/\*

\* Structure of the super block

\*/

struct ext2\_super\_block {

\_\_le32 s\_inodes\_count; /\* Inodes count \*/

\_\_le32 s\_blocks\_count; /\* Blocks count \*/

\_\_le32 s\_r\_blocks\_count; /\* Reserved blocks count \*/

\_\_le32 s\_free\_blocks\_count; /\* Free blocks count \*/

\_\_le32 s\_free\_inodes\_count; /\* Free inodes count \*/

\_\_le32 s\_first\_data\_block; /\* First Data Block \*/

\_\_le32 s\_log\_block\_size; /\* Block size \*/

\_\_le32 s\_log\_frag\_size; /\* Fragment size \*/

\_\_le32 s\_blocks\_per\_group; /\* # Blocks per group \*/

\_\_le32 s\_frags\_per\_group; /\* # Fragments per group \*/

\_\_le32 s\_inodes\_per\_group; /\* # Inodes per group \*/

\_\_le32 s\_mtime; /\* Mount time \*/

\_\_le32 s\_wtime; /\* Write time \*/

\_\_le16 s\_mnt\_count; /\* Mount count \*/

\_\_le16 s\_max\_mnt\_count; /\* Maximal mount count \*/

\_\_le16 s\_magic; /\* Magic signature \*/

\_\_le16 s\_state; /\* File system state \*/

\_\_le16 s\_errors; /\* Behaviour when detecting errors \*/

\_\_le16 s\_minor\_rev\_level; /\* minor revision level \*/

\_\_le32 s\_lastcheck; /\* time of last check \*/

\_\_le32 s\_checkinterval; /\* max. time between checks \*/

\_\_le32 s\_creator\_os; /\* OS \*/

\_\_le32 s\_rev\_level; /\* Revision level \*/

\_\_le16 s\_def\_resuid; /\* Default uid for reserved blocks \*/

\_\_le16 s\_def\_resgid; /\* Default gid for reserved blocks \*/

/\*

\* These fields are for EXT2\_DYNAMIC\_REV superblocks only. \*

\* Note: the difference between the compatible feature set and

\* the incompatible feature set is that if there is a bit set

\* in the incompatible feature set that the kernel doesn’t

\* know about, it should refuse to mount the filesystem.

\*

\* e2fsck’s requirements are more strict; if it doesn’t know

\* about a feature in either the compatible or incompatible

\* feature set, it must abort and not try to meddle with

\* things it doesn’t understand...

\*/

\_\_le32 s\_first\_ino; /\* First non-reserved inode \*/

\_\_le16 s\_inode\_size; /\* size of inode structure \*/

\_\_le16 s\_block\_group\_nr; /\* block group # of this superblock \*/

\_\_le32 s\_feature\_compat; /\* compatible feature set \*

\_\_le32 s\_feature\_incompat; /\* incompatible feature set \*/

\_\_le32 s\_feature\_ro\_compat; /\* readonly-compatible feature set \*/

\_\_u8 s\_uuid[16]; /\* 128-bit uuid for volume \*/

char s\_volume\_name[16]; /\* volume name \*/

char s\_last\_mounted[64]; /\* directory where last mounted \*/

\_\_le32 s\_algorithm\_usage\_bitmap; /\* For compression \*/

/\*

\* Performance hints. ...

\*/

\_\_u8 s\_prealloc\_blocks; /\* Nr of blocks to try to preallocate\*/

\_\_u8 s\_prealloc\_dir\_blocks; /\* Nr to preallocate for dirs \*/

\_\_u16 s\_padding1; /\* \* Journaling support valid if EXT3\_FEATURE\_COMPAT\_HAS\_JOURNAL set. \*/

\_\_u8 s\_journal\_uuid[16]; /\* uuid of journal superblock \*/

\_\_u32 s\_journal\_inum; /\* inode number of journal file \*/

\_\_u32 s\_journal\_dev; /\* device number of journal file \*/

\_\_u32 s\_last\_orphan; /\* start of list of inodes to delete \*/

\_\_u32 s\_hash\_seed[4]; /\* HTREE hash seed \*/

\_\_u8 s\_def\_hash\_version; /\* Default hash version to use \*/

\_\_u8 s\_reserved\_char\_pad;

\_\_u16 s\_reserved\_word\_pad;

\_\_le32 s\_default\_mount\_opts;

\_\_le32 s\_first\_meta\_bg; /\* First metablock block group \*/

\_\_u32 s\_reserved[190]; /\* Padding to the end of the block \*/

};