Handspring−Visor with Linux mini−HOWTO

Ryan VanderBijl

<rvbijl-howto@vanderbijlfamily.com>

Revision History

Revision v0.6	January 2003	Revised by: rvbijl
Update my contact information. Remove references to outdated software. Removed PPP stuff. Updated hotplug section.		
Revision v0.5	December, 2000 – Feb, 2001	Revised by: rvbijl39
Seperate kernel/user−space parts. Updates for modules. Add usbdevfs info. Hotpluggable info. PPP info. Updates for the linux 2.4 kernel. Added troubleshooting tips. Change the SGML to DocBook format.		
Revision v0.4	August 7, 2000	Revised by: rvbijl39
Clarifications, updates, suggested by Matt Shook, Steven Coffman, Miles Lott, and info from Greg KH.		
Revision v0.3	April or May, 2000	Revised by: rvbijl39
SGML version. Some fixes from Miles Lott.		
Revision v0.1, v0.2	April 6, 2000	Revised by: rvbijl39
Original Version, HTML version.		

This document should give you instructions on how to connect your Handspring Visor (any version) to Linux, using USB.

Table of Contents

1. Standard Information

1.1. Disclaimer

No liability for the contents of this documents can be accepted. Use the concepts, examples and other content at your own risk. As this is a new edition of this document, there may be errors and inaccuracies, that may of course be damaging to your system. Proceed with caution, and although this is highly unlikely, the author(s) do not take any responsibility for that.

1.2. Author

This document was originally written, and is maintained (occasionally) by Ryan VanderBijl. Many people have made good suggestions for the improvement of this document. Many thanks to Miles Lott and especially to Greg Kroah−Hartman. Feedback is always welcome, although I can't promise if/when any suggestions will be incorporrated.

After being out−of−date for about a year and a half, I have finally updated my contact information. Also, very occasionally their may be a new copy of this document on my website. Test your luck at: http://www.vanderbijlfamily.com/~rvbijl/visor/howto/.

I would appreciate receiving an email from you if you get your Visor up and Sync'ed using this document.

1.3. Copyright

This document is Copyright (c) 2000−2002 by Ryan VanderBijl. You may distribute freely under the terms of the GPL (http://www.gnu.org/copyleft/gpl.html).

2. Introduction

2.1. Supported Devices

The following list of PDA's are supported by the Linux USB−Visor module:

- Handspring Visor (all)
- Palm USB Devices (all)
- Sony Clie (all)

2.2. What's needed?

To syncronize your Handspring Visor onto your linux computer, you need to configure linux to know how to do the low−level communication with your device, and then a a user−space program to do the actual communication with the device.

2.3. Linux Distributions

Many distibutions now come with kernels pre−configured with appropriate support. If you have such a distro, and do not wish to re−compile the kernel, then you can probably skip ahead to the Syncronziation Software section. The following is an incomplete list of distro's that have support pre−configured:

- Mandrake 8.2 (or later)
- Redhat 7.2 (or later)
- Suse 8.0 (or later)

You may check if your distro already has support by running this command as root:

```
modprobe visor
```

If the program printed any error messages, then you will need to re−compile the kernel with the approriate support. If nothing was printed, it means that it was successful, and your distro already provides support.

3. Configuring the Kernel

3.1. Requirements

Some obvious things are required, such as a computer, USB ports, Handspring Visor (with USB cradle). You should also know how to compile and install programs and the kernel. If you do not, you have a few options: learn, get your resident expert to help you, or get a distro with everything prepared for you.

If you don't have a USB controller for your computer, then you need to either buy the serial cradle for the Visor (but you wont need this document), or buy a USB−capable device for your computer (ie. add−on card or a new motherboard).

3.2. Kernel Version

The linux kernel version v2.4 was the first to have USB support. You can get the latest v2.4 kernel from: ftp://ftp.kernel.org/pub/linux/kernel/v2.4/.

The USB code has been backported to the v2.2 kernel, starting at version v2.2.18. If you must, you should probably be able to follow the same instructions with the latest v2.2 kernel

Now would be a good time to download the latest kernel, and configure things to your desire. The following has the instructions you need to configure USB.

If you have already have compiled and installed the kernel, you should not need to reboot! All we need to do is to compile and install the appropriate modules, and we will be all set.

3.3. USB Controller Type

The first thing to do is o determine which type of USB host controller you have. The USB host controller is the hardware in your computer which handles USB input/output.

Motherboards based on Intel chipsets, are typically UHCI controllers. Most addon cards are OHCI controllers. You can determine the type of USB hardware available by using, as root, the following command:

```
lspci −v
```

If you see an entry like:

```
USB Controller: ......
Flags: .....
I/O ports at ....
```

Then you have a UHCI based controller. If you see an entry like:

```
USB Controller: .....
Flags: ....
Memory at .....
```

Then you have a OHCI based controller. You could refer to http://www.linux−usb.org for further details. The kernel documentation `/usr/src/linux/Documentation/usb/usb.txt` may also be helpful in determining which type of controller you have.

3.4. Configure/Build Kernel

Now we need to configure, and make your kernel. During configuration, make sure you enable the following entries. You may either compile them directly into your kernel, or as modules. It is highly recommended that you compile them as modules. If you compile them directly in, you will need to reboot.

- USB support (*CONFIG_USB*)
- The preliminary USB Device Filesystem (*CONFIG_USB_DEVICEFS*)
- The appropriate controller − UHCI, or OHCI (*CONFIG_USB_UHCI*, or *CONFIG_USB_OHCI*)
- USB Serial Converter support (*CONFIG_USB_SERIAL*)
- USB Handspring Visor Driver (*CONFIG_USB_SERIAL_VISOR*) (serial converter's sub−option)

There are two UHCI drivers. You do not want the "UHCI Alternate (JE)" driver. This driver does not yet support all the USB features which the Visor uses. Thus, you will be unable to sync using this driver. If you are unable to see the original driver, ensure that the UHCI−JE driver is NOT selected, you should then see the option for both drivers.

Here, you should decide if you would like to include Hotplug support. With Hotplug support, you are able to auto−magically sync your visor by only pressing the Hotsync button. Please see the hotplug section, and especially the Hotplug Kernel Config, before choosing this route.

Compile and install as required. Don't forget to run lilo. If you are installing a new kernel image, do not reboot yet.

3.5. usbdevfs

usbdevfs puts information about your USB bus into the `/proc` directory tree. Its a good thing, and can be especially useful for debugging. You can enable it by adding the following line into your `/etc/fstab`.

```
none  /proc/bus/usb  usbdevfs  defaults  0  0
```

If you installed a new kernel image, you could reboot from this point on. Remember, if you are just adding the USB info as modules, you do not need to reboot.

3.6. Making `/dev` Entries

If your linux distro does not come with USB Visor support, or if you are NOT using *devfs* (not the same thing as *usbdevfs*), you will need to make the USB tty devices.

If you are using *devfs*, these devices are automagically created under `/dev/usb/tts/{0,1,...}`; so you can skip ahead to using the modules

If you are not using *devfs*, you can create the devices by executing these commands, as root:

```
mknod /dev/ttyUSB0 c 188 0
mknod /dev/ttyUSB1 c 188 1
mknod /dev/ttyUSB2 c 188 2
mknod /dev/ttyUSB3 c 188 3
etc...
chmod 666 /dev/ttyUSB*
```

You are able to have up to 255 connections/ports/devices, but unless you have more than one USB serial device, you'll probably only need the first few. The chmod is to allow users to be able to access the Visor device. It is the opinion of the author of this document that this should be safe for a personal computer. Multiuser computers may want to look into the security for this (please let me know).

When a Visor connects, there are two "ports" opened. (For most people, this will be `/dev/ttyUSB0`, and `/dev/ttyUSB1`). The first port (zero), is a generic connection. The second port is the hotsync port. This feature allows for future developments; for example, to export a filesystem from the Visor. A useful thing to do is to create a link to the hotsync port so that synchronization software will use the appropriate device be default. You can do this by:

```
cd /dev
ln −s /dev/ttyUSB1 pilot
ln −s /dev/ttyUSB1 palm
```

The software package pilot−xfer uses `/dev/pilot` by default. coldsync defaults to `/dev/palm`. Create devices and links as appropriate. Just for fun, you might also want to create a link from `/dev/visor` to `/dev/ttyUSB1`, just, well, because we have a visor, not a pilot (or Palm(tm)). The actual device number may change, depending on how many (active?) USB−serial devices you have on your system. A message containing the device actually used is entered into syslog. Eventually, the idea is to make an entry in the `/proc` filesystem which contains the needed information.

3.7. Using the Modules

For people who compiled the USB code as modules, you will also need to insert the modules into the running kernel. When you want to use the visor, you will need to runt the following commands as root:

```
/sbin/modprobe usb−uhci
/sbin/modprobe usb−ohci
/sbin/modprobe visor
```

One important note is that the actually driver/device connection for `/dev/ttyUSB*` are not created in the kernel memory until the hotsync button is pressed. Therefore, if you try to use any software before pressing the hotsync button, it won't work.

4. Synchronization Software

4.1. General

There are two software packages which communicate with your Visor. The first is coldsync, and the second is pilot−link. These programs are command−line based, and do "low−level" synchronization of the actual databases. They provide [different] interfaces for conduits. I assume you know how to download and install programs yourself. If not, learn. (I suppose you could cheat and install a pre−compiled package, but I don't support that ;−).

4.2. coldsync

coldsync can be found at http://coldsync.org/.

4.3. pilot−link

pilot−link can be found at http://www.pilot−link.org/. If you plan on using a Palm Desktop Equivalant, you'll want to use this package.

4.4. Time to Sync

At this point we should be able to test to make sure things are working. Of course, the visor needs to be in the cradle, and the cradle needs to be plugged into your USB port. The proper modules should be loaded.

NOTE: you MUST press the hotsync button BEFORE running the software. The visor driver will make an entry in syslog, and, eventually (ie, in newer drivers), to `/proc/drivers/visor`. Assuming the driver connects to `/dev/ttyUSB0` and `/dev/ttyUSB1`, you may run your program to backup the visor (chose the appropriate program):

```
cd ~
mkdir visorbackup
coldsync   -p /dev/ttyUSB1 -mb visorbackup
pilot-xfer -p /dev/ttyUSB1 -b  visorbackup
```

If you get an error such as "unable to bind to the port", you probably need to re−read the previous paragraph. With any other errors, please refer to the Troubleshooting Guide.

4.5. Palm Desktop Equivalents

There are a few Palm desktop equivalents. I haven't had time to test them. I intend, at some point, to put something about them in here. If you want to write up something to have it included here, please feel free to send it in. To my knowledge, all of them use the pilot−link libraries to talk with the Visor.

Here is the list that I know of:

- J−Pilot
- KPilot (KDE)

- Gnome−Pilot (GNOME)

4.6. Your Done!

At this point, you are done! You are able to install, backup, and syncronize your information. Congratulations! I would appreciate an email of gratitutde! The next sections talk about setting up a PPP connection and Hotplug support.

5. Hotplug−able

You must be using a fairly recent v2.4 kernel for this. You may wish to read over the whole section before proceeding.

Hotplug will allow you to automatically run your syncronization software when you press your sync button. You will only be able to configure this to sync for one Visor. If you need to be able to sync multiple users with differen Visors, you can't. (With different Palm's and Clie's, is a separate question, to which I don't know the answer to).

5.1. Hotplug Kernel Config

You will need to enable "Support for hot−pluggable devices" under "General Options" in the kernel config. This is the kernel option *CONFIG_HOTPLUG*. Make sure you compile/re−install.

5.2. Installing Hotplug

You need to get the hotplug scripts from the linux−hotplug homepage: http://linux−hotplug.sourceforge.net. The scripts are available as a rpm package or tarball If installing the tarball package, please read the README file that is included on proper installation procedures. If all you are configuring is the Visor hotplug support, then most likely you'll just have to do this:

```
            gzip -dc hotplug-2002_08_26.tar.gz | tar xvf -
            cp hotplug-2002_08_26/sbin/hotplug /sbin/hotplug
            cp -r hotplug-2002_08_26/etc/hotplug  /etc
```

5.3. Configure the Visor Driver

When all is said and done, an (executable) script named `/etc/hotplug/usb/visor` should get run when you press the hotsync button.

```
                #!/bin/sh
                /bin/su rvbijl -c /usr/local/bin/coldsync
```

Obviously, you would want to replace the *rvbijl39* with your normal user account. And make that script executable.

6. Troubleshooting Tips

6.1. The Tips

- There is a known bug about the Visor that sometimes you need to reset the Visor before it is able to sync properly. This is the number one solution for most people having problems syncing. If a soft reset does not work, and other USB devices work on this machine on Linux, a hard reset has been known to solve the problem. (Try some of the other suggestions before doing a hard reset).
- Did you install the new kernel/modules?
- Make sure that all the modules are running. The command is *lsmod* and there should be, at least, these modules: visor, usbserial, usbcore; and one of the following: usb−uhci or usb−ohci.
- If you installed a new kernel, did you reboot?
- Is the USB hardware enabled on your computer? (Check your BIOS. Does it work in other operating systems?)
- Are things (ie "usb−serial") appearing in `/proc/devices`? In `/proc/bus/usb/devices`? If not, then the drivers aren't being loaded properly.
- Is there any information being entered in syslog when you press the hotsync button?
- Did you really remember to press the hotsync button BEFORE running your synchronization script?
- Email `<usbvisor-unix@lists.sourceforge.net>`.

---

7. Links

You may find useful information at the following webpages. Some provided the information included in this document.

- ColdSync: http://coldsync.org
- Pilot−link: http://www.pilot−link.org/
- PalmOS HOWTO: http://www.orbits.com/Palm/
- Linux Kernel: ftp.kernel.org/pub/linux/kernel/
- USB Visor page: http://usbvisor.sourceforge.net
- Hotplug info: http://linux−hotplug.sourceforge.net/.
- Documentation to figure out type type of controller was leached from the kernel documentation, Documentation/usb/scanner.txt.