|  |  |  |
| --- | --- | --- |
| Bochs User Manual | | |
| [Prev](http://docs.google.com/serial-port.html) | Chapter 8. Tips and Techniques | [Next](http://docs.google.com/enter-special-keys.html) |

8.16. BIOS Tips

8.16.1. Booting from CD-ROMs

A bootable CD-ROM has a special layout that is detected by the BIOS boot loader code, and executed if it conforms the specifications. This layout is called "El Torito Bootable CD-ROM Format Specification" and has been published by Phoenix and IBM. A copy of this spec is on [Bochs tech specs page](http://bochs.sourceforge.net/techdata.html).

El Torito specifies 3 ways to have a bootable CD:

* floppy emulation boot: A standard floppy image is burnt on the CD. In this case the BIOS has to redirect all first floppy accesses to this image and the real floppy drive becomes the second one.
* a "no emulation" boot: In this case the BIOS is instructed to load an arbitrary number of sectors straight into memory, and execute it.
* hard disk emulation: A hard disk image is burnt on the CD. The BIOS has to redirect all hard disk accesses to that image. The real hard disks are still available, with BIOS numbers 81h and up.

In Bochs 2.0, hard disk emulation is not implemented in the BIOS. There are also subtilities about multiple boot-images CD-ROMs, that are not handled by Bochs.

However, our BIOS may be more strict than real PC BIOSes, I don't know. But I would definitely be interested to know of any CD that can boot on real hardware, but does not in Bochs.

When failing to boot from CD-ROM, the BIOS outputs the reason of the failure as an error code, in the log file, and on the screen.

Here is a summary of what can happen when booting from the CD.

**Table 8-3. CD Boot error codes**

|  |  |
| --- | --- |
| Error code | Reason |
| 0x01 | no atapi device found |
| 0x02 | no atapi cdrom found |
| 0x03 | can not read cd - BRVD |
| 0x04 | cd is not eltorito (BRVD) |
| 0x05 | cd is not eltorito (ISO TAG) |
| 0x06 | cd is not eltorito (ELTORITO TAG) |
| 0x07 | can not read cd - boot catalog |
| 0x08 | boot catalog : bad header |
| 0x09 | boot catalog : bad platform |
| 0x0A | boot catalog : bad signature |
| 0x0B | boot catalog : bootable flag not set |
| 0x0C | can not read cd - boot image |

0x01 no atapi device found  
0x02 no atapi cdrom found

For the first two errors, an ata-\*: type=cdrom is probably missing from the configuration file. This is what you get if no cdrom has been defined in Bochs conf file.

0x03 can not read cd - BRVD

For this error, the cdrom support has not been compiled in Bochs, or Bochs could not open the file or device. This is what you get if Bochs is not able to read the cd.

0x04 cd is not eltorito (BRVD)  
0x05 cd is not eltorito (ISO TAG)  
0x06 cd is not eltorito (ELTORITO TAG)

For these errors, the data has been read from the cd, but the cd does not conform to the El Torito specification. This is what you get if the cd is not bootable.

0x08 boot catalog : bad header  
0x09 boot catalog : bad platform  
0x0A boot catalog : bad signature  
0x0B boot catalog : bootable flag not set

now the cd is eltorito, but the boot catalog is corrupted, or the cd was made to boot on a ppc system. This should not happen for a x86 bootable cd.

0x07 can not read cd - boot catalog  
0x0C can not read cd - boot image

here, specific part of the cd could not be read. This should definitely not happen.

8.16.2. Disk translation

Since the beginning of the PC era, disks have grown in size by a factor of 10000. Due to differences between the ATA specification and BIOSes implementations, when disks reached critical sizes, it became necessary to translate the CHS geometry (cylinders, heads, sectors per track) between the BIOS (int 13h) and the ATA interface. Please refer to the [ATA-FAQ](http://burks.brighton.ac.uk/burks/pcinfo/hardware/atafaq/atafq.htm) and [Hale Landis' document](http://www.ata-atapi.com/hiwchs.htm) for a complete discussion of the problem.

Unfortunately, there has never been any standard on the translation algorithms.

Bochs implements 4 well-known algorithms, selectable in the configuration file in the "**ataX-xxxx: ..., translation='algorithm'**" section.

**Table 8-4. Disk translation algorithms**

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | Maximum disk size | Maximum logical and physical geometry (CHS) | Description |
| none | 528MB (1032192 sectors) | LCHS:1024/16/63 PCHS:1024/16/63 | no translation is done. The CHS received at the int13h interface is sent as is to the ATA interface. |
| large | 4.2GB (8257536 sectors) | LCHS:1024/128/63 PCHS:8192/16/63 | a standard bitshift algorithm (named Extended-CHS) is used to translate the CHS between the int13h interface and the ATA interface. The translation is achieved by multiplying/dividing the cylinder/head count by a power of 2 (2, 4 or 8). (a factor of 16 could not be used because the head count would become 256, and MS-DOS thought this was 0) Note that the number of sectors per track is not changed, so a lower spt value will lead to a lower maximum disk size. |
| echs |  |  | synonym for large |
| rechs | 7.9GB (15482880 sectors) | LCHS:1024/240/63 PCHS:15360/16/63 | a revised bitshift algorithm (called Revised Extended-CHS) is used to translate the CHS between the int13h interface and the ATA interface. First the number of physical heads is forced to 15, and the number of cylinders is adjusted accordingly. Then, as in the simple extended CHS algorithm, the translation is achieved by multiplying/dividing the cylinder/head count by a power of 2 (2, 4, 8 or 16). The head count being forced to 15, it can safely be multiplied by 16 without crashing dos. Note that the number of sectors per track is not changed, so a lower spt value will lead to a lower maximum disk size. |
| lba | 8.4GB (16450560 sectors) | LCHS:1024/255/63 PCHS:16320/16/63 | a LBA-assisted algorithm is used to translate the CHS between the int13h interface and the ATA interface. The translation is achieved by first computing the physical size of the disk (LBA=C\*H\*S). Then the sectors per track is forced to 63, and the head count to 255. Then the cylinder count is computed (C=LBA/(63\*255)) Note that the number of sectors per track is forced to 63 in the logical geometry, regardless of the actual geometry reported by the disk. Also note that the LBA-assisted algorithm has nothing to do with LBA access at the ATA interface. |
| auto |  |  | the best suited algorithm between none, large and lba is used |

Setting a specific CHS translation should be done if you use a disk dump of an actual disk, or use a real disk as a block device. You need to know which geometry was used to format the disk, and which translation was used. You must not set the translation to 'auto'.

**Note:** rechs translation should only be useful for Compaq users who wants to use a disk as a block device. Please report if you know any other system that use such translation.

If you plan to create a new disk image (for example with bximage), format it and install an OS on it, select the "auto" translation for an automatic selection of the best algorithm based on the disk image size. Be warned that an image created with the "auto" translation might not be readable with previous versions of Bochs. Upward compatibility will be maintained.

**Note:** This translation applies only to int13h BIOS disk accesses. Older OSes (e.g. MS-DOS) tend to use them a lot. On modern OSes, disk accesses through BIOS int13h are limited to boot loaders. The usual rules and tricks of the installed OS still apply (ie 1024 cylinders boot limit).

|  |  |  |
| --- | --- | --- |
| [Prev](http://docs.google.com/serial-port.html) | [Home](http://docs.google.com/index.html) | [Next](http://docs.google.com/enter-special-keys.html) |
| Using the serial port | [Up](http://docs.google.com/howto.html) | How to enter special key combination |