Optimisation

Hard Drives:

By default, linux is quite conservative in terms of hard drive settings - it opts for absolute fail safe values, mainly so that people with older pc's lying around that they want to set up as a router/web server/etc. can do so without any instability problems. However, you can get them to run must faster by changing a couple of settings. This is done using the **hdparm** command. This is installed with most of the common distributions, but if you don't have it go to **http://metalab.unc.edu/pub/Linux/system/hardware/.**

To view the current hard disk settings for a particular drive, type:

hdparm *<drivename>*
e.g. **hdparm /dev/hda**

which might output something like

```
/dev/hda:
multcount    =  0 (off)
I/O support  =  0 (default 16-bit)
unmaskirq    =  0 (off)
using_dma    =  0 (off)
keepsettings =  0 (off)
nowerr       =  0 (off)
readonly     =  0 (off)
readahead    =  8 (on)
geometry     = 1870/255/63, sectors = 30043440, start = 0
```

To test the performance of your hard drive, type

hdparm -tT *<drivename>*

This will time disk reads (**-t**) and disk cache reads (**-T**) and will output something like

```
/dev/hda:
Timing buffer-cache reads:   128 MB in 1.41 seconds =90.78 MB/sec
Timing buffered disk reads:  64 MB in  9.84 seconds = 6.50 MB/sec
```

If we change some of the settings from the default value, we should be able to increase these values.

*******WARNING - SOME OF THESE OPTIONS CAN REALLY DAMAGE YOUR DRIVE. USE WITH EXTREME CAUTION********

For extra safety, it is highly advisable that you try these modifications out in ***single-user mode*** first. This is a special mode which mounts the filesystem as read only, so if your hdparm settings are dodgy your data should still be safe. To boot into single-user mode you start up linux as normal. Whenever you get to the Grub bootloader screen, select the kernel you want to boot and type **e** for edit (if you specified a Grub password during installation you will need to type **p** and enter your password before this step). Select the line that starts with the word kernel and type **e** to edit this line. Go to the end of the line and type **single** (with a space before it), then press **enter**. Back at the Grub screen, type **b** to continue booting linux. When it loads up you can then log in as **root** and try these settings out.

If you look at the man page for **hdparm**, it will explain some of the options you can pass to the command in order to change the settings. Some of the most common ones are listed here.

`-c 3` will change the I/O support to 32bit with sync information (default is 16bit). This will only affect data going over the PCI bus - traffic from the disk over the ribbon cable to the board will still only be 16bit.

`-d 1` will enable DMA support (Direct Memory Access - the hard disk can send information straight to memory instead of being controlled by the CPU) - use with caution, especially on older computers, but give it a go anyway!

`-m 16` will turn on multisector transfers. This allows the transfer of multiple sectors per I/O interrupt, with the number after the **m** determining exactly how many. In this example, 16 sectors will be sent per I/O interrupt instead of the default 1. Most drives support values of 2, 4, 8 or 16 - but some can cope with 32. The higher the better. If you do a **`hdparm -i`** *<drivename>* it should tell you what the maximum value supported by that drive is - look for **`MaxMultSect`** in the output. This is another setting you should be very, very careful about.

`-A 1` will turn on the drives read ahead feature - the drive will prefetch additional blocks of data from a large file in anticipation of them being needed by whatever task is running.

If you really want to throw caution to the wind, try **`-u 1`** which will turn on **`unmaskirq`**. This allows linux to get on with other interrupt related tasks (network traffic, etc.) while it is waiting for the hard disk to return with the information it asked for. It is also very risky and if your particular combination of drive, IDE interface or motherboard can't cope with it, it will result in **MASSIVE FILESYSTEM CORRUPTION!!!** It can also increase latency in some cases – especially with laptops.
To start with, try **`hdparm -c 3 -d 1 -m 16 -A1`** *<drivename>*
and then run **`hdparm -tT`** to run the tests again. Any difference? If it has improved the response of your drive and you want to make your settings permanent, then add them to your **`/etc/sysconfig/harddisks`** configuration files.

You can also optimize your cdrom drive in the same way - not all features will be present, but you should be able to use DMA, unmaskirq and 32bit transfers. To make these settings permanent, copy your **/etc/sysconfig/harddisks** configuration file as **/etc/sysconfig/hardisk**<*drivename*> (e.g. **/etc/sysconfig/hardiskhdc**) and change it to reflect your settings.

Low Latency patch:

You can turn on the low latency patch installed as part of the planet ccrma kernel by typing:

```
echo "1" >/proc/sys/kernel/lowlatency
```

To make it active at boot, edit your **/etc/sysctl.conf** file to include the line

```
kernel.lowlatency = 1
```

While the low latency patch is active, you must observe the following:

Avoid scrolling the fb console (if **fbcon** is installed)
Don't run **hdparm**
Don't use **blkdev_close()**
Don't switch consoles

As more don'ts are discovered, they will be added here -
http://www.zip.com.au/~akpm/linux/schedlat.html#ddt

Interrupt priority and the PCI latency timers:

Interrupts have a set order of priority, and it is a little strange - something wierd to do with the history of the development of the PC. Here it is:
0, 1, 8, 9, 10, 11, 12, 13, 14, 15, 3, 4, 5, 6, 7
0 is the highest priority
7 is the lowest priority
Makes perfect sense.....
Anyway, interrupts 9, 10, 11, 3, 4, 5, 6, 7 can potentially be made available to PCI slots on many motherboards. If you want your soundcard to perform as well as possible, you should try to get it assigned as high (in terms of priority, not numerically!!) an interrupt as possible - preferably interrupt 9. Otherwise anything with a higher priority will take away precious resources from your soundcard. If you want to see which devices are assigned to which interrupt, play a sound file (this will make sure that your soundcard is using its interrupt) and type

```
cat /proc/interrupts
```

This will display a list of interrupts. If you need to change the interrupt of your soundcard, then you have two possible options.

Option 1 - If your BIOS allows, you may be able to change your interrupt (ICQ) assignments in there

otherwise....

Option 2 - physically swap your PCI cards about!

Another problem is that all the devices on the PCI bus are fighting to send information over the same route - for example, if card A is doing something and card B suddenly decides it needs to do something else should card A stop what it is doing to allow card B to do its business? And if so, for how long?

Each device on the PCI bus has its own PCI bus latency timer, and normally the default settings are ok. This timer can be any value from **0** to **248** - a setting of 0 means that the device will give up the bus as soon as another device starts transmitting while a setting of 248 means that it will continue doing its task for longer before it frees up the bus for another device. To view your current latency settings, type

```
lspci -v
```

Typing this command will display very detailed information about all of your PCI devices. The PCI latency setting for each device is listed on the third line, right before the IRQ setting.

To set the PCI latency of your soundcard to 248, you would use

```
setpci -v -s <PCIDeviceID> latency_timer=ff
```

The <*PCIDeviceID*> are the first numbers listed by the device when you do **`lspci -v`** and will be hexadecimal. The ff is also hexadecimal for 256, which for some weird reason then gets rounded down to 248 by **`setpci`**. So your command might look like

```
setpci -v -s 00:0e:0 latency_timer=ff
```

and this would be added to your system startup scripts (in **`/etc/rc.d`**).

CDROM polling:

The computer will occasionally check the cdrom device to see if a disc has been inserted - if it finds one then it will quite often try to mount it, run a cd player, open a folder etc. **TURN THIS OFF!!!!** It is a waste of system resources and you should get into the habit of mounting your cd's manually. You can even write a script to do it for you if you are too lazy to type mount -t iso /dev/hdc /mnt/cdrom or whatever you need to use for your system!

More links:

This is where I found all this info – if you want to read up on it, check them out!

http://www-106.ibm.com/developerworks/library/l-hw2.html

http://linux.oreillynet.com/lpt/a/linux/2000/06/29/hdparm.html

http://sourceforge.net/mailarchive/forum.php?thread_id=1079993&forum_id=7073

Planet CCRMA @ Home