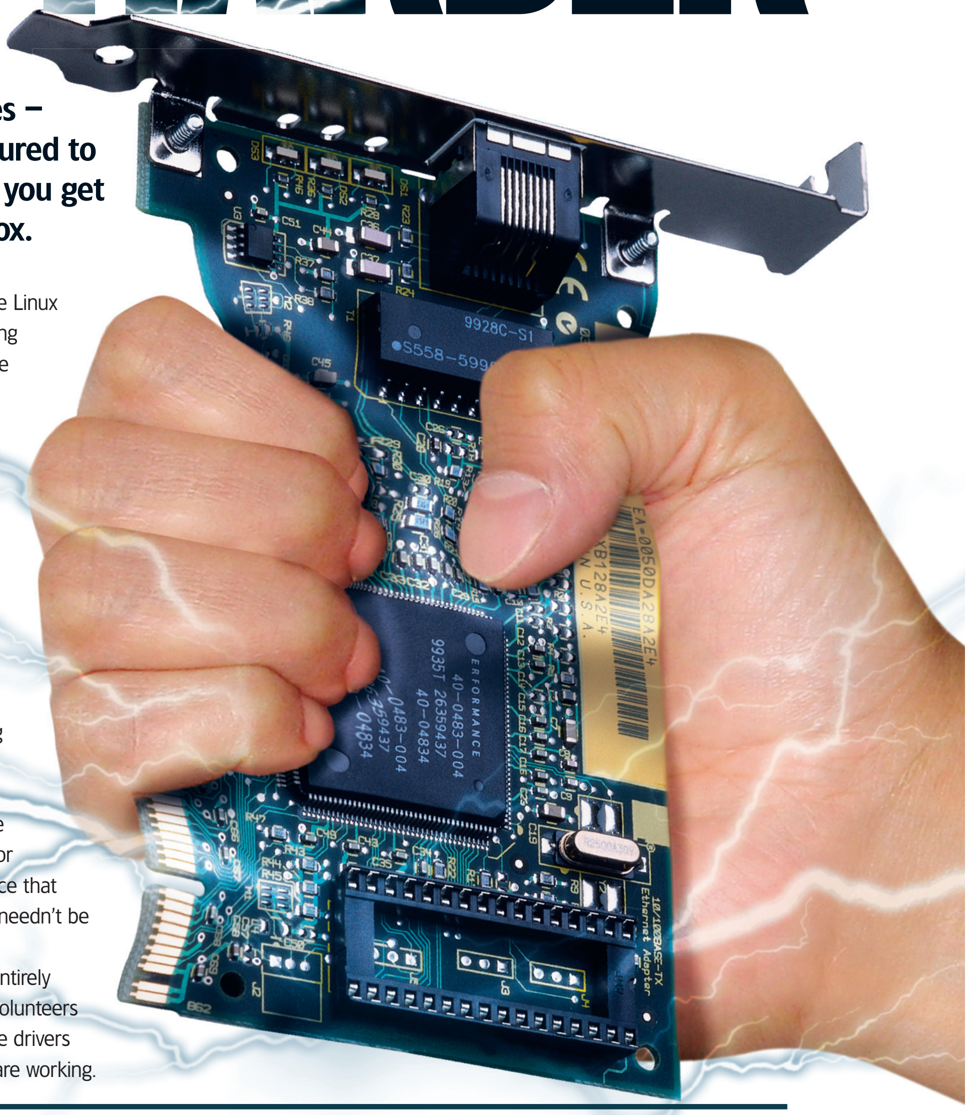
MAKE YOUR HARDWARE WORK HARDER

Computers these days are connected to so many devices – cameras, printers, iPod-alikes – that it's essential to get your machine configured to work with them all. **Graham Morrison** helps you get every last gadget working with your Linux box.

COVER FEATURE



When you choose Linux as your operating system, you give yourself a whole heap of advantages over the common-or-garden computer user. With that one choice you've protected yourself from viruses, gained an amazingly stable operating system and probably saved an awful lot of money. But yes, there are downsides, and chief among these is Linux's sometimes fraught relationship with hardware – almost everyone who uses Linux will sooner or later be frustrated by a device that refuses to cooperate. But it needn't be like this...

Linux users are almost entirely dependent on the work of volunteers to design and implement the drivers necessary to get our hardware working.

"LINUX HARDWARE IS BECOMING EASIER TO INSTALL, BUT THERE'S ALWAYS ROOM FOR IMPROVEMENT."

Without reference documentation, these developers often have to go to great lengths to figure out how a piece of hardware actually works before they can start to code for it. Then there are the proprietary drivers – closed solutions provided by companies like Nvidia and ATI. In the cut-throat world of 3D graphics, it's understandable that manufacturers are going to be very cagey about their intellectual property. But it leaves many people in a dilemma about free software alternatives, and stops the manufacturers' drivers being included in many open distributions.

Linux hardware is becoming easier to install, especially with the great improvements made with USB devices made over the last two years, but there's always room for improvement. We're going to get all your problematic peripherals working so that your Linux box and its hardware will be 100% functional. That includes graphics cards, networking devices, printers, scanners and even your iPod.

ON the DISC

CUPS Printing system.
DigiKam KDE photo management.
F-Spot Gnome photo management.
GPhoto-libs For digital cameras.
KSynaptics Touchpad driver.
Lineak Advanced keyboard support.
LIRC Infra-red drivers.
Madwifi Atheros wireless networking.
Ndiswrapper Wifi driver layer.
Sane Scanner drivers.
SynCE Link to Windows CE devices.

Input expert

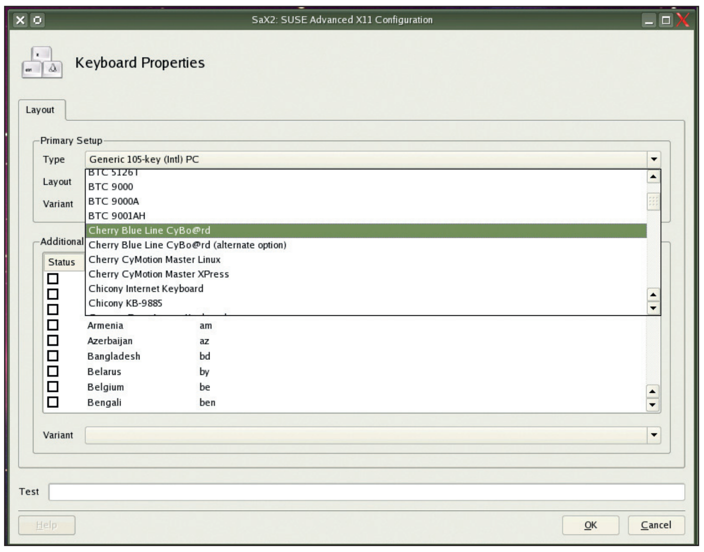
Keyboards, mice and joysticks are probably the simplest PC devices you'll ever use; they're certainly the most important. Here's how you can get the most out of them.

KEYBOARDS

Many modern keyboards have extra keys for tasks such as controlling a media player. In Linux, this keyboard mapping is done through the *X Window System*, unlikely as that might sound, and it needs to know your specific model to be able to map the new keys. Some distributions let you do this from one of their graphical configuration tools. For example, SUSE and Mandriva have their own keyboard configuration panels, available from SUSE's *Yast* and Mandriva's *Control Center*. These tools simply add the keyboard to the XkbModel field in the *X Windows* configuration file (**/etc/xorg.conf**) under the Input Device section. When you change anything in this file, you will need to restart the *X* server (log out and back in again) for the changes to have an effect.

The result is that the extra buttons (sometimes called multimedia keys) now send a recognised signal to your desktop applications. To see this in action, open the key editor for your favourite application and press one of the special keys. It should come up with a designation such as 'XF86Back' for the back key, or 'XF86AudioNext' for the play key. This means you can now assign these extra keys.

If you need more control over what functions your extra keys perform, the utility you want is *Lineak* (short for Linux support for Easy Access and internet Keyboards). This consists of a daemon that captures output from your keyboard, and graphical configuration utilities for Gnome and KDE (*Lineakconfig* and *KLineak*, respectively). *Lineak* not only supports more keyboards than the *X Windows* configuration file, it has the advantage of being able to run scripts.



SaX2: SUSE Advanced X11 Configuration

Keyboard Properties

Layout

Primary Setup

Type: Generic 105-key (Intl) PC

BTC 5126T
BTC 9000
BTC 9000A
BTC 9001AH
Cherry Blue Line CyBo@rd
Cherry Blue Line CyBo@rd (alternate option)
Cherry CyMotion Master Linux
Cherry CyMotion Master XPress
Chicony Internet Keyboard
Chicony KB-9885

Layout

Variant

Additional

Status		
☐		
☐		
☐		
☐	Armenia	am
☐	Azerbaijan	az
☐	Bangladesh	bd
☐	Belarus	by
☐	Belgium	be
☐	Bengali	ben

Variant

Test

Help OK Cancel

SUSE's *Yast* tool can get your keyboard working the way it should.

COOL HACK UNDETECTED KEYS

Sometimes not all your keys will be recognised, but they still send a signal. You can solve this by mapping the signal to a key manually, using a command called *xmodmap*, but you need to get the key code before you can do this.

To find the key code, start *xev* from a command prompt. This small utility outputs everything you press when the window is active. You'll find that when you press a key you get a huge amount of information, and the parameter you're looking for is keycode. Next, create or edit a file in your home directory called ~/.Xmodmap, and enter each keycode with a key you'd like it to be mapped to. Like this:

```
keycode 161=F13
```

To make the assignment work, just run xmodmap ~/.Xmodmap and you'll find the keys are now assigned. In the above example, pressing the new key will tell the system you've pressed F13.

MICE

The device that goes hand in hand with the keyboard is usually the mouse, and Linux is notoriously bad at supporting mice with more than the industry standard of one wheel and three buttons.

It's the *X Windows* system that's at fault here, as it only supports five buttons on a mouse (a wheel counts as two buttons because it can rotate backwards and forwards.

The mouse is configured from the same config file as keyboards, and most distributions don't trouble themselves by veering too far from the default. However, there is a certain amount you can do to change your mouse behaviour – you'll need to load **/etc/X11/xorg.conf** into your favourite text editor first.

Mouse settings can be found under the Input Device section following the keyboard configuration. First, if you need to, change Buttons to reflect the number of buttons your mouse has, counting two for each wheel. Then, the Z Axis Mapping option tells *X Windows* which two buttons to use for the wheel, but you can also add another two buttons for a second wheel.

In the code below, we've just added the **6** and **7** to the Z Axis Mapping line for wheel number two:

```
Option     "ZAxisMapping" "4 5 6 7"
```

JOYSTICKS

There are quite a number of gaming devices available that are compatible with Linux, including joysticks and steering wheels. The driver for joysticks is included with the kernel and will load automatically when you plug your device in. If you're still using an old gameport, you need the corresponding module loaded.

The joystick can be found under **/dev/input/js0** on your filesystem. Check that the device is working by printing the output from **/dev/input/js0** as you move the joystick around. Print the output with **cat /dev/input/js0** – it should give you a screen full of spurious-looking characters. A better

way to test your hardware is a small utility called *jstest*, which is part of the **input-utilities** package. Typing **jstest --normal /dev/input/js0** should produce the value of your game device in all of its axes (usually just up and down, but a few joysticks rotate, too). Games that can use a joystick and find your **js0** device will configure themselves automatically.

If you need force feedback, you're out of luck. A few projects did start to develop drivers, but right now they're all gathering dust on SourceForge. »

“Configuring a picture perfect PC

You can do all sorts of things with images on Linux – *Gimp* them up, use them in print-outs, put them up on the web – but you have to get them in to and out of the machine. Here’s how...

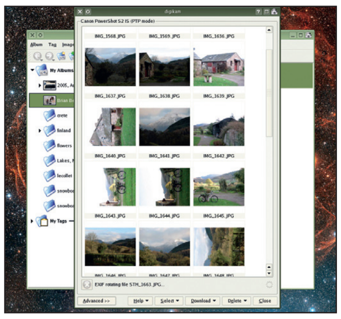
CAMERAS

It wasn’t so long ago that getting your digital camera to work in Linux was a difficult task. Each manufacturer seemed to use its own proprietary methods over USB, and gave you very little information. Things have settled down somewhat, and now most digital cameras use one of two protocols to transfer images to the computers.

The most common is USB mass storage. This is where you plug your camera into your machine, and it appears as external storage. You’re then free to copy, move and delete your images as you would on a separate hard disk.

The second possibility is PTP, the Picture Transfer Protocol, used by many newer cameras from Sony, Kodak, Nikon and Canon.

The first thing to do is plug your camera into your machine. If it uses the USB storage protocol, most modern distributions will attach the camera to the filesystem automatically, and either open a window for browsing the photos or place an icon on your desktop. If this doesn’t happen automagically, you can check the status of your camera by typing **dmesg |grep usb-storage**. This searches the system log for the string ‘usb-storage’, and the output should look something like this.



***DigiKam* can download images from your digital camera with PTP.**

```
usb-storage: device found at 8
usb-storage: waiting for device to
settle before scanning
usbcore: registered new driver usb-
storage
usb-storage: device scan complete
```

The following two commands will mount this device manually, after which you’ll be able to read your images from **/mnt/camera**. When you’ve finished with the device, there’s a corresponding unmount command, which is **umount /mnt/camera**.

```
mkdir /mnt/camera
mount -t vfat /dev/sda1 /mnt/
camera
```

If the **dmesg** command didn’t return any results, it’s likely that your camera uses PTP to communicate. If so you’ll need to use one of the limited number of PTP-compatible photo applications to access and manage the images held on your camera. The most popular is KDE-based *DigiKam*, but *F-Spot* for Gnome is similar.

To add a PTP camera to *DigiKam*, just select Add Camera from the Camera menu. If yours isn’t listed, you should try the generic USB PTP Class Camera entry. You can add your USB camera in *F-Spot* in the same way, but its PTP camera communication isn’t quite as stable as *DigiKam*’s.

SCANNERS

Scanners are still an important part of some computer setups, and the Linux protocol for communicating with them is called Sane. Unlike Windows, which has one driver for each physical piece of hardware, Linux treats the functions of multifunction devices (these offer both scanning and printing) as separate devices; you need to use Sane to get the scanner working and CUPS to get the printer working. Sane is very powerful, and offers features that aren’t available anywhere else, such as scanning across a network.

It’s composed of a back-end and a front-end – the back-end connects to the hardware and the front-end connects to the user. Many software packages including *Gimp* and *Kooka* offer support for scanning, but a quicker way to check that things are working is with the *xsane* utility, which lets you see what you’re doing when configuring your scanner.

The main problem with USB scanners – and it’s similar with other devices such as network cards and webcams – is that they often need a proprietary firmware file, usually plundered from a Windows installation.

The firmware needs to be uploaded to the device before you can start working with it. First you need to identify the scanner on the USB bus, using *lsusb* as root:

```
# lsusb
Bus 001 Device 004: ID 04a5:2060
Acer Prisa 620U
```

Check the Sane archives for any firmware you may need (see **www.sane-project.org**). For example, searching for the above scanner (Acer Prisa 620U) reveals that it needs a firmware file called **u96v121.bin**, which is available from the Windows driver package. The link also states that the scanner uses the **/etc/sane.d/snapscan.conf** file and that you need to change the second line in the file to point to the firmware file. After doing this, you just need to turn your scanner on.

Any application that uses Sane will now be able to find the scanner, and you can test it first using *xsane*. *Gimp* adds a Sane connection through its File > Acquire menu, and is probably the best tool for scanning images.

PRINTERS

There is a big difference in the level of support offered by manufacturers when it comes to creating hard copy. The problem is compounded by a new generation of multifunction devices that feature a scanner (and sometimes a fax function) alongside the normal printing hardware.

A unified Windows driver, supplied by the manufacturer, manages both functions simultaneously. But as we saw in the Scanners section, with Linux the printing functions need to be split from the scanning functions – and that can be a little tricky. In fact, the oft-repeated advice about Linux hardware – that you should know exactly how compatible a device is before you purchase it – applies more strongly to printers than any other kind of hardware.

Driver guide

Linux is a world full of wonderful acronyms, and CUPS is one of the better ones. It stands for Common Unix Printing System, and your printer needs a CUPS-compatible driver for it to work. There are two manufacturers that provide good support for their printers, and they are HP and Epson.

Drivers for many older printers will be included in your distribution, but you need to be careful if you buy the latest model – a driver may be months away from release. As a general rule, you can usually get a printer functioning adequately with a driver from a previous model in the range, but you won't be able to take advantage of any features specific to your new device.

If you want to use a newer model, you need to get the CUPS printer driver first. HP provides the best support, through the HP Linux Printing Project (HPLIP), which is hosted on SourceForge but also included in nearly all current distributions. HPLIP offers an integrated driver for many of HP's devices, including some of its multifunction devices. For other printers, you need what is known as a PPD file (Postscript Printer Description). This file contains a description of your printer's capabilities, including any specific functionality it offers, or the language it uses for printing.

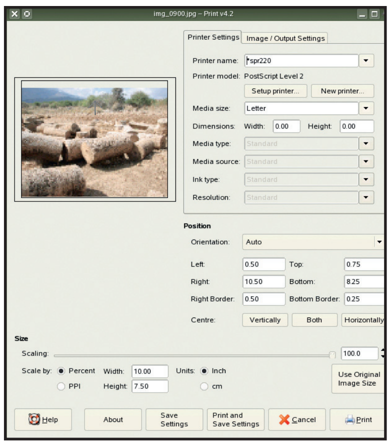
Most distributions offer their own printer configuration utilities, but CUPS has its own web-based interface for adding, removing and probing printers. CUPS is nearly always installed by default, but if your computer didn't have a printer connected when you installed your distro, you might need to install CUPS yourself.

COOL HACK ADDING CMYK TO *GIMP*

CMYK (Cyan, magenta, yellow and black) is essential if you need professional quality output, but *Gimp* doesn't natively support this colour format. Fortunately, there's a plugin to help. It's almost always included with *Gimp*, but you can get the latest version from www.blackfiveservices.co.uk/separate.shtml. Once it's installed, you can convert RGB (*Gimp*'s default colour space) into CMYK. The CMYK functions are hidden away under the Image menu, which appears when you right-click on a loaded image in *Gimp*. The menu is called Separate, and selecting Separate (Normal) will bring up a window where you can specify source and destination colour profiles (usually installed at /opt/gnome/share/color/icc). You can download several free colour profiles from Adobe's website (www.adobe.com). The resulting image will have four layers, one each for cyan, magenta, yellow and black.

COMMON PROBLEMS

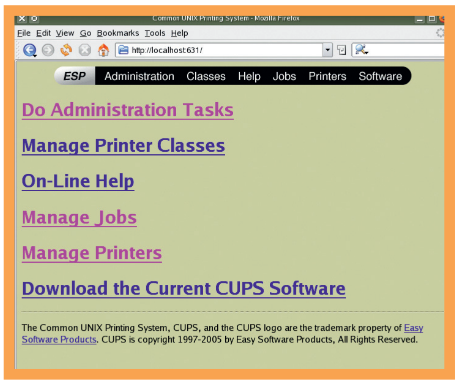
- **Symptom: Sending an image to the printer produces text.**
Cure: Get an updated PPD file or a different driver – CUPS is sending the wrong printer language to the printer. Try www.cups.org/ppd.php first.
- **Symptom: Epson driver can't add the device URI.**
Cure: Open the /etc/cups/cupsd.conf file and add 'FileDevice Yes'.
- **Symptom: Editing and adding devices with CUPS asks for a password and username.**
Cure: Open a shell and type lppasswd -g sys -a root, enter a password and re-connect as root.



Printer name: *spr220
Printer model: PostScript Level 2
Media size: Letter
Dimensions: Width: 0.00 Height: 0.00
Media type: Standard
Media source: Standard
Ink type: Standard
Resolution: Standard
Orientation: Auto
Left: 0.50 Top: 0.75
Right: 10.50 Bottom: 8.25
Right Border: 0.50 Bottom Border: 0.25
Centre: Vertically Both Horizontally
Scaling: 100.0
Scale by: [✓ Percent] PPI Width: 10.00 Height: 7.50
Units: [✓ Inch] cm

***Gimp-Print* is an alternative to CUPS, used from within *Gimp*.**

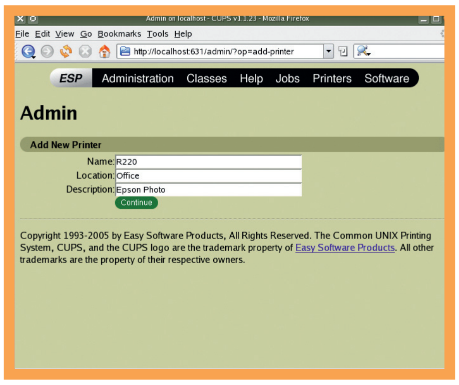
ADDING A PRINTER WITH CUPS


Common UNIX Printing System - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:631/
ESP Administration Classes Help Jobs Printers Software

Do Administration Tasks
Manage Printer Classes
On-Line Help
Manage Jobs
Manage Printers
Download the Current CUPS Software

The Common UNIX Printing System, CUPS, and the CUPS logo are the trademark property of Easy Software Products. CUPS is copyright 1997-2005 by Easy Software Products, All Rights Reserved.

1 Once CUPS is installed, open up a web browser and go to **http://localhost:631**. You will need to log in using your root account details, after which you will find yourself looking at the CUPS index page.


Admin on localhost - CUPS v1.1.23 - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:631/admin/?op=add-printer
ESP Administration Classes Help Jobs Printers Software
Continue

Admin

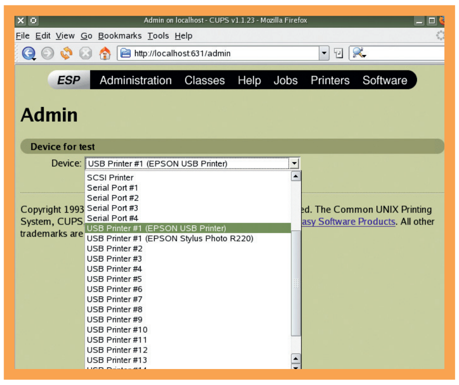
Add New Printer

Name: R220
Location: Office
Description: Epson Photo

Copyright 1993-2005 by Easy Software Products, All Rights Reserved. The Common UNIX Printing System, CUPS, and the CUPS logo are the trademark property of Easy Software Products. All other trademarks are the property of their respective owners.

2 Select the Manage Printers option from the main CUPS screen. This will usually list all your installed printers, but if there is none currently installed the list will unsurprisingly be empty. Click on Add Printer to start configuring your new device.

CUPS will ask for a name, a location and a description for the new printer. This is only descriptive data (such as 'Colour Printer' rather than a model name such as HPXJFZ-12345X), so you just need to enter enough for your own information.


Admin on localhost - CUPS v1.1.23 - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost:631/admin
ESP Administration Classes Help Jobs Printers Software

Admin

Device for test

Device: USB Printer #1 (EPSON USB Printer)

SCSI Printer
Serial Port #1
Serial Port #2
Serial Port #3
Serial Port #4
USB Printer #1 (EPSON USB Printer)
USB Printer #1 (EPSON Stylus Photo R220)
USB Printer #2
USB Printer #3
USB Printer #4
USB Printer #5
USB Printer #6
USB Printer #7
USB Printer #8
USB Printer #9
USB Printer #10
USB Printer #11
USB Printer #12
USB Printer #13

Copyright 1993 … ed. The Common UNIX Printing System, CUPS, … asy Software Products. All other trademarks are

3 The next page is where you select how your printer is connected to your system. There's a huge range of options from the drop-down list, but nine times out of ten your printer will be connected and already be listed as 'USB Printer #1'. Other common choices include older printers connected to either your computer's parallel port or SCSI port.

Once the printer is selected you need to choose the make and model of your printer. If you've added the printer correctly, an icon for the new device will show up on the printer page. You can also manage any printer jobs from here. »

LINUX-COMPATIBLE PRINTERS

	MODEL	EXPECT TO PAY
Multiuse	**HP OfficeJet 4255**	**approx £100**
	Epson Stylus Photo RX620	**£170**
Laser	**HP Color LaserJet 3000**	**approx £500**
	Epson Aculaser C1100	**approx £250**
Inkjet	**HP DeskJet 5740**	**approx £70**
	Epson Stylus Photo R800	**approx £220**

“Net works

A modern office wouldn’t run without a network, and many of us have more than one computer at home. How lucky, then, that connecting to the internet or other PCs is becoming easier and easier on Linux...

ETHERNET

The easiest way to connect one computer to another is by using the Ethernet port that nearly every machine has tucked away near the keyboard and mouse connections. Typically, an Ethernet network is a group of machines connected to a switch or hub, which sits at the centre. One thing to bear in mind is that if you’re just connecting one machine to another, you will need a ‘crossover’ cable rather than the standard Ethernet cable: this makes up for the absence of a switch or hub.

The simplest configuration is to create a server that all other machines use as a gateway to other services, whether it’s for sharing files or an internet connection. This server is usually responsible for assigning addresses to the other machines automatically. It’s also possible to use fixed addresses for each machine on the network, but this method isn’t all that versatile – it can’t accommodate changes to the network easily.

The good thing about most Ethernet networks is that provided you’ve got a single machine acting as the DHCP server, the other machines should be able to configure themselves with little intervention. You don’t need to worry too much about security, either. As long as your gateway to an external network is secure – using a firewall or a DMZ for example – any intruder would need to gain physical access to an Ethernet port to be able to connect to your network. The same can’t be said of wireless networks.

WIRELESS NETWORKING

Getting wireless hardware to work with Linux has always been a little problematic. This is because the manufacturers that develop the hardware seldom offer a driver. (Anyone thinking about buying wireless hardware should check **www.linux-wlan.org** to see if there’s a compatible driver available.)

If there’s no specific driver for your hardware, there are two paths you could follow. The first is if your hardware uses the common Atheros chipset, and that includes many products from companies like D-Link, Linksys and Netgear.

Because of the nature of wireless networking, you don’t need physical access to look at files shared over a wireless network. This is good and bad: you can move a laptop around the office and still be connected to your work; but then so could the script kiddies in the building across the road.

Going Madwifi

Madwifi is the name given to the driver for devices that include the Atheros chipset. Madwifi has improved dramatically in the last year, thanks to the continuing efforts of the original Madwifi team and the recent involvement of the Atheros development team. This has brought many advanced features such as support for Super AG, the proprietary extension to the 802.11g wireless format, with its extended range and extra speed.

You first need to be sure you’ve got a card with an Atheros chipset. Get this information from your distribution’s control panel, or by typing **lspci** as root from the shell. You should be looking for something like this:

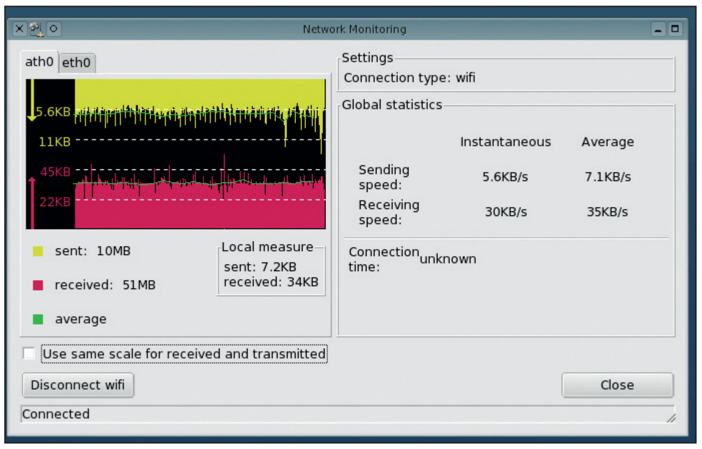
```
Atheros Communications, Inc.
AR5212 802.11abg NIC
```

To install the latest generation of the driver, download either a snapshot or an RPM from **www.madwifi.org**. To compile the driver, you install your kernel source code, which is easier than you think. Just open your distribution’s package manager, select the kernel source packages and click on Install. In Mandriva’s package manager and SUSE’s *Yast*, the package will be cunningly called **kernel-source** or similar. You’ll also need to make sure you’ve got the tools for building applications, including *GCC* and *make*.

To install *madwifi-ng* from source, untar the file, change to the new directory and run **./configure** followed by **make** and **make install** as root. You can restart your machine, or remove then add the *wlan*, *ath_hal* and *ath_pci* modules manually. The new Atheron-flavoured Madwifi drivers create a ‘wifi0’ device, and you need to create a virtual wireless interface to be able to make a connection with your access point. To do this, enter

```
wlanconfig ath0 create wlandev
wifi0 wlanmode sta
```

The final parameter **sta** creates the device in station mode, which adds a virtual wireless device on top of the wifi0 device. With the new Atheros drivers, you’re free to create other virtual devices that access the same card. This means that you can create a virtual access point and a client all on



```
Settings
Connection type: wifi

Global statistics
                 Instantaneous    Average
Sending speed:   5.6KB/s          7.1KB/s
Receiving speed: 30KB/s           35KB/s

Connection time: unknown

🟨 sent:  10MB
🟥 received:  51MB
🟩 average

Local measure
sent: 7.2KB
received: 34KB

☐ Use same scale for received and transmitted

Connected
```

Check the performance of networking hardware with a monitoring tool.

COOL HACK EXTENDING WIRELESS RANGE

If you want to extend your wireless network’s range, you’ll usually need to add a repeater to your wireless network. This is basically an access point that mirrors the signal to any neighbouring wireless devices.

Another way of extending the range is to use two wireless network devices, although bear in mind that one needs to be capable of operating as an access point (as with Atheros cards using the new drivers).

Use one card as the client, connected to the distant access point. Then use your distribution’s network configuration tools to share the network connection on this device with other connected devices (often called internet connection sharing).

This will perform a similar function to the wireless repeater. Devices can connect to the local access point and have their connection routed to the distant access point.

the same card, with the following code.

```
wlanconfig ath1 create wlandev
wifi0 wlanmode ap
iwconfig ath0 essid "Access Point
Name"
```

Open Ndiswrapper

If your wireless hardware doesn't have a native Linux driver, there is a solution: *Ndiswrapper*. This is a clever piece of programming that implements the Windows kernel API and its network driver interface so that you can use drivers designed for Windows within Linux. Even though it was designed for wireless devices, *Ndiswrapper* has been so effective that it works with other devices, such as USB serial ports.

The problem with *Ndiswrapper* is that because it re-implements some Windows API calls, it falls foul of some distributions' open software policy. It can be downloaded from **http://ndiswrapper.sourceforge.net**, but owing to its popularity, you can usually find an RPM for your system.

Once *Ndiswrapper* is installed we'll download the Windows driver for your wireless device. Knowing which driver to download isn't always easy, because you usually can't tell from its chipset or documentation which chipset your hardware is using. To find out for sure, we need the device ID. This is a hexadecimal number unique to every product, and it can be used to search for the correct driver on the *Ndiswrapper* website.

Do this in two steps from the console. First, type **lspci**. This will list each device connected to your main system bus. Find your wireless network card, and remember the number in the first column – it will look something like '**00: 0d.0**'.

Next, type **lspci -n**. This lists the same devices, but as a group of IDs rather than text. Here is an example of a single line from both commands for the same device:

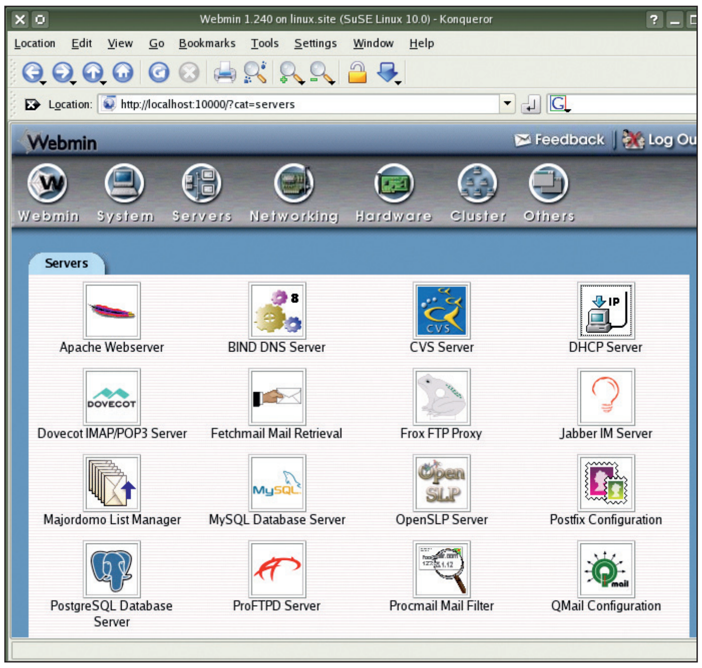
```
00:0d.0 Ethernet controller: Atheros
Communications, Inc.
00:0d.0 Class 0200: 168c:0013 (rev
01)
```

We want the PCI ID from the fourth column (**168c:0013** in this example). Use this ID to search the list of *Ndiswrapper* devices hosted on the above SourceForge page and download the linked Windows driver.

We now need to get to the separate files within the Windows driver; you might need to use *cabextract*. The file you want ends with '.inf', and you install the driver with

```
ndiswrapper -i mydriver.inf
```

List *Ndiswrapper*-installed drivers (and check you've been successful) with **ndiswrapper -l**. Finally, load the *modprobe ndiswrapper* module, which should load your Windows device driver and create the wireless device.



Configuring network services is easier using a tool like *Webmin*.

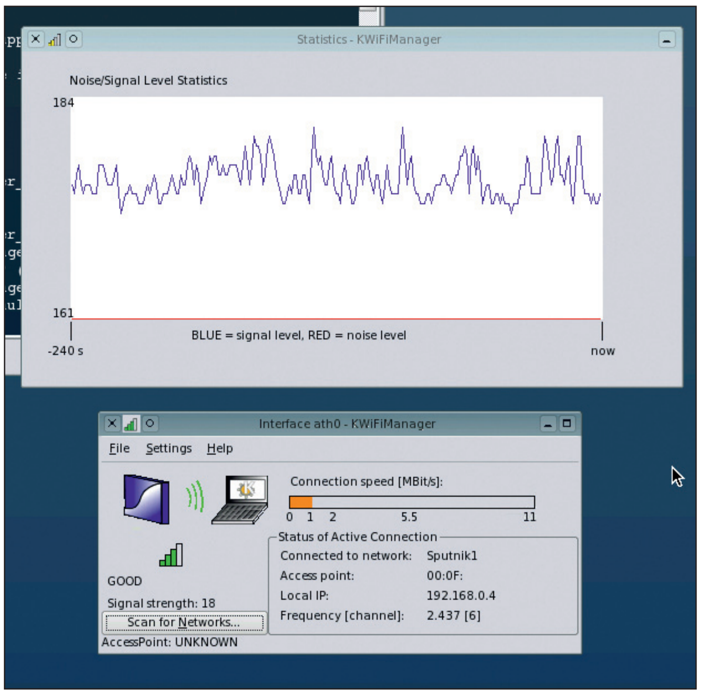
COMMON PROBLEMS

■ Symptom: Madwifi gives an 'ioctl' error message when it adds a device.
Cure: You need to make sure there are absolutely no references to ath0 in /etc/iftab.

■ Symptom: The quality of the network signal fluctuates.
Cure: Check the signal strength using the *iwconfig* wireless tool. When the link quality falls below 20/94, the connection can become unpredictable. Aerials usually transmit in a dome shape around the vertical axis. To get the best coverage, your router's aerial should be horizontal (this is especially important for routers that are high up).

■ Symptom: You've set your device into Super G mode, but the connection is no faster.
Cure: The trouble with Super G is that it's proprietary. This means you need a compatible access point or router, usually by the same manufacturer.

CONFIGURING A WIRELESS DEVICE


Statistics - KWiFiManager
Noise/Signal Level Statistics
184
161
-240 s
now
BLUE = signal level, RED = noise level
Interface ath0 - KWiFiManager
File Settings Help
Connection speed [MBit/s]:
0 1 2 5.5 11
Scan for Networks...

Status of Active Connection
Connected to network: Sputnik1
Access point: 00:0F:
Local IP: 192.168.0.4
Frequency [channel]: 2.437 [6]
GOOD
Signal strength: 18
AccessPoint: UNKNOWN

***KWifiManager* can use wireless profiles to configure a device.**

To manipulate any wireless device, you need to install the **wireless-tools** package. This includes all the commands you need to configure your wireless devices and make a connection. To view your devices, type **iwconfig**; you can scan for any available access points using the *iwlist* command. The Atheros device we created earlier is called 'ath0', while the one created by *Ndiswrapper* is called 'wlan0'. To scan for access points, just insert the device into the *iwlist* command, thus:

```
# iwlist ath0 scan
ath0        Scan completed :
            Cell 01 - Address:
00:00:00:00:00:00
            ESSID:"Sputnik1"
```

To start a connection to the **Sputnik1** access point listed above, set the wireless device to managed mode, set the WEP key and tell it which access point to connect to. WEP is a simple method for encrypting the connection as it is broadcast, and the client and access point need to be using the same key for it to work. Here are those commands in sequence, which should leave you with a working network connection:

```
iwconfig ath0 mode Managed
iwconfig ath0 key restricted s:
PASSPHRASE
iwconfig ath0 essid 'Sputnik1'
ifconfig ath0 up
```

»

"Card sharp

So many of the technological innovations that make Linux so exciting depend on a properly running graphics card. Whether it's OpenGL, transparency or newfangled 3D desktops, graphics performance is paramount.

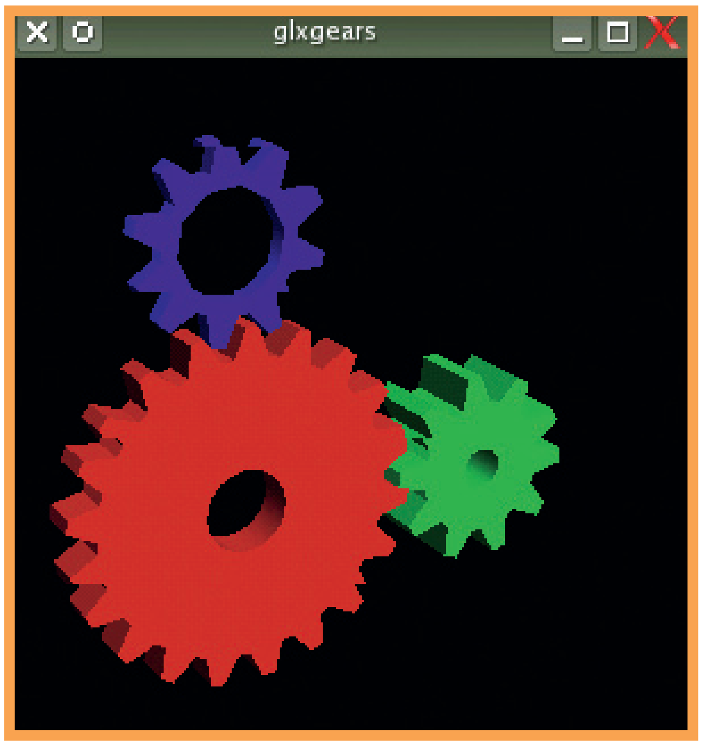
GRAPHICS CARDS

The graphics card is one of the most important and problematic parts of many Linux systems. A properly configured card can provide a high resolution, accelerated video and 3D, as well as a responsive desktop. A badly configured card produces the opposite effect – screen resolutions are too small and your desktop feels sluggish, even if you've got the most whiz-bang graphics card out there.

That's the difference between a driver that can take advantage of all the cool new hardware crammed in to your card and a driver compatible with a card built ten years ago.

There are really only two vendors if you need graphics performance in Linux: Nvidia and ATI. You may have some luck if you own an S3 Savage, but otherwise you're stuck. They perform just as well on Linux as they do on Windows, but it has to be said that the Nvidia drivers are superior to ATI's.

SETTING UP WITH NVIDIA OR ATI



1 First, we need to check how your graphics system is performing. There are as many ways of doing this as there are Linux distributions. You could just play *Crack Attack* – a slow screen update is a sign that things aren't as they should be. But the traditional way of testing your 3D graphics performance is using a tool called *glxgears*. This is an old utility, but it will quickly tell you if your graphics card is underperforming. For best results, execute the command from a shell. You should see three spinning cogs (often used as a screensaver). Every five seconds, a count of the number of frames is printed in the shell.

If your hardware isn't accelerated, you'll get in the region of 250FPS (frames per second) or less. This depends on the speed of your computer, but if everything is working you should have a dramatically higher number: expect anything in the range of 1,000 to 20,000FPS, depending on your graphics card and hardware. When you are getting results in this range, you only need to follow our instructions if you're an obsessive compulsive, or you want to update your drivers to the latest version.

"WITH A BADLY CONFIGURED GRAPHICS CARD, SCREEN RESOLUTIONS ARE TOO SMALL AND DESKTOPS FEEL SLUGGISH."

COOL HACK DUAL SCREEN

One of the cheapest ways of extending your desktop is to add another monitor. To do this, you'll need a card with two or more outputs and a driver to handle the added complexity. Both Nvidia and ATI's cards can do this, by configuring *X Windows* to open two servers on the same card. The two screens operate entirely independently, so you could run a game on one and an email client on the other. Note: this is different from Nvidia's TwinView configuration, which gives you one large desktop and was covered, along with other examples of dual screen, in *LXF68*.

The only way to achieve dual screen is by once again delving into the *X Windows* configuration file. Open **/etc/X11/xorg.conf** into your favourite text editor, and find the Device section for your Nvidia video card. Each screen needs to be assigned a separate device. We add another device by adding an extra copy of the Device section in the config file and making a couple of changes to both copies of the section.

Add **Screen 0** to the first device section and change the identifier to **nvidia0**. In the second Device section, change the identifier to **nvidia1** and add **Screen 1**. Next, we should add the name of the bus that the card is connected to inside your machine.

To find it, run **lspci |grep nVidia** as root from a shell. The first column in the output is the bus ID. This needs to be added to both device sections, making both device entries look something like the following:

```
Section "Device"
    Identifier  "nvidia0"
    Driver      "nvidia"
    BusID       "PCI:2:0:0"
    Screen      0
EndSection
```

You also need to duplicate the screen section and the monitor section from the **xorg.conf** file. Change the identifiers in the screen section to **Screen0** and **Screen1**, and make one reference the first Nvidia device (**nvidia0**), and the other **nvidia1**. You also have to copy the Monitor section, and use different identifiers in each one (**Monitor0** and **Monitor1**). Finally, update the 'Server Layout' section with the following two lines, and restart your *X* server.

```
Screen      0 "Screen0"
Screen      1 "Screen1" leftOf
"Screen0"
```

2 The ATI and Nvidia drivers both need your kernel's source code. This is because their installers build their own drivers against whichever version of the kernel you've got installed.

Next, we need to get the drivers themselves. And happily, these are downloadable directly from the Nvidia and ATI websites (**www.nvidia.com** and **www.ati.com**). One quick note here: SUSE makes things slightly awkward when it comes to graphics drivers, because it uses its own configuration tool (called *Sax2*), which can overwrite your settings whenever it feels like it. For this reason there's an extra step in configuring graphics cards for SUSE users, instructing *Sax2* to make the changes that *you* want it to rather than those that it thinks you should want.

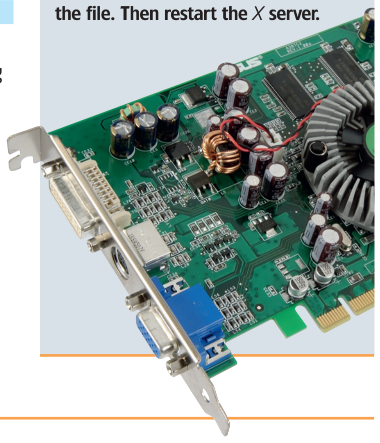
This is why, if there's an RPM or automatically installable package available for your graphics card, it's worth using that rather than trying to forge ahead on your own.

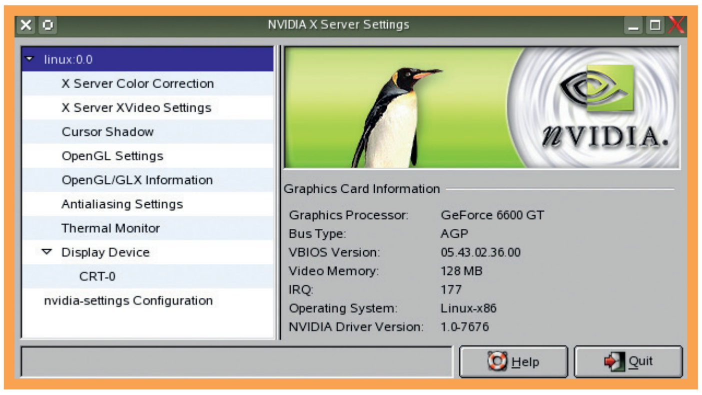
COMMON PROBLEMS

■ Symptom: *X Windows* **doesn't start**
Cure: This is most likely because the correct kernel module hasn't loaded. Look at the log file (/var/log/Xorg.0.log). Type lsmod as root, and look for either nvidia for Nvidia cards or fglrx for ATI. If it's missing, load it using the *modprobe* **command, or re-install the drivers.**

■ Symptom: A previously working system doesn't load *X Windows.*
Cure: Distributions like SUSE often use an auto-update tool that can install a new version of the kernel automatically. This breaks the current graphics card driver, which needs to be recompiled/reinstalled for the new version.

■ Symptom: The driver is loaded, *X Windows* **starts but there's no OpenGL acceleration**
Cure: This is because the *glx* **module hasn't loaded when** *X Windows* **starts. Open up /etc/X11/xorg.conf and add Load "glx" to the Module section in the file. Then restart the** *X* **server.**





Graphics Card Information

```
Graphics Processor:     GeForce 6600 GT
Bus Type:               AGP
VBIOS Version:          05.43.02.36.00
Video Memory:           128 MB
IRQ:                    177
Operating System:       Linux-x86
NVIDIA Driver Version:  1.0-7676
```

3a NVIDIA

Nvidia cards are some of the best manufacturer-supported drivers for Linux. Even if there is a debate about whether Nvidia should open the source to the community, Nvidia's drivers can bring an enormous boost to your system's performance.

They're also easier to install than ATI's drivers, thanks to the installation script, which is able to tell if there are pre-built binaries available, and even build its own if it can't find any. You'll find the drivers at **www.nvidia.com/object/unix.html**.

After you've downloaded the drivers, be sure to end your current *X Windows* session by executing **init 3** as root before running the Nvidia installer from the command line with the following code:

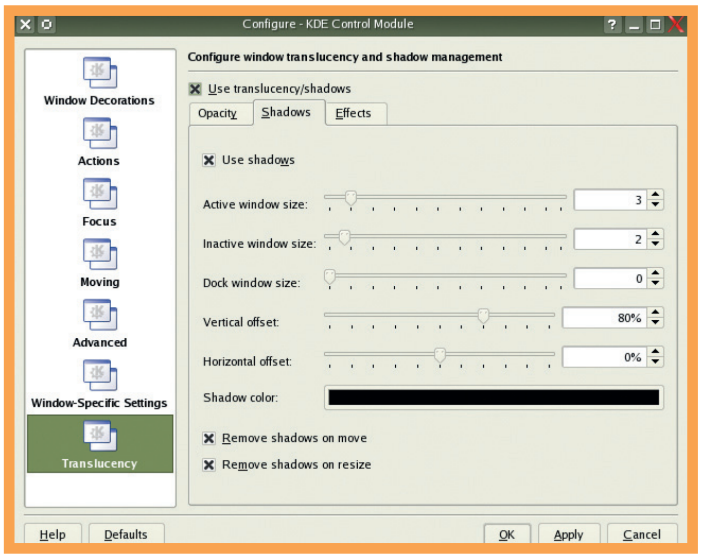
```
sh ./NVIDIA-Linux-x86.run
```

This will launch a *curses*-based installation application that will either guide you through downloading drivers for your distribution and kernel, or compile them itself. When it's finished, you need to change the driver that *X Windows* uses from the free *nv* to the new *nvidia*. SUSE users can just run the following:

```
sax2 -m 0=nvidia
```

The rest of us need to edit our *X Windows* configuration file by opening **/etc/X11/xorg.conf**. Look for the section called Device, which should contain a group of options for your Nvidia card. You need to change the Driver field from **nv** to **nvidia**.

After you've done this, restarting the *X* server (**init 5** as root) should give you OpenGL acceleration for as much *Unreal Tournament* action as you can handle.



Configure window translucency and shadow management

☒ Use translucency/shadows

☒ Use shadows

Active window size: 3
Inactive window size: 2
Dock window size: 0
Vertical offset: 80%
Horizontal offset: 0%
Shadow color:

☒ Remove shadows on move
☒ Remove shadows on resize

3b ATI

ATI's proprietary drivers only support modern ATI cards. If you have anything older than a Radeon 8500 you'll need to use the open source driver that's included with your distro's *X.org* installation. These aren't bad, and thanks to the Utah-GLX project, can even use some of the hardware-accelerated features of your card.

Back to more recent cards. You first need to download the all-inclusive driver from ATI's website, rather than the distro-specific one (which we have never been able to get to work). This is the larger of the two available, and should be around 60MB.

When it's downloaded, switch to root and type the following into a shell (we've truncated the **ati-driver** filename in the code, but the latest version is **ati-driver-installer-8.19.10-386.run**):

```
sh ./ati-driver-installer.run --get-supported
```

This will output a list of distributions supported by the driver, and should include things like 'Ubuntu/breezy' and 'SuSE/SUSE100-IA32'. The only significant omission is Mandriva, but you can get this to work by using an RPM built for Fedora (it will appear as 'Red Hat/RHEL4').

The next step is to build the package for your distribution. Do this by including your distro as a built parameter. For example, the following line would build a SUSE package:

For SUSE:

```
sh ./ati-driver-installer.run --buildpkg SuSE/SUSE100-IA32
```

For Fedora:

```
sh ./ati-driver-installer.run --buildpkg RedHat/RHEL4
```

This will build an RPM for your system. When this has finished, you need to close down your current *X Windows* session (in most systems this is done by typing **init 3**), and install the RPM (**rpm -Uvh fglrx.i386.rpm**). If all has gone well, your system should now contain the ATI driver; you just need to change the *X Windows* configuration file to use the new driver. For SUSE users, this means typing

```
sax2 -r -m 0=fglrx -b /usr/share/doc/packages/fglrx/sax2-profile
```

For the rest of us, you can use the supplied configuration utility called **fglrxconfig**, which should make the proper amendment to your *X Windows* configuration file. »

Audio heaven

Oggs, MP3s, WAVs – even the filenames sound cool. Here's how to turn your computer into a sophisticated audio centre.

ALSA

The average Linux distribution can process audio with more power than many recording studios had ten years ago – and this being Linux, it's all free. The software that controls this huge range of functions is the Advanced Linux Sound Architecture (ALSA).

The price of all this power is that it's often scarily complicated. There are many, many facets to sound production on Linux, but when you can't get even get a sound out of your machine, the first thing to do is to change the mixer settings. The best utility for this is called *Alsamixergui* (well, at least the name is clear...!), and the typical soundcard will present more than a dozen mixer channels to play with.

The most important channels to try are Master, PCM and Mix. If you use the digital rather than the analogue output on your soundcard, you need to try various levels of the channel labelled IEC958. This collection of numbers and letters is a cryptic reference to a digital audio format, but it doesn't really help the user.

Rather than acting as a volume controller, the IEC958 channels often switch between various digital output modes. For chipsets made by Creative and VIA you need to move this channel to the 25% position.

The secret to configuring your soundcard is tucked away in a file called **~/.asoundrc**. This is a hidden file located in your home directory, and it's from here that you can rename and rearrange all your card's inputs and outputs as well as perform hardware mixing and down sampling. There's no easy way to do all this; ALSA badly needs a configuration tool.

COOL HACK REORDER MULTIPLE DEVICES

When you have two audio devices, such as an internal soundcard and an external TV card or headphones, they can often be initialised the wrong way round at boot time. The TV card could initialise as the first device, for example, but when it's removed the first device will revert to the internal sound hardware. This often stops applications from producing sound, because they're expecting a specific device at a specific place.

The answer is to force them to a specific slot when the drivers are loaded. The configuration for achieving this is /etc/modprobe.conf.

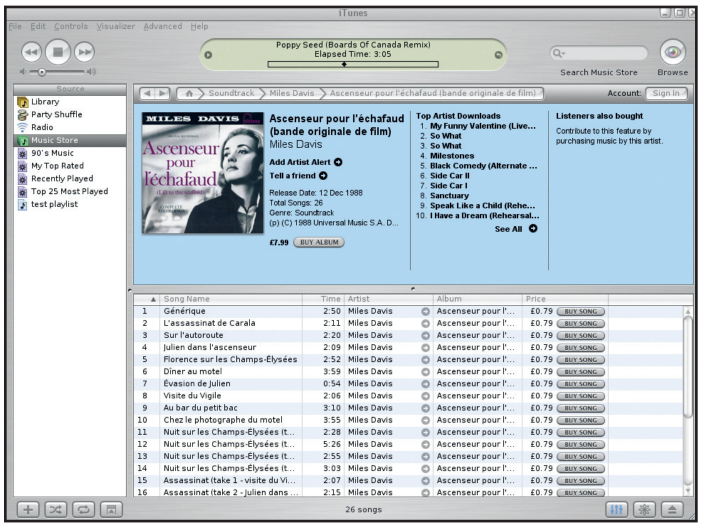
Here's an example of forcing a VIA sound chipset on a motherboard (snd-via82xx in the code below) to take the first slot (dsp0) and an external device (snd_go7007) to take the second (dsp1):

```
alias sound-card-0 snd-via82xx
alias sound-card-1 snd_go7007
```

IPOD ON LINUX

Even those of us with the strongest disliking for proprietary formats can't help but be impressed with Apple's iPod *(above right)*. The company spends millions on design, and it shows: in contrast with other media players it's a thing of beauty, and the best bit is that you can just pick it up and use it without needing anyone to show you how it works. The only bad thing about the iPod is that it was designed to work with proprietary systems, but with a bit of tinkering, you can make it work with Linux.

As the iPod uses either a Mac or a Windows filesystem, we've always been able to copy and delete files – it was just that it wouldn't update its internal database. There are now several ways to solve this problem. The easiest and most comprehensive solution is to use *CrossOver Office*. While this solution does cost money (*CrossOver Office* isn't cost-free), you get to run the real *iTunes* software and even buy music to synchronise to your iPod.



Poppy Seed (Boards Of Canada Remix)
Elapsed Time: 3:05

Ascenseur pour l'échafaud (bande originale de film)
Miles Davis

Release Date: 12 Dec 1988
Total Songs: 26
Genre: Soundtrack
(p) (C) 1988 Universal Music S.A. D...
£7.99

Top Artist Downloads
1. My Funny Valentine (Live...
2. So What
3. So What
4. Milestones
5. Black Comedy (Alternate ...
6. Side Car II
7. Side Car I
8. Sanctuary
9. Speak Like a Child (Rehe...
10. I Have a Dream (Rehearsal...

Listeners also bought
Contribute to this feature by purchasing music by this artist.

▲	Song Name	Time	Artist	Album	Price
1	Générique	2:50	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
2	L'assassinat de Carala	2:11	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
3	Sur l'autoroute	2:20	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
4	Julien dans l'ascenseur	2:09	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
5	Florence sur les Champs-Élysées	2:52	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
6	Dîner au motel	3:59	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
7	Évasion de Julien	0:54	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
8	Visite du Vigile	2:06	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
9	Au bar du petit bac	3:10	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
10	Chez le photographe du motel	3:55	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
11	Nuit sur les Champs-Élysées (t...	2:28	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
12	Nuit sur les Champs-Élysées (t...	5:26	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
13	Nuit sur les Champs-Élysées (t...	2:55	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
14	Nuit sur les Champs-Élysées (t...	3:03	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
15	Assassinat (take 1 - visite du Vi...	2:07	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG
16	Assassinat (take 2 - Julien dans ...	2:15	Miles Davis	Ascenseur pour l'...	£0.79 BUY SONG

***ITunes* will run and sync with *CrossOver Office* ($39.95 from CodeWeavers).**

It's also possible to install *iTunes* using *Wine*, but few people manage to synchronise with their device.

If you want a native solution, there are several applications that can copy files to your iPod and update the integral music database. The original iPod app for Linux is called *gtkpod*, and it's an excellent application for creating playlists and managing your music. If you're a KDE user, *Amarok* can manage your iPod music collection directly by dragging and dropping files from your collection. You can also queue tracks to be uploaded at a later time.

THE LAST WORD

There's still an awful lot that device manufacturers could do to improve Linux compatibility. Many companies provide no information whatsoever to developers who are willing to give up their own time, and often money, to develop a driver that will only make the product sell more and so make them more money.

Nonetheless, hardware support within Linux has never looked so good. Many devices will work out of the box with Linux, and every revision of the kernel supports more pieces of equipment. The trick is in knowing which devices to choose. A little research into what will and what won't work can save many hours struggling with the hardware later.

It's also worth supporting manufacturers that have shown an inclination to release driver information on their hardware to Linux developers. If you need further help with a device, it's well worth posting on the LXF forum under the Hardware section for advice from readers and the LXF team. LXF