

Partitioning Primer

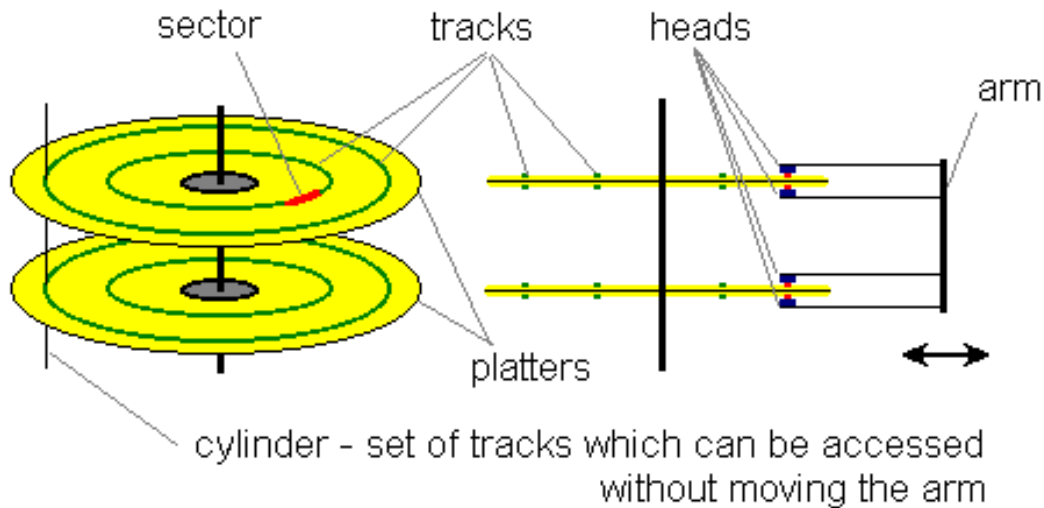
Introduction to Hard Disk Partitioning Basics

Mikhail Ranish



What is inside hard disk?

First disks had a simple design. They had one or more rotating platters and a moving arm with read/write heads attached to it - one head on each side of the platter. The arm could move and stop at the certain number of positions. When it stopped each head could read or write data on the underlying track. Every read or write had to be done in blocks of bytes, called sectors. Sectors were usually 512 bytes long and there were fixed number of sectors on each track.



The drives themselves didn't have much electronics and had to be controlled by CPU for every single step. First CPU had to issue command to position the arm. Then it had to instruct the drive which head should perform read and from which sector. After that CPU waited until the desired sector was moving under the head and started data transfer. This design was relatively simple and inexpensive. But there were several disadvantages.

First of all, each Input/Output operation involved a lot of CPU attention and, also, the disk surface was used inefficiently. It was convenient for the programmers to have fixed number of sectors on each track, but it was a waste of space, because the longer outer tracks could hold much more data than the shorter inner ones. Later, when digital electronics became cheap, hardware engineers could resolve this problem.

When IDE (Integrated Drive Electronics) disks came out they had a little processor on each drive. This helped to free up CPU time by implementing more sophisticated set of commands. The disk space

Partitioning Primer

Introduction to Hard Disk Partitioning Basics

Mikhail Ranish

was also used more efficiently. Engineers had placed more sectors on the outer tracks, but still provided software writers with a convenient "cubical" look of the disk by doing internal translation of CHS (cylinders, heads, sectors). For example my old 340M disk has only two platters = 4 heads (sides), but it reports 665 cylinders, 16 heads (sides), and 63 sectors. In reality it, probably, has more than 4*63 sectors on each outer track and a little less than 4*63 on the most inner tracks, but we could never know for sure.

With the IDE disks CPU only has to tell the CHS of the sector that it wants to read and drive's electronics will position the heads and call back the CPU when it is ready to start data transfer.

The newest drives have even simpler interface. Instead of addressing sectors by their CHS (cylinder, head, sector) address they use LBA (Logical Block Addressing) mode. In LBA mode a program has to tell only number of the sector from the beginning of the disk (all sectors on disk are numbered 0, 1, 2, 3, ...). In addition to that new disks have internal buffers, where they can store many sectors. This can speed up disk access a lot, because they could read data in buffer using all four heads at the same time.

Virtually all modern operating systems use LBA addressing, but the CHS notation is still around. First of all, MS-DOS, which is almost 20 years old, uses only CHS. Also some programs, like Partition Magic, would not work if partitions do not start at the cylinder or side boundary. And finally, it is easier to talk about hundreds of cylinders than about millions of sectors. Therefore we will be using CHS notation throughout this discussion.

There are several things that you have to know about CHS addressing. Suppose that we have a 340M disk with 665 cylinders, 16 heads, and 63 sectors per track. Then the legal values for cylinder numbers are 0..664, for head (side) numbers are 0..15, and for sector - 1..63.

The maximum allowable values for CHS addressing mode are 0..1023, 0..255, 1..63 for cylinders, heads, and sectors respectively. If you multiply these values you will see that the largest hard disk that could be addressed with CHS is 8G. Therefore, if your disk is 12G many programs will see only 8G, because they use CHS.

How disks are partitioned?

All hard disks on all IBM compatible computers have the same way of partitioning. First sector of the disk, called MBR (Master Boot Record), which we will be discussing in details later, contains partition table. This table has four records, each of them can describe one partition. In the simplest case we would have all disk space assigned to one partition, like in the following example:

Partitioning Primer

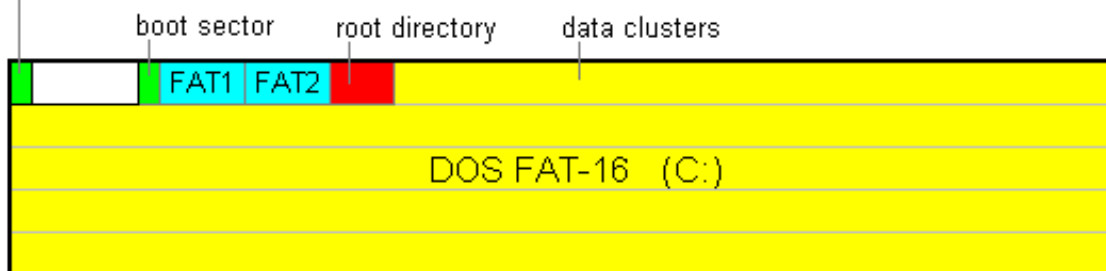
Introduction to Hard Disk Partitioning Basics

Mikhail Ranish

Example 1 Hard disk 340M [665 cyl x 16 heads x 63 sects]

MBR

#	Partition Type	Starting			Ending			Size[K]
		Cyl	Side	Sect	Cyl	Side	Sect	
1	DOS FAT-16	0	1	1	664	15	63	335,128
2	Unused	0	0	0	0	0	0	
3	Unused	0	0	0	0	0	0	
4	Unused	0	0	0	0	0	0	



Note that MBR occupies one sector at cylinder 0, side 0, sector 1 and partition starts on the cylinder 0, side 1, sector 1. The 62 sector gap between them was left unused, because we want all partitions to start at the cylinder boundary or, at least, on the side boundary. This is not required with LBA, but we need to follow this rule in order to make happy old software (MS-DOS for example).

Another important point that I want to make is that Operating System and File System are different things, which many people use interchangeably. Operating System is a piece of software which controls CPU and lets different application programs to run on the computer and use different system resources. The File System is a way to organize files and directories on the hard disk. The confusion comes because every decent operating system has one or more of its own file systems and they become clously associated.

In our example all we know is that we have FAT-16 file system. And we have no idea which operating system is installed on it. It could be MS-DOS 6.22, it could be Windows 95 or NT, or it could be all three of them installed in the different directories in the same partition. If we put additional efforts we could even install there Linux, but it is usually better to have different operating systems installed in separate partitions.

Another reason to have multiple partitions is the security against computer crashes. For example, if the system crashes in such a way that FAT tables get corrupted you will lose access to all your files, because FAT table tells where each file is located on the disk. The FAT table is so important that they decided to keep two copies of FAT table at the beginning of the disk. But in case if you have very valuable files, it, might be wise to create second partition (then it will have its own FATs) and keep copies of important files there.

However, do not rush to create the second partition. First of all, experience shows that 99% of errors damage only one copy of FAT and, secondly, for the majority of users it would be sufficient to copy their personal work to the floppy once a week and keep it in the safe place. So, in the case of a crash, they would only have to reformat the disk and reinstall all programs. Now, regardless of the reason, let's see what happens if we have multiple partitions.

Partitioning Primer

Introduction to Hard Disk Partitioning Basics

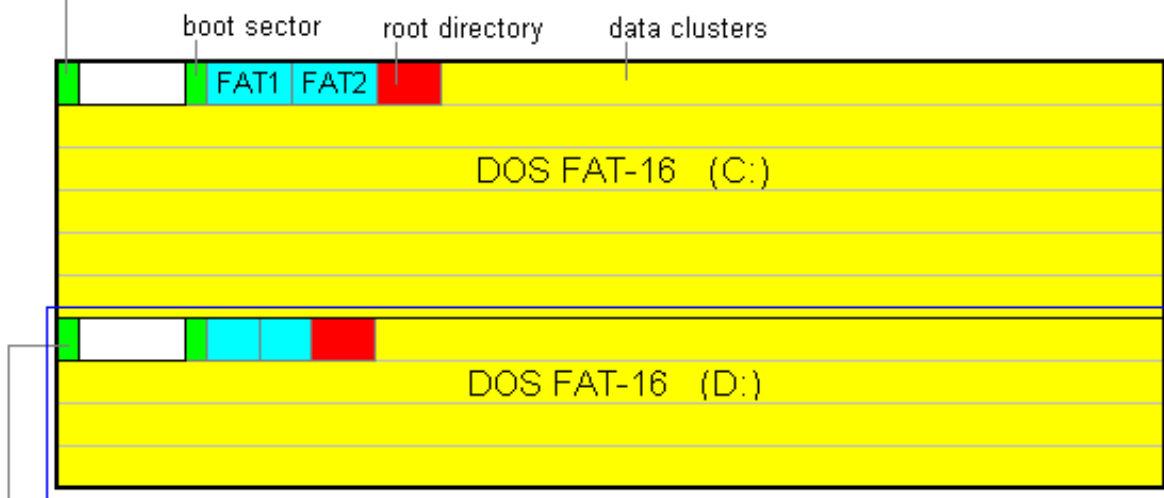
Mikhail Ranish

In the second example we have two partitions with the FAT-16 file system. For some reason makers of DOS decided that if you want to have second or third FAT partition you have to put them not in MBR but into the Extended DOS partition. Extended partition appears like an ordinary partition in MBR (it occupies space) and inside it has a table similar to partition table in MBR, called EMBR (Extended MBR), which lists partitions enclosed in the extended partition. Inside of extended partition you can have one more FAT partition and the reference to the next extended partition, then another FAT, and so forth, as long as you have drive letters for them (D:, E:, F:, ...). All those partitions have special name: logical drives, on contrary to the first FAT partition C:, listed in MBR, which is primary partition.

Example 2 Hard disk 340M [665 cyl x 16 heads x 63 sects]

MBR

#	Partition Type	Starting			Ending			Size[K]
		Cyl	Side	Sect	Cyl	Side	Sect	
1	DOS FAT-16	0	1	1	399	15	63	200,568
2	Unused	0	0	0	0	0	0	
3	Unused	0	0	0	0	0	0	
4	DOS Extended	400	0	1	664	15	63	133,560



EMBR 1

#	Partition Type	Cyl	Side	Sect	Cyl	Side	Sect	Size[K]
1	DOS FAT-16	400	1	1	664	15	63	133,528
2	Unused	0	0	0	0	0	0	

We can only speculate about the reasons for choosing such design, but there are two very obvious ones. First one is that partition table has only four records and you couldn't have more than four partitions if you didn't have extended partitions. To understand the second reason you have to know that, according to Microsoft, you can have only one primary partition on the disk and you cannot boot from the logical drives, which means that you cannot have more than one DOS-like operating system on the computer, which is another way to cut off the competitors. In fact, you can have more than one primary FAT partition and we will show later how to do that.

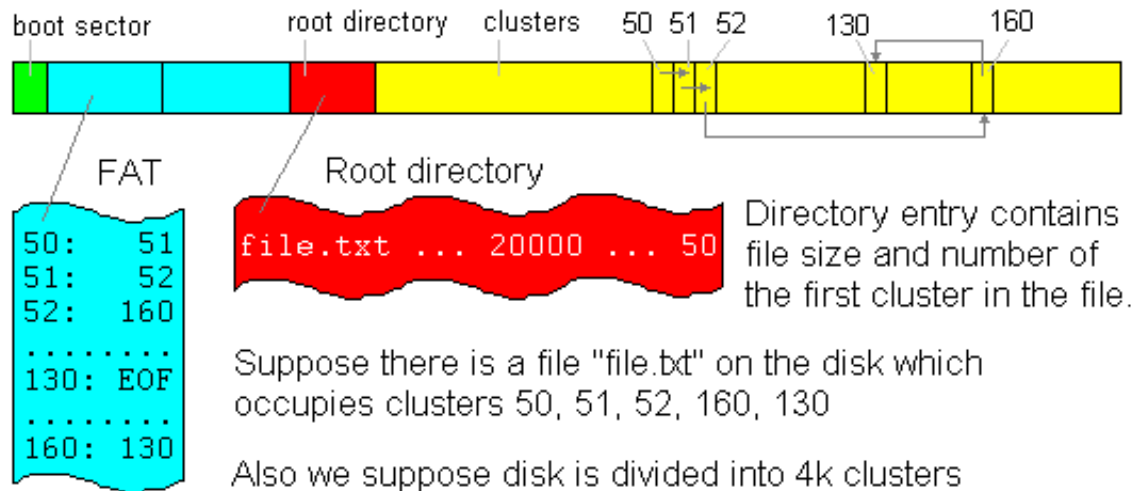
Partitioning Primer

Introduction to Hard Disk Partitioning Basics

Mikhail Ranish

Also note that FAT tables in the second partition are smaller than in the first one. It obviously happens because the second partition is smaller. The FAT tables has one entry for each cluster in the partition - it contains number of the next cluster in the chain. There is one chain of clusters for each file. Number of the first cluster of the file is stored in the directory entry for that file, along with file size, attributes, and last modification date. Space for the directories other than root is allocated among the data clusters, just as if they were ordinary files. Only root directory has special location.

FAT-16



The name of the file system is FAT-16 because it has FAT (File Allocation Table) and also because each entry in FAT is 16-bits long (2 bytes). This means that FAT-16 partition cannot have more than 65,535 clusters ($2^{16} = 65,536$). Similarly FAT-32 has 32-bit entries and could address up to 2^{32} clusters. (Actually they use only 28 bits). Based on that we can calculate maximum partition sizes for FAT file systems. Here is the table:

Cluster size	File system type		
	FAT-12	FAT-16	FAT-32
2K	8M	128M	512G
4K	16M	256M	1024G
8K	32M	512M	2048G
16K	64M	1G	2048G
32K	128M	2G	2048G

Partition size	Recommended file system / cluster size
1-16M	FAT-12 / 4K
16-256M	FAT-16 / 4K

Partitioning Primer

Introduction to Hard Disk Partitioning Basics

Mikhail Ranish

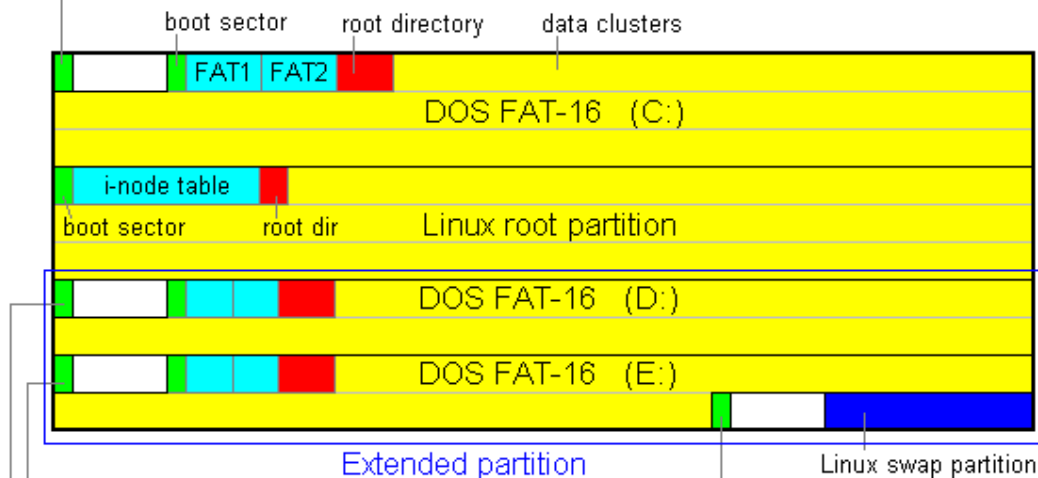
256-512M	FAT-16 / 8K
512M-1G	FAT-16 / 16K or FAT-32 / 4K
1-8G	FAT-32 / 4K
8G and up	FAT-32 / 8K

The next example shows coexistence of DOS and Linux on the same disk and some more insights on the structure of the extended partition. Beginners can skip it and go to the next section.

Example 3 Hard disk 340M [665 cyl x 16 heads x 63 sects]

MBR

#	Partition Type	Starting			Ending			Size[K]
		Cyl	Side	Sect	Cyl	Side	Sect	
1	DOS FAT-16	0	1	1	199	15	63	100,768
2	Linux native	200	0	1	399	15	63	100,800
3	Unused	0	0	0	0	0	0	
4	DOS Extended	400	0	1	664	15	63	133,560



EMBR 1

#	Partition Type	Cyl	Side	Sect	Cyl	Side	Sect	Size[K]
1	DOS FAT-16	400	1	1	527	15	63	64,480
2	DOS Extended	528	0	1	640	15	63	56,952

EMBR 2

#	Partition Type	Cyl	Side	Sect	Cyl	Side	Sect	Size[K]
1	DOS FAT-16	528	1	1	640	15	63	56,920
2	DOS Extended	641	0	1	664	15	63	12,096

EMBR 3

#	Partition Type	Cyl	Side	Sect	Cyl	Side	Sect	Size[K]
1	Linux swap	641	1	1	664	15	63	12,064
2	Unused	0	0	0	0	0	0	

Partitioning Primer

Introduction to Hard Disk Partitioning Basics

Mikhail Ranish

First of all, this configuration could be derived from Example 2 if I we shrink both FAT partitions and then install Linux. Also you probably noticed that Linux native file system uses different way of organizing files than does FAT file system. The main structure is called i-node table. There is one i-node allocated for each file. The i-node keeps file size, attributes and file creation, last modification, and last access times. Unlike FAT file system directories have only file names and i-node numbers. The space allocation also differs from FAT, but it is out of the scope of our discussion.

If you study carefully cylinder numbers in the third example you will see that extended partition has three EMBR tables each one is stored at the beginning of the extended partition and keeps a record about FAT (or other) partition and the pointer to the next extended partition. Note that first extended partition encloses all FAT and extended partitions, but all other extended partitions (level 2, 3, ...) enclose only one data partitions.

And finally I want to say, that even though this was the real partitioning scheme on my hard disk it has some drawbacks. First of all, Linux swap partition is located at the end of the disk, far from the Linux root partition. It turned out that disk heads kept going long way back and forth all the time, decreasing system performance. It is much better to place swap partition as close as possible to the partition where the OS is installed. Also some people think that if they put Windows swap file into the separate partition computer will work faster. This is true only if that partition is located on the separate hard drive, which is faster or as fast as the first one. Placing swap file on the old slow drive will not do any good. It is much better to set in the Control Panels fixed size for the swap file equal to the amount of RAM and place it on C:. Then you can run Norton Speed Disk which will optimize swap file.

How does computer boot up?

