NAME

hd − winchester hard disk

DESCRIPTION

The **hd*** family of devices refer to the Winchester hard disk drivers for the IBM XT, AT and PS/2 machines, but may also refer to the generic (and slower) BIOS based hard disk driver. These disks are arrays of 512 byte sectors, although Minix always works with two sectors at a time due to its 1024 byte block size. You can read or write any number of bytes however, Minix takes care of cutting and pasting incomplete blocks together.

The devices may be divided into three classes:

The devices with a minor device number that is a multiple of 5, i.e. **hd0** or **hd5**, refer to the whole hard disk 0 and 1. Through these devices one has access to any block on the hard disk. Most notably the partition table, that can be found in the first sector of the disk.

The devices with a minor device number that is not a multiple of 5, i.e. **hd1**, **hd2**, ..., **hd6**, ..., refer to primary partitions of the lower numbered whole hard disk device. These devices normally contain MS-DOS or Minix file systems. /dev/hd1 is often the MS-DOS C: drive.

Minor devices from 128 up may refer to Minix subpartitions within primary partitions if a subpartition table has been placed in a Minix primary partition. The subpartitions of **hd3** for instance, are named **hd3a** through **hd3d**. Their minor device numbers may be calculated as 128 + 16*drive + 4*partition + subpartition, counting the partitions from zero.

If a primary partition is an extended partition then up to four logical partitions can be accessed as subpartitions of that extended partition. This allows one to access foreign file systems of other operating systems, Minix file systems are not normally placed in logical partitions.

PARTITIONING

The first sector of a drive (or partition for subpartitioning) contains the partition table at byte offset 446. This is what each of the four entries looks like as defined in <ibm/partition.h>:

```
/* Description of entry in the partition table.  */

struct part_entry {
   unsigned char bootind;          /* boot indicator 0/ACTIVE_FLAG     */
   unsigned char start_head;       /* head value for first sector      */
   unsigned char start_sec;        /* sector value + high 2 cyl bits   */
   unsigned char start_cyl;        /* low 8 cylinder bits              */
   unsigned char sysind;           /* system indicator                 */
   unsigned char last_head;        /* h/s/c for the last sector        */
   unsigned char last_sec;
   unsigned char last_cyl;
   unsigned long lowsec;           /* logical first sector             */
   unsigned long size;             /* size of partition in sectors     */
};

#define ACTIVE_FLAG     0x80   /* value for active in bootind field    */
#define NR_PARTITIONS   4      /* number of entries in table           */
#define PART_TABLE_OFF 0x1BE   /* offset of table in boot sector       */

/* Partition types (sysind). */
#define MINIX_PART      0x81   /* Minix partition type */
#define NO_PART         0x00   /* unused entry */
#define OLD_MINIX_PART 0x80    /* created before 1.4b, obsolete */
#define EXT_PART        0x05   /* extended partition */
```

The cylinder numbers are encoded in a very strange way, bits 8 and 9 are in the high two bits of the sector number. The sector numbers count from 1, not 0! More useful are the lowsec and size fields however, they simply give the location of the partition as an absolute sector offset and length within the drive.

The partition table entry defined above is specific to IBM type disks. The device drivers use another partition entry structure to pass information on a partition. This is what <minix/partition.h> looks like:

```
struct partition {
   u64_t base;                /* byte offset to the partition start */
   u64_t size;                /* number of bytes in the partition */
   unsigned cylinders;        /* disk geometry for partitioning */
   unsigned heads;
   unsigned sectors;
};
```

The base and size fields are the byte offset and length of a partition. (These are 64 bit numbers under Minix-vmd, but only 32 bit numbers under standard Minix.) The geometry of the disk is also given for the benefit of partition table editors. This information can be obtained from an open disk device with the call:

ioctl(*fd***, DIOCGETP, &***entry***);**

One can change the placement of the device to the lowsec and size fields of *entry* by using the **DIOCSETP** call instead. Only the base and size fields are used for **DIOCSETP**.

The partition tables when read from disk by the driver are checked and truncated to fit within the primary partition or drive. The first sector should be left free for the partition table.

The partition tables are read when the in-use count (opens and mounts) changes from 0 to 1. So an idle disk is automatically repartitioned on the next access. This means that repartitioning programs only have effect if a disk stays in use, unless they reload a changed partition table.

FILES

/dev/hd[0−9], /dev/hd[1−46−9][a−d]

SEE ALSO

ioctl(2), **int64**(3), **part**(8), **repartition**(8).

BUGS

The subpartitioning is incompatible with the MS-DOS method of extended partitions. The latter does not map well to the sparse minor device number space.

The primary partition table is sorted by lowsec like MS-DOS does, subpartition tables are not. Just think about what happens when you delete a partition in the MS-DOS scheme.

Don’t move a partition that is mounted or kept open by some process. The file system may write cached blocks to the new location.

The BIOS driver is not slow at all on a buffered disk.

Some IDE disks send an interrupt when they spin down under hardware power management. The driver acknowledges the interrupt as it is supposed to do by reading the status register. The disk then spins up again... You have to disable the spin down in the computer setup to fix the problem.

AUTHOR

Kees J. Bot (kjb@cs.vu.nl)