NAME

part − partition table editor

SYNOPSIS

part [*device*] ...

DESCRIPTION

Part is a screen oriented partition table editor.

While editing you will see six lines of numbers, the first line shows the device name and its geometry (number of cylinders, heads and sectors), the second shows the start and end of the drive or partition you are working on, the last four lines show the different partitions or subpartitions. All numbers except those on the second line can be edited. Question marks are showed instead of numbers if the partition table is not loaded yet. You have to select a device and type 'r'.

Editing is a simple matter of moving around with the arrow keys and changing the values with + and − (or PgUp and PgDn), or by typing the desired value. The '?' key will give a small list of commands, the '!' key gives advice on how to make a new entry.

The spacebar toggles between showing the size of the partition and the last sector on the partition. Useful to check if a partition is adjacent to the next.

The 'm' key is "magical", it lets you cycle through a set of interesting values for the base or size of a partition. These values are: Aligned to a cylinder, taped to other partitions (inside or outside), or filling out holes. **Use this key**!

Minix subpartition tables or extended partitions may be edited after hitting the '>' key. The number of this partition will be shown after the device name on the second row, e.g. **/dev/hd0:2**. Minix subpartition tables are shown as is, but extended partition bases are translated to absolute offsets on the screen to hide the gory details of their implementation from the innocent user. (Hit 'p' if you dare.) The '<' key will bring you back to the enclosing partition table.

With arguments, **part** will use the given devices or files. Without arguments, **part** will use all interesting block devices in **/dev** sorted by device number and starting with **/dev/hd0**.

Values that are out of range, overlapping, or otherwise strange are shown in reverse video. Values that may possibly be a problem for operating systems other then Minix are shown in bold characters.

The name of the device is highlighted when it has not been read yet.

Head or sector numbers are highlighted if the partition does not start or end at a cylinder boundary.

The base and/or size field is highlighted if they fall outside the device, if they are inside some other partition, if the base equals the device's base (no room for the boot sector), or if the size is zero.

Part complies with the good old UNIX tradition of trusting the user. It will write any table, no matter how bad. You have been warned.

By the way, as far as Minix is concerned there is absolutely no reason to make partitions start precisely on a cylinder or track nor does it have to be an exact number of cylinders long. Minix only looks at the base and size of a partition, the geometry of the drive doesn't have to be correct. Other Operating systems can be very picky about partitions that are not aligned. Some partition editors may refuse to edit a table, others may even make a mess of the table. The only exception is the first partition, it traditionally starts on the first track, not the first cylinder. All editors must understand this. (Subpartition tables are Minix specific, so there is no reason at all for any alignment.)

Extended Partitions

Extended partitions are a mess that is only made slightly better by **part** by translating the base offsets to absolute numbers. It is better to use DOS **fdisk** to create them, but if you insist on using **part** then this is what they should look like:

> The extended partition entry in the primary partition table must cover the whole logical partition space within it.

The area thus created is split in segments, each segment contains a partition table in sector 0 and one (just one) logical partition.

The first entry of a segment's partition table describes this logical partition: it's partition ID, base and size.

The second entry is an extended partition that describes base and size of the next segment (partition table and logical partition). The last segment's partition table is empty, or contains one logical partition.

SEE ALSO

mkfs(1), **fd**(4), **hd**(4).

BUGS

You can have a table read, messed up, and written in no time, be careful.

You can't type head or sector numbers directly.

Sectors are counted from 0 for consistency, but the partition table counts from 1 like DOS addresses them. Most confusing.

You can't write a backup copy to a file, that's what **dd**(1) with count=1 is for.

AUTHOR

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