Description of the MBR bootloader

This is machine-translated text

```
This bootloader must be located in sector zero of the first hard disk (MBR)
and is designed to load the boot sector of the active partition or a
partition with a given number on the specified hard disk. The MBR, having
booted from the first hard drive, can choose which hard drive to continue
booting from next. On the selected hard drive, the boot sector of the active
partition, or a partition with a specific number, is loaded. The number of
the next hard drive and the partition number on it are written in two bytes
inside the MBR sector on the first hard drive. The active partition is
determined by the first byte of the partition descriptor in the Partition
table (PT). If this field is equal to 80h, then the section is considered
active, but if it is equal to 00h, then the section is inactive.
To boot from a primary or logical partition with a given number, the
BootPart field in the MBR sector is intended, having a size of byte and
located in the MBR at offset 0x1bb from the beginning of the sector. This
field is written to the MBR when installing the bootloader. If this field is
zero, this means that the bootloader must determine the active primary
partition of the hard disk and load the boot sector from it, transferring
control to it. If the BootPart field is non-zero, this is an indication to
the bootloader to force the selection of a boot partition, logical or
primary, without checking the activity byte in the partition table. In this
case, the value of the BootPart byte is checked. If BootPart has a value
from 1 to 4, then this number is interpreted as the number of the primary
partition. If its value is 5 or more, then BootPart - 4 is the number of the
logical disk inside the Extended partition.
  Thus, since the maximum value of the BootPart byte is 255, the number of
logical disks can be from 0 to 255 - 4 = 251. That is, there can be a
maximum of 3 primary partitions plus 1 extended and 251 logical disks inside
extended.
  This bootloader uses LBA to load sectors from the hard drive into memory
if LBA is supported. Otherwise, the BIOS feature is used to read sectors
using CHS.
  Before the partition table, at offset 0x1bd, there is the ForceLBA byte.
If this byte is not zero, then the LBA is forced.
  Before ForceLBA and after BootPart, at offset 0x1bc, there is the BootDev
byte. It determines the physical device on which to look for the boot
partition with the boot sector. The value of this byte is equal to the hard
disk number in the format int 13h. For example, the first hard drive has the
number 80h, the second 81h, etc. By default, this byte is 80h, that is, it
boots from the first hard drive in the system.
  The partition table is located starting from the byte next to ForceLBA, at
offset 0x1be, and, according to the standard, consists of 4 partition
descriptors, each 16 bytes in size.
  The sector ends with an MBR signature of 0x55aa. These two bytes are
intended to monitor the integrity of the MBR, as well as for convenience
when searching for MBRs and bootsectors when restoring data on a crashed
```

```
hard drive.
  This bootloader displays two messages in emergency situations.
      The first message – “R” (the first letter of the phrase “Read error”)
is displayed when there are disk read errors,
          namely, if neither the LBA nor the CHS gave a successful status
when reading a sector from the disk.
      The message “P” means “Partition not found” and is displayed when:
          No active partition found in partition table
          Extended partition not found and BootPart byte > 4
          Looking through the EBR (Extended boot records) chain, (!!! Fix
it) the bootloader did not find a link to the next EBR and the current EBR,
according to the value of the BootPart byte, is not the last one.
          The descriptor for the required section in the EBR was not found
(should be marked with activity byte 80h).
          The descriptor in the MBR of the primary partition consists of
zeros and does not describe any partition.
  The boot loader at startup (BIOS int 19h loads it at address 0x7c0:0x0)
moves itself to address 0x60:0x0 and transfers control to the new address.
The previous bootloader location (from 0x7c0:0x0 to 0x7e0:0x0 of sector
size) is used as a buffer for loading sectors. The MBR of the first hard
drive, the MBR of the next hard drive, the 1st EBR, the 2nd EBR, ..., the
nth EBR and, finally, the boot partition boot sector are alternately loaded
into this location. After this, control is transferred to the bootsector.
  The space from 0x7e0:0 to 0x800:0 is used by the MBR bootloader, as well
as the boot sector, to store some variables. The stack used by both the MBR
loader and the bootsector is as follows: ss = 0, the bottom of the stack is
initialized to 0x7bff.
  This bootloader, after loading the bootsector,
      “corrects” the hiddensectors value in BPB by adding to it the number
of the first sector of the given logical disk (the same sector where the EBR
is located, that is, 63 sectors before the bootsector).
      The MBR bootloader also enters the physical disk number and the number
corresponding to the logical disk letter in the corresponding BPB fields of
the bootsector.
      These manipulations are performed to make it possible to boot OS /2
from logical drives in the Extended partition. Similar operations are
performed by the OS /2 boot manager when loading OS /2 from logical drives.
The fact is that for logical disks the hiddensectors value is 63, and for
primary partitions it is equal to the disk offset from the beginning of the
physical disk. Therefore, normal booting is usually only possible from
primary disks. To ensure that OS /2 can boot from a logical disk and that
(if a pre-LVM version of OS /2 is used) the correct boot drive letter is
assigned, help is needed from the boot manager.
      Until now, there were three boot managers that could boot OS /2 from a
logical disk:
          OS /2 bootmanager
          butmanager of VPart from Veit Kannegieser
          AirBoot by Martin Kiewitz.
```

Now the list can be supplemented with our MBR bootloader, which acts as a mini-boot manager. But our MBR bootloader, unlike a real bootmanager, does not allow interactive selection of the boot

partition; instead, we will select the boot partition from FreeLdr, which will perform the functions of OS Loader and bootmanager at the same time and will be booted by blackbox. (That is, it will be an advanced replacement for both OS / 2 bootmanager and os2ldr at the same time. The idea of such a combined bootloader and bootmanager first appeared in GNU GRUB, and we want to follow a similar principle. The advantage of this approach is primarily that , that from one menu we can select both the OS to load and various OS parameters, which can be passed from the loader to the kernel via config.sys.In addition, we will be able to select from the loader several versions/builds of the kernel and os2ldr, which is also very convenient.).