XFree Local Multi−User HOWTO

Svetoslav Slavtchev

`<svetoslav (at) users.sourcefourge.net>`

Aug 2004

Revision History

Revision 1.4	2004−08−14	Revised by: SS
add some info on the new IsolateDevice XFree/X.org, drop some stalled comments about ruby−2.6 (a lot more must be removed − linux−2.6 is table now)		
Revision 1.3	2003−11−11	Revised by: SS
Ruby−2.6 is working :) , sync to new BRuby, add config file paths for Gentoo		
Revision 1.2	2003−09−17	Revised by: SS
Lots, lots of stuff. the most major things: XFree−PrefBusID v3, input.rc (and exp. service ruby_init), add "Known Limitations"		
Revision 1.1	2003−07−14	Revised by: SS
sync input agent with examples, a bit more about ruby, less Fix−Me's, additions to video compatibility		
Revision 1.0	2003−05−13	Revised by: TP
Initial release, reviewed by LDP.		

This HOWTO explains one of the ways to get a working, multiple, local X user−capable PC system for up to 16 users. It is based on using a modified Linux kernel with support for multiple independent users . The second way is not covered here, but on the web page of it's author, the pioneer Miguel Freitas.

Table of Contents

Table of Contents

Table of Contents

Chapter 1. Introduction

1.1. About Backstreet Ruby/ Ruby

Backstreet Ruby is a kernel patch for the Linux kernel. It is a back port to Linux−2.4 of the *Ruby kernel tree*, which is developed by the Linux Console Project. The aim of the Linux Console developers is to enhance and reorganize the input, the console and the framebuffer subsystems in the Linux kernel, so they can work independent from each other and to allow multi−desktop operation. All this is done in the Ruby kernel tree which is based on the development Linux−2.5 kernel. The new Input subsystem and the new Framebuffer layer are already integrated in Linux−2.5 kernel, but as the main developer of the Linux Console Project, James Simmons, is too busy with completing the rewrite of the framebuffer layer in Linux−2.5, the multi−desktop operation will not be integrated in the next stable Linux kernel (Linux−2.6).

So Backstreet Ruby brings to the current stable Linux kernel (Linux−2.4) the enhanced input subsystem and the ability to use multiple graphic cards and multiple keyboards independently, in order to make multiple local XFree users on a single PC system possible.

You can have multiple independent graphic cards and multiple independent mice, but in order for multiple users to interact with the system, they do need independent keyboards as well. Multiple independent keyboards is the feature that Linux−2.4 (and in the future Linux−2.6) lacks, and this is what Backstreet Ruby adds to the stable Linux kernel Linux−2.4.

The entire work on back porting Ruby to Linux−2.4 is done by Aivils Stoss. `<Aivils.Stoss (at) unibanka.lv>`

Aivils got recently his hands on Ruby, and now Ruby is fully functional[1] too, so if you prefer the Linux−2.6 kernel you might use Ruby instead of Linux−2.4 + Backstreet Ruby.

Visit his web site for more information on the patch itself, on the current status, how to build a kernel using his patch or how to build modified XFree86 server.

You can find it here: http://startx.times.lv

There are also several mirrors

1. in the United States:

   http://people.debian.org/~andreas/aivils/
2. in Germany:

   http://www.schuldei.org/aivils/
3. in the United Kingdom:

   http://karlovo.demon.co.uk/~svetlio/aivils/

The address of the Linux Console Project is: http://linuxconsole.sf.net

1.2. About this document

This document explains how to configure your system for multiple local XFree users using the enchanted console/input subsystem in the Backstreet Ruby/ Ruby−2.6 kernel .

I will use :

- "Backstreet Ruby" or "BRuby" to refer to the back port to Linux 2.4
- "Ruby" or "Ruby−2.6" to refer to the original Ruby kernel tree for Linux 2.6

Every mention of Backstreet Ruby should be replaceable by Ruby/ Ruby−2.6 unless else mentioned.

Note

Currently it is not possible to set up systems for multiple console users.

There are two ways of setting up multiple local XFree users:

1. Modify the kernel to ignore input from USB keyboards and add the handling of USB keyboards to a modified Xserver. This solution was developed by Miguel Freitas. Visit his page on the topic at http://cambuca.ldhs.cetuc.puc−rio.br/multiuser/, for instructions on how to set up such a system.
2. Use the Backstreet Ruby kernel which supports independent keyboards.

I'll concentrate on configuring a system for multiple local XFree users using the Backstreet Ruby kernel, but there are parts which can be used also on a system using the solution from Miguel Freitas.

Note

This document is not intended to be a replacement of the existing documentation on the Backstreet Ruby home page (http://startx.times.lv), but rather, this is a HOWTO, explaining the way to a working X multi−user PC system. If you encounter any problems you'll probably need to consult the more detailed information there.

The document is based on the file system layout of the Mandrake−Linux distribution, but I tried to make it distribution−independent by including information about the differences to other mainstream distributions like Debian, Red Hat and SuSE Linux.

1.3. Related Documentation

- The Linux Console Project

  http://linuxconsole.sourceforge.net
- The Backstreet Ruby home page

  http://startx.times.lv/
- XFree with support for the new input layer by Zephaniah Hull

  (seems the patches are obsolated, and were removed from the site)

  http://people.debian.org/~warp/evdev/

- Miguel Freitas' page on multiple local XFree users

  http://cambuca.ldhs.cetuc.puc−rio.br/multiuser/
- Russian multi−terminal project Gorinich

  http://www.ctc.msiu.ru/zg/main.html
- Step by step instructions by Jean−Daniel Pauget

  http://disjunkt.com/dualhead/
- Multi−seat XFree solution under Linux with framebuffers, by Frode Trydal

  http://www.itsopen.net/projects/x−hack/

1.4. New versions of this document

You can find the latest stable version of this How−To at The Linux Documentation Project web site:

http://tldp.org/HOWTO/XFree−Local−multi−user−HOWTO/

and the latest unstable version :

http://karlovo.demon.co.uk/~svetlio/ruby−contrib/how−to/XFree_local_multi−user−HOWTO/

1.5. Copyright and License

This document, *XFree−Local−multi−user−HOWTO*, is copyrighted (c) 2003 by *Svetoslav Slavtchev*.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front−Cover Texts, and with no Back−Cover Texts. A copy of the license is available at http://www.gnu.org/copyleft/fdl.html.

Linux is a registered trademark of Linus Torvalds.

NVIDIA is a registered trademark of NVIDIA Corporation.

1.6. Disclaimer

No liability for the contents of this document can be accepted. Use the concepts, examples and information at your own risk. There may be errors and inaccuracies, that could be damaging to your system. Proceed with caution, and although this is highly unlikely, the author(s) do not take any responsibility.

All copyrights are held by their by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark. Naming of particular products or brands should not be seen as endorsements.

1.7. Credits/Contributors

In this document, I have the pleasure of acknowledging:

- James Simmons <jsimmons (at) transvirtual.com >

  for working so hard on Linux console project, for developing the new framebuffer, VT/console subsystem
- Vojtech Pavlik<vojtech (at) suse.cz>

  for rewriting the input subsystem and working hard on the Linux console project
- Aivils Stoss <Aivils.Stoss (at) unibanka.lv>

  for back porting Ruby to linux−2.4 and providing his back port and experiences to the world
- Andreas Schuldei <andreas (at) schuldei.org>

  for providing Debian packages, comments about Debian

1.8. Feedback/Bug Reporting

Feedback is most certainly welcome for this document. Send your additions, comments and criticisms to the following email address : <svetoslav (at) users.sourcefourge.net>.

In case you experiance troubles in configuring the system, feel free to contact me or the linuxcosnole mailing list.

Please send as much details as possible, the most important information would be (from a running Backstreet Ruby kernel):

- output from **dmesg**
- output from **lsmod**
- output from **cat /proc/bus/console/*/***
- contents of `/proc/bus/input/devices`
- contents of `/proc/bus/usb/devices`
- contents of the XFree configuration file(s) `/etc/X11/XF86Config(−4)`
- contents of the XFree86 log files `/var/log/XFree86.[n].log`

Chapter 2. Before we begin

2.1. Known Limitations

XFree does not support DRI acceleration on multiple graphic cards. The only way to get multiple accelerated X sessions is to use Nvidia's closed source driver and GL library or a single card using DRI and multiple cards using Nvidia's closed source drivers. XFree extensions not dependent on DRI should work.

Most of the multiheaded graphic cards can be used only for a single user/display. In order to start independent X servers the heads have to be explicitly addressable (which can not be done with most of the cards). It should be possible to use cards with explicitly addressable heads for independent displays, but this has not yet been tested.

In the following cases it might be possible to use a single multiheaded card for multiple independent displays :

- The card(s) have different PCI Bus ID's for the different heads

  (for example Matrox MMS G200/G450)
- The card(s) register frame buffer device for each head (only with the Linux−2.5/2.6 patch)

  Have in mind that the XFree frame buffer driver does not support acceleration and most of the XFree extensions.

  (for example Matrox G400DH, G450DH, G550DH)

Using/configuring independent devices for the independent screens/users is pretty hard or not explored. Exceptions are the input devices, graphic cards and sound thanks artsd. Some examples of such unexplored areas:

- External storage device (USB/Firewire hard disks, CD/DVD drives, ZIP/Floppy drives, memory sticks, ...
- USB/Firewire printers, scanners, cameras ...

In case you succeed in configuring such devices for independent usage by multiple users, please share with us how you did it.

Note

This doesn't mean that the devices won't work, but that every user will have access to all devices.

Currently under Backstreet Ruby/ Ruby you can have a maximum of 16 Virtual Terminals (8 for Backstreet Ruby released before 7 Oct 2003). This means that without hotplug configured if you are using USB keyboards with fancy keys you are limited to maximum of 8 independent users (for older versions Backstreet Ruby − 4 users). So do use hotplug if you are going to setup a system for more then 4 users (even systems with 2 attached USB keyboards benefit from using hotplug).

If you are using hotplug the maximal number of independent users is 16 for Backstreet Ruby / Ruby−2.6 and 8 for older versions Backstreet Ruby.

2.2. XFree configuration files

You should configure each of your video cards to work properly with a single X server, which is actually beyond the scope of this document. You should refer to the documentation that came with your distribution, but some general hints couldn't hurt.

The easiest way would be to use the same kind of monitors & video cards, you could then configure only the first card/monitor pair, make copies of this configuration file for the number of video cards you have, and then only adjust the BusID "PCI:x:xx:x" field in the configuration file. You can do this with the help of lspci, XFree86 −scanpci −verbose , or other similar distribution−specific tools.

You could use a similar approach if you have only monitors or video cards of the same type.

Most modern distributions also have advanced tools for easier configuration of Xinerama. You can use these tools to set up the system for Xinerama and then use this configuration file for generating the configuration files for the different X servers. You can use an example configuration file, replacing video card and monitor section, by the corresponding sections from the Xinerama `XFConfig−4` file.

Other useful resources:

- The Linux XFree86 HOWTO
- XFree86 Video Timings HOWTO
- X Window System Architecture Overview HOWTO
- The X Window User HOWTO

2.3. Reusing Xinerama configured XFree

If you have a system configured for Xinerama, you can easily adjust the XFree configuration file so you can use it for multiple users.

This will allow you to easily switch between a multi−user environment and a Xinerama multi−monitor environment.

What is Xinerama and how does the system configured using this HOWTO differ from a system using the Xinerama extensions in XFree?

The Xinerama extensions were introduced to the XFree86 system in version 4.0. Xinerama is an extension to XFree86 Release 6 Version 4.0 (X4.0) which allows applications and window managers to use the two (or more) physical displays as one large virtual display. In case Xinerama is not used, applications can only reside on one of the displays and can not be moved between the two. Window managers had to be specially written to support the two displays. With Xinerama, window managers and applications don't have to be specially written to support the larger "Virtual Desktop" Xinerama creates.

Just the opposite, the primary goal of a system configured according to this HOWTO is to offer multiple independent displays for several users on a single PC system.

For more information on Xinerama read:

- Xinerama−HOWTO, Using Xinerama to MultiHead XFree86 v.4.0+

2.4. Binary packages

Binary rpms of modified XFree servers are currently available for Mandrake 8.2/ 9/ 9.1/ 9.2, Red Hat 8/ 9, SuSE 8.1. If you're running other rpm−based distributions please help me to prepare and rebuild packages, so other users can get pre−compiled binaries. Currently the binary rpm packages are not mirrored and are only available from http://karlovo.demon.co.uk/~svetlio/ruby−contrib.

Binary packages for Debian Sid are also available thanks to Andreas Schuldei at http://www.schuldei.org/debian/bruby, or as apt repository "deb http://www.schuldei.org/debian/bruby ./ ".

Chapter 3. Installing the kernel

The installation of Ruby−2.6 is not fully covered, partly because there are almost no differences compared to the installation of Linux−2.6 kernel, partly because I have not gathered enough experience with Ruby−2.6 yet, so any comments and questions are welcomed.

If you are going to use Ruby−2.6, please do install Linux−2.6 without the Ruby−2.6 patch first and configure your system for it, in order to avoid tracking "Ruby−2.6 Bugs" which are actually due to a not properly configured system because of the Linux−2.4 −> Linux−2.6 changes. A good starting point is probably http://www.codemonkey.org.uk/docs/post−halloween−2.6.txt" .

3.1. Installing the Backstreet Ruby/ Ruby−2.6 kernel

Now it's time to install the kernel.

The easiest way would be to pull an already prepared binary kernel; there are packages for some distributions (currently only Mandrake and Debian) or a source package, and rebuild it on your system.

If for some reason you cannot use them or have problems using them you can also build your own kernel with the Backstreet Ruby/ Ruby−2.6 patch. For more information how to do this visit the Backstreet Ruby page on building and installing the kernel: http://startx.times.lv (or some of the mirrors) −> Documentation −> Quick Kernel.

(If you are new to Linux, reading "The Linux Kernel HOWTO", http://tldp.org/HOWTO/Kernel−HOWTO.html, could be very helpful.)

You can find binary kernel packages for Mandrake at http://karlovo.demon.co.uk/~svetlio/ruby−contrib/.

Debian binary kernel packages are available at http://www.schuldei.org/debian/bruby, or as apt repository "deb http://www.schuldei.org/debian/bruby ./ "

Note

- If you are going to use USB input devices, it's recommended to use the hid driver instead of usbkbd/ usbmouse.
- Upon loading the hid driver, you might get the following error message:

  ```
  [root@svetljo RPM]# modprobe hid
  modprobe: Can't locate module keybdev which is needed for hid
  [root@svetljo RPM]#
  ```

  The reason is that in some version of the module−utils package this dependency is hardcoded, but overrideable, so you should override it by adding "above hid usbcore" to your `/etc/modules.conf`

3.2. Notes on building your own kernel

There are some things I would like to mention, although I won't go in details, as the Backstreet Ruby page on compiling the kernel discusses this topic.

1. You have to follow this order:

```
Input support
Virtual Terminal support
Console drivers
```

   for all required options to be available/selectable.
2. You have to use built in input support:

```
Input device support --> Input devices (needed for keyboard, mouse,..)
Input device support --> Mouse support
```

3. I would suggest you also include at least one keyboard (built in − not as a module). You can also use modules, but I find it safer to be able to use a keyboard instead of trying to find a PC with ssh (or something similar) to load the required modules.

   For AT/PS2 keyboards, turn on (not modules):

```
Input device support --> Serial i/o support
Input device support --> i8042 PC Keyboard controller
Input device support --> Keyboards
Input device support --> AT keyboard support
```

   For a USB keyboard turn on (not modules):

```
Input device support --> Keyboards
USB support --> support for USB
USB support --> USB Host Controller Drivers
USB support --> USB Human Interface Device (full HID) support
USB support --> HID input layer support
```

4. If you are new to Linux, do not try to patch an already patched kernel (heavily patched kernels like the ones that ship with most distributions). Use a kernel from www.kernel.org, and take a look at the Linux Kernel HOWTO.

Note

In Backstreet Ruby (the patch for a 2.4 Linux kernel) does not support frame buffer devices , and for that reason is disabled.

In Ruby (the patch for 2.6 Linux kernel) if you want to disable/ change to modules support for PS2 input devices, you have to first activate/enable "General setup −−> Remove kernel features (for embedded systems)"

3.3. Creating needed device files

If you are not using the devfs file system, you might need to create several device files needed for the new input sub−system in the Backstreet Ruby kernel:

Note

Most current distributions should already provide the necessary device files, so try booting Backstreet Ruby without creating the device files and in case you don't miss input devices omit this section. Any distribution that came with XFree−4.3.0 and linux−2.4.20 should provide these device files.

```
cd /dev
mkdir input.old
mv mouse js? input.old
mkdir input
cd input
mknod js0 c 13 0
mknod js1 c 13 1
mknod js2 c 13 2
mknod js3 c 13 3
mknod mouse0 c 13 32
mknod mouse1 c 13 33
mknod mouse2 c 13 34
mknod mouse3 c 13 35
mknod mice c 13 63
mknod event0 c 13 64
mknod event1 c 13 65
mknod event2 c 13 66
mknod event3 c 13 67
cd ..
ln -s input/js0 js0
ln -s input/js1 js1
ln -s input/mice mouse
```

If you use devfs, all required devices will be created automatically by devfs.

Mandrake is an example of one distribution that uses devfs. Debian does not use devfs by default, but the kernel supports devfs; in order to activate devfs you have to add "devfs=mount" to the "append" line of your boot loader and install devfsd (the devfs demon). Distributions that do not use devfs are Red Hat and SuSE.

You can check whether devfs is used by issuing the following commands:

- To check whether support for devfs is enabled in your kernel

  cat /proc/filesystems | grep devfs
- To check whether devfs is used/mounted

  mount | grep devfs

If you get an empty string this means that devfs is not used; if you get something like the following output, devfs is activated:

```
[root@mc contrib]# cat /proc/filesystems | grep devfs
nodev   devfs
nodev   usbdevfs
[root@mc contrib]# mount | grep devfs
none on /proc/bus/usb type usbdevfs (rw)
none on /dev type devfs (rw)
[root@mc contrib]#
```

3.4. Notes on using multiple VT's & VGA console

As the frame buffer layer is not back−ported to Linux−2.4, only the primary graphic card is initialized during the boot process. Secondary graphic cards can only be initialized by an X server, so under Backstreet Ruby you will have a single VGA text console on the primary graphic card.

Ruby for Linux−2.6 supports framebuffer devices and single framebuffer console (which takes over the VGA console), but support for multiple VT's through framebuffer consoles is not yet ready.

To keep VGA console properly working it is important first to be started the XFree instance which will drive the graphic card which is used for VGA console(the graphic card defined as primary in BIOS).

3.5. Keyboard numbering(order of detection)

In the following chapters you will read about 1st keyboard, 2nd keyboard and so on, so here I will explain what is meant by n−th keyboard.

When a keyboard device is found, it is bound to a free VT (given that there are free VT's). The first keyboard found will be bound to VT0 (tty1−tty16), the second to VT1 (tty17), the third to VT2 (tty18).

Note

Older versions of the bruby patch (released before Oct 7 2003) use :

first keyboard found => VT0 (tty0−tty7)

second => VT1 (tty8−tty15)

third => VT2 (tty16−tty23)

The order of detecting the keyboards depends on the configuration of your kernel :

- If you are using kernel with integrated USB input the USB keyboard devices will be registered first, then the AT/PS2 keyboards will follow when the modules are loaded
- If you are using kernel with integrated PS2 input the AT/PS2 keyboard devices will be registered first, then the USB keyboards will follow when the modules are loaded
- If you are using kernel with integrated PS2 & USB input the AT/PS2 keyboard devices will be registered first, then the USB keyboards will follow

But there are some caveats:

Most USB keyboards represent themselves as more than one keyboard; it is common that the multimedia keys or the number−pad identify themselves as a different keyboard device. So if you are running a kernel with integrated USB input and have one USB keyboard with multimedia keys and one PS2 keyboard, the USB keyboard will be bound to VT0(real keyboard) and VT1(multimedia keys), the PS2 keyboard will be bound to VT2 (in case you have enough DUMB consoles).

There are several ways to work around these issues. Here I'll explain the easiest way to follow. It's definitely not the best one, but the shortest explanation, and I just want to make it clear to you that the problem is not that big. The better solutions will follow later in their own section.

All you need to do is to start the Backstreet Ruby/ Ruby kernel with dumbcon=n , where n is the sum of your AT/PS2 keyboards plus the sum of your USB keyboards multiplied by 2 (I suppose this is the maximum number of interfaces a USB keyboard registers), so all keyboards will be bound to a VT. Now you should find out which VT's the real keyboards are bound to (the keyboards excluding the multimedia keys) and start X using the appropriate tty ranges. Thanks to the proc interface integrated in Backstreet Ruby, you can easily find the assignment of keyboards to VT's. Each VT creates a file `/proc/bus/console/[n]/keyboard` (n is the number of the VT, for VT0 n will be 00, for VT1 − 01, ... , for VT11 − 11); reading this file will give you the assigned keyboard.

```
[root@svetljo root]# cat /proc/bus/console/*/*
usb−00:10.1−1.1/input0
usb−00:10.1−1.1/input1
isa0060/serio0/input0
```

tells us that:

- USB keyboard (real) is bound to VT0
- USB keyboard (multimedia keys) is bound to VT1
- PS2 keyboard is bound to VT2

Now we can start X on the VT's with real keyboards, in this case VT0 and VT2.

Of course in this simple example with only 2 keyboards (one USB and one PS2) the problem could be easily avoided by using a kernel with primary PS2 input support. The PS2 keyboard would be found first and bound to VT0, the USB keyboard would follow and it's real keyboard interface would be bound to VT1, so there is no need for additional dumb consoles (for the multimedia interfaces of USB keyboards).

Chapter 4. Setting up the X servers

Now its time to configure XFree.

4.1. Do I need a modified X server?

Note

For some video cards you can skip this part. Before installing the modified X server check the Video Compatibility list to determine whether you need one. Currently there are reports for working configurations without using a modified X server for Voodoo Graphics as primary and Voodoo3 or Nvidia TNT2 as secondary.

"Why should a modified X server be used?" – The reason is that XFree is designed to serve a single user and this design requires a single X server to drive all available graphic cards. So when an unmodified X server starts, it disables access to graphic cards for other X servers. Hence we have to modify XFree to make it possible more then one X server to run at the same time.

- "The experimental way": you can use the "hackvideo"(ignoring pci_disable XFree commands) feature of the Backstreet Ruby kernel. This will allow you to use the XFree server that came with your distribution (no need for installing modified XFree server).

  Q: "Why experimental?"

  A: Well, you have to find out whether it works with your combination of graphic cards. There are some combination that works flawlessly, but the majority of tested combinations have problems with this setup.
- "The surer way": you have to install XFree server modified with the Prefered Bus ID patch.

  Q: "Why surer?"

  A: Because it works with all "supported graphic cards", solves a lot of stability problems and makes it possible to use VGA console on the primary graphic card.

If you decide first to try without installing a modified X server, follow these steps:

1. To enable this feature you have to add this to your XFree configuration file:
   ```
   Section "ServerFlags"
   ...
   Option "PciOsConfig" "1"
   ...
   EndSection
   ```
2. and to inform the kernel to filter unnecessary PCI commands:
   ```
   [root@mc contrib]#echo  "1"> /proc/bus/pci/hackvideo
   ```
3. If you want this to be done automatically on every boot you have to add :
   ```
   if [ -x /proc/bus/pci/hackvideo ];then
         /bin/echo "1"> /proc/bus/pci/hackvideo
   fi
   ```
   to your init scripts, preferably somewhere at the end of /etc/rc.d/rc.sysinit (so the command is executed before X is started)

4. If you want to disable this functionality you have to:

```
[root@mc contrib]# echo "0"> /proc/bus/pci/hackvideo
```

Note

This functionality exists in the Backstreet Ruby kernel since 15. May 2003 and in Ruby−2.6 since 29. Sep 2003 , but will never be added to the official linux kernel as it is a small hack to spare you installing modified X server.

It is still recommended to install modified X server.

4.2. Installing and Configuring XFree−PrefBusID

1. Install the modified XFree server.

    ♦ Install an already built, but not packaged, modified X server and create the necessary symbolic links. You can get such binaries from the Backstreet Ruby home page, at http://startx.times.lv.
    ♦ Help us (as well other people using your distribution) in building an rpm or binary for your distribution (we lack systems installed with all available distributions, so we are not able to build packages for every distribution).
    ♦ To patch and rebuild XFree from source using the instructions on the Backstreet Ruby page. Go to the Documentation section, at http://startx.times.lv (or some of the mirrors) −> Documentation −> Quick XFree.
2. Find the BusID of your graphic cards

    Note

    For AGP cards, something similar to "1:0:0"

    For PCI cards, something similar to "0:xx:0"

    ♦ In most cases you will find the BusID already set in the device section of the XFree configuration file.

      (Virtually always in case XFree is configured for Xinerama.)
    ♦ If it is missing you can use **lspci**, **XFree86 −scanpci −verbose** or other similar tools that came with your distribution.

      With **lspci** look for "VGA compatible controller" or other similar tools that came with your distribution.

      ```
      root@svetljo mnt]# lspci | grep "VGA compatible controller"
      00:0d.0 VGA compatible controller: nVidia Corporation NV17 [GeForce4 MX 420] (rev a3)
      01:00.0 VGA compatible controller: ATI Technologies Inc Radeon RV200 QW [Radeon 7500]
      [root@svetljo mnt]#
      ```

      With **XFree86 −scanpci −verbose** , or in case XFree is already running **XFree86 :1 −scanpci −verbose** and look for your graphic cards:

      ```
      [root@svetljo mnt]# XFree86 :1 −scanpci −verbose
      ```

```
......
(0:13:0) unknown card (0x1462/0x8852) using a nVidia Corporation NV17 [GeForce4 MX 420]
......
(1:0:0) unknown card (0x1002/0x0f2a) using a ATI Technologies Inc Radeon RV200 QW [Radeon 7500]
```

3. Configure XFree−PrefBusID. You have the following choices with the same effect:

♦ Use the XFree config file option "SingleCard" , requires as argument a boolean value(true/false), added in patch version 3

For use in multi−user environment set to true, for standard XFree behavior(single X server allowed) to false or comment out/ delete the line.

This functionality is inlcuded the xorg−x11 packages for Mandrake and in Debian Sid XFree86 packages.

♦ Use the XFree config file option "PrefBusID" , requires as argument a valid BusID, added in patch version 2

For use in multi−user environment include the option with a valid BusID, for standard XFree behavior(single X server allowed) comment out or delete.

This functionality is inlcuded the xorg−x11 packages for Mandrake and in Debian Sid XFree86 packages, but the option name is changed to "IsolateDevice"

♦ Use the XFree command line option **−prefbusid x:x:x** , requires as argument a valid BusID, initial release of the patch

For use in multi−user environment pass the option with a valid BusID to XFree at start−up, for standard XFree behavior(single X server allowed) don't specify the option.

This functionality is inlcuded the xorg−x11 packages for Mandrake and in Debian Sid XFree86 packages, but the option name is changed to **−isolateDevice x:x:x**

Note

♦ For Mandrake and Debian users:

the XFree configuration files are normally /etc/X11/XF86Config−4

♦ For Red Hat, Gentoo, SuSE users:

the XFree configuration files are normally /etc/X11/XF86Config

♦ using the "SingleCard" option

```
Section "ServerLayout"
        Identifier     "X0"
        Screen      0  "Screen0" 0 0
        InputDevice    "Mouse0" "CorePointer"
        InputDevice    "Keyboard0" "CoreKeyboard"
        Option         "SingleCard" "true"
EndSection
Section "ServerLayout"
        Identifier     "X1"
        Screen      0  "Screen1" 0 0
        InputDevice    "Mouse1" "CorePointer"
```

```
        InputDevice    "Keyboard0" "CoreKeyboard"
        Option         "SingleCard" "true"
EndSection
```

Note

The BusID have to be specified in the "Device" Section of the XFree configuration file.

```
Section "Device"
    Identifier  "nv"
    VendorName  ""
    BoardName   ""
    Driver      "nvidia"
    # Clock lines

    # Uncomment following option if you see a big white block
    # instead of the cursor!
    #    Option      "sw_cursor"
    Option      "NoLogo" "On"
    BusID       "PCI:0:13:0"
EndSection
```

♦ using the "PrefBusID/IsolateDevice" option (requires as argument a valid BusID)

```
Section "ServerLayout"
        Identifier     "X0"
        Screen      0  "Screen0" 0 0
        InputDevice    "Mouse0" "CorePointer"
        InputDevice    "Keyboard0" "CoreKeyboard"
        Option "PrefBusID" "1:0:0"
EndSection
Section "ServerLayout"
        Identifier     "X1"
        Screen      0  "Screen1" 0 0
        InputDevice    "Mouse1" "CorePointer"
        InputDevice    "Keyboard0" "CoreKeyboard"
        Option "PrefBusID" "0:13:0"
EndSection
```

or for Debian Sid's XFree86 and Mandrake's xorg−x11

```
Section "ServerLayout"
        Identifier     "X0"
        Screen      0  "Screen0" 0 0
        InputDevice    "Mouse0" "CorePointer"
        InputDevice    "Keyboard0" "CoreKeyboard"
        Option "IsolateDevice" "1:0:0"
EndSection
Section "ServerLayout"
        Identifier     "X1"
        Screen      0  "Screen1" 0 0
        InputDevice    "Mouse1" "CorePointer"
        InputDevice    "Keyboard0" "CoreKeyboard"
        Option "IsolateDevice" "0:13:0"
EndSection
```

♦ using the "−prefbusid" option at XFree start−up (requires as argument a valid BusID)

or for Debian Sid's XFree86 and Mandrake's xorg−x11 use "−isolateDevice" instead

Caution

For the older(version 1) Preferred Bus ID XFree Server only this choice is valid.

For example on command line

```
[root@svetljo mnt]# startx −− /usr/X11R6/bin/X0 :0 −prefbusid 1:0:0 vt7
```

, or from a display manager (gdm):

```
.............................
# Definition of the standard X server.
[server−Standard]
name=Standard server
command=/usr/X11R6/bin/X0 :0 −layout first−Xserver −deferglyphs 16 −ac −prefbusid 1:0:0 vt7
flexible=true

[server−Second]
name=Second server
command=/usr/X11R6/bin/X1 :1 −layout second−Xserver −deferglyphs 16 −prefbusid 0:13:0 vt17
flexible=true
.............................
```

and for xdm/kdm

```
:0 local /bin/nice −n −10 /usr/X11R6/bin/X0 :0 −deferglyphs 16 −prefbusid 1:0:0 vt7
:1 local /bin/nice −n −10 /usr/X11R6/bin/X1 :1 −xf86config /etc/X11/XF86Config−4.TNT2 −prefbusid 0:13:0 vt17
```

4.3. Creating symbolic links

The symbolic links are needed for properly starting several XFree instances, as well for properly exiting an X session. This applies for both starting X from console and the automatic starting of X by the display manager (kdm, gdm, xdm).

You need to create as many symbolic links to the modified X server binary (or to the original X server in case you do not need a modified one), as the number of your video cards/X sessions.

I assume that you will have to use a modified X server, but in case you do not need it, use the following commands to create the links to your original X server:

```
cd /usr/X11R6/bin/
ln −s XFree[modified] X0
ln −s XFree[modified] X1
ln −s XFree[modified] X2
```

In case you use the provided rpm packages, you'll only need this if you want more than 4 parallel running X servers/X sessions, as the rpm creates 4 symbolic links to the X server binary.

4.4. Using independent keyboards with XFree

Once you install the Backstreet Ruby/ Ruby−2.6 kernel and start it with `dumbcon=n`, you get n +1 independent consoles [1 VGA(or Framebuffer under Ruby−2.6) + n DUMB] . If you have enough keyboards connected to your PC, each of these consoles are associated with a given keyboard. This enables you to start multiple X servers on each of the consoles, using the keyboard associated with the corresponding console for input. Hence you get multiple independent X servers with independent keyboards, which in turn make it possible for one single PC to be used by several local X users simultaneously.

To start X on a given console (using a given independent keyboard) you pass it the argument **vt[N]**, where N is a number from a given tty range.

- VGA(or Framebuffer for Ruby−2.6): tty1 to tty16
- DUMB1: tty17
- DUMB2: tty18

If you have 3 video cards, 3 keyboards, and you have started the Backstreet Ruby kernel with dumbcon=2, you can start 3 independent X servers for 3 simultaneous users with the following commands:

Note

- For Mandrake and Debian users:

  the XFree configuration files are normally /etc/X11/XF86Config−4
- For Red Hat, Gentoo, SuSE users:

  the XFree configuration files are normally /etc/X11/XF86Config

Caution

Have in mind that for the older Preferred Bus ID XFree Server (version 1) you have to specify the desired graphic card with parameter "**−prefbusid x:x:x**" where x:x:x is the Bus ID of the desired graphic card. Just append "−prefbusid x:x:x" with the correct Bus ID of the card you want to start right before the last argument vt[x]. If you omit this option the modified X server will act as a not modified/ standard XFree server and you wont be able to run multiple XFree instances at the same time.

For the 1st X server with the 1st keyboard:

$ startx −− /usr/X11R6/bin/X0 :0 −xf86config /etc/X11/XF86Config−4[for your 1st video card] vt7

For the 2nd X server with the 2nd keyboard:

$ startx −− /usr/X11R6/bin/X1 :1 −xf86config /etc/X11/XF86Config−4[for your 2nd video card] vt17

For the 3rd X server with the 3rd keyboard:

$ startx −− /usr/X11R6/bin/X2 :2 −xf86config /etc/X11/XF86Config−4[for your 3rd video card] vt18

For the 1st X server you can skip the `−xf86config /etc/X11/XF86Config−4[for your 1st video card]` argument. In this case, the default configuration file, `/etc/X11/XF86Config−4`, will be used.

Note

- VGA: tty0 to tty7

- DUMB1: tty8 to tty15
- DUMB2: tty16 to tty23

This means that you have to use different **vt** parameter for the additional X servers.

For the 2nd X server with the 2nd keyboard:

$ startx −− /usr/X11R6/bin/X1 :1 −xf86config /etc/X11/XF86Config−4[for your 2nd video card] vt8

For the 3rd X server with the 3rd keyboard:

$ startx −− /usr/X11R6/bin/X2 :2 −xf86config /etc/X11/XF86Config−4[for your 3rd video card] vt16

You can also setup your display manager to start the independent X servers, once everything is properly configured. But don't rush to setup your display manager before the configuration is finished, because this could give you serious problems. When you are ready with the required configurations, you'll reach the section on configuring the display manager.

4.5. Using independent mice with XFree

To use an independent mouse for each of your independent X servers/sessions, you just have to modify the input section of the XFree configuration files to point to the proper device files.

Use `/dev/input/mouse[n]`, where n is the number of your mouse starting from 0:

- 1st mouse −−> `/dev/input/mouse0`
- 2nd mouse −−> `/dev/input/mouse1`
- 3rd mouse −−> `/dev/input/mouse2`
- 4th mouse −−> `/dev/input/mouse3`

You shouldn't use `/dev/input/mice` because it merges the input from all mouse devices.

Here is my configuration before modifications:

```
−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−

# **********************************************************************
# Pointer section
# **********************************************************************

Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/psaux"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3−button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"
```

```
EndSection

-----------------------------------------------------------------------------------------------
```

After modifications for the first X server:

```
-----------------------------------------------------------------------------------------------

# **********************************************************************
# Pointer section
# **********************************************************************

Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse0"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3-button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection

----------------------------------------------------------------------------------------------
```

For the second X server:

```
----------------------------------------------------------------------------------------------
# **********************************************************************
# Pointer section
# **********************************************************************

Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse1"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3-button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection
----------------------------------------------------------------------------------------------
```

and so on ...

4.6. For graphic cards without DRI

(or reusing Xinerama configured XFree)

There could be several reasons for not using DRI:

- As far I know only one graphic card in a system can use DRI.
- The Nvidia closed source driver does not support DRI.

In case one of this reasons applies to your system, you do not need different XFree configuration files for the different displays.

You can configure your system for Xinerama using the tools provided with your distribution and reading The Xinerama−HOWTO, so when the system is used by a single user, he/she could switch to Xinerama desktop and use all available displays for a bigger desktop.

Once configured for Xinerama, only small additions are needed to achieve multiple independent desktops. All you have to do is to add new layouts which use single screen definition and have independent input devices (well, this is actually needed only for the mouse devices, as the keyboard is managed through the `vt[n]` option).

If you have configured Xinerama in the following way:

```
Section "ServerLayout"
    Identifier  "Simple Layout"
    Screen "Screen 2"
    Screen "Screen 1" RightOf "Screen 2"
    InputDevice "Mouse1" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection
```

To achieve multiple independent desktops you only have to add layout definitions for a single screen :

```
Section "ServerLayout"
    Identifier  "first-Xserver"
    Screen "Screen 1"
    InputDevice "Mouse1" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection

Section "ServerLayout"
    Identifier  "second-Xserver"
    Screen "Screen 2"
    InputDevice "Mouse2" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection
```

Which should result in these layout definitions:

```
Section "ServerLayout"
    Identifier  "Xinerama"
    Screen "Screen 2"
    Screen "Screen 1" RightOf "Screen 2"
    InputDevice "Mouse1" "CorePointer"
```

```
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection

Section "ServerLayout"
    Identifier  "first-Xserver"
    Screen "Screen 1"
    InputDevice "Mouse1" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection

Section "ServerLayout"
    Identifier  "second-Xserver"
    Screen "Screen 2"
    InputDevice "Mouse2" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection
```

Now you can start a single X server with option `−layout Xinerama` and enjoy the Xinerama desktop, or

You can start 2 independent X servers using `−layout first−Xserver` for the first, and `−layout second−Xserver` for the second.

Since you will use a single XFree configuration file for all X servers,

- in order to use independent keyboards you have to use following command:

  For 1st X server with the 1st keyboard:

  $ startx −− /usr/X11R6/bin/X0 :0 −layout first−Xserver vt7

  For 2nd X server with the 2nd keyboard:

  $ startx −− /usr/X11R6/bin/X1 :1 −layout second−Xserver vt17

  and if you want to use Xinerama:

  $ startx −− /usr/X11R6/bin/X −layout Xinerama vt7
- **Note**

  For older versions of Backstreet Ruby you have to use:

  For 1st X server with the 1st keyboard:

  $ startx −− /usr/X11R6/bin/X0 :0 −layout first−Xserver vt7

  For 2nd X server with the 2nd keyboard:

  $ startx −− /usr/X11R6/bin/X1 :1 −layout second−Xserver vt8
- also the mice must have different identifiers:

```
-------------------------------------------------------------------------------------------------

# **********************************************************************
# Pointer section
# **********************************************************************
```

```
Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse0"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3-button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection

Section "InputDevice"

    Identifier  "Mouse2"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse1"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3-button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection
-----------------------------------------------------------------------------------------------
```

4.7. Nvidia GLX & DRI

Note

Here you will learn how to configure your system for parallel use of Nvidia's GLX and XFree's DRI. If you do not have Nvidia cards, or you have only Nvidia cards, you do not need to read this section. In the first case you do not need the Nvidia GLX at all, and in the second, you can use the standard procedure for installing GLX.

Why Nvidia? Why closed source drivers?

A bit of a mixed up answer:

1. With the open source driver it's almost impossible to bring up a secondary card, so we should use the closed source driver.
2. Why the Nvidia card? Currently these are the only available, affordable PCI video cards with some acceleration.
3. I tried to use DRI on 2 parallel X servers, but it didn't work. I posted emails to XFree, DRI and lkml list, but I only got a single answer with no valuable information on my problem. I tried to run DRI on a Matrox G550 DH AGP & SiS63xx PCI, but when enabled for both cards, I got AGP errors. When enabled only for one of the cards, I got DRI up and running. Please, someone confirm or prove me wrong!

I'll explain several ways to get configuration working for both Nvidia GLX and XFree86 DRI. There are probably a lot of other possibilities, and maybe these are not the simplest, but they are the ones I know to work.

The reasons why this is needed:

1. Nvidia should use a different module path for xf86: the glx extension module from Nvidia is incompatible with the one from XFree86.
2. Nvidia should use a different XF86Config file: because DRI should be disabled for Nvidia and enabled for others.

If you find a simpler way, please email it me and I'll include it.

Caution

This can not be used as−is on SuSE Linux. In order to make it easy for the user to switch between Mesa, XFree and Nvidia GL libraries, SuSE uses a very complicated setup for the GL libraries. To use this setup you have to switch your configuration to XFree86's GL libraries.

4.7.1. Example 1

This is the configuration that I use on my system (ATI AIW Radeon 7500 AGP and Nvidia TNT2 M64 PCI) XFree configuration files:

1. Create a directory `/usr/X11R6/libNV`:

   mkdir /usr/X11R6/libNV
2. Create links to the original `/usr/X11R6/lib`:

   cd /usr/X11R6/libNV

   lndir −silent −ignorelinks ../lib
3. Install the Nvidia driver and libraries in `/usr/X11R6/libNV`.
4. Install Nvidia's `libGLcore.so.1.0` [driver version], or better, `libGLcore.so.1`, in `/usr/lib`. Make a symbolic link from `/usr/X11R6/libNV/libGLcore.so.1` to `/usr/lib/libGLcore.so.1` (this will allow you to easily update your Nvidia drivers):

   cd /usr/lib

   ln −s ../X11R6/libNV/libGLcore.so.1 ./

   Note: the Nvidia `libGL.so` is installed `/usr/X11R6/libNV`, so it's invisible to the system unless you tell the system about the existence of `/usr/X11R6/libNV`. For this setup, you must not do this, as it will break the standard X server start−up. But you can use the XFree GL libraries with the Nvidia graphic card and Nvidia closed source drivers, with a non−Nvidia graphic card, using XFree's DRI, which the GL library from Nvidia cannot do.
5. Add a line in the XFree configuration file for the Nvidia card to point the X server to the right location of the library and module path:

```
Section "Files"

..........
ModulePath  "/usr/X11R6/libNV/modules"
..........

EndSection
```

6. Install the Nvidia kernel driver.

Now everything should be fine and you should be able to use DRI and Nvidia GLX at the same time. You will have a bit smaller performance in comparison to a setup which uses Nvidia's libGL & libGLcore, but the difference is not that big on my PC.

4.7.2. Example 2

This example will give you the full performance of both the Nvidia card(s), and the non−Nvidia card, since XFree's libGL is used for the non Nvidia card, and Nvidia's libGL is used for Nvidia cards. But this will require one more X server to be precise; a simple wrapper to add the path to the Nvidia libraries, and symbolic links to it for additional Nvidia cards.

It is almost the same as the previous scenario, with the difference that the X servers for the Nvidia cards should start with an environment where Nvidia's libGL is known, while the X servers for non Nvidia cards shouldn't know anything about the Nvidia libGL. This requires a wrapper to be used for starting the X servers driving Nvidia cards.

Install the Nvidia libraries and kernel driver like in the previous example. You may skip step 4. as `libGLcore.so.1` is installed in `/usr/X11R6/libNV`, and we'll inform the X servers driving Nvidia cards about the proper path to the Nvidia libraries.

The missing part − the wrapper :

```
#!/bin/bash
export LD_LIBRARY_PATH=/usr/X11R6/libNV
exec /usr/X11R6/bin/X0 $*
```

Copy these lines into your favourite editor and save the file as `XNV`. Make it executable:

chmod +x XNV

Copy the file to `/usr/X11R6/bin` and make symbolic links to it for additional Nvidia cards (for additional cards just add more links):

```
cp XNV /usr/X11R6/bin
cd /usr/X11R6/bin
ln −s XNV Xnv0
ln −s XNV Xnv1
ln −s XNV Xnv2
```

Remember to use `/usr/X11R6/bin/Xnv0`, `/usr/X11R6/bin/Xnv1` ..., instead of `/usr/X11R6/bin/X0`, `/usr/X11R6/bin/X1` ... for your Nvidia cards while configuring the display managers in the next chapter, or when starting X on Nvidia card(s) from console.

4.7.3. Installing the Nvidia libraries easily

Using the new Nvidia installer (note, this is a work in progress, do not use if you don't understand what happens here. To−do: write a script to perform steps 1−4. Please provide some feedback on the script in

Appendix−>Scripts):

Manually:

1. Make a backup of your XFree GL libraries:
   ```
   cd  [XFree prefix]
   ```
   On my Mandrake system I can use:
   ```
   cd $OPENWINHOME
   find lib −name "libGL.*" −o −name "libGLcore*" −o −name "libglx.*" | xargs tar  rpfv   libGL−backup.tar
   ```
2. Then run the Nvidia installer:
   ```
   ./NVIDIA−Linux−x86−1.0−4349.run −−no−opengl−headers −−xfree86−prefix=/usr/X11R6NV −−opengl−prefix=/usr/X11R6NV
   ```
3. Copy the installed files to `/usr/X11R6/libNV`:
   ```
   cd /usr/X11R6NV/lib &&  tar cv * | tar xvC /usr/X11R6/libNV/
   ```
4. Restore the backed−up GL libraries:
   ```
   cd [XFree prefix]
   tar xvfp libGL−backup.tar && ldconfig
   ```

Chapter 5. More on configuring input devices

Here you will find more details on configuring input devices and dealing with secondary keyboard interfaces found in USB multimedia keyboards.

If you feel comfortable under Linux, please review and test the experimental service ruby_init. The service, the configuration file and the README can be found in Appendix D. Any kind of feedback is highly appreciated. (Please, please drop a line with comments. :−))

Note

If you are configuring a system with two displays(2 keyboards, 2 mice) you probably can skip to Chapter 6 "Configuring display managers", but if you want to use a single system for more users you will find really useful information in this chapter.

5.1. Finding the real devices

We will need this information later on, to be able to assign a given keyboard/mouse to a given X−server/Display.

To find the PHYS ID's (the addresses) or the name(quite oft it differs from the one labeled on the device) of your input devices you have to read the file `/proc/bus/input/devices`.

Here is what I have:

```
[svetljo@svetljo How−To]# cat /proc/bus/input/devices
I: Bus=0011 Vendor=0002 Product=0002 Version=0034
N: Name="PS2++ Logitech Wheel Mouse"
P: Phys=isa0060/serio1/input0
H: Handlers=mouse0 ts0
B: EV=7
B: KEY=f0000 0 0 0 0 0 0 0 0
B: REL=103

I: Bus=0011 Vendor=0001 Product=0002 Version=ab02
N: Name="AT Set 2 keyboard"
P: Phys=isa0060/serio0/input0
H: Handlers=kbd
B: EV=120003
B: KEY=4 2000000 8061f9 fbc9d621 efdfffdf ffefffff ffffffff fffffffe
B: LED=7

I: Bus=0003 Vendor=046d Product=c303 Version=0700
N: Name="Logitech    USB Keyboard"
P: Phys=usb−00:10.1−1.1/input0
H: Handlers=kbd
B: EV=120003
B: KEY=10000 7f ffe00000 7ff ffbeffdf ffffffff ffffffff fffffffe
B: LED=7

I: Bus=0003 Vendor=046d Product=c303 Version=0700
N: Name="Logitech    USB Keyboard"
P: Phys=usb−00:10.1−1.1/input1
H: Handlers=kbd
B: EV=100003
```

```
B: KEY=1078 1800d100 1e0000 0 0 0

I: Bus=0003 Vendor=05fe Product=0011 Version=0000
N: Name="Cypress Sem. PS2/USB Browser Combo Mouse"
P: Phys=usb−00:10.1−1.2/input0
H: Handlers=mouse1 ts1
B: EV=7
B: KEY=1f0000 0 0 0 0 0 0 0 0
B: REL=103
```

Note

- `/proc/bus/input/devices` will provide the needed information for all devices except USB multimedia/office keyboards.
- For such USB multimedia/office keyboards you will have to gather additional information, for example with the help of **lsusb**.
- Under Ruby−2.6 for usb devices it will look like "usb−0000:00:10.x" not "usb−00:10.x"

- First we have to find the address of the USB keyboard:

```
[root@svetljo How−To]# lsusb
Bus 004 Device 001: ID 0000:0000
Bus 003 Device 001: ID 0000:0000
Bus 003 Device 002: ID 0409:55ab NEC Corp. Hub [iMac kbd]
Bus 003 Device 003: ID 046d:c303 Logitech, Inc.
Bus 003 Device 004: ID 05fe:0011 Chic Technology Corp. Browser Mouse
Bus 002 Device 001: ID 0000:0000
Bus 001 Device 001: ID 0000:0000
```

  Here, my USB Logitech keyboard is Device 003 on Bus 003.
- Now we run **lsusb** with arguments *`−v −s [your USB keyboard device id in form Bus:Device]`*, in my case, **lsusb −v −s 003:003**.

```
........
    Interface Descriptor:
      bLength                 9
      bDescriptorType         4
      bInterfaceNumber        0
      bAlternateSetting       0
      bNumEndpoints           1
      bInterfaceClass         3 Human Interface Devices
      bInterfaceSubClass      1 Boot Interface Subclass
      bInterfaceProtocol      1 Keyboard
      iInterface              0
........

    Interface Descriptor:
      bLength                 9
      bDescriptorType         4
      bInterfaceNumber        1
      bAlternateSetting       0
      bNumEndpoints           1
      bInterfaceClass         3 Human Interface Devices
      bInterfaceSubClass      0 No Subclass
      bInterfaceProtocol      0 None
      iInterface              0
........
```

So my USB keyboard has two interfaces (see bInterfaceNumber); the first one is the real keyboard (bInterfaceProtocol 1 Keyboard), the second (bInterfaceProtocol 0 None) – the additional keys. Hence the real

USB keyboard is:

```
.....
N: Name="Logitech    USB Keyboard"
P: Phys=usb-00:10.1-1.1/input0
H: Handlers=kbd
.....
```

The "P: Phys=" field (the physical descriptor/address) consorts of:

1. Bus type: "usb"
2. PCI function of the USB controller: "00:10.1 " (for Ruby−2.6 "0000:00:10.1")
3. USB device id: "1.1"
4. The string: "/input"
5. Interface number: "0"

5.2. Using hotplug with `input.agent` and `input.rc`

Using the input.agent will allow you to manage input devices based on their PHYS ID.

The input.rc script will run the input.agent with appropriate arguments for drivers that are built in the kernel or are loaded before hotplug is available.

5.2.1. `input.agent`

The input agent uses 3 configuration files:

- `/etc/hotplug/kbd.conf`
- `/etc/hotplug/mouse.conf`
- `/etc/hotplug/event.conf`

Note

The explanations here are using Backstreet Ruby (usb devices have PHYS_ID "usb−00:xx.x−..."), if you are running Ruby−2.6 for usb devices you should have PHYS_ID "usb−0000:00:xx.x−...." , but you should be able to use the same configuration files both under Backstreet Ruby and Ruby−2.6 by specifying "usb−0*:xx.x−..." .

To configure the keyboards you have to adjust `/etc/hotplug/kbd.conf`

If I wanted to use the PS2 keyboard for the primary Display and for the VGA console, I would have:

```
#
# keyboard configuration
#
# vt_name device_physicaly_location

VT0     isa0060/serio0/input0
VT1     usb-00:10.1-1.1/input0
```

Or, if I want to use the USB keyboard for the primary Display and for the VGA console:

```
#
# keyboard configuration
#
# vt_name device_physicaly_location

VT0     usb-00:10.1-1.1/input0
VT1     isa0060/serio0/input0
```

you could also use "*.*" instead of the pci function of the USB controller:

```
#
# keyboard configuration
#
# vt_name device_physicaly_location

VT0     usb-*.*-1.1/input0
VT1     isa0060/serio0/input0
```

For mouse devices you have to edit `/etc/hotplug/mouse.conf`

```
#
# mouse device configuration
#
# sym_link device_physicaly_location
mouse0br        usb-00:10.1-1.2/input0
mouse1br        usb-*.*-2.7.*/input0
mouse2br        isa0060/serio1/*
```

and adjust the XFree configuration file.

For the first mouse change

```
......
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse0"
    Option "ZAxisMapping" "4 5"
......
```

to :

```
......
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse0br"
    Option "ZAxisMapping" "4 5"
......
```

For the second mouse change

```
......
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse1"
    Option "ZAxisMapping" "4 5"
```

```
......
```

to :

```
......
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse1br"
    Option "ZAxisMapping" "4 5"
......
```

and so on.

For Event devices edit `/etc/hotplug/event.conf`

```
#
# input event device config file
#
# symbolic_link device_physicaly_location
event0br        isa0060/serio0/*
event1br        isa0060/serio1/input0
event2br        usb-*.*-3/input0
```

and configure the applications which use them to use the symbolic links instead of the real devices

5.2.2. `input.rc`

If you are using input drivers built into the kernel please ensure that the `input.rc` script is started/executed at system start. In case the script is not executed you will find that hotplug could not configure these input devices.

Note

This area requires user feedback. Currently I have information only about Mandrake, where hotplug is not run as service.

If your distribution runs hotplug as service this will be done automatically.

If your distribution does not run hotplug as service you will have to modify your init scripts to run `input.rc` for you before XFree is started.

You may add this at the end of your `/etc/rc.d/rc.sysinit`

```
if [[ -f /proc/bus/console -o -n tmp=`uname -r | sed -n 's:ruby::p'` ]]; then
   if [ -x /etc/hotplug/input.rc ]; then
      /etc/hotplug/input.rc start
      if [ $? = 0 ]; then
        action "Configuring cold plugged input devices" /bin/true
      else
        action "Configuring cold plugged input devices" /bin/false
      fi
   else if [ -f /etc/hotplug/input.rc ]; then
           action "Input: input.rc installed, but not executable. Please check the file permissions." /bin/false
```

```
        else
           action "Input: Failed to configure cold plugged devices - input.rc missing"  /bin/false
        fi
   fi
fi
```

5.3. Using XFree with event interface support

This will allow you:

- if you have input devices with different names, to use them with the same Xserver/screen wherever you plug or re−plug them.
- if you have input devices with the same names, to use them with the same Xserver/screen according to the USB port where you plug or re−plug them.
- the ability to use wild cards such as "*" and "?".

Caution

Currently hot−plugging doesn't seems to work properly. I have reports that it works when using the "Dev Name" option, but my primary purpose was to get it working with "Dev Phys" and this does not currently seem to work. "Why "Dev Phys"?" − because if one wants to setup a single system for 4,5 or more users it would be easier to get 4,5 or more pieces of the same keyboard/ mouse then to find the same number keyboards or mice but from different manufacturer or with different names, and i find configuring XFree for such number users is simpler when "Dev Phys" is used.

Warning

This section is somewhat abandoned and is not tested under Ruby−2.6 as it didn't work reliably for me.

For this to work you will have to use XFree with the patches for event interface support, developed by Zephaniah Hull. You can find them at the following url: http://people.debian.org/~warp/evdev/.

To build from source you will need the following patches :

- `029_lnx_evdev.diff` : The evdev core patch.
- `030_lnx_evdev_mouse.diff` : The mouse side of the patch.
- `031_lnx_evdev_keyboard.diff` : The keyboard side of the patch.

The binaries for Debian include these patches.

For hot−plugging you will also need the `/etc/hotplug/input.agent` , which you can find under the above address and in Appendix Scripts

Then you have to configure XFree to use the event devices.

The configuration section for a mouse should look something like this:

```
Section "InputDevice"
    Identifier  "Mouse1"
    Driver      "mouse"
    Option      "Protocol"      "evdev"
    Option      "Dev Name"      "A4Tech USB Optical Mouse"
```

```
    Option      "Dev Phys"      "usb-*/input0"
    Option      "Buttons"       "9"

    Option      "ZAxisMapping"  "6 7 8 9"
EndSection
```

The configuration section for a keyboard should look something like this:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "kbd"
    Option      "Protocol"      "evdev"
    Option      "Dev Name"      "SILITEK USB Keyboard"
    Option      "Dev Phys"      "usb-*/input0"
    Option      "AutoRepeat"    "250 30"
    Option      "XkbRules"      "xfree86"
    Option      "XkbModel"      "pc101"
    Option      "XkbLayout"     "dvorak"
EndSection
```

For Dev Name and Dev Phys, the wildcats "?" and "*" work, you MUST have at least one of the two, if you have both then the device must match on both, a non−existent entry is the same as one consisting of "*".

5.4. Using the "Phys" descriptor and USB devices

Using the "Phys" descriptor of input devices simplifies a lot the configuration of input devices in XFree, especially when a bigger number of displays are used.

As USB devices are connecting in a tree form, you can really easy specify the way keyboard and mice devices are bound to a specified X display. You have to use one USB hub with number of ports equal(or bigger) to the number of the X displays, to this hub are connected smaller (2−4 port) hubs (or keyboards with integrated hub). To the first port of the smaller (integrated) hub are connected the keyboards, to the second the mice (in case there are free ports you can connect usb−audio devices to them :)). This results in the following layout of the usb−id's in case the primary USB hub is the first USB device :

Note

In the following explanations and examples I use for first device on the secondary(integrated) hub keyboard device because my keyboard is internally connected to the 1st port of the integrated hub. I assume this will apply for most of the keyboards with integrated hub, but in case the one you own uses different port you will have to make small adjustments.

- on the 1st port of the primary hub
    - 1.1 USB hub (integrated)
    - 1.1.1 USB keyboard
    - 1.1.2 USB mouse
    - (1.1.3 usb−audio/other usb device)
- on the 2nd port
    - 1.2 USB hub (integrated)
    - 1.2.1 USB keyboard
    - 1.2.2 USB mouse

    - (1.2.3 usb−audio/other usb device)
- on the 3rd port
    - 1.3 USB hub (integrated)
    - 1.3.1 USB keyboard
    - 1.3.2 USB mouse
    - (1.3.3 usb−audio/other usb device)
- on the 4th port
    - 1.4 USB hub (integrated)
    - 1.4.1 USB keyboard
    - 1.4.2 USB mouse
    - (1.4.3 usb−audio/other usb device)

Based on this we can bind all devices connected to a specified USB port to a given X server.

5.4.1. ... with Input Agent

An example for a 4−user system using the "Phys" descriptor with Input Agent and USB input devices.

Note

Here the PCI function of the USB controller is masked by *.* so it should work both under Backstreet Ruby and Ruby−2.6 . If you use more then one USB controller you'll have to use different mask (for example "usb−0*:xx.x−...") or specify the full PHYS_ID's : .

For Backstreet Ruby you should have PHYS_ID "usb−00:xx.x−..."

For Ruby−2.6 you should have PHYS_ID "usb−0000:00:xx.x−...."

We'll use the "**vt[n]**" parameter when starting X and the following configuration file for the keyboards(`/etc/hotplug/kbd.conf`):

```
#
# keyboard configuration
#
# vt_name device_physicaly_location

VT0     usb-*.*-1.1.1/input0
VT1     usb-*.*-1.2.1/input0
VT2     usb-*.*-1.3.1/input0
VT3     usb-*.*-1.4.1/input0
```

For mouse devices the configuration file (`/etc/hotplug/mouse.conf`) will look like this:

```
#
# mouse device configuration
#
# sym_link device_physicaly_location
mouse0br        usb-*.*-1.1.2/input0
mouse1br        usb-*.*-1.2.2/input0
mouse2br        usb-*.*-1.3.2/input0
mouse3br        usb-*.*-1.4.2/input0
```

and we have to adjust the XFree configuration files, so XFree uses the symbolic links instead of the actual devices. If you already configured independent mice you have only to append "br" to each of the mouse devices.

Change each "/dev/input/mouse[n]" to "/dev/input/mouse[n]br".

For the first display:

```
......
# **********************************************************************
# Pointer section
# **********************************************************************

Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse0br"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3−button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection
......
```

For the second display:

```
......
# **********************************************************************
# Pointer section
# **********************************************************************

Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse1br"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3−button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection
......
```

and so on.

Or in case a single XFree configuration file is used:

```
......
# **********************************************************************
# Pointer section
# **********************************************************************

Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse0br"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3-button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection
Section "InputDevice"

    Identifier  "Mouse2"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
    Option "Device"      "/dev/input/mouse1br"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3-button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection
......
```

5.4.2. ... with XFree with event interface support

Using the "Dev Phys" option of XFree with event device support and USB input devices enables us to use almost identical configuration of the input devices for all X servers. The only difference will be in the part of the usb−id, which reflects the port of the primary USB hub.

Note

The examples below are for multiple XFree configuration files, if you use a single XFree configuration file you have to adjust the identifiers.

The configuration for the input devices for the 1st display would look something like this:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "kbd"
    Option      "Protocol"      "evdev"
    Option      "Dev Phys"      "usb-*-1.1.1/input0"
    Option      "AutoRepeat"    "250 30"
    Option      "XkbRules"      "xfree86"
    Option      "XkbModel"      "pc101"
    Option      "XkbLayout"     "dvorak"
EndSection
Section "InputDevice"
```

```
    Identifier  "Mouse1"
    Driver      "mouse"
    Option      "Protocol"      "evdev"
    Option      "Dev Phys"      "usb-*-1.1.2/input0"
    Option      "ZAxisMapping"  "4 5"
EndSection
```

For the 2nd display something like this:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "kbd"
    Option      "Protocol"      "evdev"
    Option      "Dev Phys"      "usb-*-1.2.1/input0"
    Option      "AutoRepeat"    "250 30"
    Option      "XkbRules"      "xfree86"
    Option      "XkbModel"      "pc101"
    Option      "XkbLayout"     "dvorak"
EndSection
Section "InputDevice"
    Identifier  "Mouse1"
    Driver      "mouse"
    Option      "Protocol"      "evdev"
    Option      "Dev Phys"      "usb-*-1.2.2/input0"
    Option      "ZAxisMapping"  "4 5"
EndSection
```

For the 3rd display something like this:

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "kbd"
    Option      "Protocol"      "evdev"
    Option      "Dev Phys"      "usb-*-1.3.1/input0"
    Option      "AutoRepeat"    "250 30"
    Option      "XkbRules"      "xfree86"
    Option      "XkbModel"      "pc101"
    Option      "XkbLayout"     "dvorak"
EndSection
Section "InputDevice"
    Identifier  "Mouse1"
    Driver      "mouse"
    Option      "Protocol"      "evdev"
    Option      "Dev Phys"      "usb-*-1.3.2/input0"
    Option      "ZAxisMapping"  "4 5"
EndSection
```

and so on.

You could also use the "?", so wherever you plug the primary hub, all displays will still have the desired configuration.

```
Section "InputDevice"
    Identifier  "Keyboard1"
    Driver      "kbd"
    Option      "Protocol"      "evdev"
    Option      "Dev Phys"      "usb-*-?.1.1/input0"
    Option      "AutoRepeat"    "250 30"
    Option      "XkbRules"      "xfree86"
```

```
    Option      "XkbModel"      "pc101"
    Option      "XkbLayout"     "dvorak"
EndSection
Section "InputDevice"
    Identifier  "Mouse1"
    Driver      "mouse"
    Option      "Protocol"      "evdev"
    Option      "Dev Phys"      "usb-*-?.1.2/input0"
    Option      "ZAxisMapping"  "4 5"
EndSection
```

Chapter 6. Configuring display managers

If you have successfully finished the installation and configuration of the kernel and XFree, it's time to configure your display manager(s).

Beside the graphical differences, xdm/kdm and gdm handle the X servers differently. Gdm will start the X servers in the order specified in it's configuration file (and stop them in the reverse order). Xdm/kdm will start and stop all the X servers at the same time(in case there are no opened X sessions). Also, restarting the gdm daemon means end for all X sessions, but if you restart xdm/kdm when you are under X, your session won't be closed.

- Using gdm could help you to retain the VGA console and prevents lock−ups with some graphic cards (check the compatibility list).
- Using xdm/kdm allows you to switch it's configuration retaining your opened X session(of course the changes shouldn't affect the X server you are using).

Note

Have in mind that for the older Preferred Bus ID XFree Server (version 1) you have to specify the desired graphic card with parameter "**−prefbusid x:x:x**" where x:x:x is the Bus ID of the desired graphic card. Just append "−prefbusid x:x:x" with the correct Bus ID of the card you want to start right before the last argument vt[x].

6.1. Configuring xdm and kdm

If everything is working now, it's time to setup the automatic starting of X on all displays. For xdm and kdm you have to modify one single file. For a Red Hat−like system this would be `/etc/X11/xdm/Xservers`; for other distributions check whether this file exists. If not, find your XFree86 configuration directory, and in it you'll find `xdm/Xservers`.

Note

- SuSE uses:
    - `/etc/X11/XF86Config`
    - `/etc/X11/xdm/Xservers` for xdm
    - `/etc/opt/kde3/share/config/kdm/Xservers` for kdm

  you can make a backup copy of `/etc/opt/.../kdm/Xservers` and make a symbolic link from `/etc/X11/xdm/Xservers` to `/etc/opt/../kdm/Xservers`, in order to use the same configuration file for xdm and kdm.
- Debian uses:
    - `/etc/kde3/kdm/Xservers` for kdm

  you can make a backup copy of `/etc/kde3/kdm/Xservers` and make a symbolic link from `/etc/X11/xdm/Xservers` to `/etc/kde3/kdm/Xservers`, in order to use the same configuration file for xdm and kdm.
- Red Hat and Gentoo use:
    - `/etc/X11/XF86Config`

For every additional X server you should add a single line. You can copy the existing line, change the X server binary and display number, and append `−xf86config` [your configuration file]. My original xdm/Xservers:

```
######################################################################

# $XConsortium: Xserv.ws.cpp,v 1.3 93/09/28 14:30:30 gildea Exp $
#
#
# $XFree86: xc/programs/xdm/config/Xserv.ws.cpp,v 1.1.1.1.12.2 1998/10/04 15:23:14 hohndel Exp $
#
# Xservers file, workstation prototype
#
# This file should contain an entry to start the server on the
# local display; if you have more than one display (not screen),
# you can add entries to the list (one per line).  If you also
# have some X terminals connected which do not support XDMCP,
# you can add them here as well.  Each X terminal line should
# look like:
#       XTerminalName:0 foreign
#
:0 local /bin/nice −n −10 /usr/X11R6/bin/X :0 −deferglyphs 16 vt7

#######################################################################
```

and the modified version:

```
#######################################################################

# $XConsortium: Xserv.ws.cpp,v 1.3 93/09/28 14:30:30 gildea Exp $
#
#
# $XFree86: xc/programs/xdm/config/Xserv.ws.cpp,v 1.1.1.1.12.2 1998/10/04 15:23:14 hohndel Exp $
#
# Xservers file, workstation prototype
#
# This file should contain an entry to start the server on the
# local display; if you have more than one display (not screen),
# you can add entries to the list (one per line).  If you also
# have some X terminals connected which do not support XDMCP,
# you can add them here as well.  Each X terminal line should
# look like:
#       XTerminalName:0 foreign
#

:0 local /bin/nice −n −10 /usr/X11R6/bin/X0 :0 −deferglyphs 16 vt7
:1 local /bin/nice −n −10 /usr/X11R6/bin/X1 :1 −xf86config /etc/X11/XF86Config−4.TNT2 vt17

########################################################################
```

If you have more video cards just add more lines:

```
:2 local /bin/nice −n −10 /usr/X11R6/bin/X2 :2 −xf86config /etc/X11/XF86Config−4.[your 3rd card] vt18
:3 local .....
```

If you use a single XFree configuration file :

```
#####################################################################

# $XConsortium: Xserv.ws.cpp,v 1.3 93/09/28 14:30:30 gildea Exp $
#
#
# $XFree86: xc/programs/xdm/config/Xserv.ws.cpp,v 1.1.1.1.12.2 1998/10/04 15:23:14 hohndel Exp $
#
# Xservers file, workstation prototype
#
# This file should contain an entry to start the server on the
# local display; if you have more than one display (not screen),
# you can add entries to the list (one per line).  If you also
# have some X terminals connected which do not support XDMCP,
# you can add them here as well.  Each X terminal line should
# look like:
#       XTerminalName:0 foreign
#

:0 local /bin/nice -n -10 /usr/X11R6/bin/X0 :0 -layout first-Xserver -deferglyphs 16 vt7
:1 local /bin/nice -n -10 /usr/X11R6/bin/X1 :1 -layout second-Xserver vt17

######################################################################
```

Note

Under older versions of Backstreet Ruby (released before 7. Oct 2003) each console is represented by 8 tty's, so you should use:

```
.....
:0 local /bin/nice -n -10 /usr/X11R6/bin/X0 :0 -deferglyphs 16 vt7
:1 local /bin/nice -n -10 /usr/X11R6/bin/X1 :1 -xf86config /etc/X11/XF86Config-4.TNT2 vt9
.....
```

Or if you have more video cards just add more lines:

```
:2 local /bin/nice -n -10 /usr/X11R6/bin/X2 :2 -xf86config /etc/X11/XF86Config-4.[your 3rd card] vt16
:3 local .....
```

If you use a single XFree configuration file :

```
................
:0 local /bin/nice -n -10 /usr/X11R6/bin/X0 :0 -layout first-Xserver -deferglyphs 16 vt7
:1 local /bin/nice -n -10 /usr/X11R6/bin/X1 :1 -layout second-Xserver vt9
................
```

6.2. Configuring gdm

Gdm, as a complete rewrite of xdm, uses its own configuration file, `/etc/X11/gdm/gdm.conf`. You should locate the definitions of the local X servers and add additional X servers for the number of cards you have.

Note

- SuSE uses:

    - `/etc/X11/XF86Config`
    - `/etc/opt/gnome2/gdm/gdm.conf` for gdm2
- Debian uses:
    - `/etc/gdm/gdm.conf` for gdm

      `/etc/X11/gdm/` is a symbolic link to `/etc/gdm/`, so you can use both paths.
- Red Hat and Gentoo use:
    - `/etc/X11/XF86Config`

Modifications:

6.2.1. Changes, Part 1

```
.......
[servers]
# These are the standard servers.  You can add as many you want here
# and they will always be started.  Each line must start with a unique
# number and that will be the display number of that server.  Usually just
# the 0 server is used.
0=Standard
1=Second
.......
```

Here, you'll have only 0=Standard. For each additional X server you should add a definition, like here, for 1=Second.

6.2.2. Changes, Part 2

```
..............................
# Definition of the standard X server.
[server-Standard]
name=Standard server
command=/usr/X11R6/bin/X0 :0 -deferglyphs 16 -ac vt7
flexible=true

[server-Second]
name=Second server
command=/usr/X11R6/bin/X1 :1 -deferglyphs 16 -xf86config /etc/X11/XF86Config-4.[your card] vt17
flexible=true
..............................
```

And here, the exact definition of the command line for starting the X server, very similar to the definitions in `/etc/X11/xdm/Xservers` (the definition of [server−Second] is the additional one).

If you use a single XFree configuration file:

```
..............................
# Definition of the standard X server.
[server-Standard]
```

```
name=Standard server
command=/usr/X11R6/bin/X0 :0 -layout first-Xserver -deferglyphs 16 -ac vt7
flexible=true

[server-Second]
name=Second server
command=/usr/X11R6/bin/X1 :1 -layout second-Xserver -deferglyphs 16 vt17
flexible=true
..............................
```

Note

Under older versions of Backstreet Ruby (released before 7. Oct 2003) each console is represented by 8 tty's, so you should use:

```
..............................
name=Standard server
command=/usr/X11R6/bin/X0 :0 -deferglyphs 16 -ac vt7
..............................
name=Second server
command=/usr/X11R6/bin/X1 :1 -deferglyphs 16 -xf86config /etc/X11/XF86Config-4.[your card] vt9
..............................
```

If you use a single XFree configuration file:

```
..............................
name=Standard server
command=/usr/X11R6/bin/X0 :0 -layout first-Xserver -deferglyphs 16 -ac vt7
......
name=Second server
command=/usr/X11R6/bin/X1 :1 -layout second-Xserver -deferglyphs 16 vt9
..............................
```

Chapter 7. Tweaking it

Adding customisation and automation.

7.1. Using independent sound cards

Note

In case you do not use devfs, you may need to create additional device files. Take a look at The Linux Sound HOWTO, for information on how to setup additional sound cards.

7.1.1. Using aRts daemon

We have to specify different sound devices for the different Xsessions/Displays. This is done by using the following options of aRts daemon (http://www.arts–project.org/):

- By OSS–free sound driver:
  ```
  -D /dev/dsp[n]
  ```
  where n is the number of the sound card.
- By Alsa sound driver:
  ```
  -a alsa -D hw:[n],0
  ```
  where n is sound card id.

Add these lines to your Window Manager start–up script (of course, with the proper arguments for your setup):

```
real_display=`echo $DISPLAY | sed "s/://" | sed "s/\..*//"`
case "$real_display" in
        0)
        artsd -F 10 -S 4096 -D /dev/dsp -s 5 -m artsmessage -l 3 -f &
        ;;
        1)
        artsd -F 10 -S 4096 -D /dev/dsp1 -s 5 -m artsmessage -l 3 -f &
        ;;
        2)
        artsd -F 10 -S 4096 -a alsa -D hw:4,0 -s 5 -m artsmessage -l 3 -f &
        ;;
esac
```

And at the end of the file:

```
artsshell -q terminate
```

Here is an example for `/usr/X11R6/bin/startenlightenment`:

```
#!/bin/sh
#   License: GPL

real_display=`echo $DISPLAY | sed "s/://" | sed "s/\..*//"`
case "$real_display" in
        0)
```

```
        artsd -F 10 -S 4096 -D /dev/dsp -s 5 -m artsmessage -l 3 -f &
        ;;
        1)
        artsd -F 10 -S 4096 -D /dev/dsp1 -s 5 -m artsmessage -l 3 -f &
        ;;
        2)
        artsd -F 10 -S 4096 -a alsa -D hw:2,0 -s 5 -m artsmessage -l 3 -f &
        ;;
esac

/usr/X11R6/bin/enlightenment
artsshell -q terminate
```

This will start 3 aRts daemons for 3 X servers.

1. Daemon will use the first OSS sound device for the 1st X server.
2. Daemon will use the second OSS sound device for the 2nd X server.
3. Daemon will use the Alsa sound device for the 3rd X server (requires feedback).

7.2. Customising the login screen

7.2.1. Using xdm

Copy `/etc/X11/xdm/Xsetup_0` to `/etc/X11/xdm/Xsetup_1`. For additional X servers, create the file(s) `/etc/X11/xdm/Xsetup_[n]`, where n is the number of the X server starting from 0.

1. Modify the line containing the background image, to adjust the path to your image for the 2nd X server:

   ```
   ....
   if [ -r /usr/share/mdk/backgrounds/default.png -a -x /usr/bin/qiv ]; then
       /usr/bin/qiv -z /usr/share/mdk/backgrounds/default.png
   else
       /usr/X11R6/bin/xsetroot -solid "#21449C"
   fi
   ....
   ```

   Modified:

   ```
   ....
   if [ -r /usr/share/mdk/backgrounds/flower.jpg -a -x /usr/bin/qiv ]; then
       /usr/bin/qiv -z /usr/share/mdk/backgrounds/flower.jpg
   else
       /usr/X11R6/bin/xsetroot -solid "#21449C"
   fi
   ....
   ```

   You can also specify a different background color, with:

   ```
   ....
   #if [ -r /usr/share/mdk/backgrounds/flower.jpg -a -x /usr/bin/qiv ]; then
   #    /usr/bin/qiv -z /usr/share/mdk/backgrounds/flower.jpg
   #else
       /usr/X11R6/bin/xsetroot -solid "[your color]"
   ```

```
#fi
....
```

2. Comment out the lines which may affect the primary X server, from:

```
....
if [ -x /etc/X11/xinit.d/numlock ]; then
    /etc/X11/xinit.d/numlock &
fi
....
```

to

```
....
#if [ -x /etc/X11/xinit.d/numlock ]; then
#    /etc/X11/xinit.d/numlock &
#fi
....
```

3. Inform xdm about the existence of Xsetup_1 by modifying `/etc/X11/xdm/xdm-config`:

```
......
! The following three resources set up display :0 as the console.
DisplayManager._0.setup:        /etc/X11/xdm/Xsetup_0
DisplayManager._0.startup:      /etc/X11/xdm/GiveConsole
DisplayManager._0.reset:        /etc/X11/xdm/TakeConsole
DisplayManager._0.startAttempts: 1
!
......
```

Modified:

```
......
! The following three resources set up display :0 as the console.
DisplayManager._1.setup:        /etc/X11/xdm/Xsetup_1
DisplayManager._0.setup:        /etc/X11/xdm/Xsetup_0
DisplayManager._0.startup:      /etc/X11/xdm/GiveConsole
DisplayManager._0.reset:        /etc/X11/xdm/TakeConsole
DisplayManager._0.startAttempts: 1
!
......
```

4. Repeat the procedure for each additional X server.
5. Check here for additional customising options: Linux−Journal Issue 68: Linux Apprentice: Customising the XDM Login Screen.

7.2.2. Using kdm

- I'm not really sure. This area requires feedback.

  Modify `/usr/share/config/kdm/kdmrc`, from:

```
......
[X-:0-Core]
Authorize=true
AutoLogin1st=true
AutoLoginEnable=false
Reset=/etc/X11/xdm/TakeConsole
Setup=/etc/X11/xdm/Xsetup_0
```

```
Startup=/etc/X11/xdm/GiveConsole

[X−:1−Core]
Authorize=true
......
```

to

```
......
[X−:0−Core]
Authorize=true
AutoLogin1st=true
AutoLoginEnable=false
Reset=/etc/X11/xdm/TakeConsole
Setup=/etc/X11/xdm/Xsetup_0
Startup=/etc/X11/xdm/GiveConsole

[X−:1−Core]
Authorize=true
#AutoLogin1st=true
#AutoLoginEnable=false
#Reset=/etc/X11/xdm/TakeConsole
Setup=/etc/X11/xdm/Xsetup_1
#Startup=/etc/X11/xdm/GiveConsole

#[X−:1−Core]
#Authorize=true
......
```

- Check for additional customising options at the KDE Help Center.

7.2.3. Using gdm

This requires `gdmlogin` to be used instead of `gdmgreater`, because using different themes for different displays is not yet implemented in gdm. In case you want to use gdm themes you'll have the same theme on all displays.

1. Switch to gdmlogin by making this changes in `/etc/X11/gdm/gdm.conf`

   From:

   ```
   .....
   # Greeter for local (non−xdmcp) logins.  Change gdmlogin to gdmgreeter to
   # get the new graphical greeter.
   Greeter=/usr/bin/gdmgreeter
   .....
   ```

   to

   ```
   .....
   # Greeter for local (non−xdmcp) logins.  Change gdmlogin to gdmgreeter to
   # get the new graphical greeter.
   Greeter=/usr/bin/gdmlogin
   .....
   ```

2. Copy the file `/etc/X11/gdm/Init/Default` to `/etc/X11/gdm/Init/:0`, and `/etc/X11/gdm/Init/:1`
3. Add these lines to use the background that kdm uses (you can use another image file as well, just change the full path to it):

```
if [ -r /usr/share/mdk/backgrounds/default.png -a -x /usr/bin/qiv ]; then
    /usr/bin/qiv -z /usr/share/mdk/backgrounds/default.png
else
    /usr/X11R6/bin/xsetroot -solid "#21449C"
fi
```

You can also specify a different background color, with:

```
/usr/X11R6/bin/xsetroot -solid "[your color]"
```

Modified:

`/etc/X11/gdm/Init/:0`

```
#!/bin/sh

if [ -r /usr/share/mdk/backgrounds/default.png -a -x /usr/bin/qiv ]; then
    /usr/bin/qiv -z /usr/share/mdk/backgrounds/default.png
else
    /usr/X11R6/bin/xsetroot -solid "#21449C"
fi
if [ -x /etc/X11/xinit/fixkeyboard ]; then
    /etc/X11/xinit/fixkeyboard
fi

exit 0
```

`/etc/X11/gdm/Init/:1`

```
#!/bin/sh

if [ -r /usr/share/mdk/backgrounds/flower.jpg -a -x /usr/bin/qiv ]; then
    /usr/bin/qiv -z /usr/share/mdk/backgrounds/flower.jpg
else
    /usr/X11R6/bin/xsetroot -solid "#21449C"
fi

if [ -x /etc/X11/xinit/fixkeyboard ]; then
    /etc/X11/xinit/fixkeyboard
fi

exit 0
```

4. Repeat the procedure for each additional X server, using file(s) `/etc/X11/gdm/Init/:[n]`, where n is the number of the display.
5. Check here for additional customising options: Gnome Display Manager Reference Manual.

7.3. 1st X server configuration file

A small part of the Mandrake init scripts `/etc/rc.d/rc.sysinit` (you can append it to yours if you are missing something similar):

```
-----------------------------------------------------------------------------------
# (pixel) a kind of profile for XF86Config
# if no XFree=XXX given on kernel command-line, restore XF86Config.standard
for i in XF86Config XF86Config-4; do
    if [ -L "/etc/X11/$i" ]; then
        XFree=`sed -n 's/.*XFree=\(\w*\).*/\1/p' /proc/cmdline`
        [ -n "$XFree" ] || XFree=standard
        [ -r "/etc/X11/$i.$XFree" ] && ln -sf "$i.$XFree" "/etc/X11/$i"
    fi
done
-----------------------------------------------------------------------------------
```

Move your `XF86Config-4` file (the one for standard kernel) to `XF86Config-4.standard`, create a symbolic link from it to `XF86Config-4`, and move the `XF86Config-4` file (the one for Backstreet Ruby) to `XF86Config-4.bruby`. For Ruby/Backstreet Ruby kernels, add to the append line in `/etc/lilo.conf`, or on boot prompt "XFree=bruby", leave the standard kernel as is.

Results:

Booting with "XFree=standard" or without "XFree=" (boot prompt or `lilo.conf`) will result in linking `XF86Config-4.standard` to `XF86Config-4`; booting with "XFree=bruby" will link `XF86Config-4.bruby` to `XF86Config-4`, so in both scenarios XFree can be started with the proper configuration file for the first X server.

And what about the other X servers?

Under a standard kernel you cannot use several independent X servers, so you should use the other XFree configuration files only under Ruby/Backstreet Ruby − there is no need for different configuration files under standard & bruby kernels.

7.4. Number X servers started by Display managers

Here is a modified version of the previous approach. Add this to your init scripts (I bet it's missing!):

```
---------------------------------------------------------------------------------------
#
#the same like XF86Config but for gdm.conf & Xservers
#
for i in xdm/Xservers gdm/gdm.conf; do
    if [ -L "/etc/X11/$i" ]; then
        DumbCon=`sed -n 's/.*dumbcon=\([0-9]*\).*/\1/p' /proc/cmdline`
        [ -n "$DumbCon" ] || DumbCon=0
        [ -r "/etc/X11/$i.$DumbCon" ] && ln -sf "/etc/X11/$i.$DumbCon" "/etc/X11/$i"
    fi
done
echo "Setting up display managers for `expr $DumbCon + 1` Xservers"
---------------------------------------------------------------------------------------
```

This will adjust the proper `/etc/X11/xdm/Xservers` and `/etc/X11/gdm/gdm.config` according to the boot line argument dumbcon=n (remember n+1= number of X users/sessions).

You have to create the configuration files following these assumptions:

"i" only stands for `/etc/X11/xdm/Xserver` and `/etc/X11/gdm/gdm.conf`.

- "i.0" is used for a single X server, when dumbcon=n is not specified, or dumbcon=0.
- "i.1" is used by the display manager when dumbcon=1 is specified.
- "i.2" is used by the display manager when dumbcon=2 is specified.
- "i.3" is used by the display manager when dumbcon=3 is specified.

...and so on.

Therefore:

- "i.0" should contain the definition only of your original standard X server.
- "i.1" should contain the definitions for 2 X servers.
- "i.2" should contain the definitions for 3 X servers.
- ...and so on.

If you boot without dumbcon=n or with dumbcon=0 (for example a standard kernel), your display manager will start a single X server with the corresponding XF86Config file.

If you start with dumbcon=1 the display manager will automatically start 2 X servers.

If you start with dumbcon=2, when booting is finished you'll get 3 login prompts on your 3 displays.

Keep in mind that each X server should have it's own configuration file, and it should be specified in the display manager configuration file properly. Take a look at the configuration files before restarting with an activated display manager and this addition to your init scripts.

This can also be used if you have a single XFree configuration file (see Section 4.6, "For graphic cards without DRI"). In this case you will have to specify the correct layout instead of the correct XFree configuration file.

7.5. Dynamically switching the number of X servers

There is a very experimental GUI/CLI for dynamically switching the number of running X servers. It uses the automatic configuration of the display managers (mentioned in Section 7.4), Python, dialog for the CLI, and Xdialog for the GUI.

Once it is more tested and bug−free, you could, for example, use it under Backstreet Ruby to switch between 2, 3 or more X servers and a single X server using Xinerama. So when your PC isn't used by more then one user, you could use the other monitors under Xinerama. Or one more funny example: you're simulating net gaming with a number of friends on your bruby Linux PC, you have invested a bit more in an additional graphic card which is already configured, but you don't have enough money right now to buy one more monitor and keyboard/mouse pair. One friend of yours comes and says, "Hey guys, that's cool. Can I join?" What would you answer? Using the GUI could result in the following answer from your side: "No problem, just bring your monitor,keyboard and mouse."

If you are feeling like a hacker and want to try out this BUGGY GUI/CLI, check the current status at http://karlovo.demon.co.uk/~svetlio/ruby−contrib/bruby−python/. But remember, it's not very tested, and if not configured properly it can cause you serious troubles. Please wait until it is more stable if you are not that familiar with Linux. If you feel comfortable enough under Linux, and think of yourself as a hacker, please help in testing it and making it better, bug−free and easy to configure.

Chapter 8. Known problems

8.1. Hardware problems

While not exactly problems, some graphic cards do not work well, or even at all in multi−user environments.

If you are building such a system from the beginning, check the Video Compatibility list before buying video hardware.

Sometimes secondary graphic card(s) (for now reported only by Nvidia owners) will refuse to start, even if they have worked flowlessly for months. One of the following sollutions should fix the problem:

- Running **/usr/X11R6/bin/X −probeonly** on the secondary card(s) before initializing/ starting XFree on the primary graphic card.

  Some examples:

  /usr/X11R6/bin/X −probeonly −layout X2

  /usr/X11R6/bin/X −probeonly −layout X1

  or:

  /usr/X11R6/bin/X −probeonly −xf86config /etc/X11/XF86Config−4.X2

  /usr/X11R6/bin/X −probeonly −xf86config /etc/X11/XF86Config−4.X1
- Switching the secondary graphic card to primary, starting the system (to initialize the card as primary), and reverting the card to secondary.
- In case this is the only secondary card, you can try switching the order of the graphic cards permanently.

8.2. Software problems

For details on solving software problems see Chapter 9, "Special notes on some distributions."

8.2.1. Incompatible userspace program:s

1. gpm − freezy mouse under XFree86. With the current XFree86 you are losing VGA virtual consoles anyway.

   Recommended: disable.
2. RedHat 8.0/9 − `/bin/sysfont`:

   You can use RedHat 7.3 `consolechars` instead.
3. SuSE 8.1 − `/etc/init.d/hwscan`:

   Recommended: disable. If you have to install new hardware and want to use this service, boot with standard kernel and start it manually.

4. Programs writing directly to tty's, like `vcstime`, cannot be used.

8.2.2. Tweaks needed

1. Mandrake 9.1 − `/etc/init.d/numlock`:

   You should change the lines including "`/dev/tty[0−8]`" to "`/dev/tty[0−7]`".
2. SuSE 8.1 − `/etc/init.d/kbd`:

   Add this line in the very beginning of the file:

   ```
   KBD_TTY="tty0 tty2 tty3 tty4 tty5 tty6 tty7"
   ```

Chapter 9. Special notes on some distributions

9.1. Mandrake

Almost everything is easy to accomplish (probably because I've used it for 2−3 years, at least).

1. Just a small tweak in `/etc/init.d/numlock`:

```
--- /etc/init.d/numlock.orig    2003-04-11 00:58:55.000000000 +0200
+++ /etc/init.d/numlock 2003-03-19 13:03:30.000000000 +0100
@@ -21,14 +21,14 @@
        echo
        touch $SYSCONF_FILE

-       for tty in /dev/tty[0-8]; do
+       for tty in /dev/tty[0-7]; do
                setleds -D +num < $tty
        done

        ;;
   stop)
        gprintf "Disabling numlocks on ttys: "
-       for tty in /dev/tty[0-8]; do
+       for tty in /dev/tty[0-7]; do
                setleds -D -num < $tty
        done
        echo_success
```

9.2. Red Hat

1. Replacing `sysfont` with `consolechars`.

   < needs to be written >

   Rebuild `console−tools−19990829−40.src.rpm` using **rpmbuild −−rebuild console−tools−19990829−40.src.rpm**. You can find the source rpm on http://www.rpmfind.net).

   Then install it:

   rpm −Uvh /usr/src/redhat/RPMS/i386/console−tools−19990829−40.i386.rpm

   I still seem to have some kind of problem, because on the console I always get:

   `findfont no such file or directory unable to setfont xxx`

   But everything else seems okay.

9.3. Debian

There are no known problems.

9.4. SuSE

1. In `/etc/init.d/kbd`, add this line in the very beginning of the file:

   ```
   KBD_TTY="tty0 tty2 tty3 tty4 tty5 tty6 tty7"
   ```

   After modifications:

   ```
   #! /bin/sh
   # Copyright (c) 1995-2001 SuSE GmbH Nuernberg, Germany.
   #
   # Author:   Burchard Steinbild <bs@suse.de>
   #           Werner Fink <werner@suse.de>
   #
   # /etc/init.d/kbd
   #
   #   and symbolic its link
   #
   # /sbin/rckbd
   #
   ### BEGIN INIT INFO
   # Provides:       kbd
   # Required-Start: $remote_fs
   # Required-Stop:
   # X-SuSE-Should-Start:  fbset serial
   # X-SuSE-Should-Stop:
   # Default-Start:  1 2 3 5 S
   # Default-Stop:
   # Description:    Keyboard settings (don't disable!)
   ### END INIT INFO

   . /etc/rc.status
   . /etc/sysconfig/console
   . /etc/sysconfig/keyboard

   MACHINE=`/bin/uname -m 2> /dev/null`
   if [ "$MACHINE" = "sparc" -o "$MACHINE" = "sparc64" ]; then
      # Test if we have a serial console.
      (test -c /dev/tty1 && > /dev/tty1 ) > /dev/null 2>&1 || exit 0
   fi

   # The variable NON_SUSE_KERNEL determines whether we need to chvt
   # to a console before some console settings apply.
   # We have no magic to find out about this (at boot time), so we
   # leave it to the user to read this comment and put NON_SUSE_KERNEL="yes"
   # into /etc/sysconfig/console

   KBDBASE="/usr/share/kbd"

   KBD_TTY="tty0 tty2 tty3 tty4 tty5 tty6 tty7"
   KTABLE=${KEYTABLE%.map*}
   KTABLE=${KTABLE##*/}
   #
   # first search the wanted keytable.
   #
   if [ $MACHINE = ppc -o $MACHINE = ppc64 ]; then
           test -f /proc/cpuinfo || mount -n -t proc proc /proc 2>/dev/null
           while read line; do
   ......
   ......
   ```

2. Hardware scans sometimes cause problems.

Recommended: disable. If you have to install new hardware and want to use this service, boot with standard kernel and start it manually.

Chapter 10. Final words

Have some comments? Send them to Svetoslav Slavtchev, `<svetoslav (at) users.sourcefourge.net>`.

Difficulty understanding the HOWTO? Some parts are not clear? Drop a line to the above address.

In case you experience troubles in configuring the system, feel free to contact me or the linuxconsole mailing list.

Please send as many details as possible, the most important information would be (from a running Backstreet Ruby kernel):

- output from **dmesg**
- output from **lsmod**
- output from **cat /proc/bus/console/*/***
- contents of `/proc/bus/input/devices`
- contents of `/proc/bus/usb/devices`
- contents of the XFree configuration file(s) `/etc/X11/XF86Config(-4)`
- contents of the XFree86 log files `/var/log/XFree86.[n].log`

You got it running? Congratulations! Drop a line, give some details on your configuration and attach your XFree configuration files.

Appendix A. Video Compatibility list

This is an extract from the Video Compatibility list at the Backstreet Ruby home page.

A.1. Graphic card pairs/triples that work perfectly

A.1.1. Modified X server not needed

Voodoo Graphics (glide) + Voodoo 3 (pci)(tdfx)

Voodoo Graphics (glide) + Riva TNT2 M64 (agp)(nvidia)

A.1.2. Modified X server needed

ATI Radeon 7000(AGP)+ Matrox Mystique (PCI)

AGP S3 Inc. 86c368[Trio 3D/2X]+Matrox MGA1064SG[Mystique] (PCI)

Nvidia cards

Riva TNT (PCI+PCI+AGP)

Riva TNT2 M64 (pci)(nvidia) + Riva TNT2 M64 (pci)(nvidia)

Riva TNT2 M64 (agp)(nvidia) + GeForce4MX(PCI)

Nvidia GeForce2MX(PCI) + GeForce2MX(PCI) + GeForce2MX(PCI) + GeForce2MX(PCI)

Works fine. DRI + Nvidia GLX works too.

Voodoo 3 (pci)(tdfx) + Riva TNT2 M64 (agp)(nvidia)

ATI Radeon 7500(AGP) + Nvidia TNT2(PCI)

A.2. Graphic card pairs/triples that work, but with some glitches

- Generally the X server driving the AGP card has to be started first.

  This can be acomplished by manually starting X or using gdm as desktop manager. You'll have to abstain from using xdm or kdm, as they start the X servers at the same time.
- AGP sever restart leads to system crashes. In `gdm.conf`, set `AlwaysRestartServer=false`.
- The XFree–4.3–prefbusid patch/binary fixes most of the problems. In case the X servers are started in the right order there are no lock ups.

Nvidia TNT2(AGP) + Matrox Mystique(PCI)

Nvidia TNT2(AGP) + S3 VIRGE/DX(PCI)

Nvidia GeForce4MX440(AGP) + TNT2M64(PCI) + TNT2M64(PCI)

Matrox MGAG400(AGP) + Matrox MGA1064SG[Mystique]PCI

Matrox MGAG450DH(AGP) + ATI Mach 64(PCI)

Matrox MGAG550DH(AGP) + Riva TNT2−M64(PCI)

Matrox MGAG550DH(AGP) + Geforce4 420(PCI)

Appendix B. Example configuration files

For more examples visit http://karlovo.demon.co.uk/~svetlio/examples/.

B.1. XFree86

B.1.1. 1st XFree server configuration file

```
# File generated by XFdrake.

# **********************************************************************
# Refer to the XF86Config(4/5) man page for details about the format of
# this file.
# **********************************************************************

Section "Files"

    RgbPath     "/usr/X11R6/lib/X11/rgb"

# Multiple FontPath entries are allowed (they are concatenated together)
# By default, Mandrake 6.0 and later now use a font server independent of
# the X server to render fonts.


EndSection

# **********************************************************************
# Server flags section.
# **********************************************************************

Section "ServerFlags"

    # Uncomment this to cause a core dump at the spot where a signal is
    # received.  This may leave the console in an unusable state, but may
    # provide a better stack trace in the core dump to aid in debugging
    #NoTrapSignals

    # Uncomment this to disable the <Crtl><Alt><BS> server abort sequence
    # This allows clients to receive this key event.
    #DontZap

    # Uncomment this to disable the <Crtl><Alt><KP_+>/<KP_-> mode switching
    # sequences.  This allows clients to receive these key events.
    #DontZoom

    # This  allows  the  server  to start up even if the
    # mouse device can't be opened/initialised.
    AllowMouseOpenFail

EndSection

# **********************************************************************
# Input devices
# **********************************************************************

# **********************************************************************
# Keyboard section
# **********************************************************************
```

```
Section "InputDevice"

    Identifier "Keyboard1"
    Driver      "Keyboard"
    Option "AutoRepeat"  "250 30"

    Option "XkbRules" "xfree86"
    Option "XkbModel" "pc105"
    Option "XkbLayout" "de(nodeadkeys)"

EndSection

# **********************************************************************
# Pointer section
# **********************************************************************

Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "IMPS/2"
#    Option "Device"      "/dev/psaux"
    Option "Device"      "/dev/input/mouse1"
    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3-button Logitech mice
    Option "Emulate3Buttons"
#    Option "ChordMiddle"

EndSection



Section "Module"

# This loads the DBE extension module.
    Load        "dbe"
    Load        "GLcore"
#    Load       "dga"
    Load        "glx"
    Load        "extmod"
    Load        "dri"
# pass two from mga mailing-lists
#    Load       "pex5"
#    Load       "xie"
    Load        "bitmap"
    Load        "record"
    Load        "vbe"
    Load        "int10"
# end pass two mga mailing-lists


# This loads the Video for Linux module.
    Load        "v4l"


# This loads the miscellaneous extensions module, and disables
# initialisation of the XFree86-DGA extension within that module.

    SubSection  "extmod"
        #Option "omit xfree86-dga"
```

```
    EndSubSection

# This loads the Type1 and FreeType font modules

    Load        "type1"
    Load        "freetype"
EndSection

Section "DRI"
    Mode        0666
EndSection

# **********************************************************************
# Monitor section
# **********************************************************************

# Any number of monitor sections may be present

Section "Monitor"
    Identifier "Generic Monitor that can do 1600x1200 at 70 Hz"
    VendorName "Unknown"
    ModelName  "Unknown"


# HorizSync is in kHz unless units are specified.
# HorizSync may be a comma separated list of discrete values, or a
# comma separated list of ranges of values.
# NOTE: THE VALUES HERE ARE EXAMPLES ONLY.  REFER TO YOUR MONITOR'S
# USER MANUAL FOR THE CORRECT NUMBERS.
    HorizSync  30−98


# VertRefresh is in Hz unless units are specified.
# VertRefresh may be a comma separated list of discrete values, or a
# comma separated list of ranges of values.
# NOTE: THE VALUES HERE ARE EXAMPLES ONLY.  REFER TO YOUR MONITOR'S
# USER MANUAL FOR THE CORRECT NUMBERS.
    VertRefresh 50−160


# This is a set of extended mode timings typically used for laptop,
# TV fullscreen mode or DVD fullscreen output.
# These are available along with standard mode timings.

# Sony Vaio C1(X,XS,VE,VN)?
# 1024x480 @ 85.6 Hz, 48 kHz hsync
ModeLine "1024x480"    65.00 1024 1032 1176 1344   480  488  494  563 −hsync −vsync

# 768x576 @ 79 Hz, 50 kHz hsync
ModeLine "768x576"     50.00  768  832  846 1000   576  590  595  630
# 768x576 @ 100 Hz, 61.6 kHz hsync
ModeLine "768x576"     63.07  768  800  960 1024   576  578  590  616


EndSection


Section "Monitor"
    Identifier "monitor2"
    VendorName "Unknown"
    ModelName  "Unknown"
```

```
    HorizSync   30-98
    VertRefresh 50-160
EndSection


Section "Monitor"
    Identifier "monitor3"
    VendorName "Unknown"
    ModelName  "Unknown"

    HorizSync   31.5-60.0
    VertRefresh 56.0-75.0
EndSection



# **********************************************************************
# Graphics device section
# **********************************************************************

Section "Device"
    Identifier "Generic VGA"
    Driver     "vga"
EndSection


Section "Device"
    Identifier  "g550_1"
    VendorName  ""
    BoardName   ""
    Driver      "mga"
    # Clock lines


    # Uncomment following option if you see a big white block
    # instead of the cursor!
    #    Option      "sw_cursor"
    Option      "AGPMode" "4"
#    Option      "HWCursor" "Off"
    Option      "HWCursor" "On"
    Option      "MGASDRAM" "On"
    Option      "DPMS"  "Off"
    # Screen 0
    BusID       "PCI:1:0:0"
EndSection


Section "Device"
    Identifier  "g550_2"
    VendorName  ""
    BoardName   ""
    Driver      "mga"
    # Clock lines


    # Uncomment following option if you see a big white block
    # instead of the cursor!
    #    Option      "sw_cursor"
    Option      "AGPMode" "4"
    Option      "MGASDRAM" "On"
#    Option      "HWCursor" "Off"
    Option      "HWCursor" "On"
```

```
    Option      "DPMS"  "Off"
    # Screen 1
    BusID       "PCI:1:0:0"
EndSection



# **********************************************************************
# Screen sections
# **********************************************************************

Section "Screen"
    Identifier "screen1"
    Device      "g550_1"
    Monitor     "monitor2"
    DefaultColorDepth 16
    Subsection "Display"
        Depth       8
        Modes       "1280x1024" "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       15
        Modes       "1280x1024" "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       16
        Modes       "1280x1024" "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       24
        Modes       "1280x1024" "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
EndSection


Section "ServerLayout"
    Identifier "layout1"
    Screen     "screen1"
    Option      "SingleCard" "true"

    InputDevice "Mouse1" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"
EndSection
```

B.1.2. 2nd XFree server configuration file

```
# File generated by XFdrake.

# **********************************************************************
# Refer to the XF86Config(4/5) man page for details about the format of
# this file.
# **********************************************************************

Section "Files"

    RgbPath     "/usr/X11R6/lib/X11/rgb"
```

```
    ModulePath  "/usr/X11R6/libNV/modules"
# Multiple FontPath entries are allowed (they are concatenated together)
# By default, Mandrake 6.0 and later now use a font server independent of
# the X server to render fonts.


EndSection

# **********************************************************************
# Server flags section.
# **********************************************************************

Section "ServerFlags"

    # Uncomment this to cause a core dump at the spot where a signal is
    # received.  This may leave the console in an unusable state, but may
    # provide a better stack trace in the core dump to aid in debugging
    #NoTrapSignals

    # Uncomment this to disable the <Crtl><Alt><BS> server abort sequence
    # This allows clients to receive this key event.
    #DontZap

    # Uncomment this to disable the <Crtl><Alt><KP_+>/<KP_-> mode switching
    # sequences.  This allows clients to receive these key events.
    #DontZoom

    # This  allows  the  server  to start up even if the
    # mouse device can't be opened/initialised.
    AllowMouseOpenFail

EndSection

# **********************************************************************
# Input devices
# **********************************************************************

# **********************************************************************
# Keyboard section
# **********************************************************************

Section "InputDevice"

    Identifier "Keyboard1"
    Driver      "Keyboard"
    Option "AutoRepeat"  "250 30"

    Option "XkbRules" "xfree86"
    Option "XkbModel" "pc105"
    Option "XkbLayout" "de(nodeadkeys)"

EndSection

# **********************************************************************
# Pointer section
# **********************************************************************

Section "InputDevice"

    Identifier  "Mouse1"
    Driver      "mouse"
    Option "Protocol"    "PS/2"
```

```
    Option "Device"      "/dev/input/mouse0"
    Option "Emulate3Buttons"
#    Option "ZAxisMapping" "4 5"

# ChordMiddle is an option for some 3-button Logitech mice

#    Option "ChordMiddle"

EndSection


Section "Module"

# This loads the DBE extension module.
    Load        "dbe"
    Load        "glx"
    Load        "vbe"
    Load        "int10"

# This loads the Video for Linux module.
#    Load        "v4l"


# This loads the miscellaneous extensions module, and disables
# initialisation of the XFree86-DGA extension within that module.

    SubSection  "extmod"
        #Option "omit xfree86-dga"
    EndSubSection

# This loads the Type1 and FreeType font modules

    Load        "type1"
    Load        "freetype"
EndSection

#Section "DRI"
#    Mode       0666
#EndSection

# **********************************************************************
# Monitor section
# **********************************************************************

# Any number of monitor sections may be present

Section "Monitor"
    Identifier "Generic Monitor that can do 1600x1200 at 70 Hz"
    VendorName "Unknown"
    ModelName  "Unknown"


# HorizSync is in kHz unless units are specified.
# HorizSync may be a comma separated list of discrete values, or a
# comma separated list of ranges of values.
# NOTE: THE VALUES HERE ARE EXAMPLES ONLY.  REFER TO YOUR MONITOR'S
# USER MANUAL FOR THE CORRECT NUMBERS.
    HorizSync  30-98


# VertRefresh is in Hz unless units are specified.
# VertRefresh may be a comma separated list of discrete values, or a
```

```
# comma separated list of ranges of values.
# NOTE: THE VALUES HERE ARE EXAMPLES ONLY.  REFER TO YOUR MONITOR'S
# USER MANUAL FOR THE CORRECT NUMBERS.
    VertRefresh 50-160


# This is a set of extended mode timings typically used for laptop,
# TV fullscreen mode or DVD fullscreen output.
# These are available along with standard mode timings.

# Sony Vaio C1(X,XS,VE,VN)?
# 1024x480 @ 85.6 Hz, 48 kHz hsync
ModeLine "1024x480"    65.00 1024 1032 1176 1344   480  488  494  563 -hsync -vsync

# 768x576 @ 79 Hz, 50 kHz hsync
ModeLine "768x576"     50.00  768  832  846 1000   576  590  595  630
# 768x576 @ 100 Hz, 61.6 kHz hsync
ModeLine "768x576"     63.07  768  800  960 1024   576  578  590  616


EndSection


Section "Monitor"
    Identifier "monitor2"
    VendorName "Unknown"
    ModelName  "Unknown"

#    HorizSync   30-98
#    VertRefresh 50-160

    HorizSync   30-50
    VertRefresh 50-120
EndSection



# **********************************************************************
# Graphics device section
# **********************************************************************

Section "Device"
    Identifier "Generic VGA"
    Driver     "vga"
EndSection

Section "Device"
    Identifier  "nv"
    VendorName  ""
    BoardName   ""
    Driver      "nvidia"
    # Clock lines


    # Uncomment following option if you see a big white block
    # instead of the cursor!
    #    Option      "sw_cursor"
    Option      "UseInt10Module" "off"
    Option      "ConnectedMonitor" "CRT"
    Option      "IgnoreEDID" "off"
    Option      "HWCursor" "on"
```

```
    Option      "DPMS"  "Off"
    Option      "NoLogo" "On"
    BusID       "PCI:0:10:0"
EndSection


# **********************************************************************
# Screen sections
# **********************************************************************

Section "Screen"
    Identifier "screen1"
    Device      "nv"
    Monitor     "monitor2"
    DefaultColorDepth 24
    Subsection "Display"
        Depth       8
        Modes       "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       15
        Modes       "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       16
        Modes       "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       24
        Modes       "1024x768" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
    Subsection "Display"
        Depth       32
        Modes       "1280x1024" "800x600" "640x480"
        ViewPort    0 0
    EndSubsection
EndSection


Section "ServerLayout"
    Identifier "layout1"
    Screen     "screen1"
    Option      "SingleCard" "true"

    InputDevice "Mouse1" "CorePointer"
    InputDevice "Keyboard1" "CoreKeyboard"

EndSection
```

B.2. Display managers

B.2.1. xdm and kdm

`/etc/X11/xdm/Xservers.0`

```
# $XConsortium: Xserv.ws.cpp,v 1.3 93/09/28 14:30:30 gildea Exp $
#
#
# $XFree86: xc/programs/xdm/config/Xserv.ws.cpp,v 1.1.1.1.12.2 1998/10/04 15:23:14 hohndel Exp $
#
# Xservers file, workstation prototype
#
# This file should contain an entry to start the server on the
# local display; if you have more than one display (not screen),
# you can add entries to the list (one per line).  If you also
# have some X terminals connected which do not support XDMCP,
# you can add them here as well.  Each X terminal line should
# look like:
#       XTerminalName:0 foreign
#
:0 local /bin/nice −n −10 /usr/X11R6/bin/X −deferglyphs 16 vt7
```

`/etc/X11/xdm/Xservers.1`

```
# $XConsortium: Xserv.ws.cpp,v 1.3 93/09/28 14:30:30 gildea Exp $
#
#
# $XFree86: xc/programs/xdm/config/Xserv.ws.cpp,v 1.1.1.1.12.2 1998/10/04 15:23:14 hohndel Exp $
#
# Xservers file, workstation prototype
#
# This file should contain an entry to start the server on the
# local display; if you have more than one display (not screen),
# you can add entries to the list (one per line).  If you also
# have some X terminals connected which do not support XDMCP,
# you can add them here as well.  Each X terminal line should
# look like:
#       XTerminalName:0 foreign
#
:0 local /bin/nice −n −10 /usr/X11R6/bin/X0 :0 −deferglyphs 16 vt7
:1 local /bin/nice −n −10 /usr/X11R6/bin/X1 :1 −xf86config /etc/X11/XF86Config-4.X1 −deferglyphs 16 vt17
```

(Not really used by me.)

`/etc/X11/xdm/Xservers.2`

```
# $XConsortium: Xserv.ws.cpp,v 1.3 93/09/28 14:30:30 gildea Exp $
#
#
# $XFree86: xc/programs/xdm/config/Xserv.ws.cpp,v 1.1.1.1.12.2 1998/10/04 15:23:14 hohndel Exp $
#
# Xservers file, workstation prototype
#
# This file should contain an entry to start the server on the
# local display; if you have more than one display (not screen),
# you can add entries to the list (one per line).  If you also
# have some X terminals connected which do not support XDMCP,
```

```
# you can add them here as well.  Each X terminal line should
# look like:
#       XTerminalName:0 foreign
#
:0 local /bin/nice -n -10 /usr/X11R6/bin/X0 :0 -deferglyphs 16 vt7
:1 local /bin/nice -n -10 /usr/X11R6/bin/X1 :1 -xf86config /etc/X11/XF86Config-4.X1 -deferglyphs 16 vt17
:2 local /bin/nice -n -10 /usr/X11R6/bin/X2 :2 -xf86config /etc/X11/XF86Config-4.X2 -deferglyphs 16 vt18
```

B.2.2. gdm

/etc/X11/gdm/gdm.conf.0

```
# GDM Configuration file.  You can use gdmsetup program to graphically
# edit this, or you can optionally just edit this file by hand.  Note that
# gdmsetup does not tweak every option here, just the ones most users
# would care about.  Rest is for special setups and distro specific
# tweaks.  If you edit this file, you should send the HUP or USR1 signal to
# the daemon so that it restarts: (Assuming you have not changed PidFile)
#   kill -USR1 `cat /var/run/gdm.pid`
# (HUP will make gdm restart immediately while USR1 will make gdm not kill
# existing sessions and will only restart gdm after all users log out)
#
# You can also use the gdm-restart and gdm-safe-restart scripts which just
# do the above for you.
#
# Have fun! - George

[daemon]
# Automatic login, if true the first local screen will automatically logged
# in as user as set with AutomaticLogin key.
AutomaticLoginEnable=false
AutomaticLogin=
# Timed login, useful for kiosks.  Log in a certain user after a certain
# amount of time
TimedLoginEnable=false
TimedLogin=
TimedLoginDelay=30
# A comma separated list of users that will be logged in without having
# to authenticate on local screens (not over xdmcp).  Note that 'root'
# is ignored and will always have to authenticate.
LocalNoPasswordUsers=
# If you are having trouble with using a single server for a long time and
# want gdm to kill/restart the server, turn this on
AlwaysRestartServer=false
# The gdm configuration program that is run from the login screen, you should
# probably leave this alone
Configurator=/usr/sbin/gdmsetup --disable-sound --disable-crash-dialog
GnomeDefaultSession=/usr/share/gnome/default.session
# The chooser program.  Must output the chosen host on stdout, probably you
# should leave this alone
Chooser=/usr/bin/gdmchooser
# Default path to set.  The profile scripts will likely override this
DefaultPath=/bin:/usr/bin:/usr/bin/X11:/usr/X11R6/bin:/usr/local/bin:/usr/bin
# Default path for root.  The profile scripts will likely override this
RootPath=/sbin:/usr/sbin:/bin:/usr/bin:/usr/bin/X11:/usr/X11R6/bin:/usr/local/bin:/usr/bin
DisplayInitDir=/etc/X11/gdm/Init
# Greeter for local (non-xdmcp) logins.  Change gdmlogin to gdmgreeter to
# get the new graphical greeter.
Greeter=/usr/bin/gdmgreeter
```

```
# Greeter for xdmcp logins, usually you want a less graphically intensive
# greeter here so it's better to leave this with gdmlogin
RemoteGreeter=/usr/bin/gdmlogin
# User and group that gdm should run as.  Probably should be gdm and gdm and
# you should create these user and group.  Anyone found running this as
# someone too privilaged will get a kick in the ass.  This should have
# access to only the gdm directories and files.
User=gdm
Group=gdm
# To try to kill all clients started at greeter time or in the Init script.
# doesn't always work, only if those clients have a window of their own
KillInitClients=true
LogDir=/var/lib/gdm
# You should probably never change this value unless you have a weird setup
PidFile=/var/run/gdm.pid
PostSessionScriptDir=/etc/X11/gdm/PostSession/
PreSessionScriptDir=/etc/X11/gdm/PreSession/
# Distributions:  If you have some script that runs an X server in say
# VGA mode, allowing a login, could you please send it to me?
FailsafeXServer=
# if X keeps crashing on us we run this script.  The default one does a bunch
# of cool stuff to figure out what to tell the user and such and can
# run an X configuration program.
XKeepsCrashing=/etc/X11/gdm/XKeepsCrashing
# Reboot, Halt and suspend commands, you can add different commands
# separated by a semicolon and gdm will use the first one it can find
RebootCommand=/sbin/shutdown -r now;/usr/sbin/shutdown -r now
HaltCommand=/usr/bin/poweroff;/sbin/poweroff;/sbin/shutdown -h now;/usr/sbin/shutdown -h now
SuspendCommand=
# Probably should not touch the below this is the standard setup
ServAuthDir=/var/lib/gdm
SessionDir=/etc/X11/gdm/Sessions/
# Better leave this blank and HOME will be used.  You can use syntax ~/ below
# to indicate home directory of the user
UserAuthDir=
# Fallback if home directory not writable
UserAuthFBDir=/tmp
UserAuthFile=.Xauthority
# The X server to use if we can't figure out what else to run.
StandardXServer=/usr/X11R6/bin/X
# The maximum number of flexible X servers to run.
FlexibleXServers=5
# the X nest command
Xnest=/usr/X11R6/bin/Xnest -name Xnest
# Automatic VT allocation.  Right now only works on Linux.  This way
# we force X to use specific vts.  turn VTAllocation to false if this
# is causing problems.
FirstVT=7
VTAllocation=false

[security]
# If any distributions ship with this one off, they should be shot
# this is only local, so it's only for say kiosk use, when you
# want to minimize possibility of breakin
AllowRoot=true
# If you want to be paranoid, turn this one off
AllowRemoteRoot=true
# This will allow remote timed login
AllowRemoteAutoLogin=false
# 0 is the most anal, 1 allows group write permissions, 2 allows all write permissions
RelaxPermissions=0
RetryDelay=3
```

```
# Maximum size of a file we wish to read.  This makes it hard for a user to DoS us
# by using a large file.
UserMaxFile=65536
# Maximum size of the session file.  This is larger because it matters less as we
# never keep it all in memory.  Just has an upper limit so that we don't go into too
# long of a loop
SessionMaxFile=524388

# XDMCP is the protocol that allows remote login.  If you want to log into
# gdm remotely (I'd never turn this on on open network, use ssh for such
# remote usage that).  You can then run X with -query <thishost> to log in,
# or -indirect <thishost> to run a chooser.  Look for the 'Terminal' server
# type at the bottom of this config file.
[xdmcp]
# Distributions: Ship with this off.  It is never a safe thing to leave
# out on the net.  Alternatively you can set up /etc/hosts.allow and
# /etc/hosts.deny to only allow say local access.
Enable=false
# Honour indirect queries, we run a chooser for these, and then redirect
# the user to the chosen host.  Otherwise we just log the user in locally.
HonorIndirect=true
# Maximum pending requests
MaxPending=4
MaxPendingIndirect=4
# Maximum open XDMCP sessions at any point in time
MaxSessions=16
# Maximum wait times
MaxWait=15
MaxWaitIndirect=15
# How many times can a person log in from a single host.  Usually better to
# keep at 1 to fend off DoS attacks by running many logins from a single
# host
DisplaysPerHost=1
# The port.  177 is the standard port so better keep it that way
Port=177
# Willing script, none is shipped and by default we'll send
# hostname system id.  But if you supply something here, the
# output of this script will be sent as status of this host so that
# the chooser can display it.  You could for example send load,
# or mail details for some user, or some such.
Willing=/etc/X11/gdm/Xwilling

[gui]
# The 'theme'.  By default we're using the default gtk theme
# Of course assuming that gtk got installed in the same prefix,
# if not change this.
GtkRC=/usr/share/themes/Default/gtk/gtkrc
# Maximum size of an icon, larger icons are scaled down
MaxIconWidth=128
MaxIconHeight=128

[greeter]
# Greeter has a nice title bar that the user can move
TitleBar=true
# Configuration is available from the system menu of the greeter
ConfigAvailable=true
# Face browser is enabled.  This only works currently for the
# standard greeter as it is not yet enabled in the graphical greeter.
Browser=true
# The default picture in the browser
DefaultFace=/usr/share/mdk/faces/default.png
# These are things excluded from the face browser, not from logging in
```

```
Exclude=bin,daemon,adm,lp,sync,shutdown,halt,mail,news,uucp,operator,nobody,gdm,postgres,pvm,rpm
# As an alternative to the above this is the minimum uid to show
MinimalUID=500
# If user or user.png exists in this dir it will be used as his picture
GlobalFaceDir=/usr/share/faces/
# Icon we use
Icon=/usr/share/pixmaps/gdm.png
# File which contains the locale we show to the user.  Likely you want to use
# the one shipped with gdm and edit it.  It is not a standard locale.alias file,
# although gdm will be able to read a standard locale.alias file as well.
LocaleFile=/etc/X11/gdm/locale.alias
# Logo shown in the standard greeter
Logo=/usr/share/pixmaps/gdm-screen.png
# The standard greeter should shake if a user entered the wrong username or
# password.  Kind of cool looking
Quiver=true
# The system menu is shown in the greeter
SystemMenu=true
# Note to distributors, if you wish to have a different Welcome string
# and wish to have this translated you can have entries such as
# Welcome[cs]=Vitejte na %n
# Just make sure the string is in utf-8
Welcome=Welcome to %n
# Don't allow user to move the standard greeter window.  Only makes sense
# if TitleBar is on
LockPosition=false
# Set a position rather then just centering the window.  If you enter
# negative values for the position it is taken as an offset from the
# right or bottom edge.
SetPosition=false
PositionX=0
PositionY=0
# Xinerama screen we use to display the greeter on.  Not for true
# multihead, currently only works for Xinerama.
XineramaScreen=0
# Background settings for the standard greeter:
# Type can be 0=None, 1=Image, 2=Color
BackgroundType=2
BackgroundImage=
BackgroundScaleToFit=true
BackgroundColor=#21449c
# XDMCP session should only get a color, this is the sanest setting since
# you don't want to take up too much bandwidth
BackgroundRemoteOnlyColor=true
# Program to run to draw the background in the standard greeter.  Perhaps
# something like an xscreensaver hack or some such.
BackgroundProgram=
# if this is true then the background program is run always, otherwise
# it is only run when the BackgroundType is 0 (None)
RunBackgroundProgramAlways=false
# Show the chooser (you can choose a specific saved gnome session) session
ShowGnomeChooserSession=false
# Show the Failsafe sessions.  These are much MUCH nicer (focus for xterm for
# example) and more failsafe then those supplied by scripts so distros should
# use this rather then just running an xterm from a script.
ShowGnomeFailsafeSession=false
ShowXtermFailsafeSession=false
# Always use 24 hour clock no matter what the locale.
Use24Clock=false
# Use circles in the password field.  Looks kind of cool actually
UseCirclesInEntry=false
# These two keys are for the new greeter.  Circles is the standard
```

```
# shipped theme
GraphicalTheme=mdk
GraphicalThemeDir=/usr/share/gdm/themes/

# The chooser is what's displayed when a user wants an indirect XDMCP
# session
[chooser]
# Default image for hosts
DefaultHostImg=/usr/share/pixmaps/nohost.png
# Directory with host images, they are named by the hosts: host or host.png
HostImageDir=/usr/share/hosts/
# Time we scan for hosts (well only the time we tell the user we are
# scanning actually)
ScanTime=3
# A comma separated lists of hosts to automatically add (if they answer to
# a query of course).  You can use this to reach hosts that broadcast cannot
# reach.
Hosts=
# Broadcast a query to get all hosts on the current network that answer
Broadcast=true

[debug]
# This will enable debugging into the syslog, usually not neccessary
# and it creates a LOT of spew of random stuff to the syslog.  However it
# can be useful in determining when something is going very wrong.
Enable=false

[servers]
# These are the standard servers.  You can add as many you want here
# and they will always be started.  Each line must start with a unique
# number and that will be the display number of that server.  Usually just
# the 0 server is used.
0=Standard
#1=Standard
# Note the VTAllocation and FirstVT keys on Linux.  Don't add any vt<number>
# arguments if VTAllocation is on, and set FirstVT to be the first vt
# available that your gettys don't grab (gettys are usually dumb and grab
# even a vt that has already been taken).  Using 7 will work pretty much for
# all Linux distributions.  VTAllocation is not currently implemented on
# anything but Linux since I don't own any non−Linux systems.  Feel free to
# send patches.  X servers will just not get any extra arguments then.
#
#Note: If you want to run an X terminal you could add an X server such as this
#0=Terminal −query serverhostname
# or for a chooser (optionally serverhostname could be localhost)
#0=Terminal −indirect serverhostname

# Definition of the standard X server.
[server−Standard]
name=Standard server
command=/usr/X11R6/bin/X0 :0 −deferglyphs 16 vt7

flexible=true

# To use this server type you should add −query host or −indirect host
# to the command line
[server−Terminal]
name=Terminal server
# Add −terminate to make things behave more nicely
command=/usr/X11R6/bin/X −terminate
# Make this not appear in the flexible servers (we need extra params
# anyway, and terminate would be bad for xdmcp)
```

```
flexible=false
# Not local, we do not handle the logins for this X server
handled=false
```

/etc/X11/gdm/gdm.conf.1

```
# GDM Configuration file.  You can use gdmsetup program to graphically
# edit this, or you can optionally just edit this file by hand.  Note that
# gdmsetup does not tweak every option here, just the ones most users
# would care about.  Rest is for special setups and distro specific
# tweaks.  If you edit this file, you should send the HUP or USR1 signal to
# the daemon so that it restarts: (Assuming you have not changed PidFile)
#   kill -USR1 `cat /var/run/gdm.pid`
# (HUP will make gdm restart immediately while USR1 will make gdm not kill
# existing sessions and will only restart gdm after all users log out)
#
# You can also use the gdm-restart and gdm-safe-restart scripts which just
# do the above for you.
#
# Have fun! - George

[daemon]
# Automatic login, if true the first local screen will automatically logged
# in as user as set with AutomaticLogin key.
AutomaticLoginEnable=false
AutomaticLogin=
# Timed login, useful for kiosks.  Log in a certain user after a certain
# amount of time
TimedLoginEnable=false
TimedLogin=
TimedLoginDelay=30
# A comma separated list of users that will be logged in without having
# to authenticate on local screens (not over xdmcp).  Note that 'root'
# is ignored and will always have to authenticate.
LocalNoPasswordUsers=
# If you are having trouble with using a single server for a long time and
# want gdm to kill/restart the server, turn this on
AlwaysRestartServer=false
# The gdm configuration program that is run from the login screen, you should
# probably leave this alone
Configurator=/usr/sbin/gdmsetup --disable-sound --disable-crash-dialog
GnomeDefaultSession=/usr/share/gnome/default.session
# The chooser program.  Must output the chosen host on stdout, probably you
# should leave this alone
Chooser=/usr/bin/gdmchooser
# Default path to set.  The profile scripts will likely override this
DefaultPath=/bin:/usr/bin:/usr/bin/X11:/usr/X11R6/bin:/usr/local/bin:/usr/bin
# Default path for root.  The profile scripts will likely override this
RootPath=/sbin:/usr/sbin:/bin:/usr/bin:/usr/bin/X11:/usr/X11R6/bin:/usr/local/bin:/usr/bin
DisplayInitDir=/etc/X11/gdm/Init
# Greeter for local (non-xdmcp) logins.  Change gdmlogin to gdmgreeter to
# get the new graphical greeter.
Greeter=/usr/bin/gdmgreeter
# Greeter for xdmcp logins, usually you want a less graphically intensive
# greeter here so it's better to leave this with gdmlogin
RemoteGreeter=/usr/bin/gdmlogin
# User and group that gdm should run as.  Probably should be gdm and gdm and
# you should create these user and group.  Anyone found running this as
# someone too privilaged will get a kick in the ass.  This should have
# access to only the gdm directories and files.
```

```
User=gdm
Group=gdm
# To try to kill all clients started at greeter time or in the Init script.
# doesn't always work, only if those clients have a window of their own
KillInitClients=true
LogDir=/var/lib/gdm
# You should probably never change this value unless you have a weird setup
PidFile=/var/run/gdm.pid
PostSessionScriptDir=/etc/X11/gdm/PostSession/
PreSessionScriptDir=/etc/X11/gdm/PreSession/
# Distributions:  If you have some script that runs an X server in say
# VGA mode, allowing a login, could you please send it to me?
FailsafeXServer=
# if X keeps crashing on us we run this script.  The default one does a bunch
# of cool stuff to figure out what to tell the user and such and can
# run an X configuration program.
XKeepsCrashing=/etc/X11/gdm/XKeepsCrashing
# Reboot, Halt and suspend commands, you can add different commands
# separated by a semicolon and gdm will use the first one it can find
RebootCommand=/sbin/shutdown −r now;/usr/sbin/shutdown −r now
HaltCommand=/usr/bin/poweroff;/sbin/poweroff;/sbin/shutdown −h now;/usr/sbin/shutdown −h now
SuspendCommand=
# Probably should not touch the below this is the standard setup
ServAuthDir=/var/lib/gdm
SessionDir=/etc/X11/gdm/Sessions/
# Better leave this blank and HOME will be used.  You can use syntax ~/ below
# to indicate home directory of the user
UserAuthDir=
# Fallback if home directory not writable
UserAuthFBDir=/tmp
UserAuthFile=.Xauthority
# The X server to use if we can't figure out what else to run.
StandardXServer=/usr/X11R6/bin/X0
# The maximum number of flexible X servers to run.
FlexibleXServers=5
# the X nest command
Xnest=/usr/X11R6/bin/Xnest −name Xnest
# Automatic VT allocation.  Right now only works on Linux.  This way
# we force X to use specific vts.  turn VTAllocation to false if this
# is causing problems.
FirstVT=7
VTAllocation=false

[security]
# If any distributions ship with this one off, they should be shot
# this is only local, so it's only for say kiosk use, when you
# want to minimize possibility of breakin
AllowRoot=true
# If you want to be paranoid, turn this one off
AllowRemoteRoot=true
# This will allow remote timed login
AllowRemoteAutoLogin=false
# 0 is the most anal, 1 allows group write permissions, 2 allows all write permissions
RelaxPermissions=0
RetryDelay=3
# Maximum size of a file we wish to read.  This makes it hard for a user to DoS us
# by using a large file.
UserMaxFile=65536
# Maximum size of the session file.  This is larger because it matters less as we
# never keep it all in memory.  Just has an upper limit so that we don't go into too
# long of a loop
SessionMaxFile=524388
```

```
# XDMCP is the protocol that allows remote login.  If you want to log into
# gdm remotely (I'd never turn this on on open network, use ssh for such
# remote usage that).  You can then run X with −query <thishost> to log in,
# or −indirect <thishost> to run a chooser.  Look for the 'Terminal' server
# type at the bottom of this config file.
[xdmcp]
# Distributions: Ship with this off.  It is never a safe thing to leave
# out on the net.  Alternatively you can set up /etc/hosts.allow and
# /etc/hosts.deny to only allow say local access.
Enable=false
# Honour indirect queries, we run a chooser for these, and then redirect
# the user to the chosen host.  Otherwise we just log the user in locally.
HonorIndirect=true
# Maximum pending requests
MaxPending=4
MaxPendingIndirect=4
# Maximum open XDMCP sessions at any point in time
MaxSessions=16
# Maximum wait times
MaxWait=15
MaxWaitIndirect=15
# How many times can a person log in from a single host.  Usually better to
# keep at 1 to fend off DoS attacks by running many logins from a single
# host
DisplaysPerHost=1
# The port.  177 is the standard port so better keep it that way
Port=177
# Willing script, none is shipped and by default we'll send
# hostname system id.  But if you supply something here, the
# output of this script will be sent as status of this host so that
# the chooser can display it.  You could for example send load,
# or mail details for some user, or some such.
Willing=/etc/X11/gdm/Xwilling

[gui]
# The 'theme'.  By default we're using the default gtk theme
# Of course assuming that gtk got installed in the same prefix,
# if not change this.
GtkRC=/usr/share/themes/Default/gtk/gtkrc
# Maximum size of an icon, larger icons are scaled down
MaxIconWidth=128
MaxIconHeight=128

[greeter]
# Greeter has a nice title bar that the user can move
TitleBar=true
# Configuration is available from the system menu of the greeter
ConfigAvailable=true
# Face browser is enabled.  This only works currently for the
# standard greeter as it is not yet enabled in the graphical greeter.
Browser=true
# The default picture in the browser
DefaultFace=/usr/share/mdk/faces/default.png
# These are things excluded from the face browser, not from logging in
Exclude=bin,daemon,adm,lp,sync,shutdown,halt,mail,news,uucp,operator,nobody,gdm,postgres,pvm,rpm
# As an alternative to the above this is the minimum uid to show
MinimalUID=500
# If user or user.png exists in this dir it will be used as his picture
GlobalFaceDir=/usr/share/faces/
# Icon we use
Icon=/usr/share/pixmaps/gdm.png
```

```
# File which contains the locale we show to the user.  Likely you want to use
# the one shipped with gdm and edit it.  It is not a standard locale.alias file,
# although gdm will be able to read a standard locale.alias file as well.
LocaleFile=/etc/X11/gdm/locale.alias
# Logo shown in the standard greeter
Logo=/usr/share/pixmaps/gdm-screen.png
# The standard greeter should shake if a user entered the wrong username or
# password.  Kind of cool looking
Quiver=true
# The system menu is shown in the greeter
SystemMenu=true
# Note to distributors, if you wish to have a different Welcome string
# and wish to have this translated you can have entries such as
# Welcome[cs]=Vitejte na %n
# Just make sure the string is in utf-8
Welcome=Welcome to %n
# Don't allow user to move the standard greeter window.  Only makes sense
# if TitleBar is on
LockPosition=false
# Set a position rather then just centering the window.  If you enter
# negative values for the position it is taken as an offset from the
# right or bottom edge.
SetPosition=false
PositionX=0
PositionY=0
# Xinerama screen we use to display the greeter on.  Not for true
# multihead, currently only works for Xinerama.
XineramaScreen=0
# Background settings for the standard greeter:
# Type can be 0=None, 1=Image, 2=Color
BackgroundType=2
BackgroundImage=
BackgroundScaleToFit=true
BackgroundColor=#21449c
# XDMCP session should only get a color, this is the sanest setting since
# you don't want to take up too much bandwidth
BackgroundRemoteOnlyColor=true
# Program to run to draw the background in the standard greeter.  Perhaps
# something like an xscreensaver hack or some such.
BackgroundProgram=
# if this is true then the background program is run always, otherwise
# it is only run when the BackgroundType is 0 (None)
RunBackgroundProgramAlways=false
# Show the chooser (you can choose a specific saved gnome session) session
ShowGnomeChooserSession=false
# Show the Failsafe sessions.  These are much MUCH nicer (focus for xterm for
# example) and more failsafe then those supplied by scripts so distros should
# use this rather then just running an xterm from a script.
ShowGnomeFailsafeSession=false
ShowXtermFailsafeSession=false
# Always use 24 hour clock no matter what the locale.
Use24Clock=false
# Use circles in the password field.  Looks kind of cool actually
UseCirclesInEntry=false
# These two keys are for the new greeter.  Circles is the standard
# shipped theme
GraphicalTheme=mdk
GraphicalThemeDir=/usr/share/gdm/themes/

# The chooser is what's displayed when a user wants an indirect XDMCP
# session
[chooser]
```

```
# Default image for hosts
DefaultHostImg=/usr/share/pixmaps/nohost.png
# Directory with host images, they are named by the hosts: host or host.png
HostImageDir=/usr/share/hosts/
# Time we scan for hosts (well only the time we tell the user we are
# scanning actually)
ScanTime=3
# A comma separated lists of hosts to automatically add (if they answer to
# a query of course).  You can use this to reach hosts that broadcast cannot
# reach.
Hosts=
# Broadcast a query to get all hosts on the current network that answer
Broadcast=true

[debug]
# This will enable debugging into the syslog, usually not neccessary
# and it creates a LOT of spew of random stuff to the syslog.  However it
# can be useful in determining when something is going very wrong.
Enable=false

[servers]
# These are the standard servers.  You can add as many you want here
# and they will always be started.  Each line must start with a unique
# number and that will be the display number of that server.  Usually just
# the 0 server is used.
0=Standard
1=2nd

# Note the VTAllocation and FirstVT keys on Linux.  Don't add any vt<number>
# arguments if VTAllocation is on, and set FirstVT to be the first vt
# available that your gettys don't grab (gettys are usually dumb and grab
# even a vt that has already been taken).  Using 7 will work pretty much for
# all Linux distributions.  VTAllocation is not currently implemented on
# anything but Linux since I don't own any non-Linux systems.  Feel free to
# send patches.  X servers will just not get any extra arguments then.
#
#Note: If you want to run an X terminal you could add an X server such as this
#0=Terminal -query serverhostname
# or for a chooser (optionally serverhostname could be localhost)
#0=Terminal -indirect serverhostname

# Definition of the standard X server.
[server-Standard]
name=Standard server
command=/usr/X11R6/bin/X0 :0 -deferglyphs 16 vt7
flexible=true

# Definition of the second X server.
[server-2nd]
name=2nd server
command=/usr/X11R6/bin/X1 :1 -xf86config /etc/X11/XF86Config-4.X1 -deferglyphs 16 vt17
flexible=true

# To use this server type you should add -query host or -indirect host
# to the command line
[server-Terminal]
name=Terminal server
# Add -terminate to make things behave more nicely
command=/usr/X11R6/bin/X -terminate
# Make this not appear in the flexible servers (we need extra params
# anyway, and terminate would be bad for xdmcp)
flexible=false
```

```
# Not local, we do not handle the logins for this X server
handled=false
```

B.3. Configuration files for Input Agent

B.3.1. Keyboard configuration

/etc/hotplug/kbd.conf

```
#
# keyboard configuretion
#
# vt_name device_physicaly_location
VT0     usb-00:10.1-1.1/input0
VT1     isa0060/serio0/input0
#VT2    usb-*.*-1/input0
```

B.3.2. Mouse configuration

/etc/hotplug/mouse.conf

```
# mouse device configuration
#
# sym_link device_physicaly_location
mouse0br        usb-00:10.1-1.2/input0
mouse2br        usb-00:10.1-2.7.*/input0
mouse1br        usb-00:10.1-2.2/input0
```

B.3.3. Event device configuration

/etc/hotplug/event.conf

```
#
# input event device config file
#
# symbolic_link device_physicaly_location
#event0br       isa0060/serio0/*
#event0br       isa0060/serio1/input0
#event3br       usb-*.*-1.3/*
```

Appendix C. Scripts

C.1. hotplug: `input.agent`

/etc/hotplug/input.agent

```
#!/bin/sh
# Please place this file /etc/hotplug
#
# input-specific hotplug policy agent.
#
# Kernel Input params are:
#
#       ACTION=add
#       PHYS=pysical location of device
#       NAME=Name of the device
#
# HISTORY:
#       15-JUN-2003     removed paste
#       07-MAY-2003     remake by Aivils Stoss
#                       /proc manipulation added
#                       parse kbd.conf event.conf mouse.conf.
#                       create necessary symbolic links
#       28-SEP-2002     Initial version from Andreas Schuldei
#                        andreas (at) schuldei.org
#
cd /etc/hotplug
. hotplug.functions
DEBUG=yes export DEBUG

KBD_CONFIG="./kbd.conf"
EVENT_CONFIG="./event.conf"
MOUSE_CONFIG="./mouse.conf"

setup_kbd ()
{
    while read VT_NUM PHYS_PATTERN NAME_PATTERN
    do
        if [ `echo "$VT_NUM" | grep "^#"` ]; then
            continue;
        fi
        if [ `echo "$PHYS" | grep $PHYS_PATTERN 2>/dev/null` ]; then
            VT=`echo "$VT_NUM" | sed 's/VT//' | awk '{printf "%02d", $0}'`
            if [ -d /proc/bus/console/$VT ]; then
                echo "$PHYS" > "/proc/bus/console/$VT/keyboard"
                debug_mesg "Input device $NAME on $PHYS mapping as secondary to VT$VT"
                return;
            else
                debug_mesg "Trying to configure keyboard for VT$VT , but not enough VT's available"
            fi
        fi
    done
    debug_mesg "Found no fitting VT"
}

setup_event ()
{
    while read SYM_LINK PHYS_PATTERN NAME_PATTERN
    do
```

```
        if [ `echo "$SYM_LINK" | grep "^#"` ]; then
            continue;
        fi
        if [ `echo "$PHYS" | grep $PHYS_PATTERN 2>/dev/null` ]; then
            case $ACTION in
            add)
                cd /dev/input
                rm -f $SYM_LINK
                ln -s $DEV_EVENT $SYM_LINK
                debug_mesg "Input event device $NAME on $PHYS linked to $SYM_LINK"
            ;;
            remove)
                rm -f /dev/input/$SYM_LINK
                debug_mesg "Input event device link $SYM_LINK removed"
            ;;
            esac
            return;
        fi
    done
    debug_mesg "Found no fitting event device"
}

setup_mouse ()
{
    while read SYM_LINK PHYS_PATTERN NAME_PATTERN
    do
        if [ `echo "$SYM_LINK" | grep "^#"` ]; then
            continue;
        fi
        if [ `echo "$PHYS" | grep $PHYS_PATTERN 2>/dev/null` ]; then
            case $ACTION in
            add)
                cd /dev/input
                rm -f $SYM_LINK
                ln -s $DEV_MOUSE $SYM_LINK
                debug_mesg "Input mouse device $NAME on $PHYS linked to $SYM_LINK"
            ;;
            remove)
                rm -f /dev/input/$SYM_LINK
                debug_mesg "Input mouse device link $SYM_LINK removed"
            ;;
            esac
            return;
        fi
    done
    debug_mesg "Found no fitting mouse device"
}

if [ "$ACTION" = "" ]; then
    mesg Bad input agent invocation
    exit 1
fi

DEV_HANDLERS=`grep -E 'Phys|Handlers' /proc/bus/input/devices | \
    awk '{ if(count == 0) { printf("%s\t",$0); count++; } else { print $0; count=0; } }' | \
    grep $PHYS | awk -F\t '{print $2}' | sed 's/^.*=//'`

# older grep do not support -o :-(
#DEV_EVENT=`echo $MATCHED | grep -oE event[0-9]+`
#DEV_MOUSE=`echo $MATCHED | grep -oE mouse[0-9]+`
#DEV_KBD=`echo $MATCHED | grep -o kbd`
```

```
DEV_EVENT=`echo $DEV_HANDLERS | \
            awk -F" " '{for(n=1;$n;n=n+1) if($n ~ /event/) print $n}'`
DEV_MOUSE=`echo $DEV_HANDLERS | \
            awk -F" " '{for(n=1;$n;n=n+1) if($n ~ /mouse/) print $n}'`
DEV_KBD=`echo $DEV_HANDLERS | \
            awk -F" " '{for(n=1;$n;n=n+1) if($n ~ /kbd/) print $n}'`

#
# What to do with this input device event?
#
case "$ACTION" in

add)
    if [ -n "$DEV_KBD" ]; then
        setup_kbd < $KBD_CONFIG
    fi
    if [ -n "$DEV_EVENT" ]; then
        setup_event < $EVENT_CONFIG
    fi
    if [ -n "$DEV_MOUSE" ]; then
        setup_mouse < $MOUSE_CONFIG
    fi
    ;;
remove)
    #setup_event < $EVENT_CONFIG
    #setup_mouse < $MOUSE_CONFIG
    ;;
*)
    debug_mesg "Input '$ACTION' event not supported"
    return 1
    ;;

esac
```

C.2. hotplug: `input.rc`

`/etc/hotplug/input.rc`

```
#!/bin/bash
#
# input.rc      This loads handlers for those input devices
#               that have drivers compiled in kernel
#               Currently stopping is not supported
#
# Best invoked via /etc/init.d/hotplug or equivalent, with
# writable /tmp, /usr mounted, and syslogging active.
#


PATH=/sbin:/bin:/usr/sbin:/usr/bin
PROCDIR=/proc/bus/input

# source function library
if [ -f /etc/init.d/functions ]; then
        . /etc/init.d/functions
elif [ -f /etc/rc.d/init.d/functions ]; then
        . /etc/rc.d/init.d/functions
fi
```

```
if [ -f /etc/hotplug/hotplug.functions ]; then
        . /etc/hotplug/hotplug.functions
fi

input_reset_state () {

    PRODUCT=
    NAME=
    PHYS=
    EV=
    KEY=
    REL=
    ABS=
    MSC=
    LED=
    SND=
    FF=

}

#
# "COLD PLUG" ... load input handlers for compile-in input drivers loaded
# before the OS could really handle hotplug, perhaps because /sbin or
# $HOTPLUG_DIR wasn't available or /tmp wasn't writable.  When/if the
# /sbin/hotplug program is invoked then, hotplug event notifications
# get dropped.  To make up for such "cold boot" errors, we synthesize
# all the hotplug events we expect to have seen already.  They can be
# out of order, and some might be duplicates.
#
input_boot_events ()
{
    if [ ! -r $PROCDIR/devices ]; then
            echo $"** can't synthesize input events - $PROCDIR/devices missing"
            return
    fi

    ACTION=add
    export ACTION

    export PRODUCT NAME PHYS EV KEY REL ABS MSC LED SND FF
    input_reset_state

    declare line

    #
    # the following reads from /proc/bus/input/devices. It is inherently
    # racy (esp. as this file may be changed by input.agent invocation)
    # but so far input devices do not appear in sysfs
    #
    while read line; do
        case "$line" in
            I:* )       # product ID
                eval "${line#I: }"
                PRODUCT=$Bus/$Vendor/$Product/$Version
            ;;
            N:* )       # name
                eval "${line#N: }"
                NAME="$Name"
            ;;
            P:* )       # Physical
                eval "${line#P: }"
                PHYS=$Phys
```

```
        ;;
            B:* )       # Controls supported
                line="${line#B: }"
                eval "${line%%=*}=\"${line#*=}\""
            ;;
            "" )        # End of block
                debug_mesg "Invoking input.agent"
                debug_mesg "PRODUCT=$PRODUCT"
                debug_mesg "NAME=$NAME"
                debug_mesg "PHYS=$PHYS"
                debug_mesg "EV=$EV"
                debug_mesg "KEY=$KEY"
                debug_mesg "REL=$REL"
                debug_mesg "ABS=$ABS"
                debug_mesg "MSC=$MSC"
                debug_mesg "LED=$LED"
                debug_mesg "SND=$SND"
                debug_mesg "FF=$FF"
                /etc/hotplug/input.agent < /dev/null
                input_reset_state
            ;;
        esac
    done < $PROCDIR/devices
}


# See how we were called.
case "$1" in
  start)
        input_boot_events
        ;;
  stop)
        : not supported currently
        ;;
  status)
        echo $"INPUT status for kernel: " `uname −srm`
        echo ''

        echo "INPUT devices:"
        if [ −r $PROCDIR/devices ]; then
            grep "^[INHP]:" $PROCDIR/devices
        else
            echo "$PROCDIR/devices not available"
        fi
        echo ''

        echo "INPUT handlers:"
        if [ −r $PROCDIR/handlers ]; then
            cat $PROCDIR/handlers
        else
            echo "$PROCDIR/handlers not available"
        fi

        echo ''

        ;;
  restart)
        # always invoke by absolute path, else PATH=$PATH:
        $0 stop && $0 start
        ;;
  *)
        echo $"Usage: $0 {start|stop|status|restart}"
```

```
        exit 1
esac
```

C.3. hotplug & XFree supporting event devices: `input.agent`

/etc/hotplug/input.agent

```
#!/bin/bash

cd /etc/hotplug
. hotplug.functions
#DEBUG=yes export DEBUG

export ARG_SEP='\002'
export VAL_SEP='\003'

for dev in `ls /tmp/.X11-unix/ | grep evdev`; do {
    export dev="/tmp/.X11-unix/${dev}"
    /bin/sh -c 'echo -e "2.1${ARG_SEP}NAME${VAL_SEP}${NAME}${ARG_SEP}PHYS${VAL_SEP}${PHYS}${ARG_SEP}ACTION${VAL_SEP}${ACTION}" >> ${dev}' &
}; done
```

C.4. Wrapper for starting X using Nvidia `libGL.so`

```
#!/bin/bash
#########################################################
### /usr/X11R6/bin/XNV                                ###
### script to start XFree with different LIBRARY_PATH ###
### in order to use Nvidia GL libraries and           ###
### XFree GL libraries at the same time               ###
#########################################################

export LD_LIBRARY_PATH=/usr/X11R6/libNV
exec /usr/X11R6/bin/X0 $*
```

C.5. For installing Nvidia drivers for parallel use with DRI

```
#!/bin/bash
########################################################
### /usr/sbin/ruby_NVinstaller                       ###
### for usage call it with -h or --help parameter    ###
########################################################

if [[ -z $1 ]] || [ "$1" == "-h" ] || [ "$1" == "--help" ] ;then
echo
echo "Usage :"
echo "$0 full path to Nvidia installer"
echo "eg. $0 /root/NVIDIA-Linux-x86-1.0-4349.run"
exit 1
fi

if [[ -z $OPENWINHOME ]];then
```

```
echo
echo "OPENWINHOME not defined!!!"
echo "Please set the environment variable OPENWINHOME"
echo "pointing to your XFree prefix eg. /usr/X11R6"
echo "for bash shell :"
echo "export OPENWINHOME=/usr/X11R6"
exit 1
fi

cd $OPENWINHOME
echo
echo "backup file for XFree's GL libraries "
echo "is $OPENWINHOME/libGL-backup.tar"
if [ -x libGL-backup.tar ]; then
        echo "old backup exist, deleting" && rm libGL-backup.tar
fi
echo
find lib -name "libGL.*" -o -name "libGLcore*" \
        -o -name "libglx.*" | xargs tar  rpf   libGL-backup.tar \
        && echo "backup finished"
echo
echo "now running Nvidia installer"
echo "`which $1` --no-opengl-headers --xfree86-prefix=/usr/X11R6NV --opengl-prefix=/usr/X11R6NV"
echo
`which $1` --no-opengl-headers --xfree86-prefix=/usr/X11R6NV --opengl-prefix=/usr/X11R6NV
RETVAL=$?
if [ $RETVAL -eq 0 ]; then

echo
echo "Nvidia installer finished,"
echo "now coping files to /usr/X11R6/libNV/"
echo
cd /usr/X11R6NV/lib &&  tar c * | tar xvC /usr/X11R6/libNV/
echo
echo "restoring backuped GL libraries"
echo
cd $OPENWINHOME
tar xvfp libGL-backup.tar && ldconfig && echo "GL libraries restored" && rm libGL-backup.tar
fi

if [ $RETVAL -ne 0 ]; then
        echo
        echo "installer aborted, not restoring backup"
        echo "deleting backup files" &&  rm libGL-backup.tar
fi

echo
echo "script finished"
```

Appendix D. Experimental Backstreet Ruby configuration Script/ Service

D.1. `README.ruby_init` explains how to configure and use the service.

```
well,
have to be written :-)

You'll need to install the following scripts  :
(and don't forget to make them executable :-) with chmod a+x [filename] )
/etc/hotplug/input.agent
/etc/hotplug/input.rc
/etc/init.d/ruby_init

and the configuration file:
/etc/sysconfig/ruby.conf

edit the configuration file to adjust it to your system configuration

start the service and check if everything is done as requested and configured
/etc/init.d/ruby_init start

if everything is OK activate the service by running "chkconfig --add ruby_init" .

"chkconfig --list ruby_init" will tell you in which runlevels the service is active

root@svetljo SRPMS]# chkconfig --list ruby_init
ruby_init       0:off   1:off   2:on    3:on    4:on    5:on    6:off

you can deactivate it by running "chkconfig --del ruby_init"

The input configuration uses the same syntax
as kbd.conf, mouse.conf, event.conf as explained
in the section about hotplug with input agent.

The other configurable options are:

HACK_VIDEO_ENABLE=no

If set to "yes", turns on at boot the hackvideo feature of the Backstreet Ruby kernel.
In case XFree PrefBusID is used set to "no"

HOTPLUG_RC_input=yes

If set to "yes", runs /etc/hotplug/input.rc to configure input devices which were activated
by the kernel before hotplug was available (drivers statically linked in the kernel).
Generaly needed by all systems unless you have all input drivers configured as loadable modules
and load the drivers after hotplug is accessible to the kernel.

AUTO_DM=no
AUTO_XFree=no

No description yet. Or may be ...
enables/disables the features explained in sections
"7.3. 1st X server configuration file" and "7.4. Number X servers started by Display managers"
you don't need to modify your init scripts if you use the ruby_init service and enable them
```

```
in ruby.conf.

LOAD_MODULES=yes

if set to "yes" loads certain input device drivers in case a ruby kernel is running.

#
# input modules configuration
#
# module arguments

The list of modules which should be loaded in case ruby kernel is running and
LOAD_MODULES is set to "yes".
In the example file are listed all important input drivers (keyboards & mice).
If you have some of them compiled in the kernel, you may disable the loading
of the corresponding drivers by commenting them out.

the ext−status argument can give you detailed information about the current
configuration, and valuable information in case you are in trouble.
```

D.2. The global configuration file `/etc/sysconfig/ruby.conf`

In `/etc/sysconfig/ruby.conf` are stored all configuration options for the Input subsystem of a Backstreet ruby kernel and wether certain actions/ services should be run when such kernel is used.

```
# /etc/sysconfig/ruby.conf
# This file contains defaults for bruby_init

#
# HACK_VIDEO_ENABLE controls wether the ruby kernel shoukd
# ignore certain XFree commands.
#
# Set to yes in case you are not running the XFree−PrefBusID
# but XFree from your distribution
#

HACK_VIDEO_ENABLE=yes

# HOTPLUG_RC_input controls whether the input subsystem is started by
# hotplug rc script ("cold plugging")
#

HOTPLUG_RC_input=yes

#
# AUTO_DM controls wether the display manager configuration files
# is adjusted upon the boot argument dumbcon=[number]
#
# AUTO_XFree controls wether the XFree configuration file is
# adjusted upon XFree=[string] argument.
# For Mandrake users this always activated in /etc/rc.d/rc.sysinit
# and can not be disabled.
#

AUTO_DM=no
AUTO_XFree=no

#
```

```
# LOAD_MODULES controls wether the additional modules listed below
# should be loaded when a ruby kernel is running.

LOAD_MODULES=yes

#
# input modules configuration
#
# module arguments

# needed for all mice
mousedev

# for USB input
hid

# for PS2 input
serio
i8042
atkbd
psmouse

#
# keyboard devices configuration
#
# vt_name device_physicaly_location
VT0     isa0060/serio0/input0
VT1     usb-00:10.1-1.1/input0
#VT0     isa0060/serio0/input0
#VT2    usb-*.*-1/input0

#
# multimedia keys configuration
#
# vt_name device_physicaly_location
#VT1     usb-00:10.1-1.1/input1
#VT2    usb-*.*-1/input0

#
# mouse devices configuration
#
# sym_link device_physicaly_location
mouse0br        isa0060/serio1/input0
mouse1br        usb-00:10.1-1.2/input0
#mouse2br        usb-00:10.1-2.7.*/input0
#mouse1br        usb-00:10.1-2.2/input0

#
# event devices configuration
#
# sym_link device_physicaly_location
#event0br        isa0060/serio1/input0
#event1br        usb-00:10.1-1.1/input0
#event2br        usb-00:10.1-1.1/input1
#event3br        usb-00:10.1-1.2/input0

#
# end input device configuration
#
```

D.3. The ruby_init service `/etc/init.d/ruby_init`

The `/etc/init.d/ruby_init` service configures/ activates most of the settings specific to a Backstreet Ruby system.

```
#!/bin/sh
#
# ruby_init  This scripts configures cold-pluging for bruby, \
#            loads additional input modules, manages the XFree \
#            and display manager configuration files
#
# chkconfig: 2345 02 98
# description:  Configures the Bruby input subsystem and manages \
#               XFree and display manager configuration files.
# config: /etc/sysconfig/ruby.conf
#
# TODO
#       * handle commented out "PciOsConfig"
#       *? status/ ext-status
#       - show config
#       - check wether curr. configuration matches setup ?
#       - make it work when hotplug files not installed ?
#         ( currently it will just just inform that they are not installed
#         and exit )
#
# $Id: XFree-Local-multi-user-HOWTO.xml,v 1.12 2003/11/11 14:52:42 gferg Exp $
# - don't exit silently if hackvideo or modules loading is deactivated,
#   run the script to the end
# $Id: XFree-Local-multi-user-HOWTO.xml,v 1.12 2003/11/11 14:52:42 gferg Exp $
# - show keyboard status per VT ,use sed instead of head(head is in /usr)
# - *? status/ ext-status
# - (mice & evdev links)?
# $Id: XFree-Local-multi-user-HOWTO.xml,v 1.12 2003/11/11 14:52:42 gferg Exp $
# - include hackvideo handling (long time wondering what was missing)
# $Id: XFree-Local-multi-user-HOWTO.xml,v 1.12 2003/11/11 14:52:42 gferg Exp $
# - got it actually running
# - use >& /dev/null to load variables from the config file
#   (what a mess)
# $Id: XFree-Local-multi-user-HOWTO.xml,v 1.12 2003/11/11 14:52:42 gferg Exp $
# - initial release

PATH=/sbin:/bin:/usr/sbin:/usr/bin

# source function library
if [ -f /etc/init.d/functions ]; then
. /etc/init.d/functions
elif [ -f /etc/rc.d/init.d/functions ]; then
        . /etc/rc.d/init.d/functions
fi

# source defaults
if [ -f /etc/sysconfig/ruby.conf ]; then
        #WTF, how to get rid of the ".... >& /dev/null"
        . /etc/sysconfig/ruby.conf >& /dev/null
        RUBY_CONF="/etc/sysconfig/ruby.conf"
else
        action "ruby_init: Configuration file missing" /bin/false
        exit 1
fi
```

```
function get_status() {
                run_hackvideo status
#               prefbus_used ?
#               print dumb_con=?
                get_kbds        #and curr. active VTs
}


function get_ext_status() {
                run_input_rc status
                run_hackvideo status
#               prefbus_used ?
#               print dumb_con=?
                get_kbds        #and curr. active VTs
                get_links mouse
                get_links event
}

function run_start_restart() {
                load_modules
                run_hackvideo start
                run_input_rc start
                run_auto_dm
                run_auto_xfree
}

function run_input_rc () {
    for RC in /etc/hotplug/input.rc
    do
        eval "doit=\"\$HOTPLUG_RC_input\""
        if [ "$doit" != yes -a "$1" != status ]; then
            continue
        fi
        $RC $1
        if [ "$1" != status ]; then
           action "bruby: configuring cold-pluged devices." $RC $1
        fi
    done
}


function run_hackvideo () {
        eval "doit=\"\$HACK_VIDEO_ENABLE\""
        if [ "$doit" != yes ]; then
            if [ "$1" = status ]; then
                echo "XFree hackvideo not configured."
            fi
            return ;
        fi
        if [ -f /proc/bus/pci/hackvideo ]; then
           if [ -f /etc/X11/XF86Config-4 ]; then
                   xf_file="/etc/X11/XF86Config-4"
           elif [ -f /etc/X11/XF86Config ]; then
                   xf_file="/etc/X11/XF86Config"
           else
                   action "bruby: XFree configuration file not found" /bin/false
                   exit 1
           fi
           #fi
           eval "xf_hackvideo=`sed -n '/^#/d;s/^.*"PciOsConfig"[        ]//p' $xf_file | sed -e 's/[    ]//g'`"
           if [ "$xf_hackvideo" = "1" ]; then
                case "$1" in
```

```
                        start)
                                action "bruby: Enabling XFree hackvideo workaound." /bin/true
                                /bin/echo "1" > /proc/bus/pci/hackvideo
                                ;;
                        stop)
                                action "bruby: Disabling XFree hackvideo workaound."
                                /bin/echo "0" > /proc/bus/pci/hackvideo
                                ;;
                        status)
                                eval "hack_enabled=`cat /proc/bus/pci/hackvideo`"
                                if [ $hack_enabled = 1 ]; then
                                    echo "XFree hackvideo activated"
                                else
                                    echo "XFree hackvideo not activated,"
                                    echo "but enabled in configuration. "
                                fi
                                ;;
                        *)
                                ;;
                esac
           else
                           action "bruby: Hackvideo not configured in XFree," /bin/false
                           action "bruby: but enabled in $RUBY_CONF." /bin/false

           fi
        else
                                action "bruby: XFree hackvideo configured," /bin/false
                                action "bruby: but kernel hackvideo support missing." /bin/false
        fi

}

function load_modules () {
        eval "doit=\"\$LOAD_MODULES\""
        if [ "$doit" != yes -a "$1" != status ]; then
            return ;
        fi
        action "bruby: loading additional modules." /bin/true
        cat $RUBY_CONF | sed -n "/input modules /,/config/p" | while read module args
        do
          case "$module" in
              \#*|"") continue ;;
          esac
          initlog -s "Loading module: $module"
          modprobe $module $args >/dev/null 2>&1
        done
}

function run_auto_xfree () {
        eval "doit=\"\$AUTO_XFree\""
        if [ "$doit" != yes -a "$1" != status ]; then
            return ;
        fi
        # (pixel) a kind of profile for XF86Config
        # if no XFree=XXX given on kernel command-line, restore XF86Config.standard
        for i in XF86Config XF86Config-4; do
           if [ -L "/etc/X11/$i" ]; then
              XFree=`sed -n 's/.*XFree=\(\w*\).*/\1/p' /proc/cmdline`
              [ -n "$XFree" ] || XFree=standard
              [ -r "/etc/X11/$i.$XFree" ] && ln -sf "$i.$XFree" "/etc/X11/$i"
          fi
        done
```

```
        action "bruby: configuring XFree." /bin/true
}

function run_auto_dm () {
        eval "doit=\"\$AUTO_DM\""
        if [ "$doit" != yes -a "$1" != status ]; then
            return ;
        fi
        #
        #the same like XF86Config but for gdm.conf & Xservers
        #
        for i in xdm/Xservers gdm/gdm.conf; do
           if [ -L "/etc/X11/$i" ]; then
                DumbCon=`sed -n 's/.*dumbcon=\([0-9]*\).*/\1/p' /proc/cmdline`
                [ -n "$DumbCon" ] || DumbCon=0
                [ -r "/etc/X11/$i.$DumbCon" ] && ln -sf "/etc/X11/$i.$DumbCon" "/etc/X11/$i"
           fi
        done
        action "bruby: configuring display manager." /bin/true
        echo "Setting up display managers for `expr $DumbCon + 1` Xservers"
}


function get_kbds() {
        j=0
        for i in /proc/bus/console/*
           do
                echo
                j=`expr $j + 1`
                phys=`cat $i/keyboard`
                if [ "$phys" = "" ];then
                   echo " VT-`basename $i` : keyboard not attached"
                else
                   echo " VT-`basename $i` using :"
#                  grep -n2 "$phys" /proc/bus/input/devices | sed -e '{/^B: /s;^B:.*;;};{s;^.*: ;;};' -e "/EV=/d"
                   grep -B2 -A1 "$phys" /proc/bus/input/devices | sed 's;^.*: ;;'
                fi
           done
        echo
        echo "Total of $j VT's avialable."
}

function get_links() {
                echo
                for i in  /dev/input/$1*br
                    do
                        real=`ls -l $i 2>/dev/null | sed "{s;^.*/dev/input/;;};{s; ->;;}" | cut -d " " -f 2`
                        if [ "$real" != "" ]; then
                           echo "$i"
                           sed -e '/Name/,/Handl/s/.*: //;{/^B: /s;^B:.*;;};'  /proc/bus/input/devices | grep -B2 -A1 "$real"
                        else
                            exit 0
                        fi
                    done
}

if [ -f /proc/bus/console  -o  -n tmp=`uname -r | sed -n 's:ruby::p'` ]; then
   if [ -x /etc/hotplug/input.rc  -a  -f /etc/hotplug/input.agent ]; then

        case "$1" in
                start|restart)
                        run_start_restart
```

```
                        ;;
                status)
                        get_status
                        touch /var/lock/subsys/ruby
                        ;;
                ext-status)
                        get_ext_status
                        touch /var/lock/subsys/ruby
                        ;;
                stop)
                        #run_hackvideo stop
                        run_input_rc stop
                        rm -f /var/lock/subsys/ruby
                        ;;
                force-reload)
                        run_input_rc stop
                        run_input_rc start
                        touch /var/lock/subsys/ruby
                        ;;

                *)
                        gprintf "Usage: %s {start|stop|restart|status|ext-status|force_reload}\n" "$0"
                        exit 3
                        ;;
        esac

   else if [ -f /etc/hotplug/input.rc -a -f /etc/hotplug/input.agent ]; then
           action "Input: input.rc and input.agent installed, but not executable." /bin/false
           action "Input: Please check the file permissions." /bin/false
        else
           action "Input: Failed to configure cold plugged devices."  /bin/false
           action "Input: input.rc or input.agent missing."  /bin/false
        fi
   fi
fi
```

D.4. Modified hotplug input.agent `/etc/hotplug/input.agent`

This `/etc/hotplug/input.agent` is modified to use the global configuration file.

```
#!/bin/sh
# Please place this file /etc/hotplug
#
# input-specific hotplug policy agent.
#
# Kernel Input params are:
#
#       ACTION=add
#       PHYS=pysical location of device
#       NAME=Name of the device
#
# HISTORY:
#       15-Sep-2003     modified to use single configuration file
#                       /etc/sysconfig/ruby.conf by me :-)
#                       (Svetoslav Slavtchev)
#                       added handling for multimedia keys
#                       but doesn't work as expected :(
#       15-JUN-2003     removed paste
```

```
#       07-MAY-2003     remake by Aivils Stoss
#                       /proc manipulation added
#                       parse kbd.conf event.conf mouse.conf.
#                       create necessary symbolic links
#       28-SEP-2002     Initial version from Andreas Schuldei
#                        andreas (at) schuldei.org
#
cd /etc/hotplug
. hotplug.functions
DEBUG=yes export DEBUG

RUBY_CONF="/etc/sysconfig/ruby.conf"

setup_kbd ()
{
    while read VT_NUM PHYS_PATTERN NAME_PATTERN
    do
        if [ `echo "$VT_NUM" | grep "^#"` ]; then
            continue;
        fi
        if [ `echo "$PHYS" | grep $PHYS_PATTERN 2>/dev/null` ]; then
            VT=`echo "$VT_NUM" | sed 's/VT//' | awk '{printf "%02d", $0}'`
            if [ -d /proc/bus/console/$VT ]; then
                echo "$PHYS" > "/proc/bus/console/$VT/keyboard"
                debug_mesg "Input device $NAME on $PHYS mapping as secondary to VT$VT"
                return;
            else
                debug_mesg "Trying to configure keyboard for VT$VT , but not enough VT's available"
            fi
        fi
    done
    debug_mesg "Found no fitting VT"
}


setup_mm_keys ()
{
    while read VT_NUM PHYS_PATTERN NAME_PATTERN
    do
        if [ `echo "$VT_NUM" | grep "^#"` ]; then
            continue;
        fi
        if [ `echo "$PHYS" | grep $PHYS_PATTERN 2>/dev/null` ]; then
            VT=`echo "$VT_NUM" | sed 's/VT//' | awk '{printf "%02d", $0}'`
            if [ -d /proc/bus/console/$VT ]; then
                echo "+$PHYS" > "/proc/bus/console/$VT/keyboard"
                debug_mesg "Input device $NAME on $PHYS mapping as secondary to VT$VT"
                return;
            else
                debug_mesg "Trying to configure keyboard for VT$VT , but not enough VT's available"
            fi
        fi
    done
    debug_mesg "Found no fitting VT"
}

setup_event ()
{
    while read SYM_LINK PHYS_PATTERN NAME_PATTERN
    do
        if [ `echo "$SYM_LINK" | grep "^#"` ]; then
            continue;
```

```
        fi
        if [ `echo "$PHYS" | grep $PHYS_PATTERN 2>/dev/null` ]; then
            case $ACTION in
            add)
                cd /dev/input
                rm −f $SYM_LINK
                ln −s $DEV_EVENT $SYM_LINK
                debug_mesg "Input event device $NAME on $PHYS linked to $SYM_LINK"
            ;;
            remove)
                rm −f /dev/input/$SYM_LINK
                debug_mesg "Input event device link $SYM_LINK removed"
            ;;
            esac
            return;
        fi
    done
    debug_mesg "Found no fitting event device"
}

setup_mouse ()
{
    while read SYM_LINK PHYS_PATTERN NAME_PATTERN
    do
        if [ `echo "$SYM_LINK" | grep "^#"` ]; then
            continue;
        fi
        if [ `echo "$PHYS" | grep $PHYS_PATTERN 2>/dev/null` ]; then
            case $ACTION in
            add)
                cd /dev/input
                rm −f $SYM_LINK
                ln −s $DEV_MOUSE $SYM_LINK
                debug_mesg "Input mouse device $NAME on $PHYS linked to $SYM_LINK"
            ;;
            remove)
                rm −f /dev/input/$SYM_LINK
                debug_mesg "Input mouse device link $SYM_LINK removed"
            ;;
            esac
            return;
        fi
    done
    debug_mesg "Found no fitting mouse device"
}

setup_input ()
{
    if [ −n "$DEV_KBD" ]; then
        sed −n "/keyboard devices/,/config/p" $RUBY_CONF        | setup_kbd
        sed −n "/multimedia keys/,/config/p" $RUBY_CONF         | setup_mm_keys
    fi
    if [ −n "$DEV_EVENT" ]; then
        sed −n "/event devices/,/config/p" $RUBY_CONF           | setup_event
    fi
    if [ −n "$DEV_MOUSE" ]; then
        sed −n "/mouse devices/,/config/p" $RUBY_CONF           | setup_mouse
    fi
}

if [ "$ACTION" = "" ]; then
    mesg Bad input agent invocation
```

```
    exit 1
fi

DEV_HANDLERS=`grep −E 'Phys|Handlers' /proc/bus/input/devices | \
    awk '{ if(count == 0) { printf("%s\t",$0); count++; } else { print $0; count=0; } }' | \
    grep $PHYS | awk −F\t '{print $2}' | sed 's/^.*=//'`

DEV_EVENT=`echo $DEV_HANDLERS | \
            awk −F" " '{for(n=1;$n;n=n+1) if($n ~ /event/) print $n}'`
DEV_MOUSE=`echo $DEV_HANDLERS | \
            awk −F" " '{for(n=1;$n;n=n+1) if($n ~ /mouse/) print $n}'`
DEV_KBD=`echo $DEV_HANDLERS | \
            awk −F" " '{for(n=1;$n;n=n+1) if($n ~ /kbd/) print $n}'`

#
# What to do with this input device event?
#
case "$ACTION" in

add)
    setup_input
    ;;
remove)
    #setup_input
    ;;
*)
    debug_mesg "Input '$ACTION' event not supported"
    return 1
    ;;

esac
```

Notes

[1] All the features in Backstreet Ruby are included ("/proc " interface & hot−plugging, video hack, ...), support for Framebuffer devices, support for single Framebuffer console which takes over the VGA console (support for multiple independent Framebuffer consoles is not yet implemented)