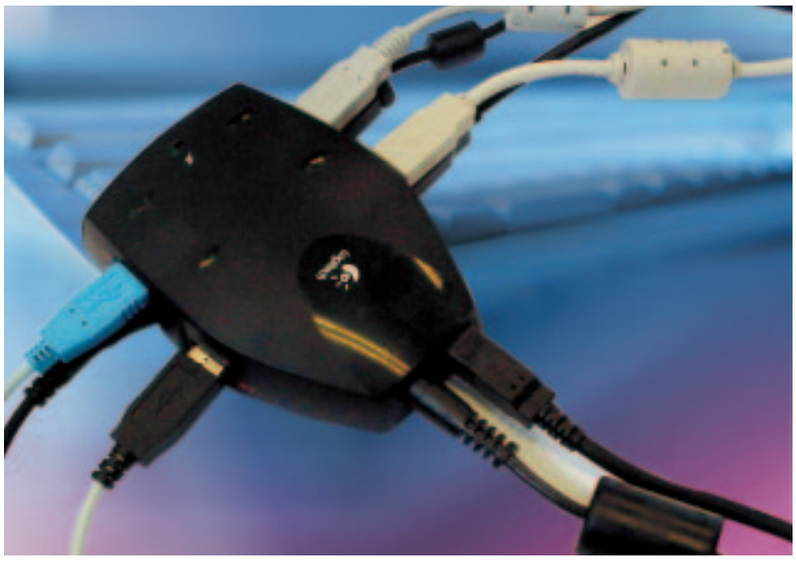
Linux USB in practice

SINGLE-FILE INPUT

CHRISTIAN REISER



This issue sees the beginning of a series of articles on USB peripheral devices. This month, we're looking at input devices: keyboards, mice, joysticks and graphics tablets.

One thing is clear right from the start, the success rate is encouragingly high. All ten devices worked. This is mainly due to the fact that both mice and keyboards use a specifically defined protocol. However, there are some differences in the case of joysticks. Many of them operate with the standard driver. Other joysticks, such as the Logitech Wingman Force, require extra modules to be loaded.

Keyboards

It's quite easy to connect a USB keyboard to your computer. Simply activate "Support USB-Keyboard" in the BIOS of your computer and skip the rest of this article. The keyboard is recognised by BIOS and then accessed under Linux like a normal keyboard. Loading the modules for the USB controller under Linux renders communication via BIOS impossible, since this is re-internalised. A kernel support then has to be installed. There are two different modules to choose from, which can cope with USB keyboards: *usbkbd* and *hid*. The sole advantage of the *usbkbd* driver is its size. As a specific keyboard driver, it only needs a quarter as much memory. The 16k of *hid.o* is not exactly enormous. The *hid* driver on the other hand also supports mice, joysticks and a few graphics tablets. Our test revealed that the special keys of the Microsoft keyboard can only be used with the *hid* driver. The module *keybdev* is also necessary in order to pass on keyboard events to the console driver.

Keysonic ACK-298

The 105 keys and the additional PS/2-USB-mouse connection of the Keysonic ACK-298 worked as expected. The integrated converter converts the PS/2 control signals to the USB mouse protocol. But at the same time "only" three mouse keys and one wheel are supported. The mouse on the Keysonic keyboard is addressed in this case like any other USB mouse – and also correspondingly installed (see section on mice). This keyboard is comfortable to use despite its low price. The keys have a firm pressure point, making it obvious whether or not a key has been pressed

Microsoft Internet Keyboard Pro

The keyboard itself actually consists of three devices: The 105 normal keys, the 19 special keys and a 4-port hub (in which, however, 2 ports are already internally occupied). The hub functions without any problems and supports both high and low-speed devices. The normal keyboard, too, posed no problems – sadly, the special keys did. But this is not exclusively a USB problem the PS/2 connector (also provided) requires at least as much additional configuration to make the keys work. The configuration necessary is also required for other keyboards, so we have devoted a separate box to this topic ("special keys under Linux").

Mice

As in the case of the keyboards, there are two modules to choose from: *usbmouse* and *hid*. Here again, when using the *hid* module you just can't go

wrong. After that, *mousedev* has to be loaded, which passes the events to the device file(s). These are created as follows (in */dev*):

```
bash> mknod mouse0 c 13 32  # 1. mouse
bash> mknod mouse1 c 13 33  # 2. mouse
bash> mknod mouse2 c 13 34  # 3. mouse
[...]
bash> mknod mice c 13 63  # Mixed signal from
all mice
```

With these files the USB mice can be addressed like their PS/2 counterparts, and in the case of the text console with, say, */usr/sbin/gpm -tps2 -m /dev/mice*.

Logitech MouseMan Wheel

In addition to the two keys and the wheel on the front, this product from Logitech has an extra key on the side. This keyboard is also suitable for larger hands. It is rubberised on the side and the somewhat angular shape the mouse sits well in the hand. There were no problems worth mentioning when it was used under Linux.

Logitech WingMan Gaming Mouse

This mouse was the only one of those tested here which did not come with a wheel. The only advantage was one less component to go wrong. That's important in the case of this mouse model, as it was built to withstand maximum loadings for gaming. Surprisingly, the connecting cable is not as sturdily designed as its MouseMan colleagues introduced above. Otherwise the device can be positioned very smoothly and easily. The buttons are very smooth and confirm every press with a clearly audible "click".

Microsoft IntelliMouse Explorer

This mouse will catch the eye on any desk or workstation. As soon as it is activated, the Intelli-Logo lights up in a subtle red to remind you of its new design. This mouse no longer works with the traditional rollerball. Instead, movements are detected directly by scanning the backing. In our test this worked superbly – there were no problems with any mats. Two of the four keys are operated with the thumbs – this worked first time. Side grip suffers slightly because of the very rounded housing.

Background: The wheel actually consists of two keys: an up key (4) and a down key (5). After restarting the X-Server, with *xev* the „Key presses"

Integrated hubs make life easier. The special keys of the Microsoft Internet Keyboard Pro, like other keyboards, require special treatment.

It worked out of the box: Logitech MouseMan Wheel

Special keys under Linux

If a key is pressed, the keyboard generates a so-called scan code, and when released, a second one. In USB this is first queried by hid (Human Interface Device) and then reaches the kernel via keybdev. But this doesn't matter at this point. The scan codes consist of one (in the case of control keys, of two or more) byte(s). On release, a value increased by 0x80 is always generated (for example 0x1e for "a" pressed, 0x9e "a" released). These scan codes can be queried directly, perhaps with showkey -s. After that, a key can be assigned a "Keycode" using setkeycodes. There are 128 of these: The first 88 are equivalent to their scan codes – the rest are dynamic. These can be queried with getkeycodes. These key codes must then be assigned an action once and for all (for example the issue of a letter to the application). This is done with the aid of loadkeys. In the following example, we will configure the "Mailkey" on the keyboard so that when it is actuated the console program pine starts:

```
bash> showkey -s

kb mode was XLATE
press any key (program terminates 10s after
last keypress)...
0x9c
0xe0 0x6c
0xe0 0xec

bash> setkeycode e06c 120 # we assign the
keycode 120 to the scancode e06c

bash> loadkeys EOF
> keycode 120 = F70
> string F70 = "pine\n\n\n\n"
> EOF
```

With XFree86 things look a bit more complicated. X-Server is one of the applications that query the scan codes directly from the kernel and then reprocess them. Unfortunately the concept is not as good as that of the kernel. With XFree86 the scan codes are permanently assigned keycodes. If these are altered the entire X-Server will have to be recompiled. If you are confident enough to do this, you can find some tips below.

The "Red Shooter" from the electronics mail order firm Conrad has a good price/performance ratio.

Leaves very little to be desired.

The unusual control concept of the Microsoft SideWinder Dual Strike can lead to damage to the device in tricky games manoeuvres.

Only a few USB graphics tablets will currently run with Linux. The Wacom Graphire USB is one of them.

Wheels under X

To be able to use the mouse wheel under XFree86 (from 3.3.2) it has to be entered into the configuration file XF86Config (either in /etc or /etc/X11). To do this, you enlarge the "Pointer section" by the following lines:

```
ZAxisMapping   4 5
# or in the case of XFree86-4.0.x:
Option "ZAxisMapping" "4 5"
```

should now be interpreted as wheel movements. After successful modification of the *XF86Config* there are now two options for "attuning applications to the wheel": *IMWheel* or *Xdefaults*. These two methods are fundamentally different in concept. *IMWheel* is a program that has to be explicitly started, while the *Xdefaults* contain standard settings for the X-Server. This means that no additional resources are needed. However, configuration is possible only to a limited extent. In the *Xdefaults*, each wheel movement is assigned an X-event – but in *IMWheel* the wheel movement can have two keys assigned (e.g. up and down arrow keys). The configuration file has the following structure:

```
"Programname"
None,    Up,      Key
None,    Down,    Key2
# And now with key combination:
Alt_L,   Up,      Key3
Alt_L,   Down,    Key4
```

Any application can simply be attuned to the mousewheel. The drawback is that regardless of where you are in the window, the same action is always performed. This could be very troublesome, especially with something like a multiframed Netscape.

There are also a few applications that can use the mousewheel without any special installation, as they directly process mouse buttons 4 and 5. These include Staroffice, Wine, XMMS, Gimp and all Xaw programs.

Joysticks

It is only since kernel version 2.2.x that support for joysticks has been firmly integrated. The result of this has been that developers of applications for which joystick support would be useful (say games), didn't usually offer any support for this important input device class until recently. Thanks to games porters such as Lokigames, the use of joysticks under Linux has now become the norm. The use of USB is a logical progression for the application. In such cases USB joysticks behave as if they were attached to the game port. The following device files will be needed for this though. These can be added with the command *mknod* (provided they're not supplied by the distributor):

```
bash> mknod js0 c 13 0  # 1. Joystick
bash> mknod js1 c 13 1  # 2. Joystick
```

On kernel modules, in this instance, you will also need, in addition to *input* and *hid*, *joydev*.

Conwheel Red Shooter USB

This joystick has four keys and six axes: two for the X/Y axes, two rotating wheels on the ground plate and two "axes" for controlling the viewing direction (hat stick). Apart from the thrust controller, five LEDs have been added, which reflect the current respective setting. The joystick is constructed exclusively for righthanders. The only problem is that the maximum input values of the X/Y axes are reached considerably before the maximum excursion, and the result of this is that the joystick is "dumb" in the marginal area.

Logitech WingMan Extreme

The exterior gives no sign at first of what is hidden inside this gamepad. Apart from the eleven buttons there is also a digital directional controller or as an alternative to that, directional control via a position sensor. If this is active, a turn to the right is interpreted as a control signal to the right. Only ten of the eleven buttons can be occupied – the eleventh is provided for toggling between motion sensor and cursor. Control by means of the sensor functions very well. The zero setting is easy to find. The only snag is that for long-time players the forced position caused by the joystick could become tiring. In any case a lot of physical input is guaranteed. There is no shortage of fun and games.

Microsoft SideWinder Dual Strike

This gamepad takes a bit of getting used to. Apart from the "normal" cursor and the nine buttons, there are also two axes. These can be actuated by turning the two halves of the SideWinder towards each other. However, you should bear in mind that not every direction is allowed! This type of control requires discipline from the player, so that nothing goes wrong. Apparently the manufacturers, Microsoft, were also aware of this. They have installed an additional sensor/ button in the axes so that the operating system can give a warning as soon as a critical condition occurs. Unfortunately this emergency backup does not function with Linux.

Wacom Graphire USB

Graphics tablets with USB connections are still relatively rare. One of the best sellers is the Wacom Graphire USB. It is also the only one which is already

SuSE 6.4 and USB

SuSE 6.4 does not use a dynamic procedure to load the requisite USB module. This means that it may no longer be possible to perform any input when using a USB keyboard. A tiny error has slipped into the file /etc/config.d/usb.rc.config. Instead of the two modules hid and keybdev, the module usb-keyboard has been loaded. Despite its intuitive name, this module does not even exist. Nor does the mouse module, mousedev would be more correct in this case. Consequently under SuSE 6.4 the line in the configuration file ought to look like this:

```
USB_DRIVERS="hid keybdev mousedev"
```

SuSE did not set a good example in Version 6.4 (which is still widely distributed) when it came to the device files. If you don't notice that the pre-installed files are wrong as soon as a mouse has been connected, you'll be in trouble. Solution: simply delete the existing files and reinstall them.

directly supported in the kernel. With the module *wacom* the tablet is recognised as a mouse. In order to still be able to use the pressure sensitivity, you have to use the XInput expansion from XFree86. The module for this can be found at the fifth Web site listed below. To connect it up, the following specifications in the "Input-Section" of the configuration file *XF86Config* are necessary.

```
Section "Module"
  Load "xf86GraphireUSB.so"
EndSection

Section "XInput"
  Subsection "gmouse"
    Port       "/dev/input/input0"
    DeviceName  "Graphire Mouse"
    Mode       Relative
    AlwaysCore
  EndSubsection
  Subsection "gstylus"
    Port       "/dev/input/input0"
    DeviceName  "Graphire Pen"
    Mode       Absolute
    AlwaysCore
  EndSubsection
  Subsection "geraser"
    Port       "/dev/input/input0"
    DeviceName  "Graphire Eraser"
    Mode       Absolute
    AlwaysCore
  EndSubsection
EndSection
```

At this point care must be taken to ensure that the "device" is definitely specified. It is advisable to ensure that the Wacom driver is loaded first and thus always sits at "input0". After restarting the X-Server all you need is an application which accesses XInput. Obviously in this case a Malprogram such as Gimp would be appropriate. Unfortunately both GTK and Gimp have to be converted again first. Refer to *http://www.gtk.org/~otaylor/xinput/* for this.

Interim results

The success rate for the USB devices so far tested under Linux is encouragingly high. You can, in principle, buy any USB mouse, any USB keyboard and any USB joystick without any worries. With a few minor modifications, these too should be capable of installation under Linux. It is only in the case of the graphics tablets that there is at present still insufficient support. In the next issue we will take a look at data storage devices with USB connections. ■

Input concept

Simultaneously with USB in the upcoming kernel 2.4, an attempt is also made to introduce a new input structure. This is based on the fact that all hardware drivers first send their signals to the central input module. The modules which process the signals and pass them on to the device files – the so-called "event handlers" are also registered there. So evdev is a very special "event handler". With its associated program evtest all events sent out by a specified device (the movement of an axis, or key press for example) can be captured. This might be a keyboard. But it also functions with all other drivers that use the input module. Unfortunately this general input architecture is currently only used by USB:

```
bash> mknod input0 c 13 64
bash> mknod input1 c 13 65
bash> mknod input2 c 13 66

bash> evtest input0
Event: time 970205127.333596, type 1 (Key),code 29 (LeftControl), value 1
Event: time 970205127.549600, type 1 (Key),code 29 (LeftControl), value 0
Event: time 970205127.997621, type 1 (Key),code 28 (Enter), value 1
Event: time 970205128.133621, type 1 (Key),code 28 (Enter), value 0
Event: time 970205128.805643, type 1 (Key),code 1 (Esc), value 1
Event: time 970205128.901644, type 1 (Key),code 1 (Esc), value 0
```

Info

[1] http://www.ics.mq.edu.au/~robbie/linux/itouch.html
[2] http://www.infosun.fmi.uni-passau.de/~nils/type6/
[3] http://www.lokigames.com/
[4] http://www.suse.cz/development/input
[5] http://www.pxh.de/fs/graphire/
[6] http://www.gtk.org/~otaylor/xinput/

■