HPFS Internals

(C) 1996, IBM Corporation All Rights Reserved

Agenda

- **Overview of High Performance File System (HPFS) features and disk layout**
  - **The Superblock and Spareblock structures**
- **Space allocation structures**
  - **The BITMAP structure**
- **Directory management structures**
  - **The DirBlock structure**
- **File/Directory information structures**
  - **The FNode structure**
- **Code Page Support**
  - **The CPInfo and CPData structures**
- **Tips and Tricks**
- **Sample Structures**

HPFS Features

- **64 GByte Partition size**
  - **(2 TByte design limit)**
- **512 byte allocation unit**
- **254 character file names**
  - **(259 character fully-qualified filespec)**
- **B-Tree directories and file allocation**
- **Multi-level cache**
  - **Complete paths**
  - **Sub-directories**
  - **Data**

Note: The information in this presentation is current with HPFS as implemented in OS/2 Warp Version 4.

(C) 1996, IBM Corporation All Rights Reserved

Terms

- **LSN: Logical Sector Number**
  - **4 bytes, top 5 bits used for volume number**
- **RSP: Redundant Sector Pointer**
  - **8 bytes, 2 LSN's**
- **SPtr: Storage Pointer**
  - **8 bytes: 4 byte count, LSN**
- **DATE: Time/Date field**
  - **4 bytes, # seconds since 1 Jan 1970**
  - **Use the C-library time() function**
  - **Min legal value is 12CEA600 (1 Jan 1980)**

NOTE: All numbers in this presentation are HEX
(Multi-byte fields in HPFS are little-endian)

HPFS in CONFIG.SYS

```
IFS=HPFS.IFS /CACHE:cc /CRECL:rr /AUTOCHECK:[+]d
```

The IFS statement loads the IFS driver (without it, you won't have support for this IFS.

- /CACHE specifies the size of the cache, in Kbytes. Minimum is 64K, maximum is 2048K. Default is 10% of FREE RAM below 16M.
- /CRECL specifies the read threshold in K bytes (range:4K-64K-1, default is 4K). Reads not bigger than this are copied to cache.
- /AUTOCHECK is the "autocheck" list. Drive letters specified here will be CHKDSK'ed on boot after improper shutdown (a "+" placed before a drive letter causes it to be CHKDSK'ed unconditionally at boot). "Dirty" drives will not mount. A "*" checks all drives (undocumented)

Undocumented CONFIG.SYS

```
IFS=HPFS.IFS /FORCE /QUIET /MAXIOW:n /F:n /N /L
```

These IFS switches are undocumented (and unsupported!).

- /FORCE allows access to drives that are dirty, but aren't in the AUTOCHECK list
- /QUIET suppresses the "drive dirty" error at boot time (useful if you use /FORCE)
- /MAXIOW:n specifies the maximum number of seconds to wait for Strategy-2 I/O completion. If it times out, a hard-error is generated.
- /F:n sets the chkdsk level (1 or 2, default is 2) for AutoCheck
- /N disables strategy-2 I/O (use with caution!)
- /L generates an IPE if an attempt is made to write low sectors (between the boot sector and the SuperBlock).

CACHE Statement

```
CACHE /LAZY:OFF|ON|n /READAHEAD:OFF|ON|n /DISKIDLE:dd
  /BUFFERIDLE:bb /MAXAGE:mm
```

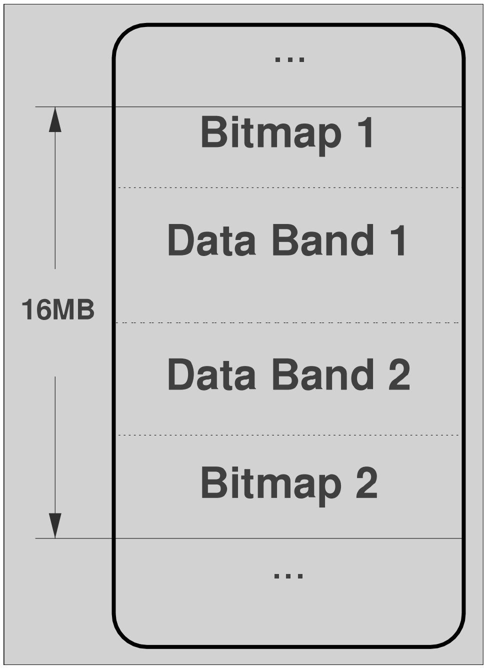
The CACHE statement sets HPFS cache parameters.

- /LAZY:OFF|ON turns "lazy write" off or on. "n" specifies the number of threads (default=3). No longer a need to detach or separate this option!
- /READAHEAD:OFF|ON turns "read ahead" off or on. "n" specifies the number of threads (default=1, which is currently the max)
- /BUFFERIDLE specifies time (in ms) since the last update before a dirty buffer is forced to disk
- /DISKIDLE specifies time (in ms) of inactivity before "idle" trigger starts writing dirty buffers
- /MAXAGE specifies time (in ms) since the last physical write before a dirty buffer is forced to disk

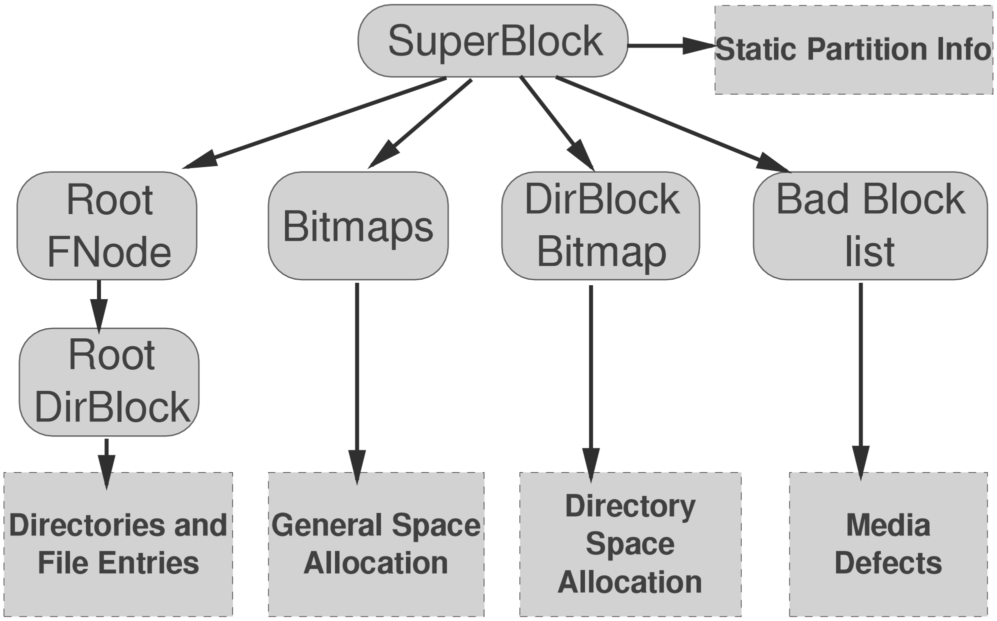
Note: DO NOT put this statement in CONFIG.SYS. Use STARTUP.CMD!

HPFS Layout

- **16MB allocation blocks**
- **2K bitmap+ 8M-2K data**
- **bands placed end-to-end**
- **Bitmaps can be anywhere**
- **Last band is <8M based on disk size**



SuperBlock Hierarchy



SuperBlock

Static Partition Info

Root FNode

Root DirBlock

Directories and File Entries

Bitmaps

General Space Allocation

DirBlock Bitmap

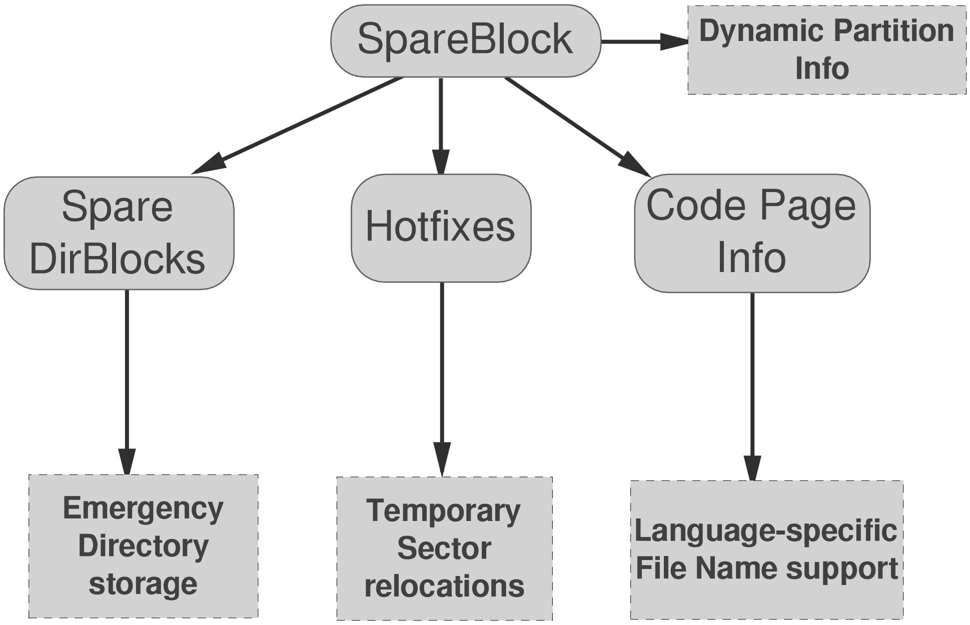
Directory Space Allocation

Bad Block list

Media Defects

(C) 1996, IBM Corporation All Rights Reserved

SpareBlock Hierarchy


SpareBlock
Dynamic Partition Info
Spare DirBlocks
Hotfixes
Code Page Info
Emergency Directory storage
Temporary Sector relocations
Language-specific File Name support

HPFS Boot Area

LSN	Size	Description
00	1 Sector	Standard Boot Sector
01	0F Sects	HPFS Boot code
10	1 Sector	Super Block
11	1 Sector	Spare Block

NOTE: These are the only sectors on an HPFS volume that can't be moved

(C) 1996, IBM Corporation All Rights Reserved

Super Block (LSN 10)

Offset	Size	Description
00	8 bytes	Signature (49 E8 95 F9 C5 E9 53 FA)
08	1 byte	HPFS Version [2]
09	1 byte	Fnct Ver [2=small, 3=(>2GB),4=386HPFS]
0A	2 bytes	unused
0C	LSN	Pointer to Root FNode
10	4 bytes	Total sectors in volume (round down 4)
14	4 bytes	Total bad sectors in volume
18	RSP	Pointer to Bitmap Indirect
20	RSP	Pointer to Bad Block list
28	Date	Date of last CHKDSK

Super Block (continued)

Offset	Size	Description
2C	Date	Date of last Disk Optimize
30	4 bytes	Sectors in DirBlock Band
34	LSN	Ptr to beginning of DirBlock Band
38	LSN	Pointer to end of DirBlock Band
3C	LSN	Pointer to DirBlock Band Bitmap
40	20 bytes	Volume Name (not used)
60	LSN	Ptr to User ID Table (8 sectors)

(C) 1996, IBM Corporation All Rights Reserved

Spare Block (LSN 11)

Offset	Size	Description
00	8 bytes	Signature (49 18 91 F9 C5 29 52 FA)
08	1 byte	Status Flag ‣ 01 = DIRTY ‣ 02 = Spare DIrblocks in use ‣ 04 = Hotfixes in use ‣ 08 = Bad Sector (corrupt disk) ‣ 10 = Bad Bitmap block detected ‣ 20 = Fast Format used ‣ 80 = Old HPFS used

Spare Block continued)

Offset	Size	Description
09	1 byte	386 HPFS Flag ‣ 01 = install DASD Limits ‣ 02 = Resych DASD Limit info ‣ 04 = DASD Limits operational ‣ 08 = Multimedia active ‣ 10 = DCE ACLs active ‣ 20 = DASD Limits dirty
0A	1 byte	MM Contiguity factor
0B	1 byte	unused
0C	LSN	Pointer to Hotfix list
10	4 bytes	Current Hotfixes in use

(C) 1996, IBM Corporation All Rights Reserved

Spare Block (continued)

Offset	Size	Description
14	4 bytes	Maximum Hotfixes available
18	4 bytes	Spare DirBlocks available for use
1C	4 bytes	Max number of spare DirBlocks
20	LSN	Pointer to CodePage Info Sector
24	4 bytes	Number of Code Pages
28	4 bytes	Super Block Checksum
2C	4 bytes	Spare Block Checksum
30	3C bytes	reserved
6C	65 LSNs	Pointers to spare Dir Blocks

Allocation Bitmaps

- **Each Bitmap is 4 sectors (2K bytes)**
  - **stored on a 4-sector boundary (for cache performance)**
  - **Each bit in the bitmap references a single sector**
  - **Each bitmap references 8M (2K bytes * 8 = 16K sectors)**
- **The DirBlock bitmap references 2K Dirblocks, not sectors.**
- **The "bitmap indirect" is a contiguous list of LSNs, which point to each bitmap.**
  - **The list is always a multiple of 4 sectors in length.**

To find the allocation bit for a sector, divide the LSN by 16K. The quotient is the band number, or the LSN offset into the bitmap indirect. Divide the remainder by 8 to get the position in the bitmap. The quotient is the byte offset, and the remainder is the bit (0=lsb, 7=msb)

(C) 1996, IBM Corporation All Rights Reserved

Hotfix Table

Offset	Size	Description
00	n LSNs	Bad sector numbers
4*n	n LSNs	Replacement sectors
8*n	n LSNs	FNodes referencing bad sectors

NOTE: The size of each LSN table (n) is defined in the SpareBlock. Current implementation sets this number at 0x64 or 1% of the partition size, whichever is less.

Bad Block List

Offset	Size	Description
00	LSN	Ptr to next BB list (0 if last block)
04	1FF LSNs	Bad sector numbers

NOTE: The Bad Block list records media defects so these sectors are never used. A full format initializes this list. A "quick" format relies on hotfix logic to develop this list.

(C) 1996, IBM Corporation All Rights Reserved

Directory Structure

- **Directories 'pre-allocated' near center of the disk**
- **Directory "Blocks" are 2K bytes long**
  - **5 for each MB of disk space (32h min) up to a maximum of 0FA0.**
  - **If Dir Band fills, additional Dir Blocks are allocated from data area.**
  - **Dir Band has its own bitmap (the normal bitmap is marked "in-use" at format time) The DirBlock Bitmap is 2K, only the first sector is used (other sectors are initialized to 0.**
- **"Spare" DirBlocks only used to complete a b-tree split operation.**

Directory Block

Offset	Size	Description
00	4 bytes	Signature (AE 0A E4 77)
04	4 bytes	Offset of first free byte
08	4 bytes	Change Count (01 bit set on top DirBlock)
0C	LSN	Ptr to Parent DirBlock (FNode if top)
10	LSN	Pointer to this DirBlock sector
14	7EC bytes	Directory Entries

(C) 1996, IBM Corporation All Rights Reserved

Directory Entry

Offset	Size	Description
00	2 bytes	Length of this entry (4-byte multiple)
02	1 byte	Flag ‣ 01 = Special ‣ 02 = ACLs present ‣ 04 =B-Tree down pointer ‣ 08 = END record ‣ 10 = EA list present ‣ 20 = Extended permissions list ‣ 40 = explicit ACL present ‣ 80 = "need" EAs present

Directory Entry (continued)

Offset	Size	Description
03	1 byte	FAT Attributes ‣ 01 = Read Only ‣ 02= Hidden ‣ 04 = System ‣ 08 = Volume Label ‣ 10 = Directory ‣ 20 = Archive
04	LSN	Pointer to FNode
08	Date	Last Modification Time
0C	4 bytes	File Size
10	Date	Last Access Time

(C) 1996, IBM Corporation All Rights Reserved

Directory Entry (continued)

Offset	Size	Description
14	Date	Creation Time
18	4 bytes	EA length
1C	1 byte	"Flex" area flag (bits 0-2=ACE count)
1D	1 byte	Code page index (>80=DBCS in use)
1E	1 byte	Length of file name
1F	<=FF bytes	File name (no trailing zero)
??	? bytes	Flex area
??	LSN	B-Tree Down Pointer (if FLAG & 04)

FNode

Offset	Size	Description
00	4 bytes	Signature (AE 0A E4 F7)
04	8 bytes	Reserved (read history)
0C	1 byte	DirBlock Name Length
0D	0F bytes	Name
1C	LSN	Pointer to Parent Directory FNode
20	SPtr	Length & pointer to external ACLs
28	2 bytes	length of ACL info in FNode
2A	1 byte	External ACL flag
2B	1 byte	reserved (history bit count)
2C	SPtr	Length & pointer to external EAs

(C) 1996, IBM Corporation All Rights Reserved

FNode (continued)

Offset	Size	Description
34	2 bytes	length of FNode EAs
36	1 byte	External EA flag
37	1 byte	FNode flag (01 = FNode is for Directory)
38	ALBLK	Allocation Block for file's data
40	60 bytes	Allocation Block Data
A0	4 bytes	File's valid data length
A4	4 bytes	number of required EAs
A8	10 bytes	reserved (user ID)
B8	2 bytes	Offset of internal EA/ACLs
BA	01 byte	% threshhold for 1st DASD limit alert

FNode (continued)

Offset	Size	Description
BB	1 byte	% delta for subsequent DASD limit alerts
BC	4 bytes	DASD limit (in sectors)
C0	4 bytes	current DASD usage (in sectors)
C4	13C bytes	EA/ACL storage

(C) 1996, IBM Corporation All Rights Reserved

Extended Attributes

Offset	Size	Description
00	1 byte	EA type: 0, 1 or 3
01	1 byte	length of EA name (not incl. NULL)
02	2 bytes	length of EA value (total)
04	?? bytes	EA name + trailing NULL
??	?? bytes	EA value or LSN + 4 byte length

NOTE: An EA type of 0 means the actual EA value follows the name. EA type 1 means the EA value is stored in another run of sectors, referenced by an SPtr following the name. Type 3 is similar to type 1, except the LSN points to an AlSec (because the EA data is fragmented).

Allocation Block (ALBLK)

Offset	Size	Description
00	1 byte	Flag: ‣ 01 = high bit of first free entry offset ‣ 20 = parent is an FNode ‣ 40 = suggest binary search (not used) ‣ 80 = ALBLK data contains NODEs
01	3 bytes	reserved
04	1 byte	number of free entries
05	1 byte	number of entries in use
06	2 bytes	offset of first free entry (from ALBLK)

(C) 1996, IBM Corporation All Rights Reserved

Allocation Block Data (ALNODE)

Offset	Size	Description
00	4 bytes	First sector offset of NEXT alnode
04	LSN	Pointer to child block

NOTE: Alnodes are used when there are too many fragments to represent the data in the AlBlk.

Allocation Block Data (ALLEAF)

Offset	Size	Description
00	4 bytes	Starting Sector Offset of data block
04	4 bytes	Length of block (in sectors)
08	LSN	pointer to data block

NOTE: Leaves are used to point at a contiguous block of data (a fragment, or an "extent").

(C) 1996, IBM Corporation All Rights Reserved

Allocation Sector (ALSEC)

Offset	Size	Description
00	4 bytes	Signature (AE 0A E4 37)
04	LSN	Pointer to this sector
08	LSN	Pointer to parent
0C	ALBLK	Allocation header
14	1E0 bytes	Allocation Data
1F4	0C bytes	unused

NOTE: ALSecs are not initialized before use, so they usually look full of "junk." Use the ALBlk header to validate the data.

Code Page Info Sector

Offset	Size	Description
00	4 bytes	Signature (F7 21 45 49)
04	4 bytes	Count of CP Info's in this sector
08	4 bytes	Index of first CPI in this sector
0C	LSN	Pointer to Next CP Info Sector
10	1F CPIs	Array of Codepage Info blocks

(C) 1996, IBM Corporation All Rights Reserved

Code Page Info Block

Offset	Size	Description
00	2 bytes	Country Code
02	2 bytes	Code page ID
04	4 bytes	Checksum of code page
08	LSN	Pointer to CP Data Sector
0C	2 bytes	Volume-specific code page ID
0E	2 bytes	Count of DBCS ranges in Code Page

Code Page Data Sector

Offset	Size	Description
00	4 bytes	Signature (F7 21 45 89)
04	2 bytes	Count of code pages in this sector
06	2 bytes	Index of first CPD in this Sector
08	4 bytes[3]	Codepage Data checksums
14	2 bytes[3]	Offsets of Codepage Data blocks
1A	3 CPDatas	Array of Codepage Data blocks

Code Page Data Block

Offset	Size	Description
00	2 bytes	Country Code
02	2 bytes	Code page ID
04	2 bytes	Count of DBCS Ranges
06	80 bytes	Case conversion table (80-FF)
86	2 bytes[]	Start/End DBCS range pairs

NOTE: There will always be at least 1 DBCS range pair, even if the count is 0.

Checksums in HPFS

To calculate a checksum for an HPFS object, use:

```
ULONG chksum (BYTE *object, USHORT size)
{
  register USHORT i;
  ULONG csum=0L;

  for (i=0; i < size; i++)
  {
    csum += (ULONG) *object++;
    csum = (csum << 7) + (csum >> (25));
  }
  return (csum);
}
```

(C) 1996, IBM Corporation All Rights Reserved

Using The Secret Password

DosRead/Write in direct-access mode will fail if the partition is >2GB. If you write an HPFS editor or other tool, you'll need to know the "secret password" that unlocks the big disks. After you use DosOpen to get a handle to the volume, use DosFSCtl function 0x9014 (FSC_SECTORIO). Route the request based on the handle (not the IFS). For the input paramter, put a pointer to the 4-byte string:

0xDEADFACE.

Doing so will put the handle in "SECTOR" mode. All offsets & sizes will refer to sectors instead of bytes, allowing you to address a partition up to 1TB in size.

(C) 1996, IBM Corporation All Rights Reserved