EXTENDED FILE SYSTEM 2

|  |  |  |  |
| --- | --- | --- | --- |
| BOOT BLOCK | BLOCK GROUP 0 | . . . | BLOCK GROUP N |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Super Block | Group Descriptors | Data Block Bitmap | I node Bitmap | I node Table | Data Blocks |

1 BLOCK N BLOCKS 1 BLOCK 1 BLOCK N BLOCKS N BLOCKS

BOOT BLOCK :- The 1st 1024 bytes of the disk, reserved for the partition boot sector and are

unused by the ext2 file system.

SUPER BLOCK :- To define the parameters of the file system & its overall state. It contains

the information such as the total number of blocks on disk, the size of a

block, the number of free blocks. etc.,

GROUP DESCRIPTORS :- This block immediately follows the super block & resides the list of

Block group descriptors. This list contains a descriptor for each

Block group on the disk.

The group descriptor tells us the location of the block/inode

Bitmaps & of the inode table.

BLOCK & INODES BITMAPS :- A bitmap is a sequence of bits. Each bit represents a specific

Block (block bitmap) or inode (inode bitmap) in the block

group. A bit value of 0 indicates that the block/inode is free,

while a value of 1 indicates that the block/inode is being used.

INODE TABLE :- The inode table consists of a series of consecutive blocks, each of which

contains a predefined number of inodes. The inode table contains

everything the OS needs to know about a file, including the type of file,

permissions, owner & location of the data blocks on disk.

In total there are 15 pointers in the i-block[] array.

* I-block [0 … 11] points directly to the 1st 12 data blocks of the file.
* I-block [12] points to a single indirect block.
* I-block [13] points to a double indirect block.
* I-block [14] points to a triple indirect block.

SUPER BLOCK STRUCTURE :-

size start end Superblock

4 1 4 Total number of inodes

4 5 8 File system size in blocks

4 9 12 Number of reserved blocks

4 13 16 Free blocks counter

4 17 20 Free inodes counter

4 21 24 Number of first useful block (always 1)

4 25 28 Block size

4 29 32 Fragment size

4 33 36 Number of blocks per group

4 37 40 Number of fragments per group

4 41 44 Number of inodes per group

4 45 48 Time of last mount operation

4 49 52 Time of last write operation

2 53 54 Mount operations counter

2 55 56 Number of mount operations before check

2 57 58 Magic signature

2 59 60 Status flag

2 61 62 Behaviour when detecting errors

2 63 64 Minor revision level

4 65 68 Time of last check

4 69 72 Time between checks

4 73 76 OS where file system was created

4 77 80 Revision level

2 81 82 Default user ID for reserved blocks

2 83 84 Default group ID for reserved blocks

4 85 88 Number of first non reserved inode

2 89 90 Size of on-disk inode structure

2 91 92 Block group number of this superblock

4 93 96 Compatible features bitmap

4 97 100 Incompatible features bitmap

4 101 104 Read-only-compatible features bitmap

16 105 120 128-bit file system identifier

16 121 136 Volume name

64 137 200 Path of last mount point

4 201 204 Used for compression

1 205 205 Number of blocks to pre allocate

1 206 206 Number of blocks to pre allocate for directories

818 207 1024 Nulls to pad out 1024 bytes

GROUP DESCRIPTOR STRUCTURE :-

size start end Group Descriptor

4 1 4 Block number of block bitmap

4 5 8 Block number of inode bitmap

4 9 12 Block number of first inode table block

2 13 14 Number of free blocks in the group

2 15 16 Number of free inodes in the group

2 17 18 Number of directories in the group

2 19 20 Alignment to word

4 21 24 Nulls to pad out 24 bytes

Finding a File :-

1. From the super block note down the value of size of inode structure (offset 89 to 90)

and Block size (offset 25 to 28).

1. From the Group Descriptor note down the value of Block number of 1st inode table block (offset 9 to 12).
2. From this we can calculate the address of 1st inode table.

Address of 1st inode table = Block number of 1st inode table block \* Block size.

1. Diskette offset for any inode in a particular inode table can be found by using the formula

Diskette offset = Address of 1st inode table + (n – 1) \* Inode size.

Where,

n -> inode’s number.

1. Go to the diskette offset address, which was calculated in the previous step and note down the value of pointer to the 1st data block (offset 41 to 44).

Data Block Address = Pointer to the 1st data block \* Block size.

The content of the data block looks like

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Size | Start | End |  | Directory Entry |
|  | | | | |
| 4 | 1 | 4 |  | Inode number |
| 2 | 5 | 6 |  | This directory entry's length |
| 1 | 7 | 7 |  | File name length |
| 1 | 8 | 8 |  | File type (1=regular file 2=directory) |
|  | 9 | ? |  | File name |

1. If we want to locate a file which is present in different inode table, note down the value of Block number of inode table from Group Descriptor and recalculate the address of inode table and hence the diskette offset.