

```
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFF
FFF
FFF
FFF
FFF
FFF
FFF
FFFFFFFFFFFF
FFFFFFFFFFFF
FFFFFFFFFFFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
FFF
```

```
111
111
111
111111
111111
111111
111
111
111
111
111
111
111
111
111
111
111
111
111
111
111
111
111111111
111111111
111111111
```

```
111
111
111
111111
111111
111111
111
111
111
111
111
111
111
111
111
111
111
111
111
111
111
111
111111111
111111111
111111111
```

```
AAAAAAAAAA
AAAAAAAAAA
AAAAAAAAAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
AAA
```

```

EEEEEEEEEE XX XX TTTTTTTTTT IIIIII DDDDDDDD XX XX
EEEEEEEEEE XX XX TTTTTTTTTT IIIIII DDDDDDDD XX XX
EE XX XX TT II DD DD XX XX
EE XX XX TT II DD DD XX XX
EE XX XX TT II DD DD XX XX
EEEEEEEE XX XX TT II DD DD XX XX
EEEEEEEE XX XX TT II DD DD XX XX
EE XX XX TT II DD DD XX XX
EE XX XX TT II DD DD XX XX
EE XX XX TT II DD DD XX XX
EEEEEEEEEE XX XX TT IIIIII DDDDDDDD XX XX
EEEEEEEEEE XX XX TT IIIIII DDDDDDDD XX XX

```

```

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

```



```

1 0001 0 MODULE EXTIDX (
2 0002 0
3 0003 0 LANGUAGE (BLISS32),
4 0004 0 IDENT = 'V04-000',
5 0005 1 BEGIN
6 0006 1
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
11 0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
12 0012 1 * ALL RIGHTS RESERVED. *
13 0013 1 *
14 0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
15 0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
16 0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
17 0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
18 0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
19 0019 1 * TRANSFERRED. *
20 0020 1 *
21 0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
22 0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
23 0023 1 * CORPORATION. *
24 0024 1 *
25 0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
26 0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *
27 0027 1 *
28 0028 1 *
29 0029 1 *****
30 0030 1
31 0031 1 **
32 0032 1
33 0033 1 FACILITY: F11ACP Structure Level 1
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1 This routine extends the volume's index file.
38 0038 1
39 0039 1 ENVIRONMENT:
40 0040 1
41 0041 1 STARLET operating system, including privileged system services
42 0042 1 and internal exec routines.
43 0043 1
44 0044 1 --
45 0045 1
46 0046 1
47 0047 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 14-Apr-1977 10:44
48 0048 1
49 0049 1 MODIFIED BY:
50 0050 1
51 0051 1 A0101 ACG0121 Andrew C. Goldstein, 16-Jan-1980 23:00
52 0052 1 Make context save and restore into subroutines
53 0053 1
54 0054 1 A0100 ACG00001 Andrew C. Goldstein, 10-Oct-1978 20:02
55 0055 1 Previous revision history moved to F11A.REV
56 0056 1 **
57 0057 1

```



```

62 0375 1 GLOBAL ROUTINE EXTEND_INDEX (FILE_NUMBER) : NOVALUE =
63 0376 1
64 0377 1 ++
65 0378 1
66 0379 1 FUNCTIONAL DESCRIPTION:
67 0380 1
68 0381 1     This routine extends the volume's index file.
69 0382 1
70 0383 1 CALLING SEQUENCE:
71 0384 1     EXTEND_INDEX (ARG1)
72 0385 1
73 0386 1 INPUT PARAMETERS:
74 0387 1     ARG1: next file number to be created
75 0388 1
76 0389 1 IMPLICIT INPUTS:
77 0390 1     CURRENT_VCB: address of volume VCB
78 0391 1
79 0392 1 OUTPUT PARAMETERS:
80 0393 1     NONE
81 0394 1
82 0395 1 IMPLICIT OUTPUTS:
83 0396 1     NONE
84 0397 1
85 0398 1 ROUTINE VALUE:
86 0399 1     NONE
87 0400 1
88 0401 1 SIDE EFFECTS:
89 0402 1     index file extended, index file window and index file FCB modified
90 0403 1
91 0404 1 --
92 0405 1
93 0406 2 BEGIN
94 0407 2
95 0408 2 LOCAL
96 0409 2     FIB          : REF BBLOCK,      : address of FIB for extend operation
97 0410 2     HEADER      : REF BBLOCK,      : address of index file header
98 0411 2     FCB         : REF BBLOCK,      : address of index file FCB
99 0412 2     WINDOW      : REF BBLOCK,      : address of index file window
100 0413 2     FREE_POINTERS, :                : number of free retrieval pointers
101 0414 2     :                : in index file window
102 0415 2     FILES_TO_GO,   :                : number of files likely to be created
103 0416 2     :                : on this volume
104 0417 2     BLOCKS_NEEDED: :                : amount to extend index file by
105 0418 2
106 0419 2 EXTERNAL
107 0420 2     CLEANUP_FLAGS : BITVECTOR,    : cleanup action flags
108 0421 2     USER_STATUS   : VECTOR,       : I/O status block of user
109 0422 2     CURRENT_VCB   : REF BBLOCK,    : VCB of volume in process
110 0423 2     PRIMARY_FCB   : REF BBLOCK,    : address of FCB in process
111 0424 2     CURRENT_WINDOW : REF BBLOCK,    : address of window in process
112 0425 2     SECOND_FIB    : BBLOCK;        : FIB for secondary operation
113 0426 2
114 0427 2 EXTERNAL ROUTINE
115 0428 2     SAVE_CONTEXT, :                : save reentrant context area
116 0429 2     RESTORE_CONTEXT, :                : restore reentrant context area
117 0430 2     READ_HEADER,  :                : read file header
118 0431 2     TURN_WINDOW,  :                : update file window

```

```
119 0432      EXTEND,           | extend a file
120 0433      CHECKSUM,        | compute file header checksum
121 0434      WRITE_HEADER,   | write back file header
122 0435      INIT_FCB;        | update file control block
123 0436
124 0437
125 0438      ! Extending the index file is a secondary operation, so we must save away the
126 0439      ! primary context, and then set up the appropriate context for this operation.
127 0440
128 0441
129 0442      SAVE_CONTEXT ();
130 0443      FIB = SECOND FIB;
131 0444      FIB[FIB$W_FIB_NUM] = 1;
132 0445      FIB[FIB$W_FIB_SEQ] = 1;
133 0446
134 0447      PRIMARY_FCB = FCB = .CURRENT_VCB[VCB$S_L_FCBFL];
135 0448      CURRENT_WINDOW = WINDOW = .FCB[FCB$S_L_W[FL]];
136 0449
137 0450      ! Now read the index file header and turn the index file window to VBN 3.
138 0451      ! Then compute the number of free retrieval pointers in the index file window,
139 0452      ! discounting pointers (if any) that only map the boot and home block.
140 0453
141 0454
142 0455      HEADER = READ_HEADER (0, .FCB);
143 0456      KERNEL_CALL (TURN_WINDOW, .WINDOW, .HEADER, 3, 1);
144 0457
145 0458      FREE_POINTERS = (.WINDOW[WCBSW_SIZE]-WCBSW_LENGTH)/6 - .WINDOW[WCBSW_NMAP];
146 0459      IF .WINDOW[WCBSL_STVBN] + .WINDOW[WCBSW_P1_COUNT] LEQU 3
147 0460      THEN
148 0461          BEGIN
149 0462              FREE_POINTERS = .FREE_POINTERS + 1;
150 0463              IF .WINDOW[WCBSL_STVBN] + .WINDOW[WCBSW_P1_COUNT] + .WINDOW[WCBSW_P2_COUNT] LEQU 3
151 0464              THEN FREE_POINTERS = .FREE_POINTERS + 1;
152 0465              END;
153 0466      IF .FREE_POINTERS LEQ 0 THEN FREE_POINTERS = 1;
154 0467
155 0468      ! Compute the number of files likely to still be created on the volume. This
156 0469      ! is the minimum of the number permitted minus the current number and a
157 0470      ! fraction of the number of free blocks on the volume. The amount to extend
158 0471      ! the index file by is this quantity divided by the number of available
159 0472      ! retrieval pointers in the index file window.
160 0473
161 0474
162 0475      FILES_TO_GO = MINU (.CURRENT_VCB[VCB$S_MAXFILES] - .FILE_NUMBER + 1,
163 0476                      .CURRENT_VCB[VCB$S_FREE] / .CURRENT_VCB[VCBSW_CLUSTER] / 4);
164 0477
165 0478      BLOCKS_NEEDED = MINU (.FILES_TO_GO / .FREE_POINTERS, 1000);
166 0479
167 0480      ! Build the extend control in the FIB and call the EXTEND routine.
168 0481
169 0482
170 0483      FIB[FIB$S_EXSZ] = .BLOCKS_NEEDED;
171 0484      FIB[FIB$S_ALCON] = 1;
172 0485      FIB[FIB$S_ALCONB] = 1;
173 0486      FIB[FIB$S_ALDEF] = 1;
174 0487      FIB[FIB$S_NOHDREXT] = 1;
175 0488
```

```

: 176 0489 2 EXTEND (.FIB, .HEADER);
: 177 0490 ~~~~~
: 178 0491 ! Now write the header, update the FCB, and restore the primary context.
: 179 0492 !
: 180 0493 ~~~~~
: 181 0494 CHECKSUM (.HEADER);
: 182 0495 WRITE_HEADER ();
: 183 0496 KERNEC_CALL (INIT_FCB, .FCB, .HEADER);
: 184 0497 ~~~~~
: 185 0498 RESTORE_CONTEXT ();
: 186 0499 USER_STATUS[1] = 0;
: 187 0500 ~~~~~
: 188 0501 1 END;

```

! end of routine EXTEND_INDEX

.TITLE EXTIDX
.IDENT \V04-000\

.EXTRN CLEANUP_FLAGS, USER_STATUS
.EXTRN CURRENT_VCB, PRIMARY_FCB
.EXTRN CURRENT_WINDOW, SECOND_FIB
.EXTRN SAVE_CONTEXT, RESTORE_CONTEXT
.EXTRN READ_HEADER, TURN_WINDOW
.EXTRN EXTEND, CHECKSUM
.EXTRN WRITE_HEADER, INIT_FCB
.EXTRN SYSSCMKRNL

.PSECT \$CODE\$,NOWRT,2

			00FC 00000	.ENTRY	EXTEND INDEX, Save R2,R3,R4,R5,R6,R7	: 0375
			9F 9E 00002	MOVAB	@#SYSSCMKRNL, R7	
0000G	CF		00 FB 00009	CALLS	#0, SAVE_CONTEXT	: 0442
	53	00000	CF 9E 0000E	MOVAB	SECOND_FIB, FIB	: 0443
04	A3	000100	8F D0 00013	MOVL	#65537, 4(FIB)	: 0444
	55	00000	DF D0 0001B	MOVL	@CURRENT_VCB, FCB	: 0447
0000G	CF		55 D0 00020	MOVL	FCB, PRIMARY_FCB	
	52	10	A5 D0 00025	MOVL	16(FCB), WINDOW	: 0448
0000G	CF		52 D0 00029	MOVL	WINDOW, CURRENT_WINDOW	
			55 DD 0002E	PUSHL	FCB	: 0455
			7E D4 00030	CLRL	-(SP)	
0000G	CF		02 FB 00032	CALLS	#2, READ_HEADER	
	56		50 D0 00037	MOVL	R0, HEADER	
			01 DD 0003A	PUSHL	#1	: 0456
			03 DD 0003C	PUSHL	#3	
		0044	8F BB 0003E	PUSHR	#*M<R2,R6>	
			04 DD 00042	PUSHL	#4	
			5E DD 00044	PUSHL	SP	
		0000G	CF 9F 00046	PUSHAB	TURN_WINDOW	
	67		07 FB 0004A	CALLS	#7, SYSSCMKRNL	
	50	08	A2 3C 0004D	MOVZWL	8(WINDOW), R0	: 0458
	50		30 C2 00051	SUBL2	#48, R0	
	50		06 C6 00054	DIVL2	#6, R0	
	54	16	A2 3C 00057	MOVZWL	22(WINDOW), FREE_POINTERS	
54	50		54 C3 0005B	SUBL3	FREE_POINTERS, R0, FREE_POINTERS	
	50	30	A2 3C 0005F	MOVZWL	48(WINDOW), R0	: 0459
	50	2C	A2 C0 00063	ADDL2	44(WINDOW), R0	
	03		50 D1 00067	CMPL	R0, #3	

: R

				10	1A	0006A	BGTRU	1\$	
				54	D6	0006C	INCL	FREE_POINTERS	0462
		52	36	A2	3C	0006E	MOVZWL	54(WINDOW), R2	0463
		52		50	C0	00072	ADDL2	R0, R2	
		03		52	D1	00075	CMPL	R2, #3	
				02	1A	00078	BGTRU	1\$	
				54	D6	0007A	INCL	FREE_POINTERS	0464
				54	D5	0007C	TSTL	FREE_POINTERS	0466
				03	14	0007E	BGTR	2\$	
		54		01	D0	00080	MOVL	#1, FREE_POINTERS	
		50	0000G	CF	D0	00083	MOVL	CURRENT_VCB, R0	0475
51	44	A0	04	AC	C3	00088	SUBL3	FILE_NUMBER, 68(R0), R1	
				51	D6	0008E	INCL	R1	
		52	3C	A0	3C	00090	MOVZWL	60(R0), R2	0476
50	40	A0		52	C7	00094	DIVL3	R2, 64(R0), R0	
		50		04	C6	00099	DIVL2	#4, R0	
		50		51	D1	0009C	CMPL	R1, R0	
				03	1B	0009F	BLEQU	3\$	
		51		50	D0	000A1	MOVL	R0, R1	
		50		51	D0	000A4	MOVL	R1, FILES_TO_GO	0475
		50		54	C6	000A7	DIVL2	FREE_POINTERS, R0	0478
		000003E8		50	D1	000AA	CMPL	R0, #1000	
				05	1B	000B1	BLEQU	4\$	
		50	03E8	8F	3C	000B3	MOVZWL	#1000, R0	
	18	A3		50	D0	000B8	MOVL	BLOCKS_NEEDED, 24(FIB)	0483
	16	A3	020B	8F	A8	000BC	BISW2	#523, 22(FIB)	0487
			0048	8F	BB	000C2	PUSHR	#M<R3,R6>	0489
		0000G	CF	02	FB	000C6	CALLS	#2, EXTEND	
				56	DD	000CB	PUSHL	HEADER	0494
		0000G	CF	01	FB	000CD	CALLS	#1, CHECKSUM	
		0000G	CF	00	FB	000D2	CALLS	#0, WRITE_HEADER	0495
			7E	55	7D	000D7	MOVQ	FCB, -(SPT)	0496
				02	DD	000DA	PUSHL	#2	
				5E	DD	000DC	PUSHL	SP	
			0000G	CF	9F	000DE	PUSHAB	INIT_FCB	
		0000G	67	05	FB	000E2	CALLS	#5, SYSSCMKRNL	
			CF	00	FB	000E5	CALLS	#0, RESTORE_CONTEXT	0498
				0000G	CF	D4	CLRL	USER_STATUS#4	0499
				04	00	000EE	RET		0501

: Routine Size: 239 bytes, Routine Base: \$CODE\$ + 0000

```

: 189      0502  1
: 190      0503  1 END
: 191      0504  0 ELUDOM

```

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	239	NOVEC,NOWRT, RD, EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
:_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	19	0	1000	00:01.8

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:EXTIDX/OBJ=OBJ\$:EXTIDX MSRCS:EXTIDX/UPDATE=(ENHS:EXTIDX)

: Size: 239 code + 0 data bytes
 : Run Time: 00:08.1
 : Elapsed Time: 00:22.6
 : Lines/CPU Min: 3742
 : Lexemes/CPU-Min: 14443
 : Memory Used: 113 pages
 : Compilation Complete

