

FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFFFFFFFFFFFFFFF	111	111	AAAAAAAAA	
FFF	111111	111111	AAA	AAA
FFF	111111	111111	AAA	AAA
FFF	111111	111111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFFFFFFFFFFFFF	111	111	AAA	AAA
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAAAAAAAAAAAAAAA	
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111	111	AAA	AAA
FFF	111111111	111111111	AAA	AAA
FFF	111111111	111111111	AAA	AAA
FFF	111111111	111111111	AAA	AAA

[illegible]

```

LL                      IIIIII                      SSSSSSSS
LL                      IIIIII                      SSSSSSSS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SSSSSS
LL                      II                      SSSSSS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SS
LL                      II                      SS
LLLLLLLLLLLLLLLL      IIIIII                      SSSSSSSS
LLLLLLLLLLLLLLLL      IIIIII                      SSSSSSSS

```

DIRF
V04-

.....


```
0001 0 MODULE DIRACC (
0002 0     LANGUAGE (BLISS32),
0003 0     IDENT = 'V04-000'
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1
0008 1 *****
0009 1 *
0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 * ALL RIGHTS RESERVED.
0013 1 *
0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 * TRANSFERRED.
0020 1 *
0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 * CORPORATION.
0024 1 *
0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 *****
0030 1
0031 1 ++
0032 1
0033 1 FACILITY: F11ACP Structure Level 1
0034 1
0035 1 ABSTRACT:
0036 1
0037 1     This routine "accesses" a directory and readies it for processing.
0038 1 ENVIRONMENT:
0039 1
0040 1     STARLET operating system, including privileged system services
0041 1     and internal exec routines.
0042 1
0043 1 --
0044 1
0045 1
0046 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 2-Jan-1977 22:53
0047 1
0048 1 MODIFIED BY:
0049 1
0050 1     V03-002 LMP51262      L. Mark Pilant,      13-Dec-1982 15:47
0051 1     Create non-contiguous directory file windows with a valid
0052 1     PID so they are not taken as shared windows.
0053 1
0054 1     V03-001 LMP0025      L. Mark Pilant,      17-May-1982 13:50
0055 1     Fix a bug that caused the system to crash if the users byte
0056 1     quota was exhausted while creating a directory file.
0057 1
```

; Ro

: 1
: 1
: 1

DIRACC
V04-000

B 3
16-Sep-1984 00:57:44
14-Sep-1984 12:29:29

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[F11A.SRC]DIRACC.B32;1 Page 2 (1)

```

: 58      0058 1 | V02-002 ACG0229      Andrew C. Goldstein,    22-Dec-1981  19:49
: 59      0059 1 |      Count directory LRU hits and misses
: 60      0060 1 |
: 61      0061 1 | V02-001 ACG33975      Andrew C. Goldstein,    13-Mar-1981  15:46
: 62      0062 1 |      Fix losing directory FCB's when LRU is disabled
: 63      0063 1 |
: 64      0064 1 | V02-000 ACG0167      Andrew C. Goldstein,    7-May-1980  18:49
: 65      0065 1 |      Previous revision history moved to F11A.REV
: 66      0066 1 |      **
: 67      0067 1 |
: 68      0068 1 |
: 69      0069 1 | LIBRARY 'SYSS$LIBRARY:LIB.L32';
: 70      0070 1 | REQUIRE 'SRC$:FCPDEF.B32';

```



```
72 0385 1 GLOBAL ROUTINE DIR_ACCESS (FIB, WRITE) : NOVALUE =
73 0386 1
74 0387 1 ++
75 0388 1
76 0389 1 FUNCTIONAL DESCRIPTION:
77 0390 1
78 0391 1 This routine "accesses" a directory and readies it for processing.
79 0392 1
80 0393 1 CALLING SEQUENCE:
81 0394 1 DIR_ACCESS (ARG1, ARG2)
82 0395 1
83 0396 1 INPUT PARAMETERS:
84 0397 1 ARG1: address of FIB
85 0398 1 ARG2: 0 if read only
86 0399 1 1 if read/write
87 0400 1
88 0401 1 IMPLICIT INPUTS:
89 0402 1 NONE
90 0403 1
91 0404 1 OUTPUT PARAMETERS:
92 0405 1 NONE
93 0406 1
94 0407 1 IMPLICIT OUTPUTS:
95 0408 1 DIR_FCB: address of FCB for directory
96 0409 1 DIR_WINDOW: address of window, if any
97 0410 1
98 0411 1 ROUTINE VALUE:
99 0412 1 NONE
100 0413 1
101 0414 1 SIDE EFFECTS:
102 0415 1 FCB and/or window may be created
103 0416 1 directory LRU may be altered
104 0417 1 directory file header may be read
105 0418 1
106 0419 1 --
107 0420 1
108 0421 2 BEGIN
109 0422 2
110 0423 2 MAP
111 0424 2 FIB : REF BBLOCK; ! FIB argument
112 0425 2
113 0426 2 LOCAL
114 0427 2 TEMP, : temp storage for file header address
115 0428 2 FCB : REF BBLOCK, ! FCB address
116 0429 2 HEADER : REF BBLOCK, ! directory header address
117 0430 2 MAP_AREA : REF BBLOCK; ! address of file header map area
118 0431 2
119 0432 2 EXTERNAL
120 0433 2 DIR_FCB : REF BBLOCK, ! global FCB address
121 0434 2 DIR_WINDOW : REF BBLOCK, ! global window address
122 0435 2 FILE_HEADER : REF BBLOCK, ! global file header address
123 0436 2 IO_PACKET : REF BBLOCK, ! I/O request packet
124 0437 2 PMS$GL_DIRHIT : ADDRESSING_MODE (GENERAL), ! count of directory LRU hits
125 0438 2 PMS$GL_DIRMISS : ADDRESSING_MODE (GENERAL); ! count of directory LRU misses
126 0439 2
127 0440 2
128 0441 2
```

```
129 0442 2 EXTERNAL ROUTINE
130 0443 2 SEARCH_FCB, ! search FCB list
131 0444 2 READ_HEADER, ! read file header
132 0445 2 CREATE_FCB, ! create an FCB
133 0446 2 CREATE_WINDOW, ! create a file window
134 0447 2 CHECK_PROTECT, ! check file protection
135 0448 2 MAKE_DIR_FCB; ! put directory FCB in LRU
136 0449
137 0450
138 0451 2 ! First find the FCB of the directory, if any exists.
139 0452 2 !
140 0453 2
141 0454 2 HEADER = 0; ! assume no header
142 0455 2 IF (FCB = .DIR_FCB) EQL 0
143 0456 2 THEN
144 0457 2 BEGIN
145 0458 2 FCB = SEARCH_FCB (FIB[FIB$W_DID]);
146 0459 2
147 0460 2 ! If there was no FCB, or it is an accessed file, read the header.
148 0461 2 ! If the FCB is in the LRU, check the sequence number since no
149 0462 2 ! one else does. If there was no FCB, create one.
150 0463 2 !
151 0464 2
152 0465 4 IF (IF .FCB EQL 0 THEN 1 ELSE (NOT .FCB[FCB$V_DIR]))
153 0466 4 THEN
154 0467 4 BEGIN
155 0468 4 PMSSGL_DIRMISS = .PMSSGL_DIRMISS + 1;
156 0469 4 TEMP = .FILE_HEADER;
157 0470 4 HEADER = READ_HEADER (FIB[FIB$W_DID], .FCB);
158 0471 4 FILE_HEADER = .TEMP; ! restore address for cleanup
159 0472 4 END
160 0473 3 ELSE
161 0474 4 BEGIN
162 0475 4 PMSSGL_DIRHIT = .PMSSGL_DIRHIT + 1;
163 0476 4 IF .FCB[FCB$W_FID_SEQ] NEQ .FIB[FIB$W_DID_SEQ]
164 0477 4 THEN ERR_EXIT (SS$_FILESEQCHK);
165 0478 4 END;
166 0479 3
167 0480 2 IF .FCB EQL 0
168 0481 2 THEN FCB = KERNEL_CALL (CREATE_FCB, .HEADER);
169 0482 2 DIR_FCB = .FCB; ! store global FCB address
170 0483 2 END;
171 0484 2
172 0485 2 ! Check directory file protection.
173 0486 2 !
174 0487 2
175 0488 2 CHECK_PROTECT (.WRITE, .HEADER, .FCB);
176 0489 2
177 0490 2 ! If we read a header, check the file attributes to make sure that
178 0491 2 ! this is a real directory file.
179 0492 2 !
180 0493 2
181 0494 2 IF .HEADER NEQ 0
182 0495 2 THEN
183 0496 2 BEGIN
184 0497 2 MAP_AREA = .HEADER + .HEADER[FH1$B_MPOFFSET]*2;
185 0498 2 IF .BBLOCK [HEADER[FH1$W_RECATTR], .FAT$B_RTYPE] NEQ FAT$C_FIXED
```



```
186 0499 3 OR .BBLOCK [HEADER[FM1$W_RECATTR], FATS$W_RSIZE] NEQ 16
187 0500 3 OR .MAP_AREA[FM1$B_EX_SEGNUM] NEQ 0
188 0501 3 OR .MAP_AREA[FM1$W_EX_FILNUM] NEQ 0
189 0502 3 OR .MAP_AREA[FM1$W_EX_FILSEQ] NEQ 0
190 0503 3 THEN ERR_EXIT (SS$_BADIRECTORY);
191 0504 3 END;
192 0505 3
193 0506 3 ! Arbitrate the access interlocks.
194 0507 3 !
195 0508 3
196 0509 3 IF .FCB[FCB$V_EXCL] ! check for exclusive access
197 0510 3 THEN ERR_EXIT (SS$_ACCONFLICT);
198 0511 3 IF .WRITE ! if write access desired
199 0512 3 AND .FCB[FCB$W_LCNT] NEQ 0 ! and file locked against writers
200 0513 3 THEN ERR_EXIT (SS$_ACCONFLICT);
201 0514 3
202 0515 3 ! If the file is not contiguous, construct a window.
203 0516 3 !
204 0517 3
205 0518 3 IF .FCB[FCB$L_STLBN] EQL 0
206 0519 3 AND .DIR_WINDOW EQL 0
207 0520 3 THEN
208 0521 3 BEGIN
209 0522 3 DIR_WINDOW = KERNEL_CALL (CREATE_WINDOW, 0, 0, .HEADER, .IO_PACKET[IRP$L_PID], .FCB);
210 0523 3 IF .DIR_WINDOW EQL 0 THEN ERR_EXIT (SS$_EXBYTLM);
211 0524 3 END;
212 0525 3
213 0526 3 ! Make the final diddles for the FCB to be useful for directory searches.
214 0527 3 !
215 0528 3
216 0529 3 KERNEL_CALL (MAKE_DIR_FCB, .FCB);
217 0530 3
218 0531 3 RETURN 1;
219 0532 3
220 0533 3 1 END; ! end of routine ACCESS_DIR
```

```
.TITLE DIRACC
.IDENT \V04-000\
```

```
.EXTRN DIR_FCB, DIR_WINDOW
.EXTRN FILE_HEADER, IO_PACKET
.EXTRN PM$SGL_DIRHIT, PM$SGL_DIRMISS
.EXTRN SEARCH_FCB, READ_HEADER
.EXTRN CREATE_FCB, CREATE_WINDOW
.EXTRN CHECK_PROTECT, MAKE_DIR_FCB
.EXTRN SY$SCMKNL
```

```
.PSECT $CODE$,NOWRT,2
```

```
56 00000000G 007C 00000
9F 9E 00002
54 D4 00009
53 0000G CF D0 0000B
5F 12 00010
52 04 AC D0 00012
OA A2 9F 00016
```

```
.ENTRY DIR_ACCESS, Save R2,R3,R4,R5,R6
MOVAB @SY$SCMKNL, R6
CLRL HEADER
MOVL DIR_FCB, FCB
BNEQ SS
MOVL FIB, R2
PUSHAB 10(R2)
```

```
: 0385
:
: 0454
: 0455
:
: 0458
:
```

0000G	CF	01	FB	00019	CALLS	#1, SEARCH_FCB	:	
	53	50	D0	0001E	MOVL	R0, FCB	:	
		04	13	00021	BEQL	1\$	:	0465
	1F	22	A3	E8	00023	BLBS	34(FCB), 2\$	
		00000000G	00	D6	00027	1\$: INCL	PMSSGL_DIRMISS	0468
	55	0000G	CF	D0	0002D	MOVL	FILE_HEADER, TEMP	0469
			53	DD	00032	PUSHL	FCB	0470
		0A	A2	9F	00034	PUSHAB	10(R2)	
0000G	CF		02	FB	00037	CALLS	#2, READ_HEADER	
	54		50	D0	0003C	MOVL	R0, HEADER	
0000G	CF		55	D0	0003F	MOVL	TEMP, FILE_HEADER	0471
			12	11	00044	BRB	3\$	0465
		00000000G	00	D6	00046	2\$: INCL	PMSSGL_DIRHIT	0475
	0C		A3	B1	0004C	CMPW	38(FCB), 12(R2)	0476
		26	05	13	00051	BEQL	3\$	
		08B8	8F	BF	00053	CHMU	#2232	0477
				04	00057	RET		
			53	D5	00058	3\$: TSTL	FCB	0480
			10	12	0005A	BNEQ	4\$	
			54	DD	0005C	PUSHL	HEADER	0481
			01	DD	0005E	PUSHL	#1	
			5E	DD	00060	PUSHL	SP	
		0000G	CF	9F	00062	PUSHAB	CREATE_FCB	
	66		04	FB	00066	CALLS	#4, SYS\$CMKRN	
	53		50	D0	00069	MOVL	R0, FCB	
0000G	CF		53	D0	0006C	4\$: MOVL	FCB, DIR_FCB	0482
			53	DD	00071	5\$: PUSHL	FCB	0488
			54	DD	00073	PUSHL	HEADER	
		08	AC	DD	00075	PUSHL	WRITE	
0000G	CF		03	FB	00078	CALLS	#3, CHECK_PROTECT	
			54	D5	0007D	TSTL	HEADER	0494
			2A	13	0007F	BEQL	7\$	
	50	01	A4	9A	00081	MOVZBL	1(HEADER), R0	0497
	50		64	40	3E	00085	MOVAB	(HEADER)[R0], MAP_AREA
	51	0E	A4	9E	00089	MOVAB	14(HEADER), R1	0498
	01		61	91	0008D	CMPB	(R1), #1	
			14	12	00090	BNEQ	6\$	
	10	02	A1	B1	00092	CMPW	2(R1), #16	0499
			0E	12	00096	BNEQ	6\$	
			60	95	00098	TSTB	(MAP_AREA)	0500
			0A	12	0009A	BNEQ	6\$	
		02	A0	B5	0009C	TSTW	2(MAP_AREA)	0501
			05	12	0009F	BNEQ	6\$	
		04	A0	B5	000A1	TSTW	4(MAP_AREA)	0502
			05	13	000A4	BEQL	7\$	
		0828	8F	BF	000A6	6\$: CHMU	#2088	0503
				04	000AA	RET		
09	22	A3	03	E0	000AB	7\$: BBS	#3, 34(FCB), 8\$	0509
		0A	AC	E9	000B0	BLBC	WRITE, 9\$	0511
			1E	A3	B5	000B4	TSTW	30(FCB)
			05	13	000B7	BEQL	9\$	0512
		0800	8F	BF	000B9	8\$: CHMU	#2048	0513
				04	000BD	RET		
		30	A3	D5	000BE	9\$: TSTL	48(FCB)	0518
			2B	12	000C1	BNEQ	10\$	
		0000G	CF	D5	000C3	TSTL	DIR_WINDOW	0519
			25	12	000C7	BNEQ	10\$	

50	0000G	53	DD	000C9	PUSHL	FCB	0522
	OC	CF	DD	000CB	MOVL	IO_PACKET, R0	
		A0	DD	000D0	PUSHL	12(R0)	
		54	DD	000D3	PUSHL	HEADER	
		7E	7C	000D5	CLRQ	-(SP)	
		05	DD	000D7	PUSHL	#5	
		5E	DD	000D9	PUSHL	SP	
	0000G	CF	9F	000DB	PUSHAB	CREATE_WINDOW	
0000G	66	08	FB	000DF	CALLS	#8, SYSSCMKRNL	
	CF	50	DD	000E2	MOVL	R0, DIR_WINDOW	
		05	12	000E7	BNEQ	10\$	0523
		2A14	8F	BF	CHMU	#10772	
			04	000ED	RET		
		53	DD	000EE	PUSHL	FCB	0529
		01	DD	000F0	PUSHL	#1	
		5E	DD	000F2	PUSHL	SP	
	0000G	CF	9F	000F4	PUSHAB	MAKE_DIR_FCB	
66		04	FB	000F8	CALLS	#4, SYSSCMKRNL	
			04	000FB	RET		0533

; Routine Size: 252 bytes, Routine Base: \$CODE\$ + 0000

:	221	0534	1
:	222	0535	1 END
:	223	0536	0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	252	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	21	0	1000	00:02.0

COMMAND QUALIFIERS

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

DIRACC
V04-000

16^H-3³-Sep-1984 00:57:44
14-Sep-1984 12:29:29

VAX-11 Bliss-32 V4.0-742 Page 8
DISK\$VMSMASTER:[F11A.SRC]DIRACC.B32;1 (2)

```

: Size:                252 code + 0 data bytes
: Run Time:            00:09.3
: Elapsed Time:        00:26.1
: Lines/CPU Min:       3473
: Lexemes/CPU-Min:    14274
: Memory Used:         130 pages
: Compilation Complete

```

DIRC
V04-

: Rd

1
1
1

•

⋮

0165 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

