your position is our focus

Application Note

Topic :	**u-blox 5 and ANTARIS 4 Receivers with Linux via USB: How To**
	GPS.G4-CS-07059-A
Author :	Daniel Ammann
Date :	30/01/2008

This document explains how to use u-blox (u-blox 5 and ANTARIS®4) receivers with Linux.

1 Linux kernel versions and drivers

The exact methods for connecting and troubleshooting a USB connection to a u-blox[1] GPS receiver differ slightly between kernel versions and Linux distributions. The basic steps involved, and the drivers used are similar over a range of different Linux flavors. The recommendations in this document are based on using examples of Ubuntu 7.04 (Feisty Fawn), running a Kernel 2.6.20 SMP on an i686 CPU. This particular example uses the `cdc_acm` driver, found in 2.6 kernels. Alternatives are `acm` (on 2.4.x kernels) or `usbserial` (both 2.4.x and 2.6.x kernels).

It is assumed that the device file will be located directly in `/dev/ttyACM*b*`. Other distributions, kernel versions and drivers may use a different name (e.g an openwrt distribution using kernel 2.4.30 on a mips CPU will create `dev/tts/n` device files). When using usbserial, device files typically are located under `/dev/ttyUSBN`.

1.1 USB Technical Implementation

u-blox[1] USB interfaces implement Communications Device Class (CDC). The behavior and power is configured using the UBX protocol command `UBX-CFG-USB`. For more information consult the UBX protocol documentation in the Protocol Specifications ([1][2]).

1.2 RS232 ports

This document focuses on USB connections to u-blox[1] receivers. When using an RS232 connection instead of USB, no special drivers are needed. The device files for RS232 connections to be used are `/dev/ttyS0` or similar.

2 Installation Steps

1. Plug in a u-blox[1] receiver through the USB interface.
2. If correctly functioning, NMEA data should be readily available using `/dev/ttyACM0`. Direct the application to read from this device file. Writing to this will send data to the GPS receiver.
3. If the Linux GPS application uses the gpsd interface, start `gpsd -f /dev/ttyACM0`.

! Note The device number 0 in `/dev/ttyACM0` might be different or change when disconnecting or reconnecting.

[1] Receivers based upon u-blox 5 or ANTARIS®4 technology

3 Troubleshooting

The following assumes a 2.6.x kernel with cdc_acm as a loadable module. For 2.4.X kernels, the steps are similar, but use acm instead of cdc_acm.

If something doesn't work, consult `/var/log/messages` or wherever kernel messages go in the distribution (e.g. `/var/log/kernel, dmesg`, ...).

When successful, the output should be similar to the following:

```
kernel: usb 2-2: new full speed USB device using uhci_hcd and address 2
kernel: usb 2-2: configuration #1 chosen from 1 choice
kernel: cdc_acm 2-2:1.0: ttyACM0: USB ACM device
kernel: usbcore: registered new interface driver cdc_acm
kernel: drivers/usb/class/cdc-acm.c: v0.25:USB Abstract Control Model driver for USB modems and
ISDN adapters
```

3.1 USB device not recognized

In case the u-blox[1] GPS receiver does not work properly, use the `lsusb` command to see attached USB devices:

```
root:/var/log# lsusb
[...]
Bus 002 Device 002: ID 1546:01a5 U-Blox AG [2]
[...]
```

If this device cannot be found, see `/proc/bus/usb/devices` and/or check the USB cabling

3.2 USB device visible, but no device file available

The following instructions are true only if the driver is available as a loadable module (standard with most distributions). If the driver is compiled into the kernel, this step is not needed.

In case the GPS device shows up with `lsusb`, but it is still not possible to access the `/dev/ttyACMN` device, check whether the kernel module has been automatically loaded:

```
root:/var/log# lsmod|grep acm
cdc_acm 15904 0
usbcore 134280 4 cdc_acm,uhci_hcd,ehci_hcd
```

If `cdc_acm` is not in the list, load it using `insmod cdc_acm` (alternatively, use `modprobe` to automatically load missing dependent modules). If this returns a failure, check the kernel / distribution for `cdc_acm` support (most modern distributions of Linux are shipped with this module precompiled).

! Note If choosing to use the `usbserial` driver rather than the `acm/cdc_acm` driver, it is necessary to specify vendor/product ID: `insmod usbserial vendor=0x1546 product=0x01a4`

If `cdc_acm` loads correctly (as verified using the `lsmod` command), but still no device file exists in `/dev`, it is necessary to use `mknod` to create these device files. For `acm / cdc_acm`, Major `166` is used, for `usbserial`, Major `188`. In both cases, Minor is the device number, starting with zero:

Once everything is up and running, check that data is actually being received from the GPS receiver. This can be controlled by performing a `cat /dev/ttyACM0.` A continuous data stream should result, starting as follows:

```
$GPTXT,01,01,02,u-blox ag - www.u-blox.com*50
$GPTXT,01,01,02,ANTARIS ATR062x HW 00040001*2E
$GPTXT,01,01,02,EXT CORE 5.00 Sep 19 2006 16:26:36*68
$GPTXT,01,01,02,INT EXT0 (RCV) M4H1.1 Sep 19 2006 16:37:08*7F
$GPTXT,01,01,02,LIC 1EBF-BD07-E83D-6BE1-0F7A*50
$GPTXT,01,01,02,ANTSUPERV=AC SD PDoS *21
$GPTXT,01,01,02,ANTSTATUS=OK*3B
$GPRMC,155410.00,V,,,,,,,210607,,,N*7B
$GPVTG,,,,,,,,,N*30
```

[2] `u-blox 5: 1546:01a5`
`ANTARIS 4: 1546:01a4`

If user root can access the data stream, but a user application can't, make sure that the permissions of the corresponding `/dev/ttyACM0` are set correctly. If not, modify with `chmod` or change ownership using `chown`.

4 Applications

4.1 NMEA capable applications

Some GPS applications under Linux directly process the NMEA data stream. These should work correctly when pointed to the right device file.

4.2 gpsd

gpsd is an intermediate driver, sitting between the GPS receiver's hardware interface (typically the `/dev/ttyACM0` device) and an application. This offers the application a simplified interface to get GPS information.

gpsd was found to work very well together with u-blox[1] receivers, through USB and RS232, in NMEA and UBX protocols.

gpsd can be started as follows: `gpsd -f /dev/ttyACM0`

4.3 Applications using gpsd

Consult the gpsd homepage at http://gpsd.berlios.de/ for applications using the gpsd interface.

The following screenshot shows the gpsdrive application, using gpsd via USB from a u-blox[1] GPS receiver:

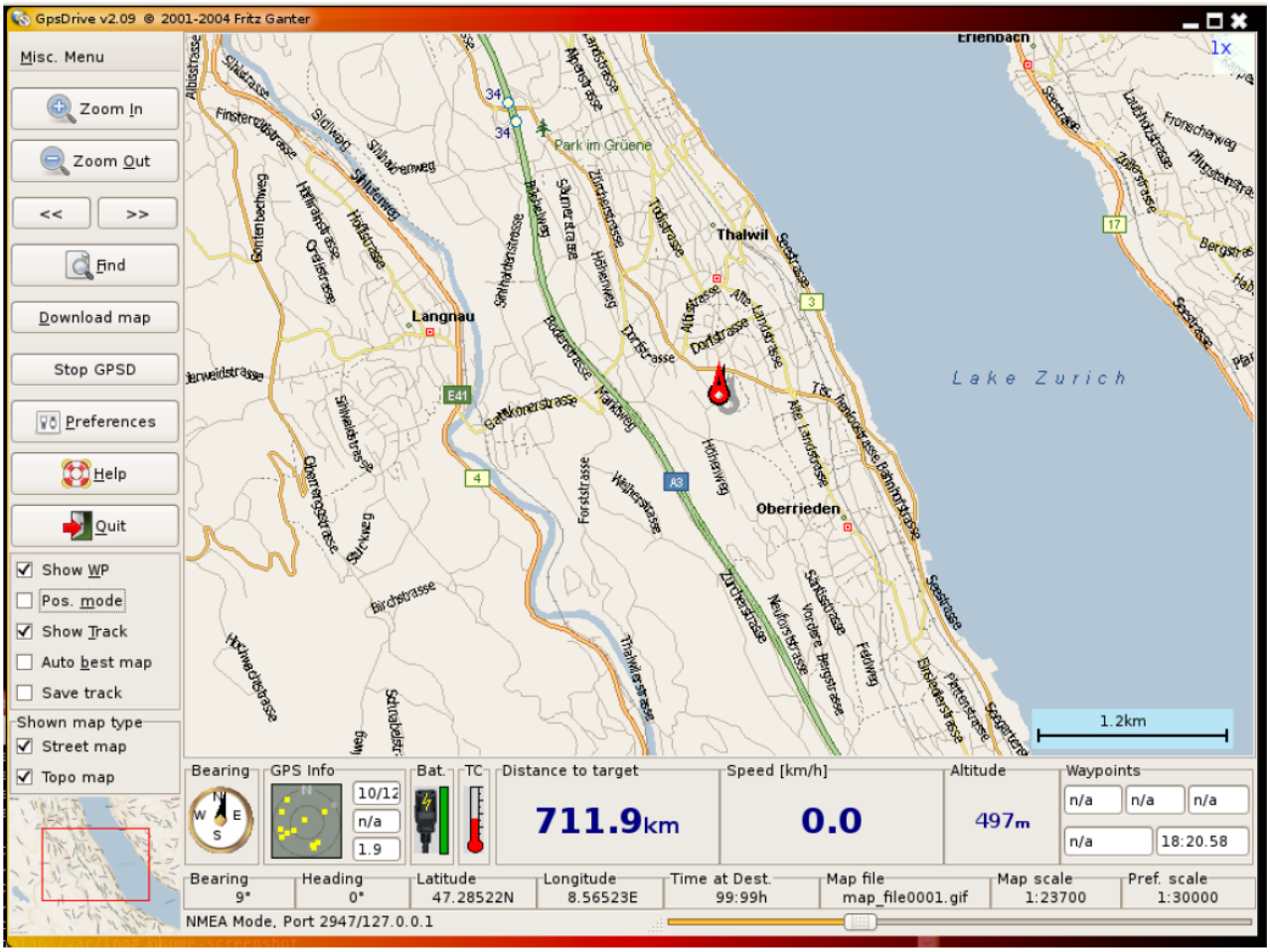

Figure 1: screenshot of gpsdrive

4.4 Additional Information

Consult the distribution manual on how to load kernel modules at startup.
The Linux kernel distribution also contains a wealth of information about the USB drivers. In the source code tree of the kernel used, begin by referring to the following files:

```
/usr/src/your kernel version/Documentation/usb/acm.txt
/usr/src/your kernel version/Documentation/devices.txt
/usr/src/your kernel version/Documentation/usb/usb-serial.txt
```

For gpsd, see the project's homepage at http://gpsd.berlios.de/.
For UBX- and NMEA protocol information on u-blox 5 and ANTARIS 4 receivers, see the Protocol Specifications ([1][2]).

Related Documents

[1] u-blox 5 Protocol Specification, Doc No GPS.G5-X-07036

[2] ANTARIS 4 Protocol Specification – CHM, Doc No GPS.G3-X-03002-D