Best viewed on NetFront (640x480)

my Zaurus SL-C3000 and SL-C3100

pdaXrom Customisations:

This page is discontinued...

Please see my new page titled pdaXii13

The pdaXrom is a complete X windows environment for the Zaurus. It makes the C3000 and C3100 a true mini laptop.

Note: Development for 2.4.20 kernel based pdaXrom 1.1.0 beta1/beta3 has been discontinued by the developers in favour of a 2.6.x kernel based release.

The following section covers implementing workarounds for bugs and improvements to the 2.4.20 kernel based system. Most of these customisations should be generic for pdaXrom running on any Zaurus model. However, there are also a few C3000 specific customisations that should not be applied to the C3100 or any other models even though these two models are very similar hardware-wise. The major difference between the C3000 version of pdaXrom to other versions is that pdaXrom does not reside on the flash but rather is entirely located on the internal MicroDrive using a pivot boot feature.

Some of the customisations require additional files which should be downloaded first and placed on a SD or CF card for easy transfer to the Zaurus.

Fixing the boot process

There are several things during the boot process that can be fixed.

The first and most important one is the pivot boot feature that comes with the C3000 beta2 release of pdaXrom 1.1.0 beta1. The improved fix allows the C3000 to boot even if an external CF is inserted during the boot process and allows both the external and internal CFs to be mounted properly.

```
# cp init /media/realroot/sbin/init
# chmod 755 /media/realroot/sbin/init
# cp halt /etc/rc.d/init.d/halt
# chmod 755 /etc/rc.d/init.d/halt
# cp ide.opts /etc/pcmcia/ide.opts
```

It probably also is a good idea to run fsck before mounting the disk just in case. The suspend feature also seems to be happier on a clean disk. Often failure to suspend is

corrected after the disk is fscked cleanly. The init script has been enhanced to do it if it can find fsck which you can copy to the initial root partition as follows:

```
# cd /media/realroot/lib
# cp /lib/libblkid.so.1.0 .
# ln -s libblkid.so.1.0 libblkid.so.1
# cp /lib/libuuid.so.1.2 .
# ln -s libuuid.so.1.2 libuuid.so.1
# cp /lib/libext2fs.so.2.4 .
# ln -s libext2fs.so.2.4 libext2fs.so.2
# cp /lib/libcom_err.so.2.1 .
# ln -s libcom_err.so.2.1 libcom_err.so.2
# cd /media/realroot/sbin
# cp /sbin/fsck .
# cp /sbin/e2fsck .
# ln -s e2fsck fsck.ext2
# ln -s e2fsck fsck.ext3
```

And finally, there are some minor fixes to eliminate and hide some annoying bootup error messages.

Hiding the *cardservices cannot find modules* message

```
# mkdir -p /media/realroot/lib/modules/2.4.20
```

Start X GUI automatically

Usually, runlevel 3 means booting into the console and runlevel 5 is for GUI mode. Since pdaXrom boots into runlevel 5, it should start X windows. The following fix will start X windows automatically during boot.

```
# head -`wc -l /etc/rc.d/rc|awk '{print $1-1}'` /etc/rc.d/rc > /etc/rc.d/rc.new
# echo "if [ \$runlevel -eq 5 ]; then" >> /etc/rc.d/rc.new
# echo " export HOME=/home/root" >> /etc/rc.d/rc.new
# echo " /usr/X11R6/bin/startx" >> /etc/rc.d/rc.new
# echo "fi" >> /etc/rc.d/rc.new
# echo "exit 0" >> /etc/rc.d/rc.new
# mv /etc/rc.d/rc.new /etc/rc.d/rc
```

The above has the same effect as adding the following towards the end of */etc/rc.d/rc*, but before the line that reads *exit 0* :

```
if [ $runlevel -eq 5 ]; then
```

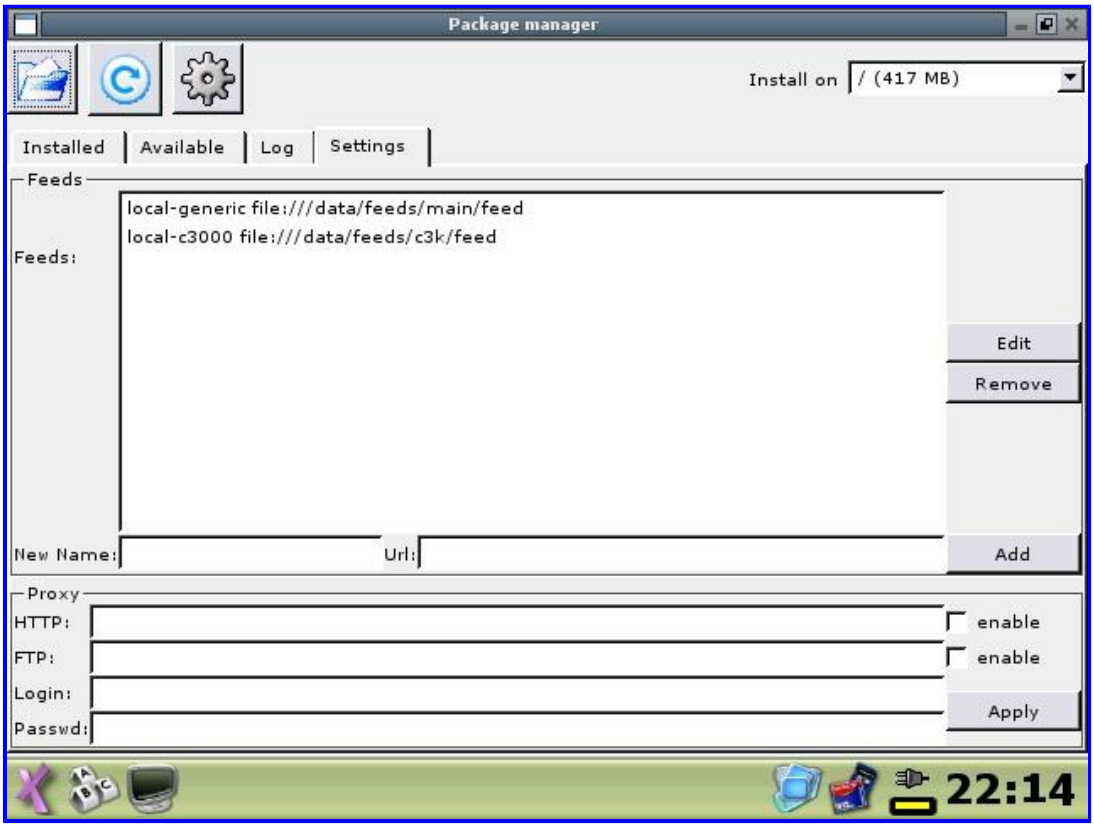
```
export HOME=/home/root
/usr/X11R6/bin/startx
fi
```

Configuring Feeds and Installing Packages

The default pre-configured feed for the package manager is very out of date and needs to be replaced. There are two feeds for the C3000, a generic feed which is common to most Zauri models and a C3x00 specific one:

- http://mail.pdaxrom.org/download/1.1.0beta1/Zaurus-C1000-C3100/feed/
- http://mail.pdaxrom.org/download/1.1.0beta1/Zaurus-7x0-860/feed/

Alternatively, you can also just download both of the feed.tar.gz files and extract the content to a card or the internal disk. Then point the package manager at the directory containing the feed contents (eg: file:///home/root/Documents/installs/C3000/feeds).



You can also install individual packages without requiring a feed, but you will need to manually install dependant packages yourself. You can either use the GUI Package Manager or use the command line **ipkg** command.

For local installs, you can also merge the two feeds (and any additional ones) onto your local disk. Just extract the files into a directory (extract the 7x0 archive first and then the C3100 one on top of it to eliminate the duplicates). Then use **ipkg-make-index** to generate

a new *Packages* file for the feed. I have included a copy of ipkg-make-index in my ipktools package [ipktools_0.2_armv5tel.ipk].

Additional Packages

There are many more packages beside the ones on the official feeds. Many user contributed packages exist and can also be installed. Below is a summary of additional packages that I installed from both the feeds and some were installed manually as stand alone packages:

required libraries:

- libstdcxx5-compat (needed for older applications)

useful command line utilities:

- vim (replace vi with vim because vi locks up all the time)
- lynx (great text based browser)
- aumix (volume controller)
- mplayer (video player)
- flite (speech synthesizer)
- bzip2 (compression utility)
- unrar (uncompression utility)
- unzip (uncompression utility)
- diffutils (text comparison utility)
- aspell (spelling engine)
- imagemagic (image conversion)
- kismet (wifi detector)

engines and runtimes:

- samba (file sharing)
- perl (interpretor)
- dosbox (DOS emulator)
- x11vnc (vnc server)
- java (classpath and jamvm)

games:

- doom2 (prboom doom engine)
- quake2 (quake2 engine)
- supertux

applications:

- gtkterm2 (tabbed terminal emulator)
- bluefish (HTML editor)
- tuxpaint (graphics)
- inkscape (graphics)
- xmmsplayer (video player plugin for xmms)

- gaim (IM client)
- tightvnc (vnc client)
- rox (file manager)
- emelfm2 (file manager)
- xarchiver (file compressor/decompressor GUI)
- gthumb (image viewer)
- gftp (ftp client)
- qbedic (dictionary)
- stardict (dictionary)
- imposter (presentation)

Some of the packages require further customisation and configuration before they can be used. Later sections of this guide show how those need to be configured.

Fixing the keymap

There are two types of changes to the keymappings. The first one are the essential fixes to make the keymap work, and the second ones are enhancements to make it more usable and similar to the keyhelper enhancements on the Sharp distro.

To customise or fix the default keymap, modify /etc/sysconfig/keyboard/kernel.map and /etc/X11/kb/akita.xmodmap and use *loadkeys* and *xmodmap* to reload the key mappings respectively. Use *xev* within X to determine the keycode for the keys.

You can also use the Input setup tool from the Systems Tools to remap the shortcut keys at the bottom of the keyboard. However, this seems to mess up the default keymapping, in particular, you will lose the . key so make sure you make a copy of kernel.map before you use the Input setup tool and then use loadkeys to reload kernel.map.

To enable sticky keys, you need to modify /usr/X11/bin/startx and replace **-kb** with **+kb +accessx** under the defaultserverargs variable. Also modify /etc/X11/kb/akita.xmodmap and uncomment the block of code around **keycode=73** except the last line. This will disable Tab to toggle Caps Lock. Now restart X and press the Shift key 5 times consecutively and sticky keys are enabled.

The keys are mapped as displayed on the keyboard except for some additional keys which have also been mapped but are not marked on the keyboard. Also some keys have been remapped as follows:

- Fn+q = Scandinavian A with loop
- Fn+o = Danish O crossed out
- Fn+p = British Pound sign
- Fn+a = AE character
- Fn+s = Euro symbol
- Fn+m = Backtick (accent grave)
- LeftKanji = Alt
- RightKanji = Backtick (same as Fn+m)
- Calendar = Control
- Address = Alt

I rewrote akita.xmodmap to mirror the Spitz/Borzoi keyboard according to the values returned by *xev*. The result is that each key on the keyboard is now clearly mapped and reflected in the new file that I called **spitz.xmodmap**. I also removed the duplicate mapping for Alt and Control from the Calendar and Address keys so that they can be reassigned, and also swapped Fn+m and Fn+q around. The right kanji key has also been remapped as a modifier key (Super L).

By defeault, there are also several predefined key combos which do the following:

- Ctrl+Alt+BS = Shutdown X
- Ctrl+Alt+Left = Switch to previous Virtual Desktop
- Ctrl+Alt+Right = Switch to next Virtual Desktop
- Shift+Alt+Left = Send to previous Desktop
- Shift+Alt+Right = Send to next Desktop
- Ctrl+Alt+d = Toggle Show Desktop
- Ctrl+Alt+m = Toggle Maximize/Restore Window
- Fn+Ctrl+Alt+Down = MoveRelativeVert
- Fn+Ctrl+Alt+Up = MoveRelativeVert
- Fn+Ctrl+Alt+Left = MoveRelativeHorz
- Fn+Ctrl+Alt+Right = MoveRelativeHorz
- Alt+Tab = Switch forward between running Applications
- Alt+Shift+Tab = Switch backward between running Applications
- Alt+0 = MaximizeFull
- Alt+5 = UnmaximizeFull
- Alt+9 = Iconify
- Alt+8 = Resize
- Alt+7 = Move
- Alt+4 = Close
- Menu = Activate Launch Menu
- Shift+Home = suspend
- Mail = sylpheed

I have assigned the Calendar and Address keys as follows:

- Calendar = aterm
- Address = xmms

Unfortunately, the following mapped keys did not work:

- Fn+1 = VGA mode (320x240)
- Fn+2 = SVGA mode (640x480)
- Fn+3 = Decrease Brightness
- Fn+4 = Increase Brightness
- Alt+F12 = ToggleShade

Hence I changed the mapping so that the Super key can be used instead of the Fn key. I also remapped Alt+F12 to Alt+6.

To reduce finger movements, I also added extra mappings so that instead of pressing Fn+Ctrl+Alt, you can also use the Home key instead, and instead of Ctrl+Alt, you can use the Super key (right kanji key). I also associated most Alt+key combos to be also activated

by Home+key as well.

In addition, I have added the following shortcuts:

- Calendar = aterm
- Shift+Calendar = xterm
- Fn+Calendar = network config
- Ctrl+Calendar = usb config
- Alt+Calendar = ppp config
- Super+Calendar = installer
- Home+Calendar = dialer
- Address = xmms
- Shift+Address = keyboard
- Fn+Address = mixer
- Ctrl+Address = calculator
- Alt+Address = image viewer
- Super+Address = pdf viewer
- Home+Address = gimp
- Mail = sylpheed
- Shift+Mail = firefox
- Fn+Mail = thunderbird
- Ctrl+Mail = dillo
- Alt+Mail = xchat
- Super+Mail = gnumeric
- Home+Mail = abiword
- Ctrl+Alt+c = screencap
- Home+5 = Refresh Screen

The above changes can be done by modifying *rc.xml* which is by default located under */etc/xdg/openbox/*, however, it may also be under /home/root/.config/openbox/

For XMMS, the following keys can be used to control it:

- X = Play
- C = Pause
- V = Stop
- Up = Increase Volume
- Down = Decrease Volume
- Left = Skip Backward
- Right = Skip Forward
- Z = Prev Song
- B = Next Song

However, those keys aren't very convennient especially if you use the Zaurus in portrait mode and are using the buttons and scrollwheel at the back. Because of that, I modified xmms to be more Zaurus friendly so that you can use OK to play, Cancel to stop, and Up and Down to switch between tracks. I have reassigned N and M to be the volume keys.

You can either replace the **xmms** binary with my hacked version if you have already installed xmms, or install xmms_1.2.10-2_armv5tel.ipk instead.

xpdf has the following key mappings:

- o = open file
- r = reload
- f = find next
- n = next page
- p + previous page
- + = zoom in
- - = zoom out
- z = zoom page
- w = zoom page width
- 0 = zoom normal
- q = quit

You can either replace the **xpdf** binary with my hacked version if you have already installed xpdf, or install xpdf_3.01-2_armv5tel.ipk instead.

When you plug in a USB keyboard, you can automatically remap the keys so that the keys on the external keyboard are mapped correctly, but it will result in the Zaurus keyboard being mismapped. When the external keyboard is unplugged, the keymapping for the zaurus keyboard is restored.

This can be done by creating a script in the */etc/hotplug* directory and catching the USB event to detect the insertion and removal of a USB keyboard and running **xmodmap** to load the appropriate keyboard map file.

Screen Resolution

The screen resolution can be switched between 640x480 and 320x240 using **xrandr** command. It can also be used to rotate the screen.

The *Home* + *1* and *Super* + *1* key combos should switch the screen to 320x240 if my custom keymapping is applied. Similarly, *Home* + *2* and *Super* + *2* will switch to 640x480 mode.

Screen Rotation

Screen rotation works on pdaXrom. Openbox will detect transformation between laptop mode (landscape) and PDA mode (portrait) and rotate the screen appropriately. You will end up with the taskbar at the correct location for the orientation you are in. However, the currently running applications are unaware of the rotation and their window size remain unchanged. I have written a script, **rotateall**, which rotates all the application windows to adjust them to the new orientation. To automatically invoke the script when the Zaurus is rotated, modify rc.xml (/etc/xdg/openbox/rc.xml) and replace *chkhinge* with *rotateall*. For best results, run the applications in full screen mode, ie maximised. I have written the script in such a way that if you run **rotateall** from a console without the screen actually rotated it will rotate the screen to the right instead of to the left. Running the script again will restore the previous orientation.

Screen Capture

A screenshot can be taken by running xwd -display :0 > screendump.xwd

A smarter script can be written that mimics the capture behaviour on the Sharp distro, ie bind the script to a key combo (eg Ctrl+Alt+c) and play a shutter sound before doing the screen dump. Also the filenaming and sequencing can also be made to mimic the Sharp snapshot tool. See the **screencap** script.

The screencap script I have written, also calls a .wav file (click.wav) to make a sound before taking the snapshots. If you install ImageMagick as well, then it will be able to convert the xwd image into png automatically.

Window Managers

pdaXrom uses openbox as its default window manager because it is lightweight and loads pretty fast.

However, there are many other alternate window managers (wm) that can be installed and used as well such as xfce, blackbox, fluxbox, matchbox, icewm, kde and enlightment. You can also setup multiple window managers and switch between them as long as you have sufficient space.

The default openbox window manager uses matchbox panels and applets for the taskbar and desktop management.

kde

KDE is also very popular, however, it is quite heavy and thus a bit slower. The simplest way of getting kde 3.5 installed is to use the kde squashfs image. Copy it to the internal harddrive and mount it as a loop device under /usr/lib/kde/kde/kde3.5. Then setup the symbolic links by installing the kdesymlink package.

Now modify .xinitrc and startx to run the selected default window manager. My customised scripts allow you to run either window managers depending on the WM variable, ie ob or kde. You can also start kde by running startxkde.

icewm

xfce

enlightment

Desktop and Menu Customisation

The .desktop files defining the applications are located under */usr/share/applications* and the icons are under */usr/share/pixmaps*

As mentioned before, the desktop is by default managed by matchbox desktop. You can either access the application icons via the menu or the desktop panels.

However, you can replace the matchbox desktop with rox filer to get an integrated file manager and desktop manager. To do that, modify */home/root/.xinitrc* and replace the following:

```
eval "matchbox-desktop $MBBGND" 2>/dev/null >/dev/null &
```

with this:

```
eval "rox --pinboard=MYPINBOARD" &
```

and then restart X.

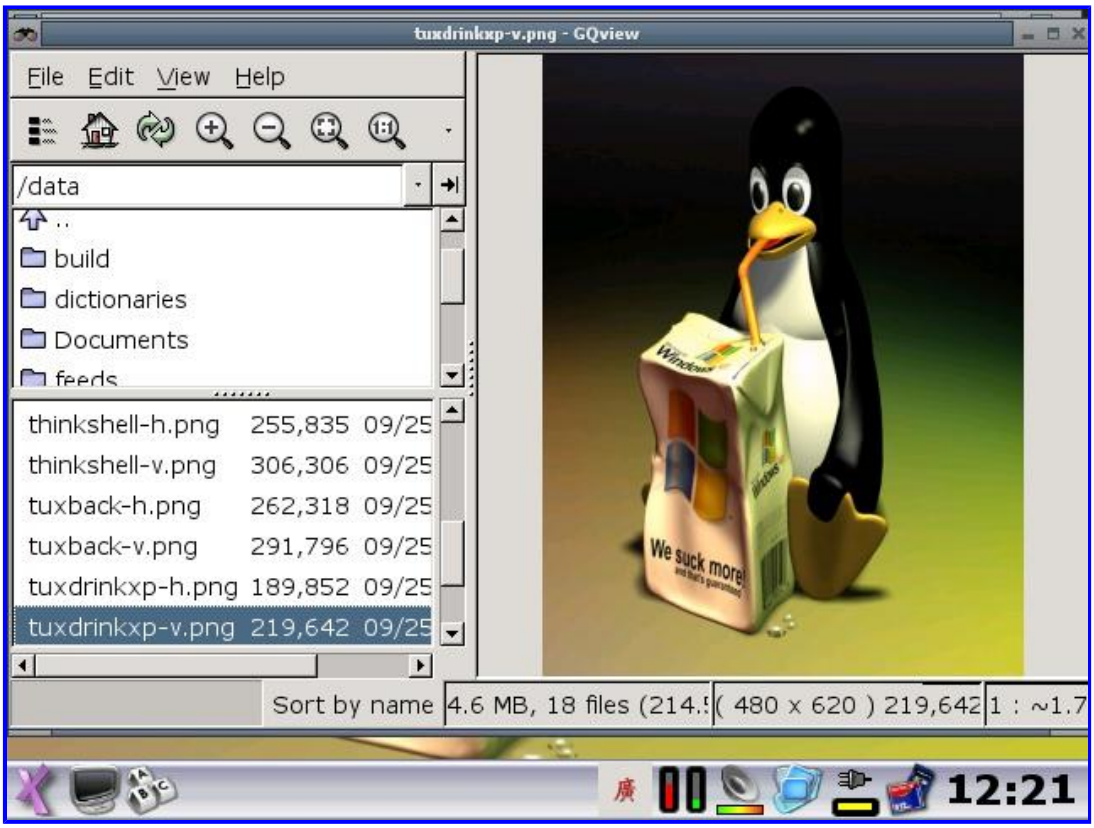
To change the theme, you need to edit rc.xml and modify the theme section. The themes are located under */usr/share/themes*. Each directory under there contains a theme. Place the name of the theme in the theme section inside the **name** tag. To apply the change, either restart X or run the following command:

```
# openbox --replace 2>/dev/null &
```

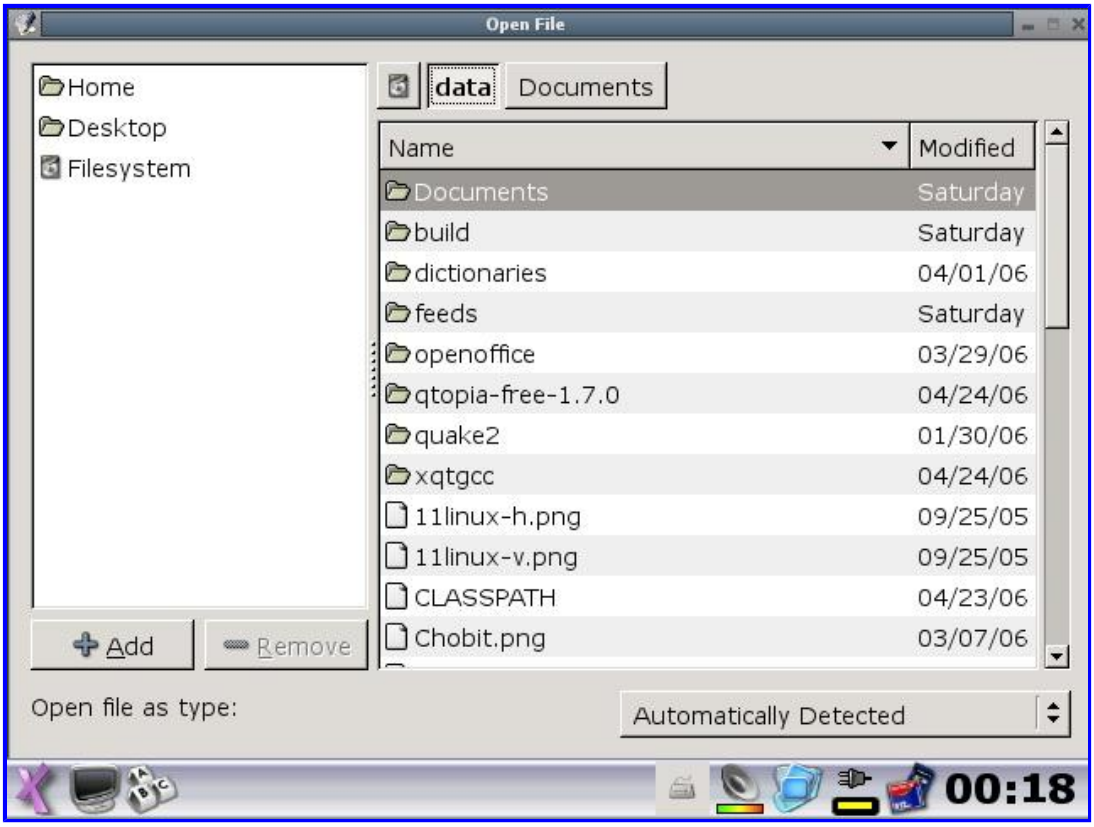
Window and Dialog Sizes

There are quite a few applications which open up in a window that is way too big for the Zaurus. This can usually be worked around by maximising the window in question (press *Home+m* or *Super+m*).

Some application can be tailored to a specific window size by editing their config files. Other applications have the size hardcoded. I have recompiled a few such applications, ie GQView and AbiWord.

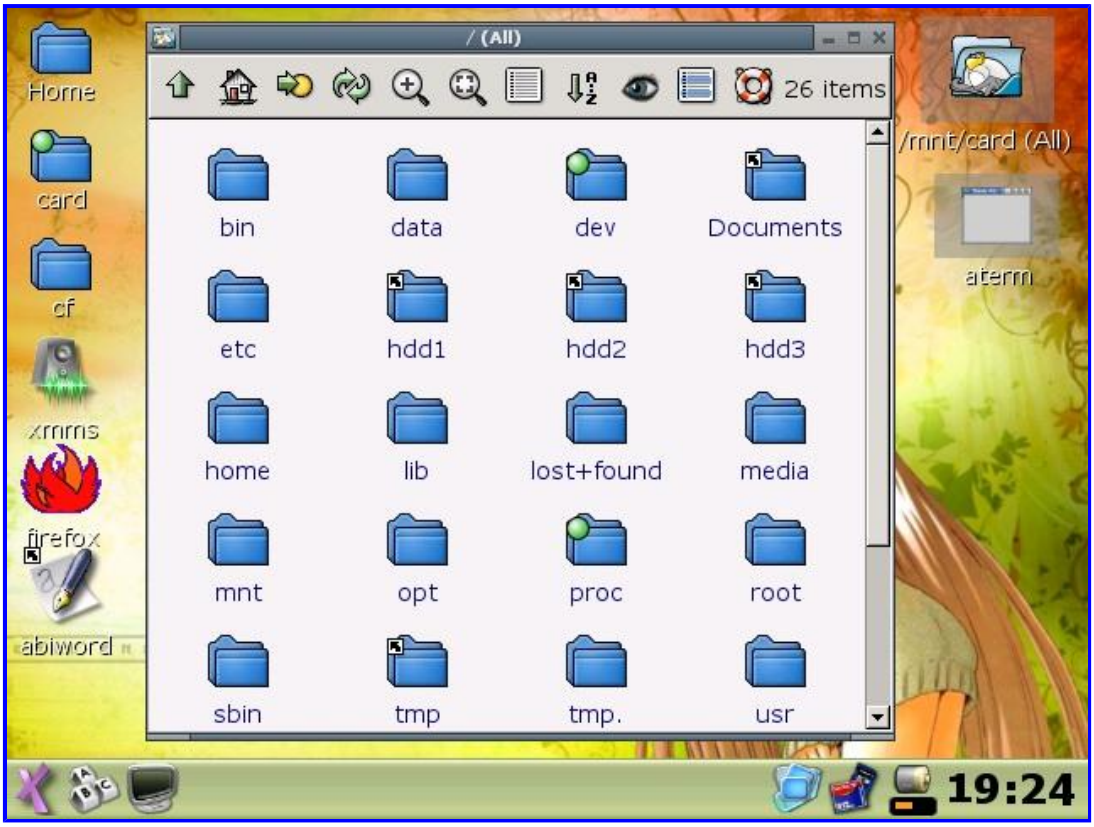


The File Open Dialog is also quite big and this problem is across most GTK2 applications. I have recompiled GTK2 and hacked the FileChooser to be more suitable for the Zaurus screen. You can either install my updated GTK2 package or replace libgtk-2.0.so.0.600.2 manually.

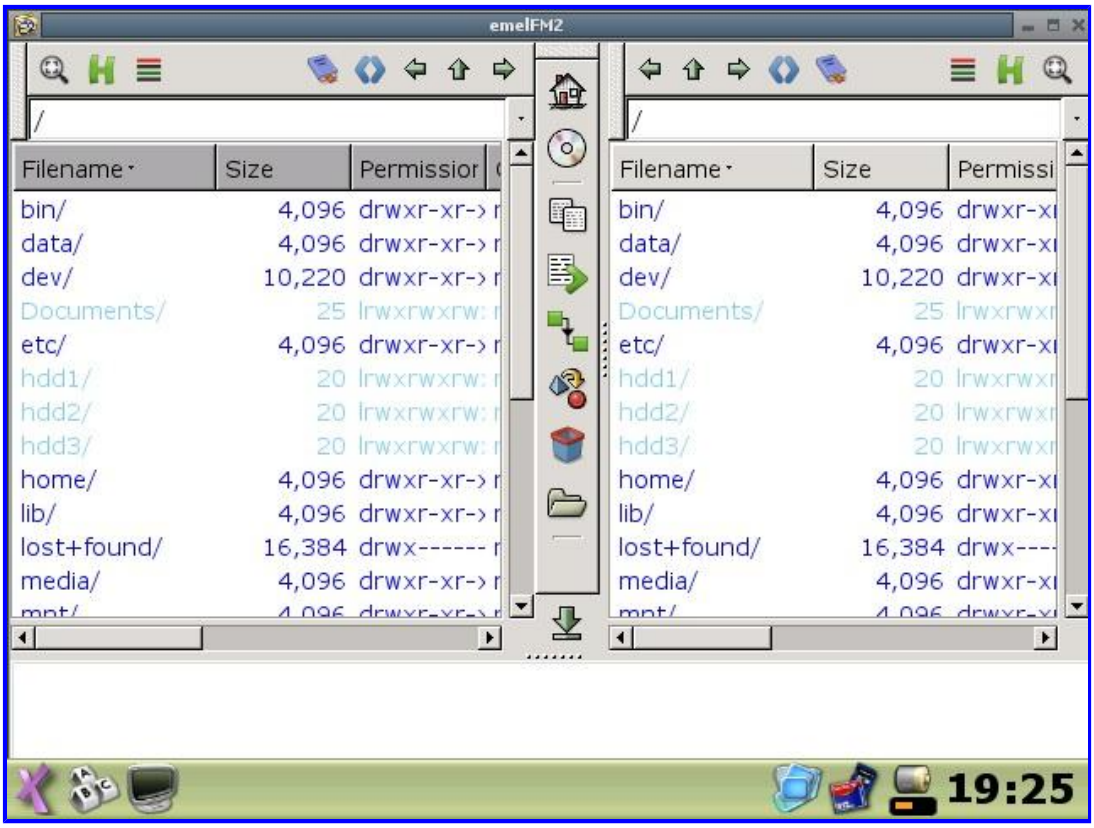


File Managers

There is no default file manager in pdaXrom, however, rox filer is a good candidate for this function. It has a similar interface to the konqueror for file management.

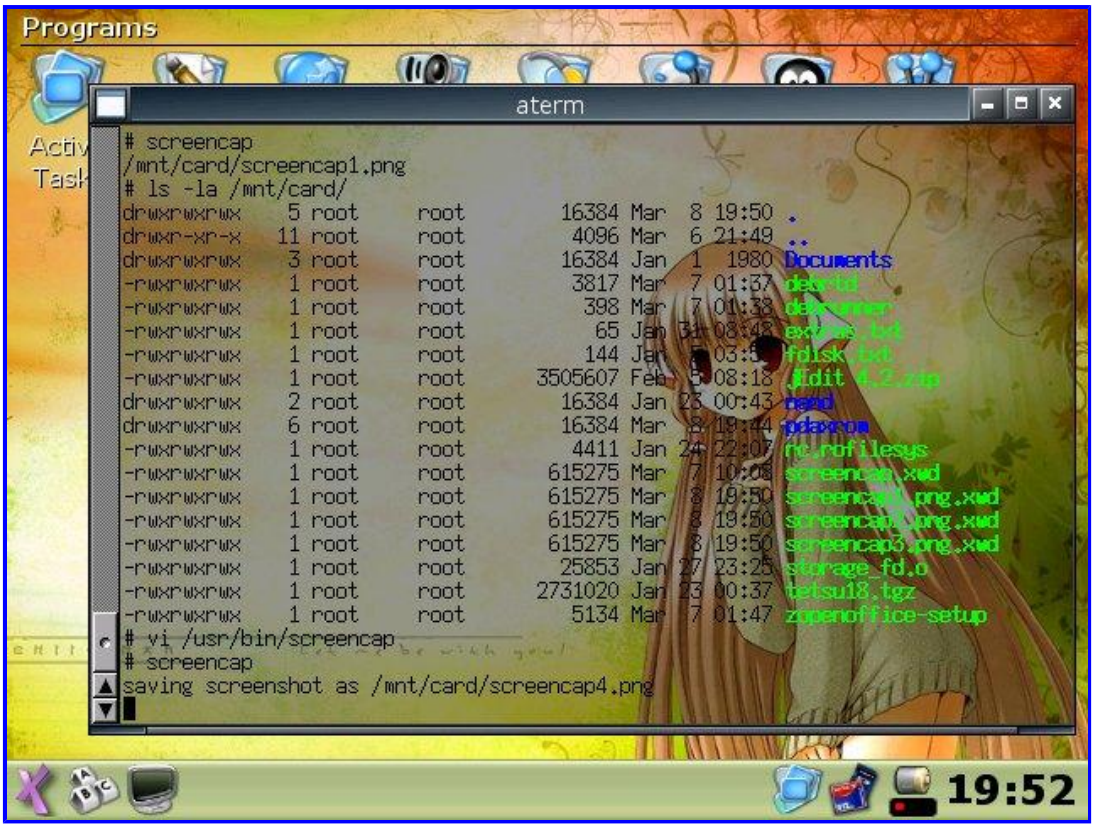


In addition, emelFM2 is also a good file manager with two panels.



Terminals

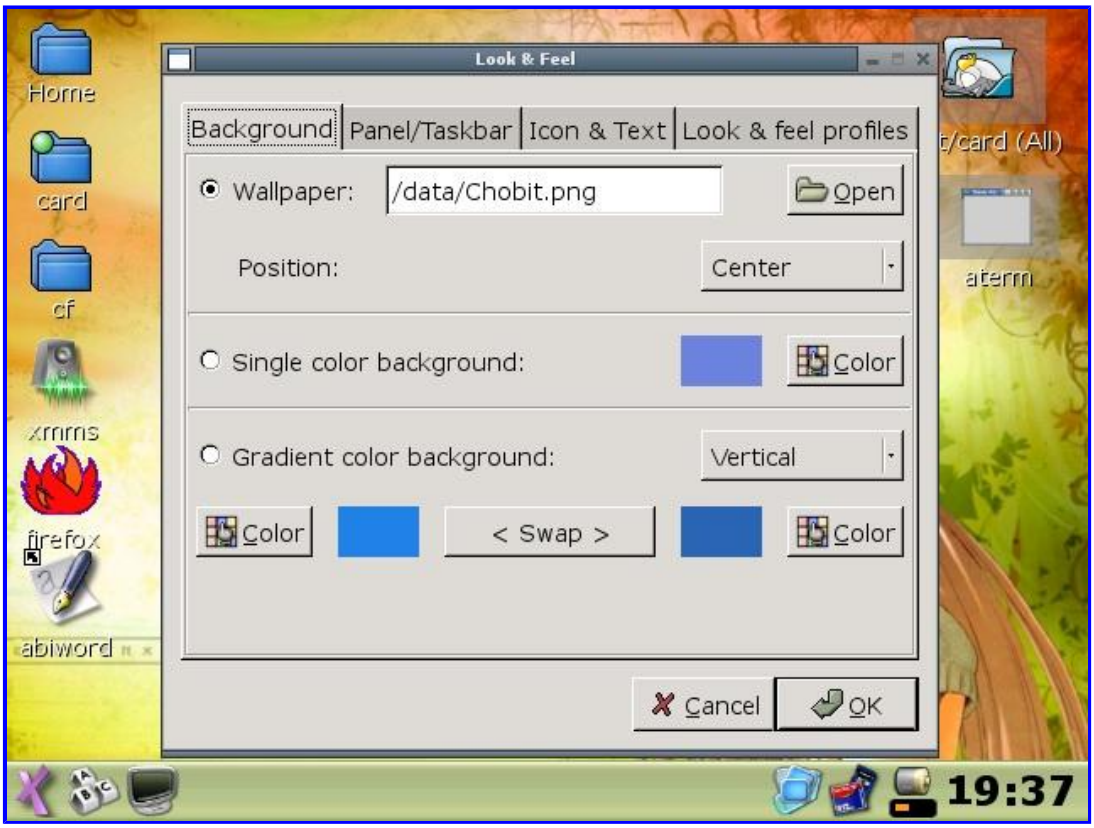
aterm is the default terminal in pdaXrom. It also supports transparent background. To enable transparency, copy Xdefaults to /home/root as .Xdefaults



Use rxvt as the terminal if you need to enter or view UTF8 encoded unicode characters.

Wallpaper Customisation

The Look & Feel tool in the Systems Tools can be used to change the default wallpaper as well as change many other aspects of how the GUI looks like. The wallpaper can be any png image file, but a resolution of 640x440 is ideal.



However, if you changed your desktop to use rox filer instead of the matchbox desktop, then you can right click (Fn+tap) to activate the rox filer properties where you can drag and drop an image into the background section.

ScreenSaver Customisation

For a X screensaver, install xscreensaver_4.24_armv5tel.ipk

zaurus user

pdaXrom by default does not have a zaurus user and thus runs everything as root. This would not be a problem if you don't connect to the net as the Zaurus is a single user environment and you are the only one using it. However, if you intend to connect your Zaurus to another computer or a network, or even the internet, then creating a zaurus user to run things as would be a good idea. You should also consider a firewall. To create a zaurus user, use the **adduser** command to add the zaurus user and then set the suid bit on busybox so the zaurus user can su and do other things.

Resume and Suspend

The suspend and resume is a bit flakey. Sometimes, the hard disk light does not turn off, but a subsequent resume and suspend will make it suspend with lights off. fscking the disk will often allow clean suspends.

Also, when the Z is suspended while xmms is active, then the sound device is locked until a reboot. Please ensure that you add esdctl suspend into the apm suspend script and have esdctl resume in the corresponding resume script to automatically suspend and resume the sound daemon.

MMC driver

There is a new OpenSource SD/MMC driver for the C3100/C1000. In order to test it on the C3000 as well, install mmc-driver_2.4.20_arm.ipk. This driver works, but currently does not automount nor does it unmount cleanly from the taskbar. Use **/etc/sdcontrol** [insert|eject] to mount and unmount the SD card.

Alternatively, a fully functional SD driver that supports 2GB+ cards has been made by Sharp and released with the SL-C3200. This driver [sdmmc-module_2.4.20_arm.ipk] also works on pdaXrom (beta1) because it is using the same kernel as the Sharp ROM, ie 2.4.20

auto mounting

Automatic mounting via /etc/fstab does not work in pdaXrom. Also, there are only two loop devices by default just like Sharp ROM. I have created an automnt script to automaticaly create additional loop devices and mount cramfs and squashfs images as well as enabling swap. I have linked /etc/rc.d/init.d/automnt to /etc/rc.d/rc5.d/S90mount and /etc/rc.d/rc6.d/K30unmount

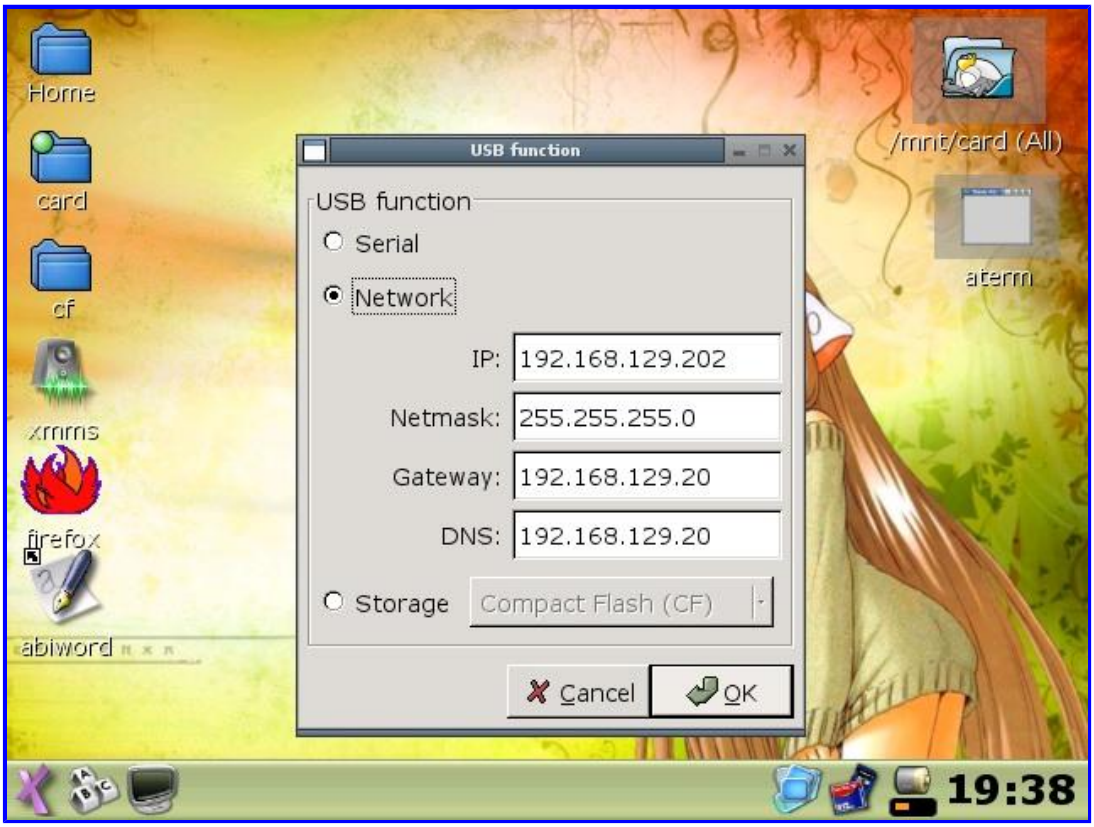
ftp and telnet

The C3000 version of pdaXrom does not have telnet nor ftp pre-installed. The version of busybox that comes with it does not include telnet and the binary on the pre-pivot root system is compiled with a newer version of glibc and thus won't run. I have copied the binaries for ftp and telnet from the Sharp distro and they work if placed in the right location. Since telnet really is busybox, I had to rename the busybox binary to telnet and it needs to be placed into /bin. I placed ftp into /usr/bin.

Alternatively, you can also install inetutils-clients_1.4.2_armv5tel.ipk from the unstable feed as well.

USB network

The USB network module works in two modes similar to the Sharp distro. It can either allow the Zaurus to share a disk or used as a virtual network adaptor. The Zaurus can be mounted as a USB storage device when connected with the USB sync cable. Copy the updated **usbfunction.py** to /usr/bin which will allow you to either share the SD, CF or internal MicroDrive. Sharing the internal MicroDrive might be a bit tricky with a Windows machine since it is formatted as ext3.



Alternatively, the USB network mode will allow you to connect the Zaurus via a virtual ethernet network and thus will allow you to share files via Samba if configured and also allow you to use any TCP/IP dependant network services if enabled. If for example the PC or Laptop you connect to via USB cable has internet sharing (Windows ICS or iptables on Linux) configured, then the Zaurus can access the internet through that connection. (see the networking section of my customisation guide for Sharp ROM on how to do that. I have also written a script **usbnet** which automatically configures the usbd0 interface to use the Windows ICS (Internet Connection Sharing).

Samba

You need to install the Samba package and configure */etc/samba/smb.conf* to share the desired shares. It is also a good idea to create a user such as zaurus and assign a password to that user using **adduser** -H. Then enable and create the zaurus user for samba using **smbpasswd** -L -a -e options. Then restart samba service using **/etc/rc.d/init.d/samba** stop and start as arguments.

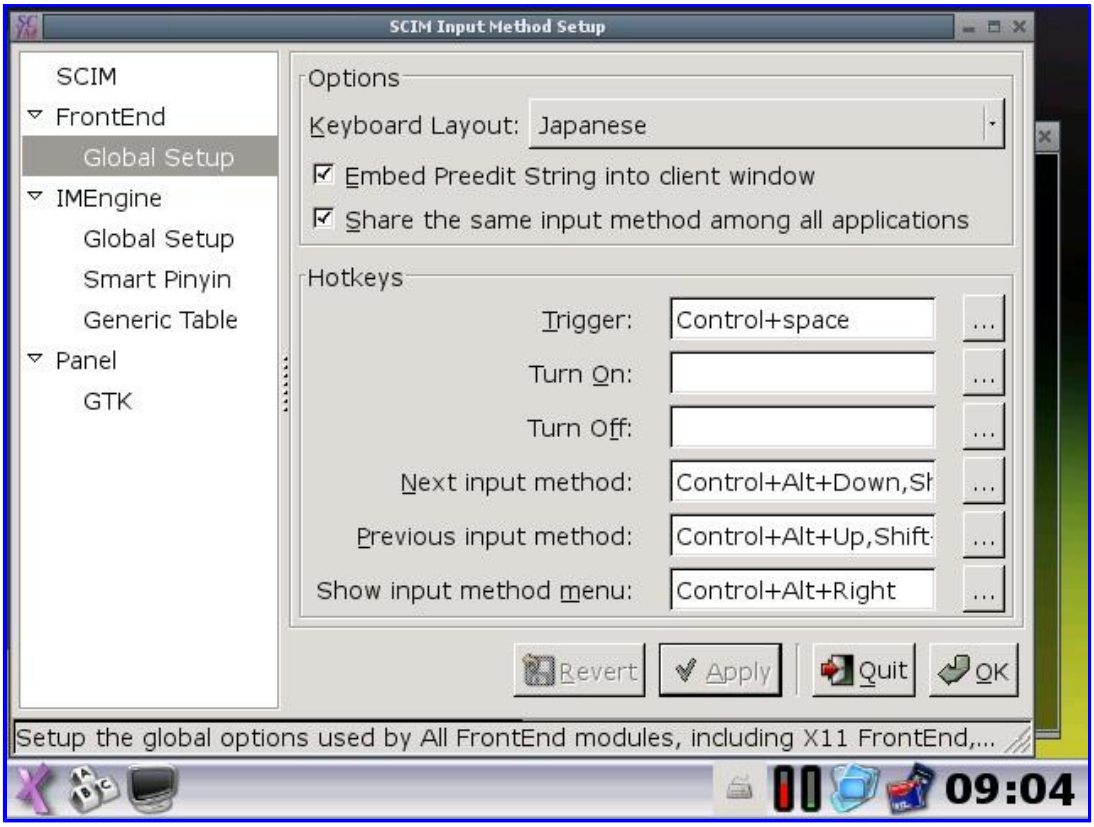
You can manually start and stop the samba service by running the script /etc/rc.d/init.d/samba which gets run at bootup. This script is very chatty and displays too many errors. I have modified it to be less chatty and also added a restart option.

If you enable the USB network, then you need to restart samba so it can pickup and listen on the new interface. You can also automate that by modifying /etc/hotplug/usbd.func and force samba to restart after the USB networking module is loaded.

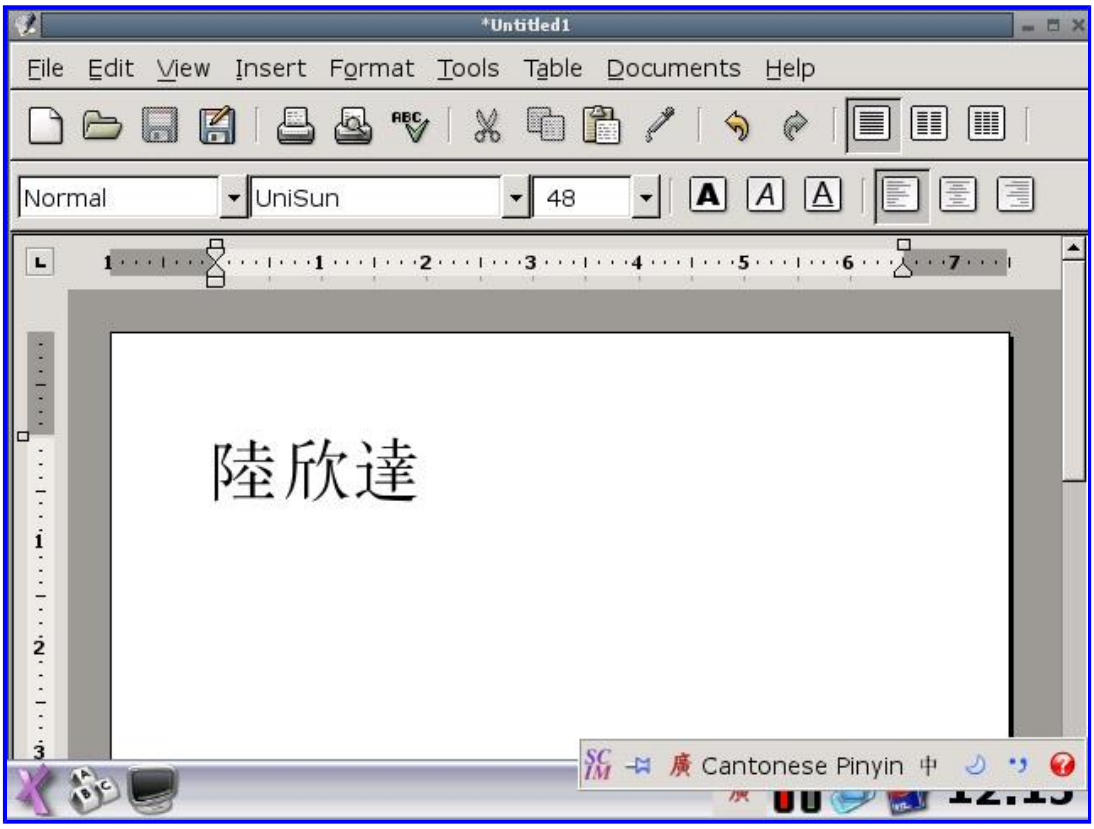
Enabling Chinese and Japanese

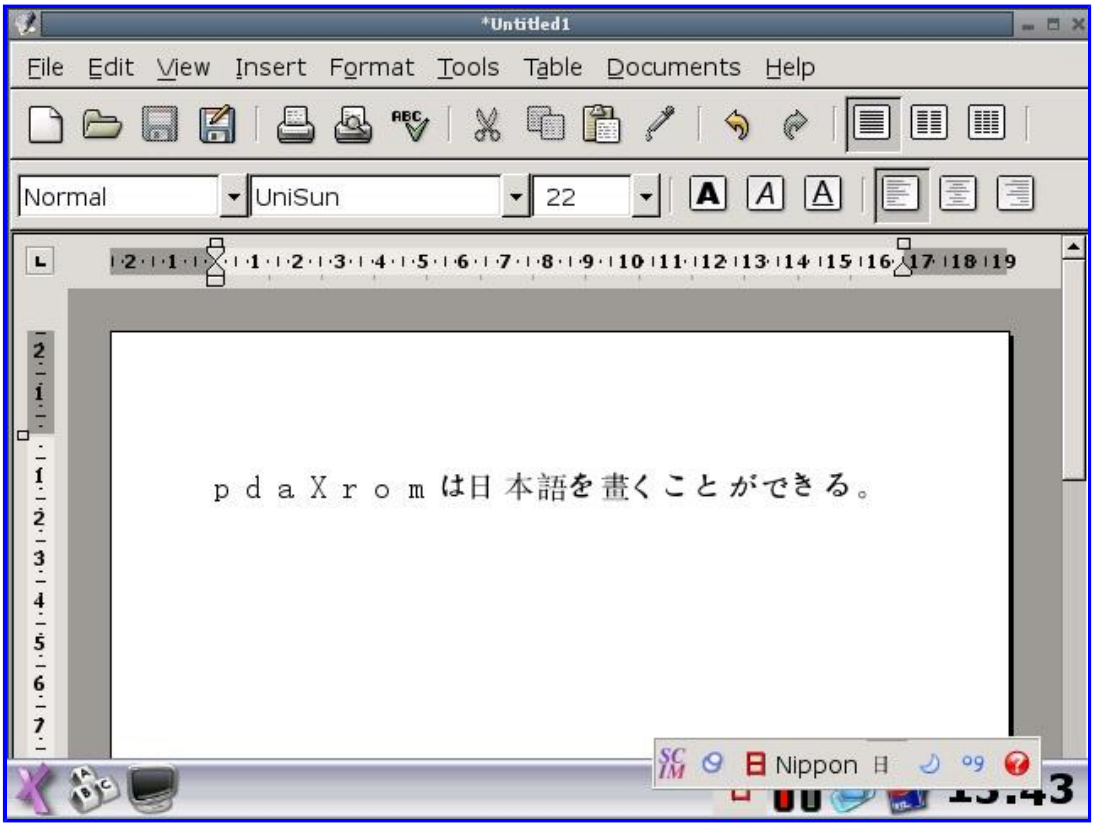
Chinese and Japanese menus can be enabled on pdaXrom by installing a Chinese or Japanese locale and installing Chinese and Japanese fonts. TTF fonts go into /usr/X11/lib/X11/fonts/TTF and pcf fonts go into /usr/X11/lib/X11/fonts/misc. You also need to install an input method engine for entering Chinese and Japanese characters. I installed scim, scim-pinyin and scim-tables for that.

Once scim is installed, you can configure it by running **scim-setup**.



Once scim is configured, you can run **scim -d** to start the scim engine and frontend as a daemon. Now for applications that scim can hook into, you can activate and switch input methods by pressing *Ctrl+space*. AbiWord for example can be used with scim:

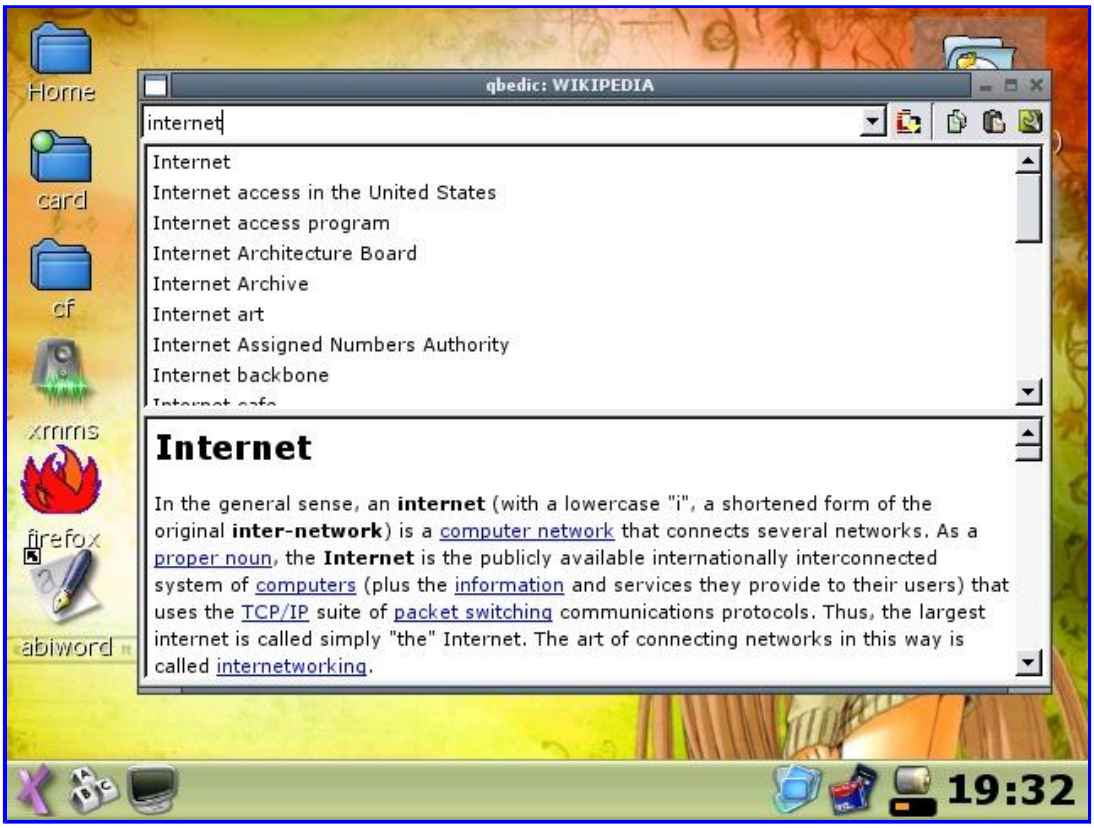




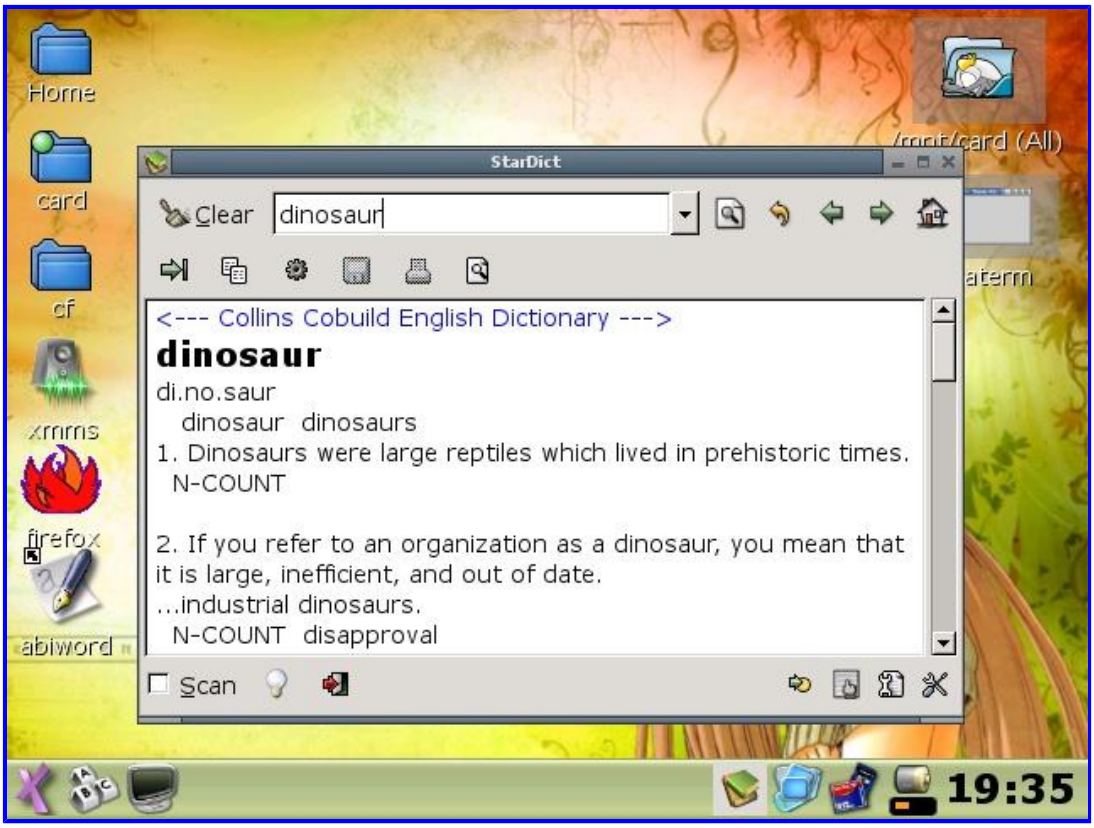
Remember to start scim each time you start X, or load scim from xinitrc to start scim automatically.

Dictionaries

There are several dictionary packages available for pdaXrom. QBEdic is a popular dictionary between different languages that is also available for the Sharp ROM (called ZBEDict), so the same dictionary files can be used. There is also a complete wikipedia that can be installed into QBEDic for reference. QBEDic dictionary files can be placed into any directory, but by default they are located under /usr/share/bedic. You can configure QBEDic to look in specific directories for dictionary files.



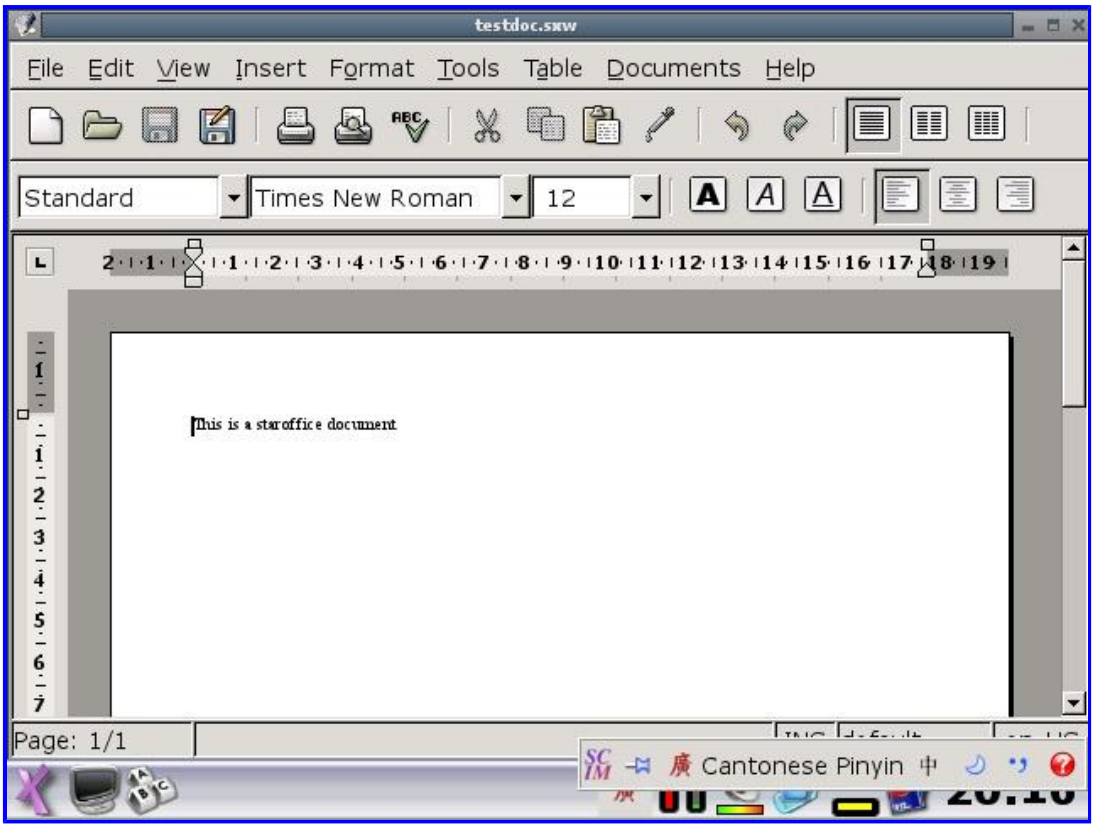
Another popular dictionary package for pdaXrom is stardict which also runs on Windows. There are many dictionary files available for stardict, and there is also a sound package for stardict to pronounce the words in English. Stardict expects its dictionary files to be located under /usr/share/stardict/dic. The sound files have to go under /usr/share



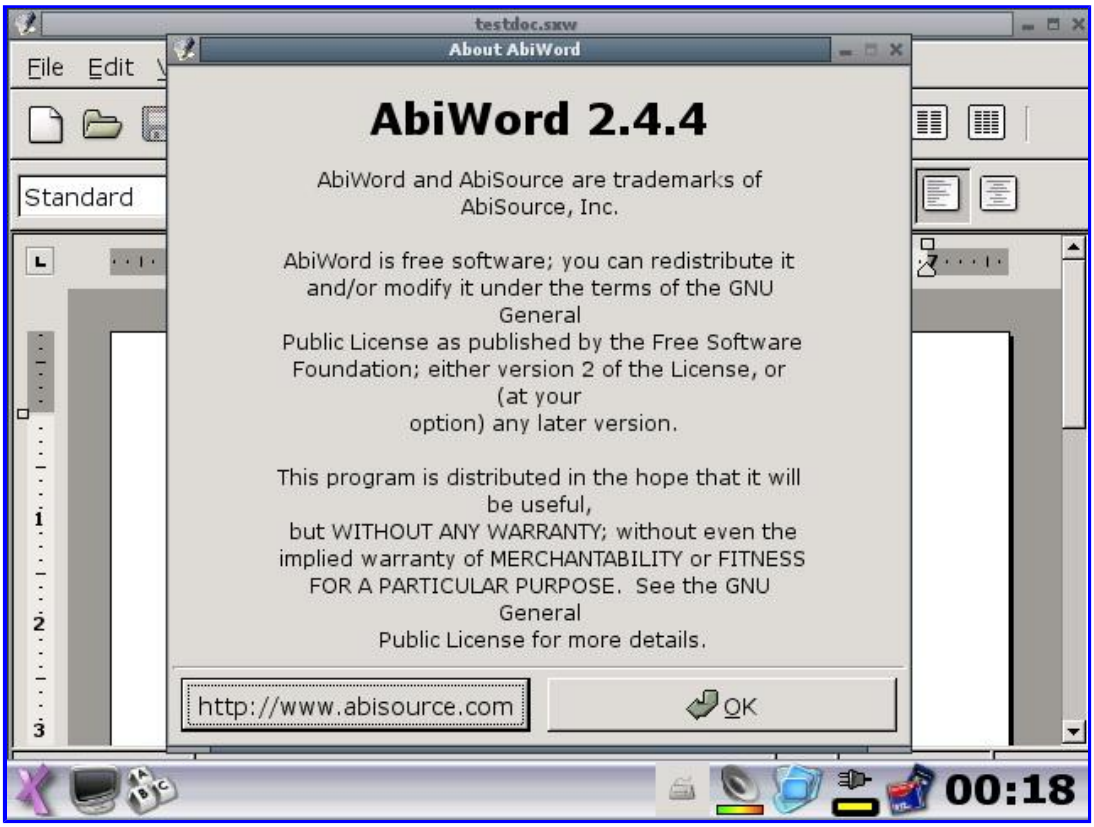
I have created a squashfs image containing several qbedic and stardict dictionaries as well as the sound files. Since the dictionary files are quite huge, placing them into a squashfs image can save quite a lot of space. I have also created a script called linkdics.sh which created the necessary links for stardict so that the files on the squashfs image can be located by stardict.

AbiWord

Abiword is a nice word processor, but it defaults to a 800x600 screen resolution which is not very usable for a Zaurus. For the C3000, it should be changed to 640x480. This can be done by editing */home/root/.AbiSuite/AbiWord.Profile*



I have compiled Abiword 2.4.4 and fixed all the window sizes so now every window in Abiword is sized correctly. I also compiled some text import/export plugins, so now I can open and edit StarOffice/OpenOffice documents.

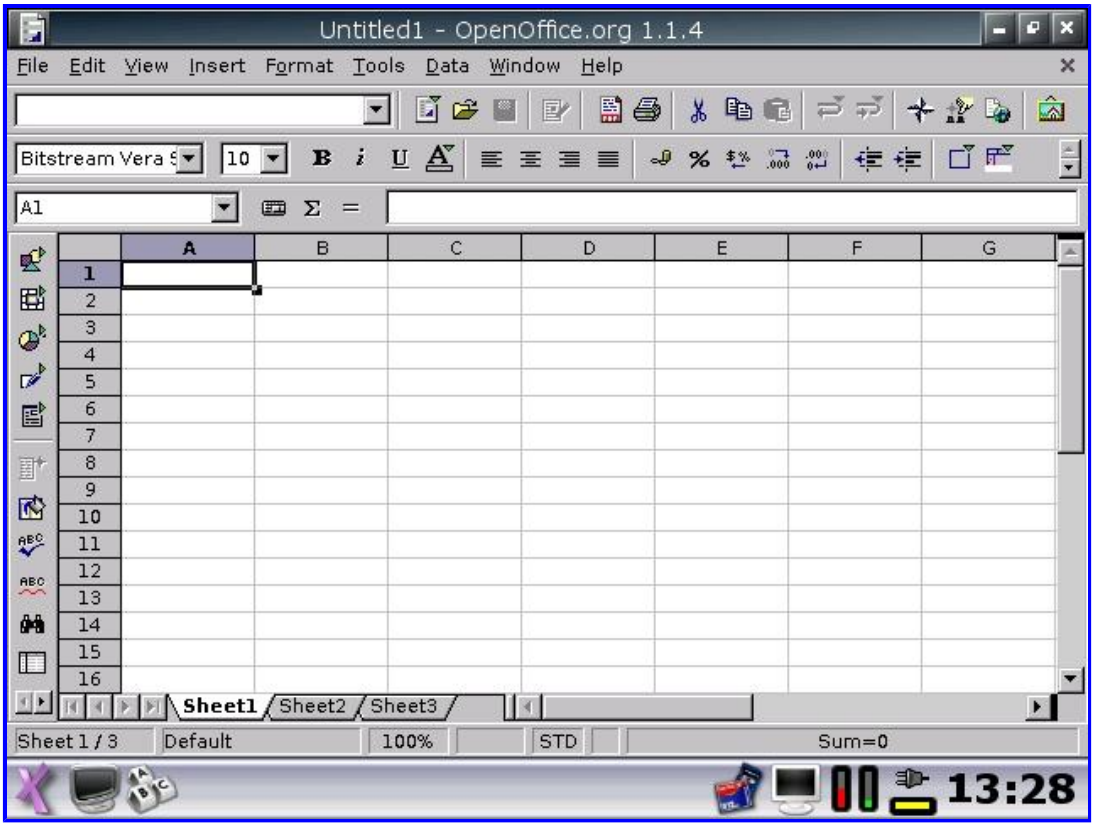


Installing aspell will enable spell checking in abiword.

OpenOffice

The Debian arm port of OpenOffice 1.1.4 works on the Zaurus. However, it is compiled against an experimental version of glibc 2.3 whereas most the Zaurus distros use a stable 2.2.x version of glibc. The usual way around this is to install PocketWorkstation (Debian arm) in chroot mode and install OpenOffice under PocketWorkstation. This certainly is inconvennient and quite a waste of space if you only want to run OpenOffice and none of the other Debian apps that come with PocketWorkstation since you would need at least 512MB of space for the two.

As an alternative, I have created a pre-installed image of OpenOffice on a cramfs as well as squashfs compressed image file which contains a bundled minimalistic version of Debian (minideb) and a pre-installed image of OpenOffice 1.1.4 binaries that can be run off the compressed image.



To install it, copy the cramfs or squashfs file to the internal harddisk or SD/CF card and mount the cramfs/squashfs file as a loop device. Then run the zopenoffice-setup script contained in the mounted image.

I have also included a tgz file, so if you have an ext2/ext3 formatted disk, you can extract the file and run zopenoffice-setup from the location you extracted the files to.

In addition, I also created openoffice-cn.squashfs which is preconfigured for usage on Chinese Zauri and includes Chinese fonts.

Lastly, you need to modify */usr/X11R6/bin/startx* and remove "-nolisten tcp" from the defaultserverargs and restart X. You should see a new OpenOffice icon in the menu. Alternatively, you can also start it from the command line by issuing **runoo** in a terminal.

Quake and Quake2

Quake as well as QuakeII plays pretty well and smoothly on the Z with pdaXrom, but don't change the resolution or display settings from within Quake/QuakeII or you will get a white screen and nothing else will happen anymore.

If you own Quake2, then you can copy the entire baseq2 directory to /home/root/.quake2

mplayer

The pdaXrom feed for C3x00 contains a mplayer package which you can install and it works quite well. Some people claim that the mplayer package from Cacko which was compiled by AGAWA Koji (atty) is much faster because it is optimised to use bvdd and iwmmx [mplayer-bvdd-iwmmxt_1.1.5-1_arm.ipk]. However, analysing the mplayer package on the C3x00 feed [mplayer_1.0pre7try2_armv5tel.ipk] it is in fact a slightly newer package based on atty's latest code and compiled with the same options, ie bvdd and iwmmx.

I have repackaged [mplayer-bvdd-iwmmxt_1.1.5-1_arm.ipk] as [mplayer-bvdd-iwmmxt_1.1.5-1_armv5tel.ipk] and changed the file structure to reflect the pdaXrom file structure and moved things to /usr/share instead of /opt/QtPalmtop and I also renamed the mplayer binary to mplayer2 so you can install both versions and compare speed.

The following mplayer command line will allow you to watch video:

```
# mplayer -vo bvdd -ac mad -ao ss -framedrop -really-quiet file
```

To force fullscreen playback, add the -vm option:

```
# mplayer -vo bvdd -ac mad -ao ss -framedrop -really-quiet -vm file
```

The xmmsmplayer addon to xmms can be used as a frontend for mplayer.

It uses the -vo x11 option by default to play the video in a gtk window. You can make it use the bvdd driver instead and play fullscreen also. The config file used by the plugin is */usr/share/mplayer/mplayer.conf* and contains the following config:

```
framedrop = yes
cache = 1024
dr = yes
af=resample=44100
```

Add the following to make mplayer use the bvdd driver and play fullscreen:

```
vo = bvdd
vm = yes
```

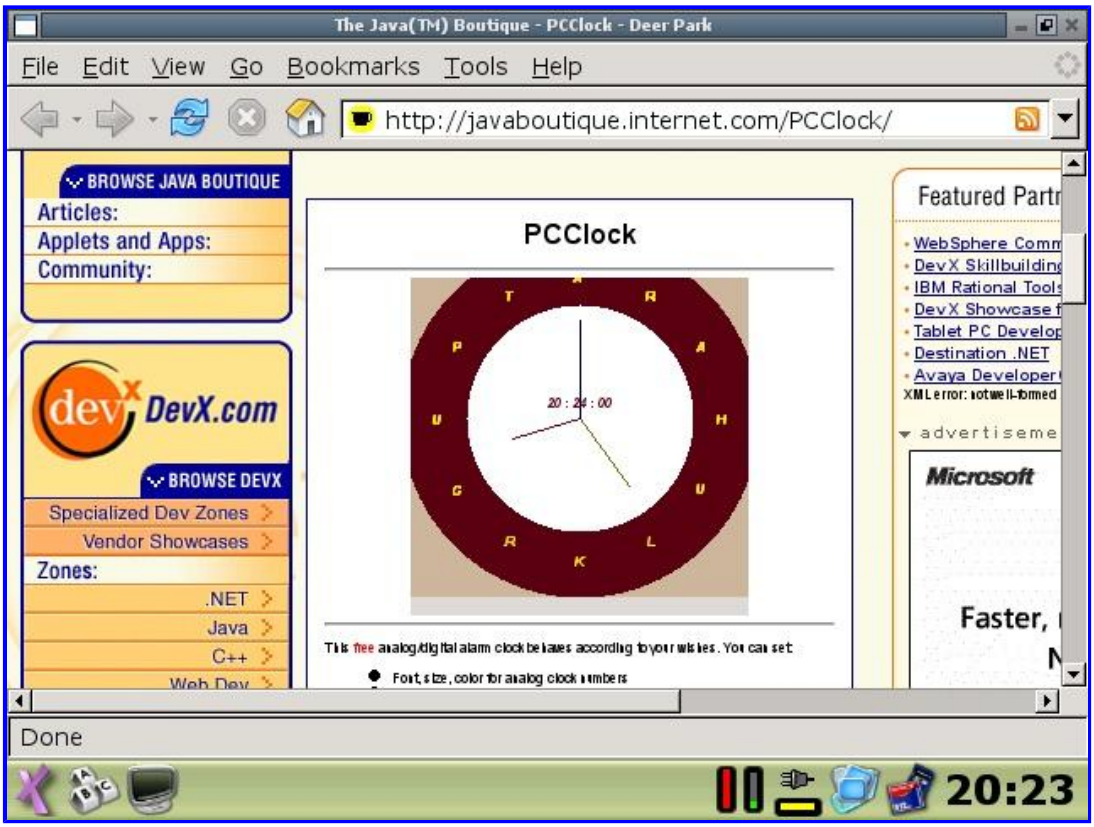
There is a bug with mplayer that it does not refresh the screen properly once it exits from fullscreen mode. As a workaround, you can run **xrandr -o normal** to refresh the screen. Alternatively, if you are using my customised rc.xml file for openbox, then you can simply press *Home+5* to refresh the screen.

I have also compiled the latest version of mplayer, 1.0pre8 with gmplayer enabled.

Java

Java is available through the jamvm and classpath packages. Using these packages, a JRE 1.4.x compatible environment can be achieved, however, due to copyright and licensing restrictions, jamvm cannot be called Java nor does it guarantee full compatability although it can run most Java applications. Using jamvm and classpath, I was able to run jedit on my Zaurus.

In addition, I managed to build a Java Plugin for Firefox which uses jamvm and classpath. Just install gcjwebplugin-0.3.2_armv5tel.ipk and Firefox 1.5 will be java enabled. It will be able to run most applets, however, not all applets will work. The plugin uses the classpath libraries so watch out for updates of the classpath package which should allow more applets to work.



The jamvm and classpath packages gives you a Java compatable runtime, however, in order to develop Java applications, you will need a Java compiler and other tools like jar, javah, javap and javadoc. Jikes can be used as a replacement Java compiler, and there is also a classpath tools package which has some of the Java tools you need.

If you want to be very sneaky, then you can copy the tools.jar file from a 1.4.x JDK and use it instead.

gcc

There is an on-board gcc 3.4.5 compiler to allow native development on the Zaurus. Furthermore, there are two different gcc images, one that generate soft float binaries (pdaXrom native) and one that generates hard float binaries (cross compile apps for Sharp/Cacko).

To use the gcc compiler, you first need to install the following packages that it requires:

- pcre
- xml-parser
- perl

Next put **zgcc-3.4.5.img** under */home/root*. This will make it being automatically mounted on boot. You can also mount it manually as follows:

```
# mount -o loop /home/root/zgcc-3.4.5.img /opt/native/arm/3.4.5-xscale-softvfp
```

To compile a little test application, create a file called hello.c with the following content:

```
#include <stdio.h>
int main()
{
printf("Hello World\n");
}
```

Then run the following command which will generate an executable called **hello**.

```
# gcc -o hello hello.c
```

To compile source packages, first download and extract the source tarball. Then do the following:

```
# ./configure
# ./make
```

You can also use **strip** to strip the binary to make it smaller once it is built. Usually, there is a DESTDIR variable defined in the Makefile which can help you turn the compiled binaries into a package.

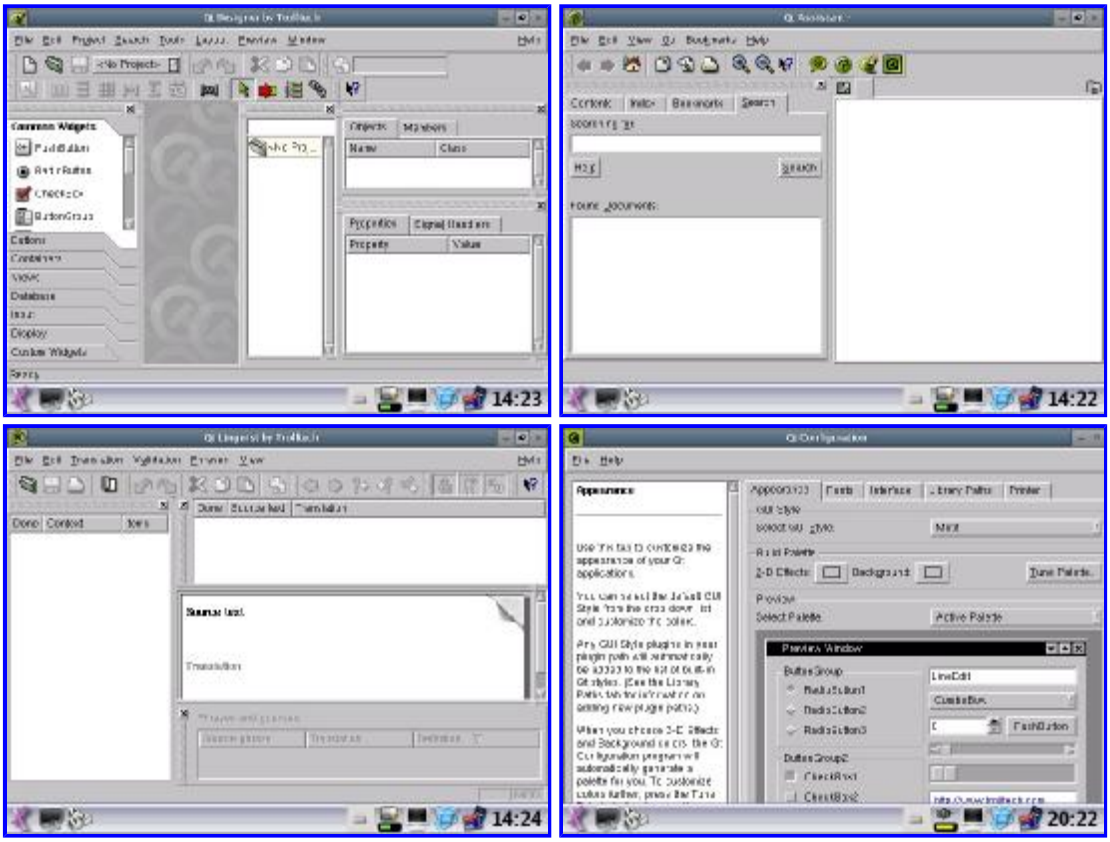
```
# cd /data/build/pkg
# newipk testapp
# cd /data/build/src/testapp
```

```
# make DESTDIR=/data/build/pkg/testapp/data install
# cd /data/build/pkg
# makeipk testapp
```

This example uses ipk-tools to create a package. See the ipkg section below for further info.

You can use gcc 3.4.5 to compile both console and X based applications, however, you cannot use it to compile the kernel or glibc. You will need gcc 2.95-2 instead.

I have slightly modified the gcc image and added a few missing symlinks and also added some binaries and scripts required for compiling some gnome packages. In addition, I also re-compressed it as a squashfs image rather than cramfs.



Furthermore, I also added QT designer, assistant and linguists, and I recompiled qtconfig so it won't startup oversized.

Building your own Packages

You can also build your own packages (ipk files) if you have written some useful scripts or written some applications that you want to distribute and let others install easily with the standard package manager.

I have build a package ipktools [ipktools_0.2_armv5tel.ipk] which has a set of tools for manipulating ipk files on pdaXrom:

- newipk - creates a package template structure for you to add files to for packaging
- makeipk - package up a directory that contains files in an ipk structure into an ipk file
- unpackipk - extracts the contents of an ipk file into a directory structure for repackaging
- deb2ipk - converts a deb file into an ipk file format (needs perl)
- zipipk - zip up an ipk file and remove ipk file afterwards
- ipkg-make-index - creates the Packages file for a feed

There currently are two ipk file formats. One uses the tar and gz format, wheras the other one uses a different binary format that is the same as the Debian .deb format. The Zaurus with default Sharp ROM (and Cacko) as well as pdaXrom uses the tar and gz format, which basically is a gzipped tarball (.tgz or .tar.gz) with a control structure and renamed to .ipk. If you extract this ipk file, you will find 3 files inside it - a text file called debian-binaries which just contains the string *2.0*, and two .tar.gz files called control and data. The control.tar.gz file contains a text file called control which has informaton about the package such as the Maintainer's name, dependencies, version, description, etc. There may also be some optional shell scripts for doing some pre and post configuration tasks during install and uninstall. Finally, the file data.tar.gz contains all the files and directory structure of the files for their destination location. The other ipk format is used by OpenZaurus and PocketWorkstation (Debian) and requires the **ar** command to extract the files.

To unpack an ipk file to see what is inside it, do the following:

```
# unpackipk somefile.ipk
```

To create your own ipk file, do the following to create the ipk file structure:

```
# newipk myproject
```

Then once you add your files in the correct locations and also update the control file with the information about your application, you can create your ipk file with the following command:

```
# makeipk myproject
```

If you want to convert a .deb or OZ ipk file so you can unpack it with **unpackipk**, then convert it using the **deb2ipk** perl script first, or ue the **ar** tool instead.

Zaurus Backup

You should always backup your system since that is the only way to recover if something

goes wrong. The easiest way to backup is to archive everything using **tar** and then use **gzip** to compress the archive. I have also created a script which does that.

Back to Main Page

DISCLAIMER: The information contained on this site is provided AS IS. No assurance is given to the accuracy of the information or instructions provided. You may use this as a guide but **do not blame me** if anything bad happens to your system or your data. Use anything described on this site at your own risk. I shall not be made responsible for anything you do.