

Advanced Partition Notes

Svend Olaf Mikkelsen

Setting ATA Disks To Not Spin Up At Power On

Newer ATA harddisks have a feature, which make it possible to set the disk to not spin up at power on using software commands.

One problem is that some systems will hang at boot if a disk does not spin up. If that happens there is no other work around than mounting the disk in a system, which will boot with the disk present, and send commands to disable the feature in that system.

In tests, a system with Award BIOS would boot with the disk on the motherboard controller and the disk set to none in BIOS, but not with the disk on a Highpoint Controller in the same system.

If the disk has a jumper setting for enabling the feature, it can be tested if the system will boot by temporarily setting the jumpers to nospinup.

Operating systems may not have features to start the disk, and mount the partitions if the disk was not available at boot.

Using Findpart for DOS, a disk can be set to 'nospinup' using these commands (example for secondary master):

```
set findpart=edit
findpart feature sm nospinup
```

To set the disk to 'spinup' at next power on:

```
set findpart=edit
findpart feature sm spinup
```

To spin up the disk, without changing the power on setting:

```
set findpart=edit
findpart feature sm spinupnow
```

The disk location codes for primary master, primary slave, secondary master and secondary slave are: pm, ps, sm, ss.

If a disk is accidently set to not spin up at power on, I guess it could happen that no one would be able to figure out what is wrong.

Ontrack Disk Manager Behavior

No full description, just a few notes. Since I do not understand the Disk Manager program, and cannot figure out how to use it, these tests are based on a Disk Manager installation done this way:

On a system with no important data, I managed to get Disk Manager 9.55 installed using the Disk Manager program. The first 63 sectors from the disk were then copied to a file. In the system, where I would test disk manager, the content of the file was copied to the first 63 sectors of that disk, with this exception: The 64 bytes in the MBR containing the partition table were not overwritten, meaning that the partition tables in the system were kept.

Using this method, it is important that nothing is written to the MBR boot program, meaning that fdisk /mbr should not be run, Linux Lilo should not be installed in MBR, and Windows 95/98/ME should not

Advanced Partition Notes

Svend Olaf Mikkelsen

be installed afterwards. Unless the user knows what he is doing, and can copy back the 63 sectors or the MBR.

Also boot directly to floppy should be avoided, since the full disk size will not be visible if ATA Max Address is set, or the BIOS does not support the full disk size.

The advantage is that there is no partition table confusion and no 63 sectors offset. Very convenient.

One use for this would be to bring a little more life into a system with a BIOS limited to 8 GB disks.

And the behavior observations:

The disk with Disk Manager is primary master. If another disk is inserted as primary slave, and set to none in BIOS, it will not get a BIOS disk number from Disk Manager. If the same disk is inserted as secondary slave, still as none in BIOS, it will get a BIOS disk number from Disk Manager. I assume, but did not test, that the behavior for secondary master and secondary slave will be the same. If a disk is inserted as primary slave, and set to something in BIOS, it will get a disk number from Disk Manager showing the correct disk size.

It seems as Disk Manager version 9.55 will detect ATA Max Address settings, and set the disks to full size during boot. May not work for disks larger than 127 GB.

Disk manager only needs to be installed on the boot disk.

It seems as Windows 95/98/ME may have a similar behavior: A disk set to none on the secondary port will be seen by Windows. The disk geometry provided by Windows however will be confusing and wrong, so if a disk cannot be set to correct values in BIOS, it should not be used with Windows 95/98/ME without Disk Manager on disk 1, or similar.

How Disk Manager works? The MBR boot code loads a program located in CHS 0/0/10 and following sectors. That program installs a TSR program, which replaces the BIOS code for interrupt 13h.

Intel Application Accelerator (disk drivers) reporting version 2.2.0.2126 in "About" does not correctly support disks larger than 128 GB in Windows 98 SE. A version reporting 2.3.0.2160 seems to do. May apply to other Windows versions too. With the earlier version the disk size is reported correctly, but read above 128 GB fails.

Windows NT Boot

It seems as the MS Windows NT line of operating systems numbers partitions in the boot.ini file as present in the partition table. If a boot partition is in entry 2, and entry 1 is deleted, Windows cannot boot.