

## NAME

**fdisk** – manipulate disk partition table

## SYNOPSIS

**fdisk** [options] *device*

**fdisk -l** [*device...*]

## DESCRIPTION

**fdisk** is a dialog-driven program for creation and manipulation of partition tables. It understands GPT, MBR, Sun, SGI and BSD partition tables.

Block devices can be divided into one or more logical disks called *partitions*. This division is recorded in the *partition table*, usually found in sector 0 of the disk. (In the BSD world one talks about ‘disk slices’ and a ‘disklabel’.)

All partitioning is driven by device I/O limits (the topology) by default. **fdisk** is able to optimize the disk layout for a 4K-sector size and use an alignment offset on modern devices for MBR and GPT. It is always a good idea to follow **fdisk**’s defaults as the default values (e.g. first and last partition sectors) and partition sizes specified by the +<size>{M,G,...} notation are always aligned according to the device properties.

Note that **partx**(8) provides a rich interface for scripts to print disk layouts, **fdisk** is mostly designed for humans. Backward compatibility in the output of **fdisk** is not guaranteed. The input (the commands) should always be backward compatible.

## OPTIONS

**-b, --sector-size** *sectorsize*

Specify the sector size of the disk. Valid values are 512, 1024, 2048, and 4096. (Recent kernels know the sector size. Use this option only on old kernels or to override the kernel’s ideas.) Since util-linux-2.17, **fdisk** differentiates between logical and physical sector size. This option changes both sector sizes to *sectorsize*.

**-B, --protect-boot**

Don’t erase the begin of the first disk sector when create a new disk label. This feature is supported for GPT and MBR.

**-c, --compatibility**[=*mode*]

Specify the compatibility mode, ‘dos’ or ‘nondos’. The default is non-DOS mode. For backward compatibility, it is possible to use the option without the *mode* argument -- then the default is used. Note that the optional *mode* argument cannot be separated from the **-c** option by a space, the correct form is for example ‘-c=dos’.

**-h, --help**

Display a help text and exit.

**-L, --color**[=*when*]

Colorize the output. The optional argument *when* can be **auto**, **never** or **always**. If the *when* argument is omitted, it defaults to **auto**. The colors can be disabled; for the current built-in default see the **--help** output. See also the **COLORS** section.

**-l, --list**

List the partition tables for the specified devices and then exit. If no devices are given, those mentioned in */proc/partitions* (if that file exists) are used.

**-o, --output** *list*

Specify which output columns to print. Use **--help** to get a list of all supported columns.

The default list of columns may be extended if *list* is specified in the format *+list* (e.g. **-o +UUID**).

**-s, --getsz**

Print the size in 512-byte sectors of each given block device. This option is DEPRECATED in favour of **blockdev(1)**.

**-t, --type *type***

Enable support only for disklabels of the specified *type*, and disable support for all other types.

**-u, --units[=*unit*]**

When listing partition tables, show sizes in 'sectors' or in 'cylinders'. The default is to show sizes in sectors. For backward compatibility, it is possible to use the option without the *unit* argument -- then the default is used. Note that the optional *unit* argument cannot be separated from the **-u** option by a space, the correct form is for example '**-u=cylinders**'.

**-C, --cylinders *number***

Specify the number of cylinders of the disk. I have no idea why anybody would want to do so.

**-H, --heads *number***

Specify the number of heads of the disk. (Not the physical number, of course, but the number used for partition tables.) Reasonable values are 255 and 16.

**-S, --sectors *number***

Specify the number of sectors per track of the disk. (Not the physical number, of course, but the number used for partition tables.) A reasonable value is 63.

**-w, --wipe *when***

Wipe filesystem, RAID and partition-table signatures from the device, in order to avoid possible collisions. The argument *when* can be **auto**, **never** or **always**. When this option is not given, the default is **auto**, in which case signatures are wiped only when in interactive mode. In all cases detected signatures are reported by warning messages before a new partition table is created. See also **wipefs(8)** command.

**-W, --wipe-partition *when***

Wipe filesystem, RAID and partition-table signatures from a newly created partitions, in order to avoid possible collisions. The argument *when* can be **auto**, **never** or **always**. When this option is not given, the default is **auto**, in which case signatures are wiped only when in interactive mode and after confirmation by user. In all cases detected signatures are reported by warning messages before a new partition is created. See also **wipefs(8)** command.

**-V, --version**

Display version information and exit.

## DEVICES

The *device* is usually */dev/sda*, */dev/sdb* or so. A device name refers to the entire disk. Old systems without libata (a library used inside the Linux kernel to support ATA host controllers and devices) make a difference between IDE and SCSI disks. In such cases the device name will be */dev/hd\** (IDE) or */dev/sd\** (SCSI).

The *partition* is a device name followed by a partition number. For example, */dev/sda1* is the first partition on the first hard disk in the system. See also Linux kernel documentation (the *Documentation/devices.txt* file).

## SIZES

The "last sector" dialog accepts partition size specified by number of sectors or by +<size>{K,B,M,G,...} notation.

If the size is prefixed by '+' then it is interpreted as relative to the partition first sector. In this case the size is expected in bytes and the number may be followed by the multiplicative suffixes KiB=1024, MiB=1024\*1024, and so on for GiB, TiB, PiB, EiB, ZiB and YiB. The "iB" is optional, e.g. "K" has the same meaning as "KiB".

The relative sizes are always aligned according to device I/O limits. The +<size>{K,B,M,G,...} notation is recommended.

For backward compatibility fdisk also accepts the suffixes KB=1000, MB=1000\*1000, and so on for GB, TB, PB, EB, ZB and YB. These 10<sup>N</sup> suffixes are deprecated.

## SCRIPT FILES

**fdisk** allows to read (by 'I' command) sfdisk compatible script files. The script is applied to in-memory partition table, and then it is possible to modify the partition table before you write it to the device.

And vice-versa it is possible to write the current in-memory disk layout to the script file by command 'O'.

The script files are compatible between cfdisk, sfdisk, fdisk and another libfdisk applications. For more details see **sfdisk(8)**.

## DISK LABELS

### GPT (GUID Partition Table)

GPT is modern standard for the layout of the partition table. GPT uses 64-bit logical block addresses, checksums, UUIDs and names for partitions and an unlimited number of partitions (although the number of partitions is usually restricted to 128 in many partitioning tools).

Note that the first sector is still reserved for a **protective MBR** in the GPT specification. It prevents MBR-only partitioning tools from mis-recognizing and overwriting GPT disks.

GPT is always a better choice than MBR, especially on modern hardware with a UEFI boot loader.

### DOS-type (MBR)

A DOS-type partition table can describe an unlimited number of partitions. In sector 0 there is room for the description of 4 partitions (called 'primary'). One of these may be an extended partition; this is a box holding logical partitions, with descriptors found in a linked list of sectors, each preceding the corresponding logical partitions. The four primary partitions, present or not, get numbers 1-4. Logical partitions are numbered starting from 5.

In a DOS-type partition table the starting offset and the size of each partition is stored in two ways: as an absolute number of sectors (given in 32 bits), and as a **Cylinders/Heads/Sectors** triple (given in 10+8+6 bits). The former is OK -- with 512-byte sectors this will work up to 2 TB. The latter has two problems. First, these C/H/S fields can be filled only when the number of heads and the number of sectors per track are known. And second, even if we know what these numbers should be, the 24 bits that are available do not suffice. DOS uses C/H/S only, Windows uses both, Linux never uses C/H/S. The **C/H/S addressing is deprecated** and may be unsupported in some later fdisk version.

**Please, read the DOS-mode section if you want DOS-compatible partitions.** **fdisk** does not care about cylinder boundaries by default.

### BSD/Sun-type

A BSD/Sun disklabel can describe 8 partitions, the third of which should be a ‘whole disk’ partition. Do not start a partition that actually uses its first sector (like a swap partition) at cylinder 0, since that will destroy the disklabel. Note that a **BSD label** is usually nested within a DOS partition.

### IRIX/SGI-type

An IRIX/SGI disklabel can describe 16 partitions, the eleventh of which should be an entire ‘volume’ partition, while the ninth should be labeled ‘volume header’. The volume header will also cover the partition table, i.e., it starts at block zero and extends by default over five cylinders. The remaining space in the volume header may be used by header directory entries. No partitions may overlap with the volume header. Also do not change its type or make some filesystem on it, since you will lose the partition table. Use this type of label only when working with Linux on IRIX/SGI machines or IRIX/SGI disks under Linux.

A `sync()` and an `ioctl(BLKRRPART)` (rereading the partition table from disk) are performed before exiting when the partition table has been updated.

### DOS mode and DOS 6.x WARNING

**Note that all this is deprecated. You don’t have to care about things like geometry and cylinders on modern operating systems. If you really want DOS-compatible partitioning then you have to enable DOS mode and cylinder units by using the ‘-c=dos -u=cylinders’ fdisk command-line options.**

The DOS 6.x `FORMAT` command looks for some information in the first sector of the data area of the partition, and treats this information as more reliable than the information in the partition table. `DOS FORMAT` expects `DOS FDISK` to clear the first 512 bytes of the data area of a partition whenever a size change occurs. `DOS FORMAT` will look at this extra information even if the `/U` flag is given -- we consider this a bug in `DOS FORMAT` and `DOS FDISK`.

The bottom line is that if you use **fdisk** or **cdisk** to change the size of a DOS partition table entry, then you must also use **dd(1)** to **zero the first 512 bytes** of that partition before using `DOS FORMAT` to format the partition. For example, if you were using **fdisk** to make a DOS partition table entry for `/dev/sda1`, then (after exiting **fdisk** and rebooting Linux so that the partition table information is valid) you would use the command `"dd if=/dev/zero of=/dev/sda1 bs=512 count=1"` to zero the first 512 bytes of the partition.

**fdisk** usually obtains the disk geometry automatically. This is not necessarily the physical disk geometry (indeed, modern disks do not really have anything like a physical geometry, certainly not something that can be described in the simplistic Cylinders/Heads/Sectors form), but it is the disk geometry that MS-DOS uses for the partition table.

Usually all goes well by default, and there are no problems if Linux is the only system on the disk. However, if the disk has to be shared with other operating systems, it is often a good idea to let an `fdisk` from another operating system make at least one partition. When Linux boots it looks at the partition table, and tries to deduce what (fake) geometry is required for good cooperation with other systems.

Whenever a partition table is printed out in DOS mode, a consistency check is performed on the partition table entries. This check verifies that the physical and logical start and end points are identical, and that each partition starts and ends on a cylinder boundary (except for the first partition).

Some versions of MS-DOS create a first partition which does not begin on a cylinder boundary, but on sector 2 of the first cylinder. Partitions beginning in cylinder 1 cannot begin on a cylinder boundary, but this is unlikely to cause difficulty unless you have OS/2 on your machine.

For best results, you should always use an OS-specific partition table program. For example, you should make DOS partitions with the `DOS FDISK` program and Linux partitions with the Linux `fdisk` or Linux

cfdisk programs.

## COLORS

Implicit coloring can be disabled by an empty file */etc/terminal-colors.d/fdisk.disable*.

See **terminal-colors.d(5)** for more details about colorization configuration. The logical color names supported by **fdisk** are:

**header** The header of the output tables.

**help-title**

The help section titles.

**warn** The warning messages.

**welcome**

The welcome message.

## AUTHORS

Karel Zak [kzak@redhat.com](mailto:kzak@redhat.com)

Davidlohr Bueso [dave@gnu.org](mailto:dave@gnu.org)

The original version was written by Andries E. Brouwer, A. V. Le Blanc and others.

## ENVIRONMENT

**FDISK\_DEBUG=all**

enables fdisk debug output.

**LIBFDISK\_DEBUG=all**

enables libfdisk debug output.

**LIBBLKID\_DEBUG=all**

enables libblkid debug output.

**LIBSMARTCOLS\_DEBUG=all**

enables libsmartcols debug output.

**LIBSMARTCOLS\_DEBUG\_PADDING=on**

use visible padding characters. Requires enabled **LIBSMARTCOLS\_DEBUG**.

## SEE ALSO

**cdisk(8)**, **mkfs(8)**, **partx(8)**, **sfdisk(8)**

## AVAILABILITY

The **fdisk** command is part of the **util-linux** package and is available from <https://www.kernel.org/pub/linux/utils/util-linux/>.