Using Linux USB

Reading the Linux USB Device Filesystem output

The USB device filesystem is a dynamically generated filesystem that complements the normal device node system, and can be used to write user space device drivers. Writing of user space device drivers is covered in the programmer's section of this guide. In addition to the device nodes, there are two files that are also generated - the `drivers` and `devices` files. If you followed the instructions in the installation chapter, you should find them as `/proc/bus/usb/drivers` and `/proc/bus/usb/device` respectively. If the `/proc/bus/usb` directory is empty, you have not mounted the filesystem, or you have mounted it in the wrong location.

`/proc/bus/usb/drivers` just lists the currently registered drivers (even if the driver is not being used by any device). This is most useful when testing module installation, and checking for USB support in an unknown kernel. Here is an example of its use:

```
[bradh@rachel bradh]$ more /proc/bus/usb/drivers
hid
ov511
cpia
printer
hub
```

`/proc/bus/usb/devices` lists information about the devices currently attached to the USB bus. This is very useful when trying to figure out if the device is correctly enumerated. Here is an example of its use, showing the root hub, a hub, a mouse and a camera:

```
T:  Bus=00 Lev=00 Prnt=00 Port=00 Cnt=00 Dev#=  1 Spd=12  MxCh= 2
B:  Alloc= 28/900 us ( 3%), #Int=  2, #Iso=  0
D:  Ver= 1.00 Cls=09(hub  ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=0000 ProdID=0000 Rev= 0.00
C:* #Ifs= 1 Cfg#= 1 Atr=40 MxPwr=  0mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00 Driver=hub
E:  Ad=81(I) Atr=03(Int.) MxPS=   8 Ivl=255ms
T:  Bus=00 Lev=01 Prnt=01 Port=01 Cnt=01 Dev#=  2 Spd=12  MxCh= 4
D:  Ver= 1.00 Cls=09(hub  ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=0451 ProdID=1446 Rev= 1.00
C:* #Ifs= 1 Cfg#= 1 Atr=e0 MxPwr=100mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=09(hub  ) Sub=00 Prot=00 Driver=hub
E:  Ad=81(I) Atr=03(Int.) MxPS=   1 Ivl=255ms
T:  Bus=00 Lev=02 Prnt=02 Port=00 Cnt=01 Dev#=  3 Spd=12  MxCh= 0
D:  Ver= 1.00 Cls=00(>ifc ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=0553 ProdID=0002 Rev= 1.00
C:* #Ifs= 1 Cfg#= 1 Atr=80 MxPwr=400mA
I:  If#= 1 Alt= 0 #EPs= 1 Cls=ff(vend.) Sub=00 Prot=ff Driver=cpia
E:  Ad=81(I) Atr=01(Isoc) MxPS=   0 Ivl=  1ms
I:  If#= 1 Alt= 1 #EPs= 1 Cls=ff(vend.) Sub=00 Prot=ff Driver=cpia
E:  Ad=81(I) Atr=01(Isoc) MxPS= 448 Ivl=  1ms
I:  If#= 1 Alt= 2 #EPs= 1 Cls=ff(vend.) Sub=00 Prot=ff Driver=cpia
E:  Ad=81(I) Atr=01(Isoc) MxPS= 704 Ivl=  1ms
I:  If#= 1 Alt= 3 #EPs= 1 Cls=ff(vend.) Sub=00 Prot=ff Driver=cpia
E:  Ad=81(I) Atr=01(Isoc) MxPS= 960 Ivl=  1ms
T:  Bus=00 Lev=02 Prnt=02 Port=02 Cnt=02 Dev#=  5 Spd=1.5 MxCh= 0
D:  Ver= 1.00 Cls=00(>ifc ) Sub=00 Prot=00 MxPS= 8 #Cfgs=  1
P:  Vendor=046d ProdID=c001 Rev= 1.10
S:  Manufacturer=Logitech
S:  Product=USB-PS/2 Mouse
C:* #Ifs= 1 Cfg#= 1 Atr=a0 MxPwr= 50mA
I:  If#= 0 Alt= 0 #EPs= 1 Cls=03(HID  ) Sub=01 Prot=02 Driver=hid
E:  Ad=81(I) Atr=03(Int.) MxPS=   8 Ivl= 10ms
```

The information in the `/proc/bus/usb/devices` output is arranged in groups:

- The line that starts with `T:` is the topology. `Bus` indicates which bus the device is on. `Lev` indicates the level of the device, starting at level 00 for the root hub, level 01 for any device attached to the root hub, level 02 for devices attached to hubs at level 01, and so on. `Prnt` is the parent device for this device (always 00 for the root hub, and 01 for the devices attached to the root hub). `Port` is the port on the parent device, starting at 00 for the first port on each device. `Prnt/Port` is unique per bus. `Cnt` indicates what number device this is, at this level, based on the enumeration order within that level of the topology, starting at 01 for the first device. `Dev#` indicates what number device this is, irrespective of level, based on the bus enumeration order. This is unique per bus. `Spd` indicates what speed this device is running at, in Mbps (either 1.5 or 12 with the current version of USB). `MxCh` indicates how many devices can be connected to this device, and is 00 for anything except a hub. `Driver` indicates which device driver is being used for this device - an entry of `(none)` indicates that no driver is being used.

- The line that starts with `D:` is information from the device descriptor. `Ver` indicates which USB specification version the device claims to meet. `Cls` indicates which device class the device is claiming to meet, in both hexadecimal and as a string. A `Cls` entry of `00(>ifc)` indicates that the device class specification compliance is interface dependent, and the interface descriptor should be read for device class information. `Sub` indicates which sub-class (within the `Cls` entry), the device meets. `Prot` indicates which protocol within a class or sub-class the device claims to meet. `MxPS` indicates how big the packets from Endpoint 0 are. `#Cfgs` indicates how many configurations this device has.

- Much like `D:`, the line that starts with `P:` is information from the device descriptor, and is seperated mainly because it wouldn't all fit on one line. `Vendor` indicates the Vendor Identification code for the device, and `ProdID` indicates the Product Identification code for the device. `Rev` indicates the product revision number.

- Refer to the USB specification clause 9.7.1 for further information on device descriptors.

- The lines that start with `S:`, if any, are the vendor and product strings that the device returned.

- The line that starts with `C:` is information from the configuration descriptor - the number of `C:`lines per device is given by `#Cfgs`, and the entry followed by an asterisk is the current configuration. `#If` indicates how many interfaces the device has. `Cfg#` indicates which configuration is being described. `Atr` is a hexadecimal indication of the device attributes (0x80 for bus-powered, 0x40 for self-powered, 0x20 for remote wake-up capable). `MxPwr` is the maximum power draw for this device configuration, in milliamps. Refer to USB specification clause 9.7.2 for further information on configuration descriptors.

- The line that starts with `I:` is information from the interface descriptor - the number of `I:` lines per `C:` line is given by the `#Ifs` entry. `If#` indicates which interface is being described within a given device configuration. `Alt` indicates which alternate setting of this interface is being described. `#EPs` indicates how many endpoints there are within the alternate setting for this endpoint. `Cls` indicates which class the alternate setting of the interface corresponds to, in both hexadecimal and as a character string. `Sub` indicates which sub-class the alternate setting of the interface belongs to. `Prot` indicates which interface protocol (within a class and sub-class tuple) the alternate setting of the interface conforms to. `Driver` indicates which of the various USB drivers has claimed this interface. See USB specification clause 9.7.3 for further information.

- The line that starts with `E:` is information from the endpoint descriptor - the number of `E:` lines per `I:` line is given by the `#EPs` entry. Endpoint 0 is not displayed. `Ad` indicates the endpoint address, with a letter to indicate whether the endpoint is an In or Out endpoint. `Atr` indicate the attribute (transfer type) associated with the endpoint, followed by a string translating the transfer type. `MxPS` indicates the maximum packet size this endpoint is capable of sending or receiving, as appropriate. `Ivl` indicates the interval, in milliseconds, between polling of interrupt endpoints. `Ivl` is ignored for bulk and control transfers, and is set to 1 for isochronous transfers. See USB specification clause 9.7.4 for further information on endpoint descriptors.

Refer to `linux/Documentation/usb/proc_usb_info.txt` for more information on using the USB device filesystem information.