STATUS, 4\$	:	
	50		01	D0	00063	3\$: MOVL	#1, R0	:	1521
			04	00066	4\$: RET			:	1523

; Routine Size: 103 bytes, Routine Base: \_FDL\$CODE + 0381

```

809 1524 1 XSBTTL 'CHECK_XAB'
810 1525 1 ROUTINE CHECK_XAB ( XAB_POINTER : REF BLOCK [ ,BYTE ] ) =
811 1526 1 ++
812 1527 1
813 1528 1 Functional Description:
814 1529 1
815 1530 1     This routine xxxxxxxxxxxxxxxx
816 1531 1
817 1532 1 Calling Sequence:
818 1533 1
819 1534 1     fdl$$gen_line( fdl_string )
820 1535 1
821 1536 1 Input Parameters:
822 1537 1
823 1538 1     fdl_desc      - descriptor of the fdl file name string (required)
824 1539 1     file_name     - descriptor file name to override the name specified
825 1540 1                   in the fdl file (optional)
826 1541 1     default_name  - descriptor default file name to override the default
827 1542 1                   name specified in the fdl file (optional)
828 1543 1
829 1544 1     flags         - address of flags longword (optional)
830 1545 1                   FDL$V_SIGNAL      signal errors instead of returning
831 1546 1                   FDL$V_FDL_STRING  input fdl-spec is a char string
832 1547 1                   FDL$V_$CALLBACK   used by EDF
833 1548 1
834 1549 1 Implicit Inputs:
835 1550 1     none
836 1551 1
837 1552 1 Output Parameters:
838 1553 1
839 1554 1     result_name  - descriptor to receive the file name which was created
840 1555 1                   (optional)
841 1556 1     fid_block    - address of a 3 longword used block to receive the fid
842 1557 1                   of the file created (optional)
843 1558 1
844 1559 1     stmt-num     - address of longword to receive statement
845 1560 1                   number (optional)
846 1561 1
847 1562 1 Implicit Outputs:
848 1563 1     none
849 1564 1
850 1565 1 Routine Value:
851 1566 1
852 1567 1     success or error code
853 1568 1
854 1569 1 Side Effects:
855 1570 1     none
856 1571 1
857 1572 1 --
858 1573 1
859 1574 2 BEGIN
860 1575 2     LOCAL TEMP;
861 1576 2
862 1577 3 WHILE ( .XAB_POINTER NEQU 0 )
863 1578 2 DO
864 1579 3     BEGIN
865 1580 3

```

```

: 866 1581 3      ! Keep the old next-link around
: 867 1582 3
: 868 1583 3      SAVE_POINTER = .XAB_POINTER [ XAB$L_NXT ];
: 869 1584 3
: 870 1585 3
: 871 1586 3      ! See if it's a useful one and save it's address if it is
: 872 1587 3
: 873 1588 4
: 874 1589 5      IF (
: 875 1590 4      ( .XAB_POINTER [ XAB$B_BLN ] EQLU 0 )
: 876 1591 5      OR
: 877 1592 4      ( .XAB_POINTER [ XAB$B_BLN ] LSS 0 )
: 878 1593 5      OR
: 879 1594 4      ( .XAB_POINTER [ XAB$B_COD ] EQLU 0 )
: 880 1595 5      OR
: 881 1596 3      ( .XAB_POINTER [ XAB$B_COD ] LSS 0 )
: 882 1597 4      ) THEN
: 883 1598 4      BEGIN
: 884 1599 4          IF .FDL$GL_INVBLK_PTR NEQU 0
: 885 1600 4          THEN
: 886 1601 4              .FDL$GL_INVBLK_PTR = .XAB_POINTER;
: 887 1602 4
: 888 1603 4          SIGNAL ( FDL$ _INVBLK,1,.XAB_POINTER );
: 889 1604 4
: 890 1605 4      END
: 891 1606 3      ELSE
: 892 1607 4      BEGIN
: 893 1608 4
: 894 1609 4          ! Only bother if we're not deallocating
: 895 1610 4          !
: 896 1611 4          IF NOT .FDL$AB_CTRL [ FDL$V_DEALLOC ]
: 897 1612 4          THEN
: 898 1613 5              BEGIN
: 899 1614 5
: 900 1615 5          ! It's not an obviously BAD XAB, is it an important one?
: 901 1616 5          !
: 902 1617 5          ! SELECTONE .XAB_POINTER [ XAB$B_COD ] OF
: 903 1618 5          !
: 904 1619 5          SET
: 905 1620 5          [ XAB$C_KEY ] : IF .XABKEY_PTR EQLU 0
: 906 1621 5          THEN
: 907 1622 5              XABKEY_PTR = .XAB_POINTER;
: 908 1623 5
: 909 1624 5          [ XAB$C_ALL ] :
: 910 1625 5          BEGIN
: 911 1626 6
: 912 1627 6          IF .XAB_POINTER [ XAB$B_BKZ ] NEQU 0
: 913 1628 6          THEN
: 914 1629 6              FDL$AB_AREA_BKZ [ .XAB_POINTER [ XAB$B_AID ] ] =
: 915 1630 6                  .XAB_POINTER [ XAB$B_BKZ ]
: 916 1631 6          ELSE
: 917 1632 6              FDL$AB_AREA_BKZ [ .XAB_POINTER [ XAB$B_AID ] ] = BUCKET_DEFAULT;
: 918 1633 6
: 919 1634 6          IF .XABALL_PTR EQLU 0
: 920 1635 6          THEN
: 921 1636 6              XABALL_PTR = .XAB_POINTER;
: 922 1637 6

```

```

: 923
: 924
: 925
: 926
: 927
: 928
: 929
: 930
: 931
: 932
: 933
: 934
: 935
: 936
: 937
: 938
: 939
: 940
: 941
: 942
: 943
: 944
: 945
: 946
: 947
: 948
: 949
: 950
: 951
: 952
: 953
: 954
: 955
: 956
: 957
: 958
: 959
: 960
: 961
: 962

```

```

1638 6
1639 5
1640 5
1641 5
1642 5
1643 5
1644 5
1645 5
1646 5
1647 5
1648 5
1649 5
1650 4
1651 5
1652 5
1653 5
1654 5
1655 6
1656 7
1657 6
1658 7
1659 5
1660 5
1661 5
1662 5
1663 5
1664 3
1665 4
1666 4
1667 3
1668 3
1669 3
1670 3
1671 3
1672 3
1673 2
1674 2
1675 2
1676 2
1677 1

```

```

END;
[ XAB$C_DAT ] : XABDAT_PTR = .XAB_POINTER;
[ XAB$C_PRO ] : XABPRO_PTR = .XAB_POINTER;
[ XAB$C_RDT ] : XABRDT_PTR = .XAB_POINTER;
[ XAB$C_JNL ] : XABJNL_PTR = .XAB_POINTER;
[ OTHERWISE ] : 0;

TES;
END
ELSE
BEGIN
! Now get rid of it if we're 'RELEASING'
!
IF (
( .XAB_POINTER [ XAB$B_COD ] EQLU XAB$C_KEY )
AND
( .XAB_POINTER [ XAB$L_KNM ] NEQU 0 )
) THEN
FDL$$FREE_VM ( 32, .XAB_POINTER [ XAB$L_KNM ] );

TEMP = .XAB_POINTER [ XAB$B_BLN ];
FDL$$FREE_VM ( .TEMP, .XAB_POINTER );

END;

END;
! Point to the next
!
XAB_POINTER = .SAVE_POINTER;

END;

RETURN S$$_NORMAL;

END;

```

```

003C 00000 CHECK_XAB:
55 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5
54 00000000' 00 9E 00009 MOVAB FDL$$FREE_VM, R5
52 04 AC D0 00010 1$: MOVAB SAVE_POINTER, R4
03 12 00014 MOVL XAB_POINTER, R2
00C5 31 00016 BNEQ 2$
64 04 A2 D0 00019 2$: BRW 17$
01 A2 95 0001D MOVL 4(R2), SAVE_POINTER
04 13 00020 TSTB 1(R2)
62 95 00022 BEQL 3$
1F 12 00024 TSTB (R2)
BNEQ 5$

```

```

: 1525
:
: 1577
:
: 1583
: 1590
: 1593
:

```

50	00000000G	00	D0	00026	3\$:	MOVL	FDL\$GL_INVBLK_PTR, R0	1599																				
		03	13	0002D		BEQL	4\$																					
60		52	D0	0002F		MOVL	R2, (R0)	1601																				
		52	DD	00032	4\$:	PUSHL	R2	1603																				
		01	DD	00034		PUSHL	#1																					
	00000000G	8F	DD	00036		PUSHL	#FDL\$ INVBLK																					
		03	FB	0003C		CALLS	#3, LIB\$SIGNAL																					
		73	11	00043		BRB	13\$	1588																				
6D	00000000G	00	E0	00045	5\$:	BBS	#5, FDL\$AB_CTRL+2, 14\$	1611																				
		15	62	91	0004D	CMPB	(R2), #21	1621																				
			0B	12	00050	BNEQ	6\$																					
			FC	A4	D5	00052	TSTL	XABKEY_PTR																				
				61	12	00055	BNEQ	13\$																				
	FC	A4		52	D0	00057	MOVL	R2, XABKEY_PTR	1623																			
				7A	11	0005B	BRB	16\$	1621																			
				14	62	91	0005D	6\$:	1625																			
				2C	12	00060	BNEQ	9\$																				
	50	00000000G	00	D0	00062	MOVL	FDL\$AB_AREA_BKZ, R0	1630																				
			51	17	A2	9E	00069	MOVAB	23(R2), R1																			
				16	A2	95	0006D	TSTB	22(R2)	1628																		
				0A	13	00070	BEQL	7\$																				
				51	61	9A	00072	MOVZBL	(R1), R1	1630																		
	6140			16	A2	90	00075	MOVB	22(R2), (R1)[R0]	1631																		
					07	11	0007A	BRB	8\$	1630																		
					51	61	9A	0007C	7\$:	1633																		
	6140				02	90	0007F	MOVB	#2, (R1)[R0]																			
					F8	A4	D5	00083	8\$:	1635																		
					4F	12	00086	TSTL	XABALL_PTR																			
					F8	A4	52	D0	00088	BNEQ	16\$	1637																
							49	11	0008C	MOVL	R2, XABALL_PTR	1617																
							12	62	91	0008E	9\$:	1641																
								06	12	00091	CMPB	(R2), #18																
								52	D0	00093	BNEQ	10\$																
								F0	A4	3E	11	00097	MOVL	R2, XABDAT_PTR														
										62	91	00099	BRB	16\$														
										13	62	91	00099	10\$:	1642													
											06	12	0009C	CMPB	(R2), #19													
											E8	A4	52	D0	0009E	BNEQ	11\$											
													33	11	000A2	MOVL	R2, XABPRO_PTR											
													62	91	000A4	BRB	16\$											
													1E	62	91	000A4	11\$:	1643										
														06	12	000A7	CMPB	(R2), #30										
														52	D0	000A9	BNEQ	12\$										
														EC	A4	28	11	000AD	MOVL	R2, XABRDT_PTR								
																62	91	000AF	BRB	16\$								
																22	62	91	000AF	12\$:	1644							
																	23	12	000B2	CMPB	(R2), #34							
																	F4	A4	52	D0	000B4	BNEQ	16\$					
																			1D	11	000B8	MOVL	R2, XABJNL_PTR					
																			62	91	000BA	BRB	16\$					
																			15	62	91	000BA	13\$:	1656				
																				0D	12	000BD	14\$:					
																				38	A2	D5	000BF	CMPB	(R2), #21			
																					08	13	000C2	BNEQ	15\$	1658		
																					38	A2	DD	000C4	BEQL	15\$		
																						20	DD	000C7	PUSHL	56(R2)	1660	
																						02	DD	000C7	PUSHL	#32		
																						65	02	FB	000C9	CALLS	#2, FDL\$\$FREE_VM	
																						53	01	A2	9A	000CC	15\$:	1662
																							52	DD	000D0	MOVZBL	1(R2), TEMP	
																							53	DD	000D2	PUSHL	R2	1663
																							02	FB	000D4	PUSHL	TEMP	
																										CALLS	#2, FDL\$\$FREE_VM	



FDLGEN  
V04-000

VAX-11 FDL Utilities  
CHECK\_XAB

E 15  
16-Sep-1984 01:41:00  
14-Sep-1984 12:31:18

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

Page 31  
(8)

04	AC	64	DO 000D7	16\$:	MOVL	SAVE_POINTER, XAB_POINTER	:	1671
		FF32	31 000DB		BRW	1\$	:	1577
	50	01	DO 000DE	17\$:	MOVL	#1, R0	:	1675
			04 000E1		RET		:	1677

; Routine Size: 226 bytes, Routine Base: \_FDL\$CODE + 03E8

```

964 1678 1 %SBTTL 'FDL$$CHECK_BLOCKS'
965 1679 1 GLOBAL ROUTINE FDL$$CHECK_BLOCKS ( FAB_POINTER : REF BLOCK [ ,BYTE ],
966 1680 1                                     RAB_POINTER : REF BLOCK [ ,BYTE ] ) =
967 1681 1 |++
968 1682 1 |
969 1683 1 | Functional Description:
970 1684 1 |
971 1685 1 |     This routine xxxxxxxxxxxxxxxx
972 1686 1 |
973 1687 1 | Calling Sequence:
974 1688 1 |
975 1689 1 |     fdl$$gen_line( fdl_string )
976 1690 1 |
977 1691 1 | Input Parameters:
978 1692 1 |
979 1693 1 |     fdl_desc      - descriptor of the fdl file name string (required)
980 1694 1 |     file_name     - descriptor file name to override the name specified
981 1695 1 |                   in the fdl file (optional)
982 1696 1 |     default_name  - descriptor default file name to override the default
983 1697 1 |                   name specified in the fdl file (optional)
984 1698 1 |
985 1699 1 |     flags         - address of flags longword (optional)
986 1700 1 |                   FDL$V_SIGNAL      signal errors instead of returning
987 1701 1 |                   FDL$V_FDL_STRING  input fdl-spec is a char string
988 1702 1 |                   FDL$V_SCALLBACK   used by EDF
989 1703 1 |
990 1704 1 | Implicit Inputs:
991 1705 1 |     none
992 1706 1 |
993 1707 1 | Output Parameters:
994 1708 1 |
995 1709 1 |     result_name   - descriptor to receive the file name which was created
996 1710 1 |                   (optional)
997 1711 1 |     fid_block     - address of a 3 longword used block to receive the fid
998 1712 1 |                   of the file created (optional)
999 1713 1 |
1000 1714 1 |     stmt-num     - address of longword to receive statement
1001 1715 1 |                   number (optional)
1002 1716 1 |
1003 1717 1 | Implicit Outputs:
1004 1718 1 |     none
1005 1719 1 |
1006 1720 1 | Routine Value:
1007 1721 1 |
1008 1722 1 |     success or error code
1009 1723 1 |
1010 1724 1 | Side Effects:
1011 1725 1 |     none
1012 1726 1 |
1013 1727 1 | --
1014 1728 1 |
1015 1729 2 | BEGIN
1016 1730 2 |
1017 1731 2 | LOCAL
1018 1732 2 |     BYTES          : LONG,
1019 1733 2 |     NAM_POINTER    : REF BLOCK [ ,BYTE ];
1020 1734 2 |

```

```
1021 1735 2 ! Allocate the area bucketsize array
1022 1736 2 !
1023 1737 2 BYTES = 256;
1024 1738 2 IF NOT LIB$GET_VM ( BYTES, FDL$AB_AREA_BKZ )
1025 1739 2 THEN
1026 1740 2     SIGNAL_STOP ( FDL$ INSVIRMEM );
1027 1741 2     CH$FILL ( 0, .BYTES, .FDL$AB_AREA_BKZ );
1028 1742 2
1029 1743 2 ! Clear the cells that indicate where important XABs were found
1030 1744 2 !
1031 1745 2 XABALL_PTR = _CLEAR;
1032 1746 2 XABDAT_PTR = _CLEAR;
1033 1747 2 XABPRO_PTR = _CLEAR;
1034 1748 2 XABRDT_PTR = _CLEAR;
1035 1749 2 XABJNL_PTR = _CLEAR;
1036 1750 2 XABKEY_PTR = _CLEAR;
1037 1751 2
1038 1752 2 ! If not defaulted, FAB_POINTER must point to a FAB
1039 1753 2 !
1040 1754 2 IF ( .FAB_POINTER NEQU 0 ) AND ( .FAB_POINTER [ FAB$B_BID ] NEQU FAB$C_BID )
1041 1755 2 THEN
1042 1756 2     BEGIN
1043 1757 2
1044 1758 2         IF .FDL$GL_INVBLK_PTR NEQU 0
1045 1759 2         THEN
1046 1760 2             .FDL$GL_INVBLK_PTR = .FAB_POINTER;
1047 1761 2
1048 1762 2         SIGNAL ( FDL$ INVBLK,1,.FAB_POINTER );
1049 1763 2
1050 1764 2     END;
1051 1765 2
1052 1766 2 ! If not defaulted, RAB_POINTER must point to a RAB
1053 1767 2 !
1054 1768 2 IF ( .RAB_POINTER NEQU 0 ) AND ( .RAB_POINTER [ RAB$B_BID ] NEQU RAB$C_BID )
1055 1769 2 THEN
1056 1770 2     BEGIN
1057 1771 2
1058 1772 2         IF .FDL$GL_INVBLK_PTR NEQU 0
1059 1773 2         THEN
1060 1774 2             .FDL$GL_INVBLK_PTR = .RAB_POINTER;
1061 1775 2
1062 1776 2         SIGNAL ( FDL$ INVBLK,1,.RAB_POINTER );
1063 1777 2
1064 1778 2     END;
1065 1779 2
1066 1780 2 ! Check the FAB attachments
1067 1781 2 !
1068 1782 2 IF .FAB_POINTER NEQU 0
1069 1783 2 THEN
1070 1784 2     BEGIN
1071 1785 2
1072 1786 2         ! Deallocate the NAM if it exists
1073 1787 2         !
1074 1788 2         NAM_POINTER = .FAB_POINTER [ FAB$L_NAM ];
1075 1789 2
1076 1790 2         IF .NAM_POINTER NEQU 0
1077 1791 2         THEN
```

```
1078 1792 4 BEGIN
1079 1793 4
1080 1794 4 ! If it exists, it must be a NAM block
1081 1795 4 !
1082 1796 4 IF .NAM_POINTER [ NAMS_BID ] EQLU NAM$C_BID
1083 1797 4 THEN
1084 1798 5 BEGIN
1085 1799 5
1086 1800 5 IF .FDL$AB_CTRL [ FDL$V_DEALLOC ]
1087 1801 5 THEN
1088 1802 6 BEGIN
1089 1803 6
1090 1804 6 IF .NAM_POINTER [ NAMS_L_ESA ] NEQU 0
1091 1805 6 THEN
1092 1806 6 FDL$$FREE_VM ( .NAM_POINTER [ NAMS_B_ESS ],
1093 1807 6 .NAM_POINTER [ NAMS_C_ESA ] );
1094 1808 6
1095 1809 6 IF .NAM_POINTER [ NAMS_L_RSA ] NEQU 0
1096 1810 6 THEN
1097 1811 6 FDL$$FREE_VM ( .NAM_POINTER [ NAMS_B_RSS ],
1098 1812 6 .NAM_POINTER [ NAMS_C_RSA ] );
1099 1813 6
1100 1814 6 FDL$$FREE_VM ( .NAM_POINTER [ NAMS_B_BLN ], .NAM_POINTER );
1101 1815 6
1102 1816 6 END
1103 1817 5 END
1104 1818 4 ELSE
1105 1819 5 BEGIN
1106 1820 5
1107 1821 5 IF .FDL$GL_INVBLK_PTR NEQU 0
1108 1822 5 THEN
1109 1823 5 .FDL$GL_INVBLK_PTR = .NAM_POINTER;
1110 1824 5
1111 1825 5 SIGNAL ( FDL$_INVBLK,1,.NAM_POINTER );
1112 1826 5
1113 1827 4 END;
1114 1828 4
1115 1829 3 END;
1116 1830 3
1117 1831 3 ! Deallocate the filename (and default filename) if present
1118 1832 3 !
1119 1833 3 NAM_POINTER = .FAB_POINTER [ FAB$L_FNA ];
1120 1834 3
1121 1835 4 IF ( .NAM_POINTER NEQU 0 ) AND ( .FDL$AB_CTRL [ FDL$V_DEALLOC ] )
1122 1836 3 THEN
1123 1837 3 FDL$$FREE_VM ( .FAB_POINTER [ FAB$B_FNS ], .NAM_POINTER );
1124 1838 3
1125 1839 3 NAM_POINTER = .FAB_POINTER [ FAB$L_DNA ];
1126 1840 3
1127 1841 4 IF ( .NAM_POINTER NEQU 0 ) AND ( .FDL$AB_CTRL [ FDL$V_DEALLOC ] )
1128 1842 3 THEN
1129 1843 3 FDL$$FREE_VM ( .FAB_POINTER [ FAB$B_DNS ], .NAM_POINTER );
1130 1844 3
1131 1845 3 ! Now check and possibly deallocate any XABs
1132 1846 3 !
1133 1847 3 CHECK_XAB ( .FAB_POINTER [ FAB$L_XAB ] );
1134 1848 3
```

```

1135 1849      IF .FDL$AB_CTRL [ FDL$V_DEALLOC ]
1136 1850      THEN
1137 1851          FDL$$FREE_VM ( .FAB_POINTER [ FAB$B_BLN ], .FAB_POINTER );
1138 1852
1139 1853      END;
1140 1854
1141 1855      ! Check the RAB attachments
1142 1856      !
1143 1857      IF .RAB_POINTER NEQU 0
1144 1858      THEN
1145 1859          BEGIN
1146 1860              ! Deallocate any existing XABs
1147 1861              !
1148 1862              CHECK_XAB ( .RAB_POINTER [ RAB$L_XAB ] );
1149 1863
1150 1864              IF .FDL$AB_CTRL [ FDL$V_DEALLOC ]
1151 1865              THEN
1152 1866                  FDL$$FREE_VM ( .RAB_POINTER [ RAB$B_BLN ], .RAB_POINTER );
1153 1867
1154 1868              END;
1155 1869
1156 1870              ! Done with the bucketsize array
1157 1871              !
1158 1872              BEGIN
1159 1873                  LOCAL STATUS;
1160 1874
1161 1875                  IF NOT ( STATUS = LIB$FREE_VM ( BYTES, FDL$AB_AREA_BKZ ) )
1162 1876                  THEN
1163 1877                      SIGNAL_STOP ( .STATUS );
1164 1878
1165 1879                  END;
1166 1880
1167 1881          RETURN SSS_NORMAL;
1168 1882
1169 1883      END;

```

		OFFC 00000		.ENTRY	FDL\$\$CHECK_BLOCKS, Save R2,R3,R4,R5,R6,R7,-	1679
					R8,R9,R10,R11	
		5B 00000000G	00 9E 00002	MOVAB	LIB\$SIGNAL, R11	
		5A 00000000G	8F D0 00009	MOVL	#FDL\$ INVBLK, R10	
		59 00000000G	00 9E 00010	MOVAB	FDL\$GC_INVBLK_PTR, R9	
		58 00000000'	00 9E 00017	MOVAB	XABPRO_PTR, R8	
		57 00000000G	00 9E 0001E	MOVAB	FDL\$AB_CTRL, R7	
		56 00000000C	00 9E 00025	MOVAB	FDL\$\$FREE_VM, R6	
		7E 0100	8F 3C 0002C	MOVZWL	#256, BYTES	1737
		00000000G	00 9F 00031	PUSHAB	FDL\$AB_AREA_BKZ	1738
		04	AE 9F 00037	PUSHAB	BYTES	
	00000000G	00	02 FB 0003A	CALLS	#2, LIB\$GET_VM	
	0D	00	50 EB 00041	BLBS	R0, 1\$	
	00000000G	00	8F DD 00044	PUSHL	#FDL\$ INSVIRMEM	1740
	00000000G	00	01 FB 0004A	CALLS	#1, LIB\$STOP	
	50 00000000G	00	D0 D0 00051	MOVL	FDL\$AB_AREA_BKZ, R0	1741
6E	00	6E	00 2C 00058	MOVCS	#0, (SP), #0, BYTES, (R0)	



			52	DD	000F4	10\$:	PUSHL	NAM_POINTER	: 1825	
			01	DD	000F6		PUSHL	#1		
			5A	DD	000F8		PUSHL	R10		
		6B	03	FB	000FA		CALLS	#3, LIB\$SIGNAL		
		52	2C	A3	DD	000FD	11\$:	MOVL	44(R3), NAM_POINTER	: 1833
			0E	13	00101		BEQL	12\$	: 1835	
09	02	A7	05	E1	00103		BBC	#5, FDL\$AB_CTRL+2, 12\$		
			52	DD	00108		PUSHL	NAM_POINTER	: 1837	
		7E	34	A3	9A	0010A		MOVZBL	52(R3), -(SP)	
		66	02	FB	0010E		CALLS	#2, FDL\$\$FREE_VM		
		52	30	A3	DD	00111	12\$:	MOVL	48(R3), NAM_POINTER	: 1839
			0E	13	00115		BEQL	13\$	: 1841	
09	02	A7	05	E1	00117		BBC	#5, FDL\$AB_CTRL+2, 13\$		
			52	DD	0011C		PUSHL	NAM_POINTER	: 1843	
		7E	35	A3	9A	0011E		MOVZBL	53(R3), -(SP)	
		66	02	FB	00122		CALLS	#2, FDL\$\$FREE_VM		
			24	A3	DD	00125	13\$:	PUSHL	36(R3)	: 1847
	FDF1	CF	01	FB	00128		CALLS	#1, CHECK_XAB		
09	02	A7	05	E1	0012D		BBC	#5, FDL\$AB_CTRL+2, 14\$	: 1849	
			53	DD	00132		PUSHL	R3	: 1851	
		7E	01	A3	9A	00134		MOVZBL	1(R3), -(SP)	
		66	02	FB	00138		CALLS	#2, FDL\$\$FREE_VM		
		16	55	E9	0013B	14\$:	BLBC	R5, 15\$	: 1857	
			40	A4	DD	0013E		PUSHL	64(R4)	: 1863
	FDD8	CF	01	FB	00141		CALLS	#1, CHECK_XAB		
09	02	A7	05	E1	00146		BBC	#5, FDL\$AB_CTRL+2, 15\$	: 1865	
			54	DD	0014B		PUSHL	R4	: 1867	
		7E	01	A4	9A	0014D		MOVZBL	1(R4), -(SP)	
		66	02	FB	00151		CALLS	#2, FDL\$\$FREE_VM		
			00	9F	00154	15\$:	PUSHAB	FDL\$AB_AREA_BRZ	: 1876	
			04	AE	9F	0015A		PUSHAB	BYTES	
	00000000G	00	02	FB	0015D		CALLS	#2, LIB\$FREE_VM		
		09	50	E8	00164		BLBS	STATUS, 16\$		
			50	DD	00167		PUSHL	STATUS	: 1878	
	00000000G	00	01	FB	00169		CALLS	#1, LIB\$STOP		
		50	01	DD	00170	16\$:	MOVL	#1, R0	: 1881	
			04	00173			RET		: 1883	

; Routine Size: 372 bytes, Routine Base: \_FDL\$CODE + 04CA

```

1171 1884 1 %SBTTL 'FETCH_FIELD'
1172 1885 1 ROUTINE FETCH_FIELD (
1173 1886 1 LINE : REF BLOCK [ FDL$C_SECBLK_SIZE, LONG ] FIELD (SECTAB_FIELDS) ) =
1174 1887 1 ++
1175 1888 1
1176 1889 1 Functional Description:
1177 1890 1
1178 1891 1 This routine xxxxxxxxxxxxxxxx
1179 1892 1
1180 1893 1 Calling Sequence:
1181 1894 1
1182 1895 1 fdl$$gen_line( fdl_string )
1183 1896 1
1184 1897 1 Input Parameters:
1185 1898 1
1186 1899 1 fdl_desc - descriptor of the fdl file name string (required)
1187 1900 1 file_name - descriptor file name to override the name specified
1188 1901 1 in the fdl file (optional)
1189 1902 1 default_name - descriptor default file name to override the default
1190 1903 1 name specified in the fdl file (optional)
1191 1904 1
1192 1905 1 flags - address of flags longword (optional)
1193 1906 1 FDL$V_SIGNAL signal errors instead of returning
1194 1907 1 FDL$V_FDL_STRING input fdl-spec is a char string
1195 1908 1 FDL$V_$CALLBACK used by EDF
1196 1909 1
1197 1910 1 Implicit Inputs:
1198 1911 1 none
1199 1912 1
1200 1913 1 Output Parameters:
1201 1914 1
1202 1915 1 result_name - descriptor to receive the file name which was created
1203 1916 1 (optional)
1204 1917 1 fid_block - address of a 3 longword used block to receive the fid
1205 1918 1 of the file created (optional)
1206 1919 1
1207 1920 1 stmt-num - address of longword to receive statement
1208 1921 1 number (optional)
1209 1922 1
1210 1923 1 Implicit Outputs:
1211 1924 1 none
1212 1925 1
1213 1926 1 Routine Value:
1214 1927 1
1215 1928 1 success or error code
1216 1929 1
1217 1930 1 Side Effects:
1218 1931 1 none
1219 1932 1
1220 1933 1 --
1221 1934 1
1222 1935 2 BEGIN
1223 1936 2
1224 1937 2 LOCAL
1225 1938 2 TEMP_SWITCH : BYTE,
1226 1939 2 TEMP_BYTE : BYTE,
1227 1940 2 TEMP_WORD : WORD,

```



```

: 1228 1941 2      TEMP_LONG      : LONG,
: 1229 1942 2      TEMP_QUAD     : VECTOR [ 2, LONG ],
: 1230 1943 2      TEMP_OCTA     : VECTOR [ 4, LONG ],
: 1231 1944 2      TEMP_STRING    : DESC_BLK,
: 1232 1945 2      TEMP_QUALIFIER : BYTE,
: 1233 1946 2      TEMP_SPECIAL   : LONG,
: 1234 1947 2      BLK            : REF BLOCK [ , BYTE ],
: 1235 1948 2      BOFF           : LONG,
: 1236 1949 2      POS            : LONG;
: 1237 1950
: 1238 1951 2      *****
: 1239 1952 2      !
: 1240 1953 2      FOR EDF TO CALL FDL$$FORMAT LINE DIRECTLY - FETCH_FIELD WILL
: 1241 1954 2      HAVE TO BE CONDITIONALIZED TO GET THE VALUES FROM
: 1242 1955 2      FDL$GL NUMBER, FDL$GL_SWITCH, ETC. INSTEAD OF FROM THE
: 1243 1956 2      RMS CONTROL BLOCKS
: 1244 1957 2      !
: 1245 1958 2      *****
: 1246 1959 2
: 1247 1960 2      ! Get the address of the BLK and the offsets within it
: 1248 1961 2      !
: 1249 1962 2      BLK = .FDL$AB BLOCK BLK [ .LINE [ FDL$V_BLK_TYPE ] ];
: 1250 1963 2      BOFF = .LINE [ FDL$V_SEC_BOFF ];
: 1251 1964 2      POS = .LINE [ FDL$V_SEC_POS ];
: 1252 1965 2
: 1253 1966 2      ! Save the value according to its datatype
: 1254 1967 2      !
: 1255 1968 2      CASE .LINE [ FDL$V_DATA_TYPE ] FROM FDL$C_DUMMY TO FDL$C_SPECIAL OF
: 1256 1969 2
: 1257 1970 2      SET
: 1258 1971 2
: 1259 1972 2      [ FDL$C_DUMMY ] : 0;
: 1260 1973 2
: 1261 1974 2      [ FDL$C_BYTE ] :
: 1262 1975 2      BEGIN
: 1263 1976 2
: 1264 1977 2      IF .FDL$GL_SECONDARY NEQU FDL$C_SEGLEN
: 1265 1978 2      THEN
: 1266 1979 2      FAO_PARAM = .BLK [ .BOFF, .POS, 8, 0 ]
: 1267 1980 2      ELSE
: 1268 1981 2      BEGIN
: 1269 1982 2
: 1270 1983 2      FAO_PARAM = (CASE .FDL$GL_SECNUM FROM 0 TO 7 OF
: 1271 1984 2      SET
: 1272 1985 2      [ 0 ] : .BLK [ XAB$B_SIZ0 ];
: 1273 1986 2      [ 1 ] : .BLK [ XAB$B_SIZ1 ];
: 1274 1987 2      [ 2 ] : .BLK [ XAB$B_SIZ2 ];
: 1275 1988 2      [ 3 ] : .BLK [ XAB$B_SIZ3 ];
: 1276 1989 2      [ 4 ] : .BLK [ XAB$B_SIZ4 ];
: 1277 1990 2      [ 5 ] : .BLK [ XAB$B_SIZ5 ];
: 1278 1991 2      [ 6 ] : .BLK [ XAB$B_SIZ6 ];
: 1279 1992 2      [ 7 ] : .BLK [ XAB$B_SIZ7 ];
: 1280 1993 2      TES);
: 1281 1994 2
: 1282 1995 2      END;
: 1283 1996 2
: 1284 1997 2      END;

```

```

1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341

```

```

[ FDL$C_WORD ] :
BEGIN
SELECTONE .FDL$GL_SECONDARY OF
SET
! For INDEX_FILL and DATA_FILL, convert the Fill Numbers to
! Fill Percents. The extra 1/2 measure is to fight roundoff error
[ FDL$C_DFILL ] :
FAO_PARAM =
(( .BLK [ XABS$W_DFL ] * 100 ) + ( .BLK [ XABS$W_DFL ] / 2 )) /
( BLOCK_SIZE * .FDL$AB_AREA_BKZ [ .BLK [ XABS$B_DAN ] ] );
[ FDL$C_IFILL ] :
FAO_PARAM =
(( .BLK [ XABS$W_IFL ] * 100 ) + ( .BLK [ XABS$W_IFL ] / 2 )) /
( BLOCK_SIZE * .FDL$AB_AREA_BKZ [ .BLK [ XABS$B_IAN ] ] );
! Look at all the segments
[ FDL$C_SEGPOS ] :
BEGIN
FAO_PARAM = (CASE .FDL$GL_SECNUM FROM 0 TO 7 OF
SET
[ 0 ] :: .BLK [ XABS$W_POS0 ];
[ 1 ] :: .BLK [ XABS$W_POS1 ];
[ 2 ] :: .BLK [ XABS$W_POS2 ];
[ 3 ] :: .BLK [ XABS$W_POS3 ];
[ 4 ] :: .BLK [ XABS$W_POS4 ];
[ 5 ] :: .BLK [ XABS$W_POS5 ];
[ 6 ] :: .BLK [ XABS$W_POS6 ];
[ 7 ] :: .BLK [ XABS$W_POS7 ];
TES);
END;
[ OTHERWISE ] : FAO_PARAM = .BLK [ .BOFF,.POS,16,0 ];
TES;
END;
[ FDL$C_LONG ] :
BEGIN
FAO_PARAM = .BLK [ .BOFF,.POS,32,0 ];
END;
[ FDL$C_QUAD ] :
BEGIN
0;

```

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000

```

: 1342 2055 3
: 1343 2056 2
: 1344 2057 2
: 1345 2058 [ FDL$C_OCTA ] :
: 1346 2059 BEGIN
: 1347 2060
: 1348 2061 0:
: 1349 2062
: 1350 2063 END:
: 1351 2064
: 1352 2065 [ FDL$C_SWITCH ] :
: 1353 2066 BEGIN
: 1354 2067
: 1355 2068 ! Output yes or no depending upon the bit setting
: 1356 2069 ! 4 of the secondaries have inverted sense
: 1357 2070 !
: 1358 2071 IF (
: 1359 2072 (.FDL$GL_SECONDARY EQL FDL$C_DATKC)
: 1360 2073 OR
: 1361 2074 (.FDL$GL_SECONDARY EQL FDL$C_DATRC)
: 1362 2075 OR
: 1363 2076 (.FDL$GL_SECONDARY EQL FDL$C_IDXC)
: 1364 2077 OR
: 1365 2078 (.FDL$GL_SECONDARY EQL FDL$C_BLKSPN)
: 1366 2079 ) THEN
: 1367 2080 BEGIN
: 1368 2081 IF NOT .BLK [ .BOFF,.POS,1,0 ]
: 1369 2082 THEN
: 1370 2083 FAO_PARAM = UPLIT BYTE (%ASCIC 'yes')
: 1371 2084 ELSE
: 1372 2085 FAO_PARAM = UPLIT BYTE (%ASCIC 'no');
: 1373 2086 END
: 1374 2087 ELSE
: 1375 2088 BEGIN
: 1376 2089 IF .BLK [ .BOFF,.POS,1,0 ]
: 1377 2090 THEN
: 1378 2091 FAO_PARAM = UPLIT BYTE (%ASCIC 'yes')
: 1379 2092 ELSE
: 1380 2093 FAO_PARAM = UPLIT BYTE (%ASCIC 'no');
: 1381 2094 END:
: 1382 2095 END:
: 1383 2096
: 1384 2097 [ FDL$C_SPECIAL ] :
: 1385 2098 BEGIN
: 1386 2099
: 1387 2100 LOCAL
: 1388 2101 TIME_ADDR : LONG,
: 1389 2102 TIME_LEN : WORD;
: 1390 2103
: 1391 2104 SELECTONE .FDL$GL_SECONDARY OF
: 1392 2105
: 1393 2106 SET
: 1394 2107
: 1395 2108 [ FDL$C_BACKUP, FDL$C_CREAT, FDL$C_EXPR, FDL$C_REV ] :
: 1396 2109 BEGIN
: 1397 2110
: 1398 2111 LOCAL

```

```

: 1399      2112  4          TIME_VEC      : REF VECTOR [ 2, LONG ];
: 1400      2113  4
: 1401      2114  4          TIME_ADDR = .BLK + .BOFF;
: 1402      2115  4          TIME_VEC = .TIME_ADDR;
: 1403      2116  4
: 1404      2117  4          ! If the time is null, don't bother putting it out
: 1405      2118  4          !
: 1406      2119  5          IF (
: 1407      2120  6          ( .TIME_VEC [ 0 ] EQLU 0 )
: 1408      2121  5          AND
: 1409      2122  6          ( .TIME_VEC [ 1 ] EQLU 0 )
: 1410      2123  4          ) THEN
: 1411      2124  4              RETURN 0;
: 1412      2125  4
: 1413      2126  4          RET_ON_ERROR ( SYSS$CTIM ( 0, TIME_BUF, .TIME_ADDR, 0 ) );
: 1414      2127  4          FAO_PARAM = TIME_BUF;
: 1415      2128  4
: 1416      2129  3          END;
: 1417      2130  3
: 1418      2131  3          [ FDL$C_PROT ] :
: 1419      2132  4          BEGIN
: 1420      2133  4
: 1421      2134  4          LOCAL
: 1422      2135  4          PROTECTION : LONG;
: 1423      2136  4
: 1424      2137  4          PROTECTION = .BLK [ XABS$W_PRO ];
: 1425      2138  4          FAO_PARAM = .PROT_VALUES [ .PROTECTION<0,4> ];
: 1426      2139  4          FAO_PARAM2 = .PROT_VALUES [ .PROTECTION<4,4> ];
: 1427      2140  4          FAO_PARAM3 = .PROT_VALUES [ .PROTECTION<8,4> ];
: 1428      2141  4          FAO_PARAM4 = .PROT_VALUES [ .PROTECTION<12,4> ];
: 1429      2142  4
: 1430      2143  3          END;
: 1431      2144  3
: 1432      2145  3          [ FDL$C_POSI ] :
: 1433      2146  3          BEGIN
: 1434      2147  4
: 1435      2148  4
: 1436      2149  4          TEMP_BYTE = 23;
: 1437      2150  4          CH$FILL ( 0, .TEMP_BYTE, TIME_TEMP );
: 1438      2151  4          TEMP_DESC [ DSC$A_POINTER ] = TIME_TEMP;
: 1439      2152  4          TEMP_DESC [ DSC$W_LENGTH ] = .TEMP_BYTE;
: 1440      2153  4
: 1441      2154  5          IF (
: 1442      2155  6          ( .BLK [ XABS$W_RF10 ] NEQU 0 )
: 1443      2156  5          OR
: 1444      2157  6          ( .BLK [ XABS$W_RF12 ] NEQU 0 )
: 1445      2158  5          OR
: 1446      2159  6          ( .BLK [ XABS$W_RF14 ] NEQU 0 )
: 1447      2160  4          ) THEN
: 1448      2161  5              BEGIN
: 1449      2162  5
: 1450      2163  5          RET_ON_ERROR ( SYSS$FAO (
: 1451      2164  5              XASCII 'file_ID (!UW,!UW,!UW)',
: 1452      2165  5              TEMP_WORD,
: 1453      2166  5              TEMP_DESC,
: 1454      2167  5              .BLK [ XABS$W_RF10 ],
: 1455      2168  5              .BLK [ XABS$W_RF12 ] );

```

1456	2169	5	.BLK [ XABS\$W_RF14 ] ));
1457	2170	5	
1458	2171	5	TEMP_DESC [ DSC\$W_LENGTH ] = .TEMP_WORD;
1459	2172	5	FAO_PARAM = TEMP_DESC;
1460	2173	5	
1461	2174	5	END
1462	2175	4	ELSE
1463	2176	5	BEGIN
1464	2177	5	
1465	2178	5	SELECTONE .BLK [ XABS\$B_ALN ] OF
1466	2179	5	
1467	2180	5	SET
1468	2181	5	
1469	2182	5	[ XABS\$C_CYL ] :
1470	2183	6	BEGIN
1471	2184	6	
1472	2185	6	RET_ON_ERROR ( SYSS\$FAO (
1473	2186	6	XASCID 'cylinder !UL',
1474	2187	6	TEMP_WORD,
1475	2188	6	TEMP_DESC,
1476	2189	6	.BLK [ XABS\$L_LOC ] ));
1477	2190	6	
1478	2191	6	TEMP_DESC [ DSC\$W_LENGTH ] = .TEMP_WORD;
1479	2192	6	FAO_PARAM = TEMP_DESC;
1480	2193	6	
1481	2194	5	END;
1482	2195	5	
1483	2196	5	[ XABS\$C_LBN ] :
1484	2197	6	BEGIN
1485	2198	6	
1486	2199	6	RET_ON_ERROR ( SYSS\$FAO (
1487	2200	6	XASCID 'logical !UL',
1488	2201	6	TEMP_WORD,
1489	2202	6	TEMP_DESC,
1490	2203	6	.BLK [ XABS\$L_LOC ] ));
1491	2204	6	
1492	2205	6	TEMP_DESC [ DSC\$W_LENGTH ] = .TEMP_WORD;
1493	2206	6	FAO_PARAM = TEMP_DESC;
1494	2207	6	
1495	2208	5	END;
1496	2209	5	
1497	2210	5	[ XABS\$C_VBN ] :
1498	2211	6	BEGIN
1499	2212	6	
1500	2213	6	RET_ON_ERROR ( SYSS\$FAO (
1501	2214	6	XASCID 'virtual !UL',
1502	2215	6	TEMP_WORD,
1503	2216	6	TEMP_DESC,
1504	2217	6	.BLK [ XABS\$L_LOC ] ));
1505	2218	6	
1506	2219	6	TEMP_DESC [ DSC\$W_LENGTH ] = .TEMP_WORD;
1507	2220	6	FAO_PARAM = TEMP_DESC;
1508	2221	6	
1509	2222	5	END;
1510	2223	5	
1511	2224	5	[ 0 ] :
1512	2225	6	BEGIN

```

: 1513 2226 6
: 1514 2227 6          IF NOT .BLK [ XAB$V_ONC ]
: 1515 2228 6          THEN
: 1516 2229 6          FAO_PARAM = %ASCID '      none';
: 1517 2230 6
: 1518 2231 5          END;
: 1519 2232 5
: 1520 2233 5          [ OTHERWISE ] : 0;
: 1521 2234 5
: 1522 2235 5          TES;
: 1523 2236 5
: 1524 2237 5          IF .BLK [ XAB$V_ONC ]
: 1525 2238 5          THEN
: 1526 2239 5          FAO_PARAM = %ASCID '      any_cylinder';
: 1527 2240 5
: 1528 2241 4          END;
: 1529 2242 4
: 1530 2243 3          END;
: 1531 2244 3
: 1532 2245 3          [ OTHERWISE ] : FAO_PARAM = 0;
: 1533 2246 3
: 1534 2247 3          TES;
: 1535 2248 3
: 1536 2249 3          END;
: 1537 2250 2
: 1538 2251 2          [ FDL$C_STRING ] :
: 1539 2252 2          BEGIN
: 1540 2253 2
: 1541 2254 2          ! Assume it won't be found
: 1542 2255 2          !
: 1543 2256 2          TEMP_DESC [ DSC$W_LENGTH ] = 0;
: 1544 2257 2
: 1545 2258 2          SELECTONE .FDL$GL_SECONDARY OF
: 1546 2259 2
: 1547 2260 2          SET
: 1548 2261 2
: 1549 2262 3          [ FDL$C_ACE ] :
: 1550 2263 4          BEGIN
: 1551 2264 4
: 1552 2265 4          TEMP_DESC [ DSC$A_POINTER ] = UPLIT BYTE ( %ASCII 'Ace' );
: 1553 2266 4          TEMP_DESC [ DSC$W_LENGTH ] = 3;
: 1554 2267 4
: 1555 2268 3          END;
: 1556 2269 3
: 1557 2270 3          [ FDL$C_DFNAM ] :
: 1558 2271 4          BEGIN
: 1559 2272 4
: 1560 2273 4          IF .BLK [ FAB$S_DNA ] NEQU 0
: 1561 2274 4          THEN
: 1562 2275 5          BEGIN
: 1563 2276 5
: 1564 2277 5          TEMP_DESC [ DSC$A_POINTER ] = .BLK [ FAB$S_DNA ];
: 1565 2278 5          TEMP_DESC [ DSC$W_LENGTH ] = .BLK [ FAB$S_DNS ];
: 1566 2279 5
: 1567 2280 4          END;
: 1568 2281 4
: 1569 2282 3          END;
```

```

: 1570 2283 3
: 1571 2284 3 [ FDL$C_NAME ] :
: 1572 2285 4 BEGIN
: 1573 2286 4
: 1574 2287 4 IF .BLK [ FAB$L_FNA ] NEQU 0
: 1575 2288 4 THEN
: 1576 2289 5 BEGIN
: 1577 2290 5
: 1578 2291 5 TEMP_DESC [ DSC$A_POINTER ] = .BLK [ FAB$L_FNA ];
: 1579 2292 5 TEMP_DESC [ DSC$W_LENGTH ] = .BLK [ FAB$B_FNS ];
: 1580 2293 5
: 1581 2294 4 END;
: 1582 2295 4
: 1583 2296 3 END;
: 1584 2297 3 [ FDL$C_AFTNAM ] :
: 1585 2298 3 BEGIN
: 1586 2299 4
: 1587 2300 4
: 1588 2301 4 IF .BLK [ XAB$L_AIA ] NEQU 0
: 1589 2302 4 THEN
: 1590 2303 5 BEGIN
: 1591 2304 5
: 1592 2305 5 TEMP_DESC [ DSC$A_POINTER ] = .BLK [ XAB$L_AIA ];
: 1593 2306 5 TEMP_DESC [ DSC$W_LENGTH ] = .BLK [ XAB$B_AIS ];
: 1594 2307 5
: 1595 2308 4 END;
: 1596 2309 4
: 1597 2310 3 END;
: 1598 2311 3 [ FDL$C_AUDNAM ] :
: 1599 2312 3 BEGIN
: 1600 2313 4
: 1601 2314 4
: 1602 2315 4 IF .BLK [ XAB$L_ATA ] NEQU 0
: 1603 2316 4 THEN
: 1604 2317 5 BEGIN
: 1605 2318 5
: 1606 2319 5 TEMP_DESC [ DSC$A_POINTER ] = .BLK [ XAB$L_ATA ];
: 1607 2320 5 TEMP_DESC [ DSC$W_LENGTH ] = .BLK [ XAB$B_ATS ];
: 1608 2321 5
: 1609 2322 4 END;
: 1610 2323 4
: 1611 2324 3 END;
: 1612 2325 3 [ FDL$C_BEFNAM ] :
: 1613 2326 3 BEGIN
: 1614 2327 4
: 1615 2328 4
: 1616 2329 4 IF .BLK [ XAB$L_BIA ] NEQU 0
: 1617 2330 4 THEN
: 1618 2331 5 BEGIN
: 1619 2332 5
: 1620 2333 5 TEMP_DESC [ DSC$A_POINTER ] = .BLK [ XAB$L_BIA ];
: 1621 2334 5 TEMP_DESC [ DSC$W_LENGTH ] = .BLK [ XAB$B_BIS ];
: 1622 2335 5
: 1623 2336 4 END;
: 1624 2337 4
: 1625 2338 3 END;
: 1626 2339 3

```

```

: 1627      2340  3      [ FDL$C_KYNAME ] :
: 1628      2341  4          BEGIN
: 1629      2342  4
: 1630      2343  4          IF .BLK [ XAB$L_KNM ] NEQU 0
: 1631      2344  4          THEN
: 1632      2345  5              BEGIN
: 1633      2346  5
: 1634      2347  5              TEMP_DESC [ DSC$A_POINTER ] = .BLK [ XAB$L_KNM ];
: 1635      2348  5              TEMP_DESC [ DSC$W_LENGTH ] = 32;
: 1636      2349  5
: 1637      2350  4          END;
: 1638      2351  4
: 1639      2352  3          END;
: 1640      2353  3
: 1641      2354  3      TES;
: 1642      2355  3
: 1643      2356  3      ! If the string length is 0, don't bother putting it out
: 1644      2357  3      !
: 1645      2358  3      IF .TEMP_DESC [ DSC$W_LENGTH ] EQLU 0
: 1646      2359  3      THEN
: 1647      2360  3          RETURN 0;
: 1648      2361  3
: 1649      2362  3      ! Add Quotes or Apostrophes to the output string
: 1650      2363  3      !
: 1651      2364  4      BEGIN
: 1652      2365  4
: 1653      2366  4          LOCAL
: 1654      2367  4              QCHAR      : BYTE,
: 1655      2368  4              OIDX       : WORD,
: 1656      2369  4              ICHAR      : REF VECTOR [ ,BYTE ];
: 1657      2370  4
: 1658      2371  4      ! Get a buffer big enough to hold the result
: 1659      2372  4      !
: 1660      2373  4      STRBYTES = ( .TEMP_DESC [ DSC$W_LENGTH ] * 2 ) + 2;
: 1661      2374  4      IF NOT LIB$GET_VM ( STRBYTES, QCHAR )
: 1662      2375  4      THEN
: 1663      2376  4          SIGNAL STOP ( FDL$_INSVIRMEM );
: 1664      2377  4      CH$FILL ( 0, .STRBYTES, QCHAR );
: 1665      2378  4      ICHAR = .TEMP_DESC [ DSC$A_POINTER ];
: 1666      2379  4
: 1667      2380  4      ! Clear the flags
: 1668      2381  4      !
: 1669      2382  4      FDL$AB_CTRL [ FDL$V_APOST_PRE ] = _CLEAR;
: 1670      2383  4      FDL$AB_CTRL [ FDL$V_QUOTE_PRE ] = _CLEAR;
: 1671      2384  4
: 1672      2385  4      ! Scan the buffer for quotes/apostrophes
: 1673      2386  4      !
: 1674      2387  5      INCR X FROM 0 TO (.TEMP_DESC[DSC$W_LENGTH]-1)
: 1675      2388  4      DO
: 1676      2389  5          BEGIN
: 1677      2390  5
: 1678      2391  5              IF .ICHAR [ .X ] EQLU "'"
: 1679      2392  5              THEN
: 1680      2393  5                  FDL$AB_CTRL [ FDL$V_APOST_PRE ] = _SET;
: 1681      2394  5
: 1682      2395  5              IF .ICHAR [ .X ] EQLU '"'
: 1683      2396  5              THEN
```



```

: 1684 2397 5
: 1685 2398 5
: 1686 2399 4
: 1687 2400 4
: 1688 2401 4
: 1689 2402 4
: 1690 2403 4
: 1691 2404 4
: 1692 2405 4
: 1693 2406 4
: 1694 2407 4
: 1695 2408 5
: 1696 2409 5
: 1697 2410 5
: 1698 2411 5
: 1699 2412 5
: 1700 2413 5
: 1701 2414 6
: 1702 2415 6
: 1703 2416 6
: 1704 2417 6
: 1705 2418 6
: 1706 2419 6
: 1707 2420 6
: 1708 2421 7
: 1709 2422 6
: 1710 2423 6
: 1711 2424 6
: 1712 2425 7
: 1713 2426 7
: 1714 2427 7
: 1715 2428 7
: 1716 2429 7
: 1717 2430 7
: 1718 2431 6
: 1719 2432 7
: 1720 2433 7
: 1721 2434 7
: 1722 2435 6
: 1723 2436 6
: 1724 2437 6
: 1725 2438 6
: 1726 2439 6
: 1727 2440 6
: 1728 2441 5
: 1729 2442 5
: 1730 2443 5
: 1731 2444 6
: 1732 2445 6
: 1733 2446 6
: 1734 2447 6
: 1735 2448 6
: 1736 2449 6
: 1737 2450 6
: 1738 2451 7
: 1739 2452 6
: 1740 2453 7

```

```

        FDL$AB_CTRL [ FDL$V_QUOTE_PRES ] = _SET;
    END;
! Add quotes to a 'vanilla' string
! Add apostrophes to a string with quotes
! Add quotes to a string with apostrophes
! Add quotes to a string with both - and double the quotes
IF .FDL$AB_CTRL [ FDL$V_QUOTE_PRES ]
THEN
    BEGIN
        IF .FDL$AB_CTRL [ FDL$V_APOST_PRES ]
        THEN
            ! Quotes AND Apostrophes were found
            BEGIN
                QCHAR = '''';
                OIDX = 0;
                OCHAR [ .OIDX ] = .QCHAR;
                OIDX = .OIDX + 1;

                INCR I FROM 0 TO (.TEMP_DESC[DSC$W_LENGTH]-1)
                DO
                    IF .ICHAR [ .I ] EQLU .QCHAR
                    THEN
                        BEGIN
                            OCHAR [ .OIDX ] = .QCHAR;
                            OIDX = .OIDX + 1;
                            OCHAR [ .OIDX ] = .QCHAR;
                            OIDX = .OIDX + 1;
                        END
                    ELSE
                        BEGIN
                            OCHAR [ .OIDX ] = .ICHAR [ .I ];
                            OIDX = .OIDX + 1;
                        END;

                OCHAR [ .OIDX ] = .QCHAR;
                OIDX = .OIDX + 1;
            END
        ELSE
            ! Quotes were found, Apostrophes were not
            BEGIN
                QCHAR = '''';
                OIDX = 0;
                OCHAR [ .OIDX ] = .QCHAR;
                OIDX = .OIDX + 1;

                INCR I FROM 0 TO (.TEMP_DESC[DSC$W_LENGTH]-1)
                DO
                    BEGIN

```

```

: 1741 2454 7
: 1742 2455 7          OCHAR [ .OIDX ] = .ICHAR [ .I ];
: 1743 2456 7          OIDX = .OIDX + 1;
: 1744 2457 7
: 1745 2458 6          END;
: 1746 2459 6
: 1747 2460 6          CCHAR [ .OIDX ] = .QCHAR;
: 1748 2461 6          OIDX = .OIDX + 1;
: 1749 2462 6
: 1750 2463 5          END;
: 1751 2464 5
: 1752 2465 5          END
: 1753 2466 4          ELSE
: 1754 2467 5          BEGIN
: 1755 2468 5
: 1756 2469 5          ! If Quotes were not found, it doesn't make
: 1757 2470 5          ! any difference if Apostrophes were
: 1758 2471 5          !
: 1759 2472 5          QCHAR = '';
: 1760 2473 5          OIDX = 0;
: 1761 2474 5          OCHAR [ .OIDX ] = .QCHAR;
: 1762 2475 5          OIDX = .OIDX + 1;
: 1763 2476 5
: 1764 2477 6          INCR I FROM 0 TO (.TEMP_DESC[DSC$W_LENGTH]-1)
: 1765 2478 5          DO
: 1766 2479 6          BEGIN
: 1767 2480 6
: 1768 2481 6          OCHAR [ .OIDX ] = .ICHAR [ .I ];
: 1769 2482 6          OIDX = .OIDX + 1;
: 1770 2483 6
: 1771 2484 5          END;
: 1772 2485 5
: 1773 2486 5          OCHAR [ .OIDX ] = .QCHAR;
: 1774 2487 5          OIDX = .OIDX + 1;
: 1775 2488 5
: 1776 2489 4          END;
: 1777 2490 4
: 1778 2491 4          ! Make the new string the result
: 1779 2492 4          !
: 1780 2493 4          TEMP_DFSC [ DSC$A_POINTER ] = .OCHAR;
: 1781 2494 4          TEMP_DESC [ DSC$W_LENGTH ] = .OIDX;
: 1782 2495 4
: 1783 2496 3          END;
: 1784 2497 3
: 1785 2498 3          ! The final string that resulted
: 1786 2499 3          !
: 1787 2500 3          FAO_PARAM = TEMP_DESC;
: 1788 2501 3
: 1789 2502 2          END;
: 1790 2503 2
: 1791 2504 2          [ FDL$C_QUALIFIER ] :
: 1792 2505 2          BEGIN
: 1793 2506 2
: 1794 2507 2          SELECTONE .FDL$GL_SECONDARY OF
: 1795 2508 2
: 1796 2509 2          SET
: 1797 2510 2

```

```

: 1798      2511      3      [ FDL$C_CARCTRL ] :
: 1799      2512      3
: 1800      2513      4      BEGIN
: 1801      2514      4
: 1802      2515      4      IF .BLK [ FAB$V_CR ]
: 1803      2516      4      THEN
: 1804      2517      4      FAO_PARAM = UPLIT BYTE ( %ASCIC 'carriage_return' )
: 1805      2518      4      ELSE IF .BLK [ FAB$V_FTN ]
: 1806      2519      4      THEN
: 1807      2520      4      FAO_PARAM = UPLIT BYTE ( %ASCIC 'FORTRAN' )
: 1808      2521      4      ELSE IF .BLK [ FAB$V_PRN ]
: 1809      2522      4      THEN
: 1810      2523      4      FAO_PARAM = UPLIT BYTE ( %ASCIC 'print' )
: 1811      2524      4      ELSE
: 1812      2525      4      FAO_PARAM = UPLIT BYTE ( %ASCIC 'none' );
: 1813      2526      4
: 1814      2527      3      END;
: 1815      2528      3
: 1816      2529      3      [ FDL$C_ORG ] :
: 1817      2530      3
: 1818      2531      4      BEGIN
: 1819      2532      4
: 1820      2533      4      SELECT ONE .BLK [ FAB$B_ORG ] OF
: 1821      2534      4
: 1822      2535      4      SET
: 1823      2536      4
: 1824      2537      4      [ FAB$C_IDX ] : FAO_PARAM = UPLIT BYTE ( %ASCIC 'indexed' );
: 1825      2538      4      [ FAB$C_REL ] : FAO_PARAM = UPLIT BYTE ( %ASCIC 'relative' );
: 1826      2539      4      [ FAB$C_SEQ ] : FAO_PARAM = UPLIT BYTE ( %ASCIC 'sequential' );
: 1827      2540      4      [ OTHERWISE ] : 0;
: 1828      2541      4
: 1829      2542      4      TES;
: 1830      2543      4
: 1831      2544      3      END;
: 1832      2545      3
: 1833      2546      3      [ FDL$C_RU ] :
: 1834      2547      3
: 1835      2548      4      BEGIN
: 1836      2549      4
: 1837      2550      4      IF .BLK [ XAB$V_RU ]
: 1838      2551      4      THEN
: 1839      2552      4      FAO_PARAM = UPLIT BYTE ( %ASCIC 'if_in_recovery_unit' )
: 1840      2553      4      ELSE IF .BLK [ XAB$V_ONLY_RU ]
: 1841      2554      4      THEN
: 1842      2555      4      FAO_PARAM = UPLIT BYTE ( %ASCIC 'necessary_to_write' )
: 1843      2556      4      ELSE IF .BLK [ XAB$V_NEVER_RU ]
: 1844      2557      4      THEN
: 1845      2558      4      FAO_PARAM = UPLIT BYTE ( %ASCIC 'never_RU_journal' )
: 1846      2559      4      ELSE
: 1847      2560      4      FAO_PARAM = UPLIT BYTE ( %ASCIC 'none' );
: 1848      2561      4
: 1849      2562      3      END;
: 1850      2563      3
: 1851      2564      3      [ FDL$C_FMT ] :
: 1852      2565      3
: 1853      2566      4      BEGIN
: 1854      2567      4
```

```

: 1855      2568  4      SELECTONE .BLK [ FAB$B_RFM ] OF
: 1856      2569  4
: 1857      2570  4      SET
: 1858      2571  4
: 1859      2572  4      [ FAB$C_STM ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'stream' );
: 1860      2573  4      [ FAB$C_STMCR ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'stream_CR' );
: 1861      2574  4      [ FAB$C_STMLF ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'stream_LF' );
: 1862      2575  4      [ FAB$C_UDF ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'undefined' );
: 1863      2576  4      [ FAB$C_VAR ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'variable' );
: 1864      2577  4      [ FAB$C_VFC ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'VFC' );
: 1865      2578  4      [ FAB$C_FIX ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'fixed' );
: 1866      2579  4      [ OTHERWISE ] : 0;
: 1867      2580  4
: 1868      2581  4      TES;
: 1869      2582  4
: 1870      2583  3      END;
: 1871      2584  3
: 1872      2585  3      [ FDL$C_SEGTyp ] :
: 1873      2586  3
: 1874      2587  4      BEGIN
: 1875      2588  4
: 1876      2589  4      SELECTONE .BLK [ XAB$B_DTP ] OF
: 1877      2590  4
: 1878      2591  4      SET
: 1879      2592  4
: 1880      2593  4      [ XAB$C_BN2 ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'bin2' );
: 1881      2594  4      [ XAB$C_BN4 ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'bin4' );
: 1882      2595  4      [ XAB$C_BN8 ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'bin8' );
: 1883      2596  4      [ XAB$C_PAC ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'decimal' );
: 1884      2597  4      [ XAB$C_IN2 ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'int2' );
: 1885      2598  4      [ XAB$C_IN4 ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'int4' );
: 1886      2599  4      [ XAB$C_IN8 ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'int8' );
: 1887      2600  4      [ XAB$C_STG ] : FAO_PARAM = UPLIT BYTE (%ASCIC 'string' );
: 1888      2601  4      [ OTHERWISE ] : 0;
: 1889      2602  4
: 1890      2603  4      TES;
: 1891      2604  4
: 1892      2605  3      END;
: 1893      2606  3
: 1894      2607  3      [ OTHERWISE ] : 0;
: 1895      2608  3
: 1896      2609  3      TES;
: 1897      2610  3
: 1898      2611  2      END;
: 1899      2612  2
: 1900      2613  2      TES;
: 1901      2614  2
: 1902      2615  2      RETURN $$$_NORMAL;
: 1903      2616  2
: 1904      2617  1      END;

```

.PSECT \_FDL\$PLIT, NOWRT, NOEXE, SHR, PIC, 2

```

73 65 79 03 000BC P.ARG: .ASCII <3>\yes\
6F 6E 02 000CO P.ABH: .ASCII <2>\no\

```

:

55	21	2C	57	55	21	28	20	44	49	5F	65	6C	69	66	000C3	P.ABI:	.ASCII	<3>\yes\
						00	00	00	29	57	55	21	2C	57	000C7	P.ABJ:	.ASCII	<2>\no\
															000CA		.BLKB	2
															000CC	P.ABL:	.ASCII	\file_ID (.UW,.UW,!UW)\<0><0><0>
															000DB			
															010E0015	P.ABK:	.LONG	17694741
															00000000		.ADDRESS	P.ABL
															00000000	P.ABN:	.ASCII	\cylinder !UL\
															010E000C	P.ABM:	.LONG	17694732
															00000000		.ADDRESS	P.ABN
															010E000B	P.ABP:	.ASCII	\logical !UL\<0>
															00000000	P.ABO:	.LONG	17694731
															00000000		.ADDRESS	P.ABP
															010E000B	P.ABR:	.ASCII	\virtual !UL\<0>
															00000000	P.ABQ:	.LONG	17694731
															00000000		.ADDRESS	P.ABR
															010E0005	P.ABT:	.ASCII	<9>\none\<0><0><0>
															00000000	P.ABS:	.LONG	17694725
															00000000		.ADDRESS	P.ABT
															010E000D	P.ABV:	.ASCII	<9>\any_cylinder\<0><0><0>
															00000000			
															00000000	P.ABU:	.LONG	17694733
															00000000		.ADDRESS	P.ABV
															65 63 41	P.ABW:	.ASCII	\Ace\
															61 63 0F	P.ABX:	.ASCII	<15>\carriage_return\
															6E			
															4E 41 52	P.ABY:	.ASCII	<7>\FORTRAN\
															54 52 4F	P.ABZ:	.ASCII	<5>\print\
															6E 69 72	P.ACA:	.ASCII	<4>\none\
															65 6E 6F	P.ACB:	.ASCII	<7>\indexed\
															6E 69 07	P.ACC:	.ASCII	<8>\relative\
															64 65 78	P.ACD:	.ASCII	<10>\sequential\
															65 64 6E	P.ACE:	.ASCII	<19>\if_in_recovery_unit\
															65 75 71			
															65 73 0A	P.ACF:	.ASCII	<18>\necessary_to_write\
															66 69 13	P.ACG:	.ASCII	<16>\never_RU_journal\
															6E 75 5F			
															001A1			
															001A6			
															001B5			
															65 74 69			
															65 6E 10			
															6C 61			
															001C8	P.ACH:	.ASCII	<4>\none\
															65 6E 6F	P.ACI:	.ASCII	<6>\stream\
															6E 72 74	P.ACJ:	.ASCII	<9>\stream_CR\
															74 73 09	P.ACK:	.ASCII	<9>\stream_LF\
															74 73 09	P.ACL:	.ASCII	<9>\undefined\
															001EA	P.ACM:	.ASCII	<8>\variable\
															001F4	P.ACN:	.ASCII	<3>\VFC\
															43 46 56	P.ACO:	.ASCII	<5>\fixed\
															03	P.ACP:	.ASCII	<4>\bin2\
															00201	P.ACQ:	.ASCII	<4>\bin4\
															64 65 78	P.ACR:	.ASCII	<4>\bin8\
															69 66 05	P.ACS:	.ASCII	<7>\decimal\
															00207	P.ACT:	.ASCII	<4>\int2\
															32 6E 69	P.ACU:	.ASCII	<4>\int4\
															69 62 04	P.ACV:	.ASCII	<4>\int8\
															34 6E 69	P.ACW:	.ASCII	<6>\string\
															0020C			
															00211			
															00216			
															0021E			
															00223			
															00228			
															0022D			

.....

.PSECT \_FDL\$CODE, NOWRT, SHR, PIC, 2

		OFFC 00000		FETCH_FIELD:					
		5B	00000000G	00	9E	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	1885
		5A	00000000G	00	9E	00009	MOVAB	FDL\$AB_CTRL, R11	
		59	00000000'	00	9E	00010	MOVAB	FDL\$GL_SECONDARY, R10	
		58	00000000'	00	9E	00017	MOVAB	P.ABG, R9	
		5E		24	C2	0001E	MOVAB	FAO_PARAM, R8	
		50		04	AC	00021	SUBL2	#36, SP	
		51		07	AO	00025	MOVAB	LINE, RO	1962
		56	00000000G0041	00	9A	00029	MOVZBL	7(RO), R1	
		51		08	AO	00031	MOVZBL	FDL\$AB_BLOCK_BLK[R1], BLK	
		52		0A	AO	00035	MOVZWL	8(RO), BOFF	1963
		00		05	AO	00039	MOVZWL	10(RO), POS	1964
012A	09	0016		05E5		0003E	CASEB	5(RO), #0, #9	1968
02C0	0071	05E5		05E5		00046	.WORD	128\$-1\$,-	
	0132	017A		047B		0004E		2\$-1\$,-	
								15\$-1\$,-	
								30\$-1\$,-	
								128\$-1\$,-	
								128\$-1\$,-	
								32\$-1\$,-	
								58\$-1\$,-	
								87\$-1\$,-	
								38\$-1\$	
				0F	11	00052	BRB	3\$	
		00000085		6A	D1	00054	CMPL	FDL\$GL_SECONDARY, #133	1977
				09	13	0005B	BEQL	4\$	
68				52	EF	0005D	EXTZV	POS, #8, (BOFF)[BLK], FAO_PARAM	1979
	6146	08		05BD	31	00063	BRB	128\$	
								FDL\$GL_SECNUM, #0, #7	
		00	00000000G	00	CF	00066	CASEL		1983
0022	07	0016		0010		0006E	.WORD	6\$-5\$,-	
003A	001C	002E		0028		00076		7\$-5\$,-	
	0034							8\$-5\$,-	
								9\$-5\$,-	
								10\$-5\$,-	
								11\$-5\$,-	
								12\$-5\$,-	
								13\$-5\$	
		50		2E	A6	9A 0007E	MOVZBL	46(BLK), RO	1985
					28	11 00082	BRB	14\$	
		50		2F	A6	9A 00084	MOVZBL	47(BLK), RO	1986
					22	11 00088	BRB	14\$	
		50		30	A6	9A 0008A	MOVZBL	48(BLK), RO	1987
					1C	11 0008E	BRB	14\$	
		50		31	A6	9A 00090	MOVZBL	49(BLK), RO	1988
					16	11 00094	BRB	14\$	
		50		32	A6	9A 00096	MOVZBL	50(BLK), RO	1989
					10	11 0009A	BRB	14\$	
		50		33	A6	9A 0009C	MOVZBL	51(BLK), RO	1990
					0A	11 000A0	BRB	14\$	
		50		34	A6	9A 000A2	MOVZBL	52(BLK), RO	1991
					04	11 000A6	BRB	14\$	
		50		35	A6	9A 000A8	MOVZBL	53(BLK), RO	1992
					00AC	31 000AC	BRW	28\$	1983
		50			6A	D0 000AF	MOVL	FDL\$GL_SECONDARY, RO	2002
		00000079		8F	50	D1 000B2	CMPL	RO, #121	2009

				1B	12	000B9		BNEQ	16\$			
		53	1C	A6	3C	000BB		MOVZWL	28(BLK), R3		2011	
		53	00000064	8F	C4	000BF		MULL2	#100, R3			
		50	1C	A6	3C	000C6		MOVZWL	28(BLK), R0			
		50		02	C6	000CA		DIVL2	#2, R0			
		53		50	C0	000CD		ADDL2	R0, R3			
		50	0A	A6	9A	000D0		MOVZBL	10(BLK), R0		2012	
				22	11	000D4		BRB	17\$			
		0000007F		50	D1	000D6	16\$:	CMPL	R0, #127		2014	
				2D	12	000DD		BNEQ	18\$			
		53	1A	A6	3C	000DF		MOVZWL	26(BLK), R3		2016	
		53	00000064	8F	C4	000E3		MULL2	#100, R3			
		50	1A	A6	3C	000EA		MOVZWL	26(BLK), R0			
		50		02	C6	000EE		DIVL2	#2, R0			
		53		50	C0	00CF1		ADDL2	R0, R3			
		50	08	A6	9A	000F4		MOVZBL	8(BLK), R0		2017	
		50	00000000G	03	C0	000F8	17\$:	ADDL2	FDL\$AB AREA_BKZ, R0			
		50		60	9A	000FF		MOVZBL	(R0), R0			
50		50		09	78	00102		ASHL	#9, R0, R0			
68		53		50	C7	00106		DIVL3	R0, R3, FAO_PARAM			
				62	11	0010A		BRB	31\$		2015	
		00000086		50	D1	0010C	18\$:	CMPL	R0, #134		2021	
				4B	12	00113		BNEQ	29\$			
		07	00	00000000G	00	CF	00115	CASEL	FDL\$GL SECNUM, #0, #7		2024	
0022	001C	0016		0010			0011D	.WORD	20\$-19\$,-			
003A	0034	002E		0028			0C 25		21\$-19\$,-			
									22\$-19\$,-			
									23\$-19\$,-			
									24\$-19\$,-			
									25\$-19\$,-			
									26\$-19\$,-			
									27\$-19\$,-			
		50	1E	A6	3C	0012D	20\$:	MOVZWL	30(BLK), R0		2026	
				28	11	00131		BRB	28\$			
		50	20	A6	3C	00133	21\$:	MOVZWL	32(BLK), R0		2027	
				22	11	00137		BRB	28\$			
		50	22	A6	3C	00139	22\$:	MOVZWL	34(BLK), R0		2028	
				1C	11	0013D		BRB	28\$			
		50	24	A6	3C	0013F	23\$:	MOVZWL	36(BLK), R0		2029	
				16	11	00143		BRB	28\$			
		50	26	A6	3C	00145	24\$:	MOVZWL	38(BLK), R0		2030	
				10	11	00149		BRB	28\$			
		50	28	A6	3C	0014B	25\$:	MOVZWL	40(BLK), R0		2031	
				0A	11	0014F		BRB	28\$			
		50	2A	A6	3C	00151	26\$:	MOVZWL	42(BLK), R0		2032	
				04	11	00155		BRB	28\$			
		50	2C	A6	3C	00157	27\$:	MOVZWL	44(BLK), R0		2033	
		68		50	D0	0015B	28\$:	MOVL	R0, FAO_PARAM		2024	
				56	11	0015E		BRB	37\$		2002	
68	6146		10	52	EF	00160	29\$:	EXTZV	POS, #16, (BOFF)[BLK], FAO_PARAM		2038	
				4E	11	00166		BRB	37\$		1968	
68	6146		20	52	EF	00168	30\$:	EXTZV	POS, #32, (BOFF)[BLK], FAO_PARAM		2047	
				46	11	0016E	31\$:	BRB	37\$		1968	
		50		6A	D0	00170	32\$:	MOVL	FDL\$GL SECONDARY, R0		2072	
		0000007A		8F	50	D1	00173	CMPL	R0, #122			
				1B	13	0017A		BEQL	33\$			
		0000007B		50	D1	0017C		CMPL	R0, #123		2074	





			18	A6	B5	00252		TSTW	24(BLK)		2155
				0A	12	00255		BNEQ	46\$		
			1A	A6	B5	00257		TSTW	26(BLK)		2157
				05	12	0025A		BNEQ	46\$		
			1C	A6	B5	0025C		TSTW	28(BLK)		2159
				26	13	0025F		BEQL	47\$		
	7E		1C	A6	3C	00261	46\$:	MOVZWL	28(BLK), -(SP)		2169
	7E		1A	A6	3C	00265		MOVZWL	26(BLK), -(SP)		
	7E		18	A6	3C	00269		MOVZWL	24(BLK), -(SP)		
			CC	A8	9F	0026D		PUSHAB	TEMP_DESC		
			10	AE	9F	00270		PUSHAB	TEMP_WORD		
			28	A9	9F	00273		PUSHAB	P.ABR		
00000000G	00			06	FB	00276		CALLS	#6, SYSSFAO		
	52			50	E9	0027D		BLBC	STATUS, 51\$		
	CC	A8		6E	B0	00280		MOVW	TEMP_WORD, TEMP_DESC		
				022B	31	00284		BRW	86\$		
	50		09	A6	9A	00287	47\$:	MOVZBL	9(BLK), R0		
	01			50	91	0028B		CMPB	R0, #1		2182
				17	12	0028E		BNEQ	49\$		
			0C	A6	DD	00290		PUSHL	12(BLK)		2189
			CC	A8	9F	00293		PUSHAB	TEMP_DESC		
			08	AE	9F	00296		PUSHAB	TEMP_WORD		
			3C	A9	9F	00299		PUSHAB	P.ABM		
00000000G	00			04	FB	0029C	48\$:	CALLS	#4, SYSSFAO		
	30			50	E8	002A3		BLBS	STATUS, 52\$		
				04	002A6			RET			2191
	02			50	91	002A7	49\$:	CMPB	R0, #2		2196
				0E	12	002AA		BNEQ	50\$		
			0C	A6	DD	002AC		PUSHL	12(BLK)		2203
			CC	A8	9F	002AF		PUSHAB	TEMP_DESC		
			08	AE	9F	002B2		PUSHAB	TEMP_WORD		
			50	A9	9F	002B5		PUSHAB	P.AB0		
				E2	11	002B8		BRB	48\$		
	03			50	91	002BA	50\$:	CMPB	R0, #3		2210
				21	12	002BD		BNEQ	53\$		
			0C	A6	DD	002BF		PUSHL	12(BLK)		2217
			CC	A8	9F	002C2		PUSHAB	TEMP_DESC		
			08	AE	9F	002C5		PUSHAB	TEMP_WORD		
			64	A9	9F	002C8		PUSHAB	P.AB0		
00000000G	00			04	FB	002CB		CALLS	#4, SYSSFAO		
	01			50	E8	002D2	51\$:	BLBS	STATUS, 52\$		
				04	002D5			RET			
	CC	A8		6E	B0	002D6	52\$:	MOVW	TEMP_WORD, TEMP_DESC		2219
	68		CC	A8	9E	002DA		MOVAB	TEMP_DESC, FAO_PARAM		2220
				0D	11	002DE		BRB	54\$		2178
				50	D5	002E0	53\$:	TSTL	R0		2224
				09	12	002E2		BNEQ	54\$		
09	08	A6		01	E0	002E4		BBS	#1, 8(BLK), 55\$		2227
	68		74	A9	9E	002E9		MOVAB	P.ABS, FAO_PARAM		2229
09	08	A6		01	E1	002ED	54\$:	BBC	#1, 8(BLK), 57\$		2237
	68		008C	C9	9E	002F2	55\$:	MOVAB	P.ABU, FAO_PARAM		2239
				02	11	002F7		BRB	57\$		2104
				68	D4	002F9	56\$:	CLRL	FAO_PARAM		2245
				0325	31	002FB	57\$:	BRW	128\$		1968
			CC	A8	B4	002FE	58\$:	CLRW	TEMP_DESC		2256
	50			6A	D0	00301		MOVL	FDL\$GL_SECONDARY, R0		2258
	08			50	D1	00304		CMPB	R0, #8		2262

			0D	12	00307	BNEQ	59\$		
	DO	A8	0094	C9	9E 00309	MOVAB	P,ABW, TEMP_DESC+4		2265
	CC	A8		03	B0 0030F	MOVW	#3, TEMP_DESC		2266
				0083	31 00313	BRW	65\$		2258
	0000004F	8F		50	D1 00316	CMPL	R0, #79		2270
				14	12 0031D	BNEQ	61\$		
			30	A6	D5 0031F	TSTL	48(BLK)		2273
				03	12 00322	BNEQ	60\$		
				0080	31 00324	BRW	67\$		
	DO	A8	30	A6	DC 00327	MOVL	48(BLK), TEMP_DESC+4		2277
	CC	A8	35	A6	9B 0032C	MOVZBW	53(BLK), TEMP_DESC		2278
				7F	11 00331	BRB	68\$		2258
	0000005E	8F		50	D1 00333	CMPL	R0, #94		2284
				11	12 0033A	BNEQ	62\$		
			2C	A6	D5 0033C	TSTL	44(BLK)		2287
				71	13 0033F	BEQL	60\$		
	DO	A8	2C	A6	D0 00341	MOVL	44(BLK), TEMP_DESC+4		2291
	CC	A8	34	A6	9B 00346	MOVZBW	52(BLK), TEMP_DESC		2292
				65	11 0034B	BRB	68\$		2258
	00000071	8F		50	D1 0034D	CMPL	R0, #113		2298
				11	12 00354	BNEQ	63\$		
			18	A6	D5 00356	TSTL	24(BLK)		2301
				57	13 00359	BEQL	68\$		
	DO	A8	18	A6	D0 0035B	MOVL	24(BLK), TEMP_DESC+4		2305
	CC	A8	14	A6	9B 00360	MOVZBW	20(BLK), TEMP_DESC		2306
				4B	11 00365	BRB	68\$		2258
	00000073	8F		50	D1 00367	CMPL	R0, #115		2312
				11	12 0036E	BNEQ	64\$		
			20	A6	D5 00370	TSTL	32(BLK)		2315
				3D	13 00373	BEQL	68\$		
	DO	A8	20	A6	D0 00375	MOVL	32(BLK), TEMP_DESC+4		2319
	CC	A8	1C	A6	9B 0037A	MOVZBW	28(BLK), TEMP_DESC		2320
				31	11 0037F	BRB	68\$		2258
	00000075	8F		50	D1 00381	CMPL	R0, #117		2326
				11	12 00388	BNEQ	66\$		
			10	A6	D5 0038A	TSTL	16(BLK)		2329
				23	13 0038D	BEQL	68\$		
	DO	A8	10	A6	D0 0038F	MOVL	16(BLK), TEMP_DESC+4		2333
	CC	A8	0C	A6	9B 00394	MOVZBW	12(BLK), TEMP_DESC		2334
				17	11 00399	BRB	68\$		2258
	00000081	8F		50	D1 0039B	CMPL	R0, #129		2340
				0E	12 003A2	BNEQ	68\$		
			38	A6	D5 003A4	TSTL	56(BLK)		2343
				09	13 003A7	BEQL	68\$		
	DC	A8	38	A6	D0 003A9	MOVL	56(BLK), TEMP_DESC+4		2347
	CC	A8		20	B0 003AE	MOVW	#32, TEMP_DESC		2348
		50	CC	A8	3C 003B2	MOVZWL	TEMP_DESC, R0		2358
				03	12 003B6	BNEQ	69\$		
				026C	31 003B8	BRW	129\$		
	10	A8		01	78 003BB	ASHL	#1, R0, STRBYTES		2373
				02	C0 003C0	ADDL2	#2, STRBYTES		
			14	A8	9F 003C4	PUSHAB	OCHAR		2374
			10	A8	9F 0C3C7	PUSHAB	STRBYTES		
	00000000G	00		02	FB 003CA	CALLS	#2, LIB\$GET_VM		
		0D		50	E8 003D1	BLBS	R0, 70\$		
	00000000G	00	00000000G	8F	DD 003D4	PUSHL	#FDL\$ INSVIRMEM		2376
				01	FB 003DA	CALLS	#1, LIB\$STOP		

10	AB	00	57 6E	14	AB 00 67	DO 2C 003E1 003E5 003EB	70\$:	MOVL MOVCS	QCHAR, R7 #0, (SP), #0, STRBYTES, (R7)	2377
			50 AB 55 51	01	DO CO CC AB 01 16	DO 8A 003F0 003F5 003F9 003FC		MOVL BICB2 MOVZWL MNEGL BRB	TEMP_DESC+4, ICHAR #192, FDL\$AB_CTRL+1 TEMP_DESC, R5 #1, X 73\$	2378 2383 2387 2391
			27		6140 05	91 12	71\$:	CMPB BNEQ	(X)[ICHAR], #39 72\$	
	AB		22	01	40 6140	8F 91	72\$:	BISB2 CMPB	#64, FDL\$AB_CTRL+1 (X)[ICHAR], #34	2393 2395
		E6	51	01	80	8F 55	73\$:	BISB2 AOBLSS	#128, FDL\$AB_CTRL+1 R5, X, 71\$	2397 2387
					01	52 AB		CLRW TSTB	OIDX FDL\$AB_CTRL+1	2417 2406
		36	51	01	06	E1		BGEQ BBC	81\$ #6, FDL\$AB_CTRL+1, 78\$	2410 2416
			53 6347		22 52	90 3C		MOVB MOVZWL	#34, QCHAP OIDX, R3	2418
			54		51	90		MOVZWL MOVZWL	QCHAR, (R3)[R7] QCHAR, (R3)[R7]	2419 2421
			53 51		52 6440	3C 91	74\$:	MNEGL BRB MOVZWL	#1, I 77\$ OIDX, R3	2426 2423
			6347		0F	i2		CMPB BNEQ	(I)[ICHAR], QCHAR 75\$	2426 2427
			53 6347		51	90		MOVZWL MOVZWL	QCHAR, (R3)[R7] QCHAR, (R3)[R7]	2428
			6347		05	11		BRB	76\$	2423
			54		6440	90	75\$:	MOVZWL MOVZWL	(I)[ICHAR], (R3)[R7] (I)[ICHAR], (R3)[R7]	2433 2429
		DD	54		52	B6	76\$:	INCW AOBLSS	OIDX R5, I, 74\$	2427 2423
			51		47	11	77\$:	BRB	84\$	2437
			53 6347		27	90	78\$:	MOVZWL MOVZWL	#39, QCHAR OIDX, R3	2446 2448
			53		51	90		MOVZWL MOVZWL	QCHAR, (R3)[R7] QCHAR, (R3)[R7]	2449 2451
			6447		52	B6	79\$:	INCW MNEGL	OIDX #1, I	2455
			53 6347		0A	11		BRB	80\$	2456
		F2	53 6347		52	3C	80\$:	MOVZWL MOVZWL	OIDX, R4 (I)[ICHAR], (R4)[R7]	2455 2456
			51		55	F2	81\$:	AOBLSS MOVZWL	R5, I, 79\$ OIDX, R3	2451 2460
			53 6347		52	3C		MOVZWL MOVZWL	QCHAR, (R3)[R7] QCHAR, (R3)[R7]	2406 2472
			53		26	11		BRB	85\$	2474
			54		22	90	81\$:	MOVZWL MOVZWL	#34, QCHAR OIDX, R3	2475 2477
					52	3C		MOVZWL MOVZWL	QCHAR, (R3)[R7] QCHAR, (R3)[R7]	2481
					51	90		INCW MNEGL	OIDX #1, I	
					0A	11		BRB	83\$	
					52	3C	82\$:	MOVZWL	OIDX, R4	

	6447		6340	90	00496		MOV B	(1)[ICHAR], (R4)[R7]		
			52	B6	00498		INC W	OIDX	2482	
F2	53		55	F2	0049D	83\$:	AOBLSS	R5, 1, 82\$	2477	
	50		52	3C	004A1	84\$:	MOVZWL	OIDX, R0	2486	
	6047		51	90	004A4		MOV B	QCHAR, (R0)[R7]		
			52	B6	004A8	85\$:	INC W	OIDX	2438	
		DO	57	DO	004AA		MOV L	R7, TEMP_DESC+4	2493	
		CC	52	B0	004AE		MOV W	OIDX, TEMP_DESC	2494	
			68	CC	9E	004B2	86\$:	MOVAB	TEMP_DESC, -FAO_PARAM	2500
			0085	31	004B6		BRW	99\$	1968	
			50	6A	DO	004B9	87\$:	MOV L	FDL\$GL_SECONDARY, R0	2507
	00000089		8F	50	D1	004BC		CMPL	R0, #137	2511
				2A	12	004C3		BNEQ	93\$	
07	1E	A6	01	E1	004C5		BBC	#1, 30(BLK), 88\$	2515	
			68	0097	C9	9E	004CA	MOVAB	P.ABX, FAO_PARAM	2517
				09	11	004CF		BRB	89\$	
			07	1E	A6	E9	004D1	88\$:	BLBC	30(BLK), 90\$
			68	00A7	C9	9E	004D5	MOVAB	P.ABY, FAO_PARAM	2520
				76	11	004DA	89\$:	BRB	102\$	
07	1E	A6	02	E1	004DC	90\$:	BBC	#2, 30(BLK), 91\$	2521	
			68	00AF	C9	9E	004E1	MOVAB	P.ABZ, FAO_PARAM	2523
				05	11	004E6		BRB	92\$	
			68	00B5	C9	9E	004E8	91\$:	MOVAB	P.ACA, FAO_PARAM
				7D	11	004ED	92\$:	BRB	104\$	2525
	00000062		8F	50	D1	004EF	93\$:	CMPL	R0, #98	2507
				28	12	004F6		BNEQ	97\$	2529
			50	1D	A6	9A	004F8	MOVZBL	29(BLK), R0	2533
			20	50	91	004FC		CMPB	R0, #32	2537
				08	12	004FF		BNEQ	94\$	
			68	00BA	C9	9E	00501	MOVAB	P.ACB, FAO_PARAM	
				0089	31	00506		BRW	108\$	
			10	70	91	00509	94\$:	CMPB	R0, #16	2538
				07	12	0050C		BNEQ	95\$	
			68	00C2	C9	9E	0050E	MOVAB	P.ACC, FAO_PARAM	
				09	11	00513		BRB	96\$	
				50	D5	00515	95\$:	TSTL	R0	2539
				05	12	00517		BNEQ	96\$	
			68	00CB	C9	9E	00519	MOVAB	P.ACD, FAO_PARAM	
				7F	11	0051E	96\$:	BRB	110\$	
	00000076		8F	50	D1	00520	97\$:	CMPL	R0, #118	2546
				2C	12	00527		BNEQ	103\$	
07	08	A6	01	E1	00529		BBC	#1, 8(BLK), 98\$	2550	
			68	00D6	C9	9E	0052E	MOVAB	P.ACE, FAO_PARAM	2552
				09	11	00533		BRB	99\$	
			07	08	A6	E9	00535	98\$:	BLBC	8(BLK), 100\$
			68	00EA	C9	9E	00539	MOVAB	P.ACF, FAO_PARAM	2555
				77	11	0053E	99\$:	BRB	113\$	
08	08	A6	05	E1	00540	100\$:	BBC	#5, 8(BLK), 101\$	2556	
			68	00FD	C9	9E	00545	MOVAB	P.ACG, FAO_PARAM	2558
				0083	31	0054A		BRW	115\$	
			68	010E	C9	9E	0054D	101\$:	MOVAB	P.ACH, FAO_PARAM
				0087	31	00552	102\$:	BRW	117\$	2560
	0000008B		8F	50	D1	00555	103\$:	CMPL	R0, #139	2507
				5B	12	0055C		BNEQ	114\$	2564
			50	1F	A6	9A	0055E	MOVZBL	31(BLK), R0	2568
			04	50	91	00562		CMPB	R0, #4	2572
				08	12	00565		BNEQ	105\$	

68	0115	C9	9E	00567	MOVAB	P.ACI, FAO_PARAM	
		0085	31	0056C 104\$:	BRW	120\$	
06		50	91	0056F 105\$:	CMPB	RO, #6	2573
		08	12	00572	BNEQ	106\$	
68	011A	C9	9E	00574	MOVAB	P.ACJ, FAO_PARAM	
		0084	31	00579	BRW	122\$	
05		50	91	0057C 106\$:	CMPB	RO, #5	2574
		08	12	0057F	BNEQ	107\$	
68	0124	C9	9E	00581	MOVAB	P.ACK, FAO_PARAM	
		0083	31	00586	BRW	124\$	
		50	D5	00589 107\$:	TSTL	RO	2575
		08	12	0058B	BNEQ	109\$	
68	012E	C9	9E	0058D	MOVAB	P.ACL, FAO_PARAM	
		0083	31	00592 108\$:	BRW	126\$	
02		50	91	00595 109\$:	CMPB	RO, #2	2576
		07	12	00598	BNEQ	111\$	
68	0138	C9	9E	0059A	MOVAB	P.ACM, FAO_PARAM	
		77	11	0059F 110\$:	BRB	126\$	
03		50	91	005A1 111\$:	CMPB	RO, #3	2577
		07	12	005A4	BNEQ	112\$	
68	0141	C9	9E	005A6	MOVAB	P.ACN, FAO_PARAM	
		76	11	005AB	BRB	128\$	
01		50	91	005AD 112\$:	CMPB	RO, #1	2578
		71	12	005B0	BNEQ	128\$	
68	0145	C9	9E	005B2	MOVAB	P.ACO, FAO_PARAM	
		6A	11	005B7 113\$:	BRB	128\$	
00000087	8F	50	D1	005B9 114\$:	CMPL	RO, #135	2585
		61	12	005C0	BNEQ	128\$	
50	13	A6	9A	005C2	MOVZBL	19(BLK), RO	2589
02		50	91	005C6	CMPB	RO, #2	2593
		07	12	005C9	BNEQ	116\$	
68	014B	C9	9E	005CB	MOVAB	P.ACP, FAO_PARAM	
		51	11	005D0 115\$:	BRB	128\$	
04		50	91	005D2 116\$:	CMPB	RO, #4	2594
		07	12	005D5	BNEQ	118\$	
68	0150	C9	9E	005D7	MOVAB	P.ACQ, FAO_PARAM	
		45	11	005DC 117\$:	BRB	128\$	
07		50	91	005DE 118\$:	CMPB	RO, #7	2595
		07	12	005E1	BNEQ	119\$	
68	0155	C9	9E	005E3	MOVAB	P.ACR, FAO_PARAM	
		39	11	005E8	BRB	128\$	
05		50	91	005EA 119\$:	CMPB	RO, #5	2596
		07	12	005ED	BNEQ	121\$	
68	015A	C9	9E	005EF	MOVAB	P.ACS, FAO_PARAM	
		2D	11	005F4 120\$:	BRB	128\$	
01		50	91	005F6 121\$:	CMPB	RO, #1	2597
		07	12	005F9	BNEQ	123\$	
68	0162	C9	9E	005FB	MOVAB	P.ACT, FAO_PARAM	
		21	11	00600 122\$:	BRB	128\$	
03		50	91	00602 123\$:	CMPB	RO, #3	2598
		07	12	00605	BNEQ	125\$	
68	0167	C9	9E	00607	MOVAB	P.ACU, FAO_PARAM	
		15	11	0060C 124\$:	BRB	128\$	
06		50	91	0060E 125\$:	CMPB	RO, #6	2599
		07	12	00611	BNEQ	127\$	
68	016C	C9	9E	00613	MOVAB	P.ACV, FAO_PARAM	
		09	11	00618 126\$:	BRB	128\$	

```

50 D5 0061A 127$: TSTL R0 ; 2600
05 12 0061C BNEQ 128$ ;
68 0171 C9 9E 0061E MOVAB P,ACW, FAO_PARAM ;
50 01 D0 00623 128$: MOVL #1, R0 ; 2615
04 00626 RET ;
50 D4 00627 129$: CLRL R0 ; 2617
04 00629 RET ;

```

: Routine Size: 1578 bytes, Routine Base: \_FDL\$CODE + 063E

```

: 1905 2618 1
: 1906 2619 0 END ELUDOM

```

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
_FDL\$OWN	168	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, PIC, ALIGN(2)
_FDL\$PLIT	564	NOVEC, NOWRT, RD, NOEXE, SHR, LCL, REL, CON, PIC, ALIGN(2)
_FDL\$CODE	3176	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	----- Symbols -----		Pages Mapped	Processing Time
	Total	Loaded Percent		
_S255\$DUA28:[SYSLIB]STARLET.L32;1	9776	110 1	581	00:00.9

COMMAND QUALIFIERS

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

```

: Size: 3176 code + 732 data bytes
: Run Time: 01:02.9
: Elapsed Time: 02:59.2
: Lines/CPU Min: 2499
: Lexemes/CPU-Min: 17141
: Memory Used: 527 pages
: Compilation Complete

```

This image displays a grid of 144 small, dark panels, each representing a different type of system output or diagnostic data. The panels are arranged in a 12x12 grid. Several panels are clearly labeled with titles, including:

- FOLDATA LIS
- FDLUTL REQ
- FDLSHR MAP
- FDLDEF SDL
- FDL CALL LIS
- CREATED LIS
- FDLDRIVER LIS
- FDLGEN LIS

The remaining panels contain various types of data, including text-based logs, tables, and graphical representations, all rendered in a low-contrast, monochrome style typical of early computer terminal outputs.

This page contains a grid of code listings for various VAX/VMS utilities. Each utility name is followed by 'LIS'. The utilities listed include:

- FDL\_PARSE LIS
- FDL\_PARSE DEF LIS
- FDL\_SDLMSG LIS
- FDLTABES LIS
- FDL\_GENTAB LIS
- FDLMSG LIS

The grid consists of multiple columns and rows of code, with some utilities appearing in multiple columns. Each listing typically includes a header line (e.g., 'FDL\_PARSE LIS'), followed by several lines of code, and ends with a '\*\*\*' line. The code is printed in a monospaced font and is somewhat faded and difficult to read in detail.