Sharp SL-series Zaurus
“Qtopia” Development Start-up Guide

(ver.1.11 2003.02.28)

TRADEMARK NOTICE

XScale is a registered trademark of Intel Corporation.

StrongARM® is a registered trademark of ARM Ltd.

Linux™ is a registered trademark owned by Linus Torvalds.

Java™ and all Java-related marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Embedix™ is a registered trademark of Lineo Inc.

Microsoft®, Windows® are registered trademarks of Microsoft Corporation in the United States and/or other countries.

Corporate names and product names belong to various corporates as trademarks.

REVISION HISTORY

Oct 16, 2001	Version 0.94, first release to public
Nov 20, 2001	Version 0.95, Qt/E and Qtopia version clarified.
April 4, 2002	Version 1.00, incorporating SL-5500 changes.
Feb 28, 2003	Version 1.11 Refreshed the document, incorporating changes and information about the SL-5600.

TABLE OF CONTENTS

Introduction

SL-series Zaurus Product Overview

The SL-series Zaurus, Sharp's Linux™/Java™ based multimedia PDA, utilizes Linux 2.4.x as its operating system. Qt/Embedded by Trolltech is the C++ embedded GUI development toolkit which provides developers the ability to create stunning graphical user interfaces for embedded devices. Qt/Embedded runs on any device using embedded Linux - without using X11.

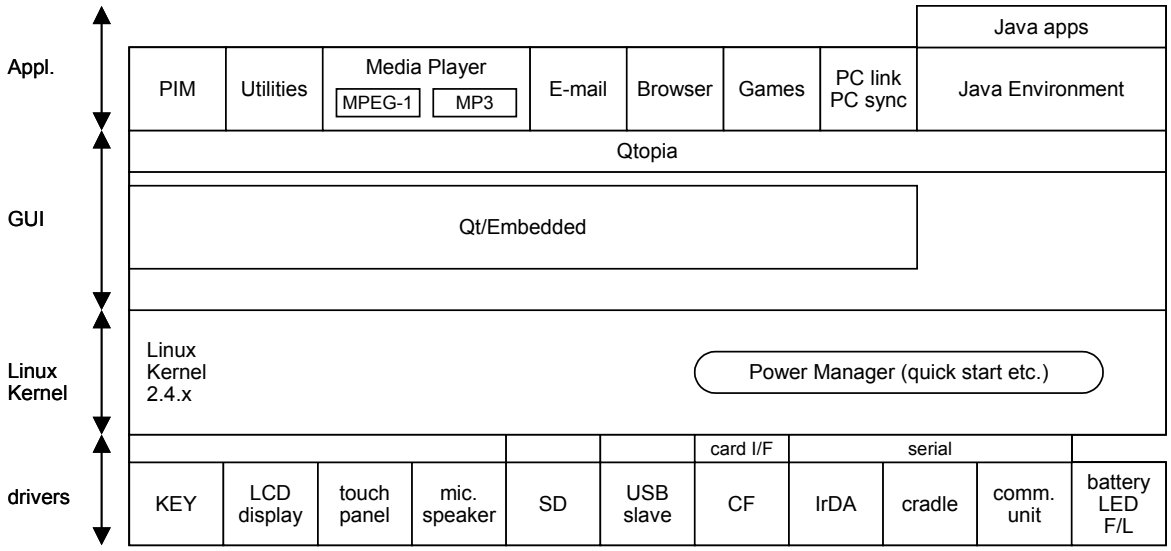
Qt/Embedded features the same API as the well known Qt/Windows and Qt/X11 versions. Qt is also the basis for the popular KDE desktop environment included in every major Linux distribution. This allows developers to write Qt applications in the favorite desktop environment with only minor changes and a recompile for usage on the SL-series Zaurus. This saves you significant development effort as it allows you to work productively on your favorite programming environment from day one.

The SL-series Zaurus further implements Qtopia by Trolltech, a complete Window System, Window Manager, Application Launcher, Input Methods (software keyboard, etc.), a GUI toolkit, and a collection of useful applications. By implementing Qt/Embedded and Qtopia, a well-known C++ embedded application GUI toolkit with complete a Window System, anyone can develop application software for the SL-Series Zaurus using existing and available Qt tools.

SL-Series Zaurus Architecture Overview

The figure below is a brief overview of the SL-Series Zaurus system architecture. The SL-Series Zaurus implements Qt/Embedded and Qtopia on Linux 2.4.x. (see figure below). Applications developed with Qt or possibly applications developed for KDE may run on the SL-Series Zaurus with less engineering effort. Only minor adjustments such as fitting the user interface for the SL-Series Zaurus's screen size is needed thanks to Qt/Embedded API.

With regards to the available commands in Linux, the SL-Series Zaurus supports the commands in BUSYBOX (http://www.busybox.net/). In addition, by using the network or a Compact Flash card, useful Linux tools can be added.



SL-5500 and SL-5600 Difference Summary

Hardware Specifications Differences

The following table summarizes the hardware difference between the SL-5500 and the SL-5600. Please also consult the information on the website (http://www.sharpusa.com) for the features and details.

	ITEMS	SL-5500	SL-5600
HARDWARE	**CPU**	Intel StrongARM (SA-1110, 206MHz)	Intel Xscale (PXA250, 400MHz)
	Memory	**ROM:** 16MB Flash (for OS, Applications, Driver) **RAM:** 64MB SDRAM	**NAND Flash:** 64MB **ROM:** 512KB **RAM:** 32MB SDRAM
	Dimensions (with slide-cover closed)	Excluding the display protection cover: Approx. 74(W) x 138(D) x 18(H) mm Including the display protection cover: Approx. 74(W) x 138(D) x 21(H) mm	Excluding the display protection cover: Approx. 74(W) x 138(D) x 18(H) mm (H: 22.7 at battery cover top) Including the display protection cover: Approx. 74(W) x 138(D) x 20.6(H) mm (H: 25.3 at battery cover top)
	Weight	Excluding the display protection cover: 194g Including the display protection cover: 212g	Excluding the display protection cover: 210g Including the display protection cover: 228g
	Audio in	Mic and stereo headphone jack (external mic needed for audio input) Sampling Rate: 8k, 11.025k, 16k, 22.05k	Built-in Microphone Sampling Rate: 8k, 11.025k, 16k, 22.05k, 24k, 32k, 44.1k, 48k
	Audio out	Mic and stereo headphone jack	Built-in speaker (mono) Stereo headphone jack
	Buzzer /dev/sharp_buz (key click, screen tap, alarm)	Supported	Supported (No buzzer sound when /dev/dsp device is open)
	Battery	<conditions > SL-5500 turned off at ≤ 25℃, a brand-new battery fully charged. approx. 10 days <conditions> The operating battery exhausted, and the power can be turned on. approx. 1 day <condition> The operating battery is not inserted, after the rechargeable back-up battery fully charged. approx. 5 minutes	<conditions > Display the "day view" of the Calendar application continuously, with the frontlight turned off. approx. 18 hours <conditions> Play an MPEG1 file continuously with the frontlight turned on at brightest llevel approx. 2 hours (Note: above values are all measured with the SL-5600 turned on at ≤ 25℃, without any peripheral devices plugged in, and with the rechageable battery fully charged.)
	Data Backup	RAM backup battery required	Not necessary (stores data in NAND Flash)
	Reset button	work as full reset.	work as reset. *1)

(NOTE)
*1) All user data is stored in the Flash memory, and thus user data will not be lost by the RESET button nor the battery discharge. Users must format the flash memory from its "maintenance menu" to fully reset the device.

Software Specifications Differences

The following table summarizes the overall software difference between the SL-5500 and the SL-5600. Please also consult the following sections and additional documents on the developer website (http://www.zaurus.com/dev/).

	ITEMS	SL-5500	SL-5600
SOFTWARE	**Linux Kernel**	Linux 2.4.6 (Embedix)	Linux 2.4.18 (Embedix)
	File System	Internal RAM: ext2	Internal Flash: JFFS2 *2)
	Memory Area	**Internal Storage:** approx. 31.8MB **Program Memory (work area):** approx. 28.1MB	**Internal Flash Storage:** approx. 35MB **Program Memory (work area):** approx. 29.1KB
	PIM database	XML based	DTM *3) based
	quickexec (*4)	Not supported	Supported
	Process Owner	Root	user (as default *4)) (root privilege possible)
	USB	Network	USB IO / Network
	Screen Saver	Not supported	Supported
	Support style (Theme)	- Windows - Light - QPE	- Windows - Light - QPE - SLStyle - ZIvory - ZBlue - Zpurple

(NOTE)

*2) JFFS2 file system is not a block device and thus swap cannot be created on the file system.

*3) DTM (DaTa Manager) is a set of modules that provide database functions to the applications. User applications can utilize the SL data manager class to easily implement store, sort, and search data entities. Applications that utilize the XML file format for the SL-5500 PIM information can also run on the SL-5600. However, all of the default PIM applications on the SL-5600 now adopt and are managed by the DTM (PIM database). Thus, any 3rd party PIM applications that access the XML files of the SL-5500 PIM applications will not run as expected. Developers need to incorporate the DTM access for those PIM applications.

*4) Processes are executed with user privileges, and thus some directories may not be accessed depending on the permissions granted. User applications must be executed with root privilege if the application will have access to the directories at which only root privileges are granted. See section 3 and Appendix A for the details of the SL-5600 directories from "/".

Target Reader of this Programming Guide

This document is intended for users who wish to develop applications with the C++ programming language within Qt/Embedded and Qtopia for SL-series Zaurus. It is therefore assumed that you, the reader, have obtained:

- Some C++ programming skills
- Some introductory knowledge of, or experience with Qt and/or Qt/Embedded
- Basic knowledge of SL-Series Zaurus PDA

Overview of this Guidebook

This document is intended to provide developers with the information on the tools, the workflow and tips to install developed application to the SL-Series Zaurus, and the SL-series Zaurus specific file system so that developed application can run on a variety of the SL-Series Zaurus.

Because there already exists many books on C++ development, Qt development, and because detailed documentation on Qt/Embedded and/or Qtopia will come with these corresponding tools themselves, this document is not intended to provide details on C++ programming or Qt programming, except for a few basic tutorials for those who are not familiar with the Qt related tools as well as Qt specific programming. You are encouraged to refer to any commercially available C++ programmer guidebook, and/or Qt programmer's guidebooks, if necessary.

Section 1 provides a brief explanation on the OS, tools, and equipment that developers need to for a development environment utilizing Qt/Embedded / Qtopia applications on their desktop.

Section 2 illustrates the basic workflow to develop and package Qtopia applications for the SL-series Zaurus. It also includes a brief introduction of tools, Qt/Embedded specific tips, examples of necessary commands, and files needed for packaging the application.

Section 3 guides you through the ways of adding a newly developed Qtopia applications to SL-series Zaurus.

Reference and URLs

Sharp Linux/Java PDA Developers Website:

http://www.zaurus.com/dev/

Trolltech

The following is the official WEB page URL:
http://www.trolltech.com/

Qt/Embedded

The following is the official WEB page URL:
http://www.trolltech.com/products/embedded/index.html

Qt/Embedded Whitepaper

Qt/Embedded Whitepaper can be found at:
http://www.trolltech.com/products/embedded/whitepaper.html

1. Qt Application Development Tools and Compiler Setup

This chapter provides information on tools and equipment required to build the development environment for Qt/Embedded and Qtopia applications on your desktop.

1.1. Building Development Environment for Qt/Embedded Applications

The following are the tools and equipment needed to build a development environment on your desktop PC:

1.1.1. PC-Linux

The development PC should have a Linux distribution pre-loaded on it. Preferably one that natively supports RPM packages, such as RedHat, SuSE, Mandrake or Caldera. You can also use distributions such as Slackware and Debian as well, but you may need to use a RPM conversion utility to support the RPM format. In this document, RedHat 7.3 is used for the OS and "bash" is used for its shell.

Please also note that it is recommended that your PC that supports a PC card slot/reader, for use with a CF memory card or SD memory card to install the developed applications onto the SL-series Zaurus.

If your Linux machine is equipped with USB, you may use the docking station and its USB connector to transfer the developed applications to the SL-series Zaurus by installing the USB driver for the Linux PC. The USB driver for the PC-Linux can be downloaded from the following URL. Note that this driver is also provided in the RPM package and thus one may have to convert this package if your Linux distribution does not support the RPM package by default.

http://www.zaurus.com/dev/tools/downloads/tools/kernel-zaurus-2.4.18.5-4a.i386.rpm

The following are some reference for the HDD, RAM, and CPU in preparing the PC to build your development environment:

HDD: approx. 400MB free space or more
CPU: appropriate CPU to run the Linux distribution
RAM: appropriate CPU to run the Linux distribution

1.1.2. Setting up the cross-compiler for ARM/XScale

Because the SL-series Zaurus has an Intel StrongARM® or Intel Xscale® as its CPU, one needs to have an ARM cross compiler for application development. The ARM cross compiler and the related tools for the SL-series Zaurus can be found at the Sharp developer website (http://www.zaurus.com/dev). The following chapters will provide you with the information on how to install and set up the tools required for development. All of the directories and files explained in the following chapters are the default configuration of the tools. If you change the directories or file names, you may have to make necessary adjustment in accordance with your changes.

Online documents for the cross compiler can be found at following URL:

http://www.gnu.org/

http://www.gnu.org/manual/binutils-2.10.1/binutils.html

http://www.gnu.org/manual/ld-2.9.1/ld.html

http://www.gnu.org/software/gcc/onlinedocs/

http://www.gnu.org/manual/glibc-2.2.3/libc.html

Once the target PC is ready, the following packages should be downloaded from the Sharp developer website:

- gcc-cross-sa1100-2.95.2-0.i386.rpm (gcc compiler for ARM architecture)
- binutils-cross-arm-2.11.2-0.i386.rpm (binary utilities for ARM architecture)
- glibc-arm-2.2.2-0.i386.rpm (GNU C libraries for ARM architecture)
- linux-headers-arm-sa1100-2.4.6-3.i386.rpm (linux header files for ARM architecture)

Each of the RPM files need to be installed from a command line prompt using the following command:

```
% rpm -Uvh filename.rpm
```

For example, to install the arm gcc compiler, one should execute:

```
% rpm -Uvh gcc-cross-sa1100-2.95.2-0.i386.rpm
```

By default, RPM installs the ARM toolchain in the `/opt/Embedix/` folder. Please note that you need to be root to install RPM packages. You may either have to login as root or, change your user privilege to root by executing `su` command:

```
$ su
Password: (enter root password)
$ whoami
root
%
```

1.1.3. Qt/Embedded / Qtopia Build Environment

Native development for the Zaurus is done using C++ and Qt by TrollTech. Qtopia comes with a virtual frame buffer (qvfb) so that you can test applications under X11 without having to have a Zaurus. To run applications on the Zaurus (and the qvfb) you need to link against QPE rather then Qt. See section 4 for more details about qvfb.

To start development you need to obtain the Qtopia SDK from Trolltech. GPL edition (qtopia-free-1.5.0-1.i386.rpm) can be located at the following URL. Note that if you use this GPL version, you have to follow the terms and conditions set forth in the GPL (GNU Public License, see http://www.gnu.org/ for details). If you are doing commercial development you need to obtain the commercial SDK:

http://www.trolltech.com/developer/download/qtopia.html. (GPL edition Qtopia SDK)

https://www.regnow.com/softsell/nph-softsell.cgi?item=7131-1 (Commercial Edition)

Install the rpm in the same manner as to how the cross compiler was installed:

```
rpm -Uvh qtopia-free-1.5.0-1.i386.rpm
```

By default, RPM installs the Qtopia SDK in the `/opt/Qtopia/` directory.

1.1.4. Configuring Your Compiler Environment

After the toolchain and the SDK are installed, you should create the two batch files in your home directory. One sets up the environment variables for compiling x86 versions of SL-series Zaurus applications (using the qvfb) and the other for setting the environment variables for doing native ARM cross-compiling for the SL-series Zaurus. More information can be found at (http://docs.zaurus.com/linux_compiler_setup_howto.shtml) The following are examples of the scripts:

Batch file #1, dev-x86-qpe.sh

```
#!/bin/bash
# dev-x86-qpe.sh
# location : /usr/bin

if [ -z ${ORG_PATH} ]
then
ORG_PATH=${PATH}
export ORG_PATH
fi

if [ -z ${ORG_LD_LIBRARY_PATH} ]
then
ORG_LD_LIBRARY_PATH=${LD_LIBRARY_PATH}
export ORG_LD_LIBRARY_PATH
fi

CROSSCOMPILE=/opt/Embedix/tools:/usr/local/x86/2.95.3/bin:/opt/Embedix/tools
QPEDIR=/opt/Qtopia
QTDIR=/opt/Qtopia
PATH=/usr/local/x86/2.95.3/bin:$QTDIR/bin:$QPEDIR/bin:${ORG_PATH}:/opt/Embedix/tools/bin
TMAKEPATH=/opt/Qtopia/tmake/lib/qws/linux-x86-g++/
LD_LIBRARY_PATH=$QTDIR/lib:${ORG_LD_LIBRARY_PATH}
export QPEDIR QTDIR PATH TMAKEPATH LD_LIBRARY_PATH PS1
echo "Altered environment for Sharp Zaurus Development x86"
```

Batch file #2, dev-arm-qpe.sh

```
#!/bin/bash
# dev-arm-qpe.sh script
# location : /usr/bin
#

if [ -z ${ORG_PATH} ]
then
ORG_PATH=${PATH}
export ORG_PATH
fi

if [ -z ${ORG_LD_LIBRARY_PATH} ]
then
ORG_LD_LIBRARY_PATH=${LD_LIBRARY_PATH}
export ORG_LD_LIBRARY_PATH
fi

CROSSCOMPILE=/opt/Embedix/tools:/usr/local/x86/2.95.3/bin:/opt/Embedix/tools
QPEDIR=/opt/Qtopia/sharp
QTDIR=/opt/Qtopia/sharp
PATH=$QTDIR/bin:$QPEDIR/bin:$CROSSCOMPILE/bin:${ORG_PATH}
TMAKEPATH=/opt/Qtopia/tmake/lib/qws/linux-sharp-g++/
LD_LIBRARY_PATH=$QTDIR/lib:${ORG_LD_LIBRARY_PATH}
export QPEDIR QTDIR PATH LD_LIBRARY_PATH TMAKEPATH PS1
echo "Altered environment for Sharp Zaurus Development ARM"
```

When you want to compile and test x86 applications run source dev-x86-qpe.sh from your home directory. Conversely, run source dev-arm-qpe.sh when you want to cross compile to run on the Zaurus.

1.1.5. tmake, a cross-platform makefile tool

The tmake tool is an easy-to-use tool from Trolltech to create and maintain makefiles for software projects. It can be a painful task to manage makefiles manually, especially if you develop for more than one platform or use more than one compiler. The tmake tool automates and streamlines this process and lets you spend your valuable time writing code, not makefiles.

The tmake tool can be found at http://www.trolltech.com/developer/download/tmake.html.

1.1.6. Confirming Installation of the tools

Once you install the above tools, you will have the following directories for each tool. If you change the installation directories, you need to change the PATH in accordance with your changes.

<Qtopia Envirpnment>

All of the related files are installed to `/opt/Qtopia`.

<cross compiler related>

All of the related files are installed to `/opt/Embedix/`.

<Batch files for configuring your development environment>

These files should be created in the user’s home directory. If the user name is “`user1`”, these files should be created in `/home/user1`. You may want to execute these script files to configure compiler, libraries, and tmake related PATH before you compile your applications depending on the target. You may create these batch files in any directories where the PATH is already configured.

<tmake>

The tmake tool itself is a part of Qtopia, can be located in `/opt/Qtopia/bin/tmake`.

1.2. Testing the cross compiler

To test the compiler, you will want to build the example application in the/opt/Qtopia/example/ directory. The following are brief instructions to test the cross compiler.

1.2.1. Check compiler setup for x86

First, run the x86 environment script in your home directory from a shell session to configure the variety of environment variables:

```
$ cd /home/[user_home_directory]
$ ./dev-x86-qpe.sh
```

Next, within the same session and in the /opt/Qtopia/example/ directory, run tmake -o Makefile example.pro. This creates the Makefile.

```
$ cd /opt/Qtopia/example
$ tmake -0 Makefile example.pro
```

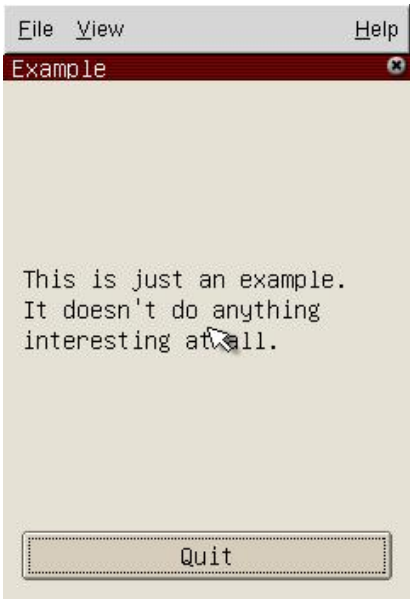
To actually build the application, run make within that same directory.

```
$ make
```

Start qvfb (Qtopia Virtual Frame Buffer) and then start the example application to see if you have successfully compiled the example for x86. The command "qvfb &" will launch the simulated SL-series Zaurus display on your PC. Any Qtopia applications compiled for x86 you launch will now display in this window. If you see the following screen displayed on your development machine, you have successfully complied the example.

```
$ qvfb &
$ ./example -qws example
```

Note that you could alternatively run the qpe application from /opt/Qtopia/bin to simulate an actual Qtopia environment and then run the example app in non-server mode.



1.2.2. Check compiler setup for ARM

Alternatively, you may want to build an ARM version to run on the SL-series Zaurus.First, run the ARM environment script in your home directory from a shell session.

```
$ ./dev-arm-qpe.sh
```

If you are building an ARM version right after x86 build, you should have a Make file in the example directory. If there is a Makefile, you need to remove it. It is essential that you do this, or make sure you do not have Makefile before you execute tmake. Also, make sure to run “make clean” within /opt/Qtopia/example to clean out the old temporary files from the x86 configuration.

```
$ cd /opt/Qtopia/example
$ make clean
$ rm Makefile
```

Run tmake again to create the Makefiles for ARM compiling.

```
$ tmake -o Makefile example.pro
```

To build the ARM binary, run make from within that directory.

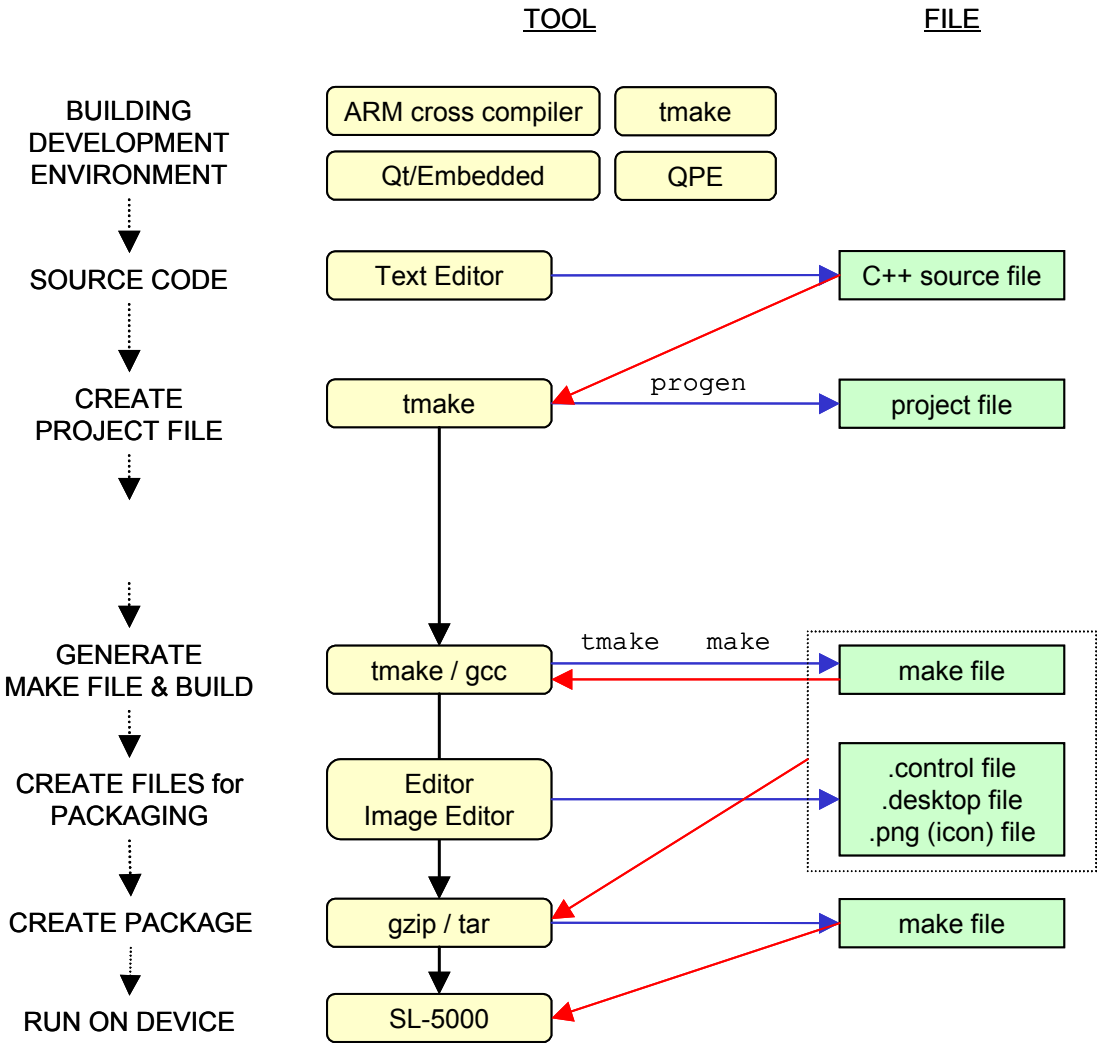
```
$ make
```

Once it has built to run (you should have “example” file in the same directory), it will need to be copied over to the Zaurus (e.g. /home/QtPalmtop/bin) and execute it from the console.

2. Qt/Embedded Application Development Overview

2.1. Application Development Workflow

The software development procedure workflow is summarized below:



As described above, once you built the development environment described in Section 1, the overall workflow is the same as other platforms – creating project, source coding, build, and install package creation. For creating the install packages, there are useful tools already made available in the Zaurus developer community. (http://killefiz.de/zaurus/) You may want to take a look at the "Development" section to find some useful tools for your development.

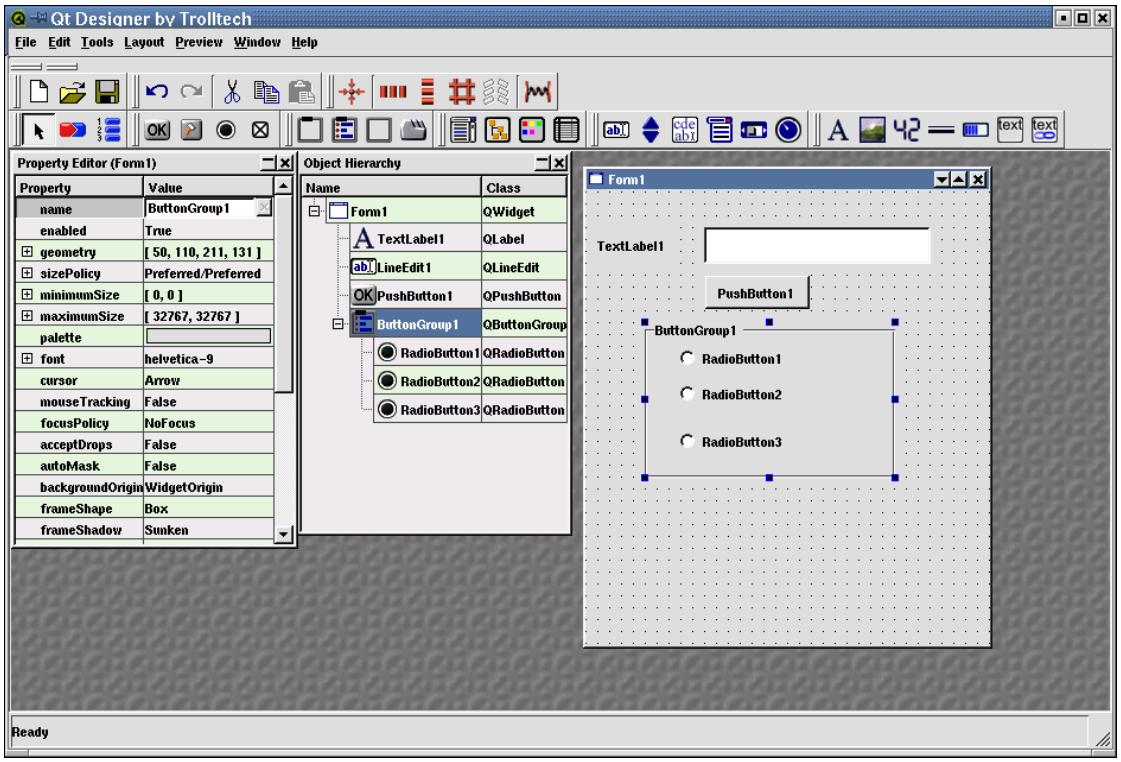
2.2. Useful Tools for the Development

Once you have an ARM cross compiler, tmake, and the Qtopia SDK installed, you are ready to start developing your Qtopia applications. The following are the tools and an overview that will be useful to your development.

2.2.1. The Qt Designer

If your desktop already has Qt/X11 installed, the Qt Designer tool is useful to assist your development efforts. It helps you to organize functionality visually so you can easily create a small yet logical and functional GUI, such as list, buttons, combo-boxes. It also allows you to configure the GUI component attributes, including SIGNAL/SLOT processes corresponding to the GUI component operation.

The Qt Designer tool is in $QTDIR/tools/designer (or /opt/Qtopia/bin/designer). You may find more information on the Qt Designer at http://doc.trolltech.com/2.3/designer.html, and you will see how effectively you may organize the functionality with the GUI. The following is a screenshot of the Qt Designer.



Once you design your application GUI using Qt Designer, you will have *.ui file created. You need to execute the "uic" to these *.ui file in order for the compiler to handle.

2.2.2. uic

The “uic” is a “User Interface Compiler” tool that generates source file(s) for the C++ compiler from the *.ui GUI information file(s) created by Qt Designer.

In order to generate a header file, you may want to execute;

```
$ uic baseform.ui –o baseform.h
```

and in order to generate a source file, you may want to execute;

```
$ uic baseform.ui –i baseform.h –o baseform.cpp
```

2.2.3. moc

The “moc” is a “Meta Object Compiler” tool that generates source file(s) for the C++ compiler from the file(s) that defines Qt event process (SIGNAL/SLOT).

The C++ compiler cannot handle keywords such as “Q_OBJECT”, “signal” or “slot”, while the class libraries provided by Qt already defines SIGNAL and SLOT, or these keywords are usually used to include SIGNAL or SLOT in the class definition. The “moc” is used here so that it generates source file(s) for the C++ compiler to handle these definitions correctly. The following is a header file example that uses the keywords mentioned above.

```
class MyTestClass : public Qobject
{
Q_OBJECT
...
signals:
        // SIGNAL
public slots:
        // public SLOT
private slots:
```

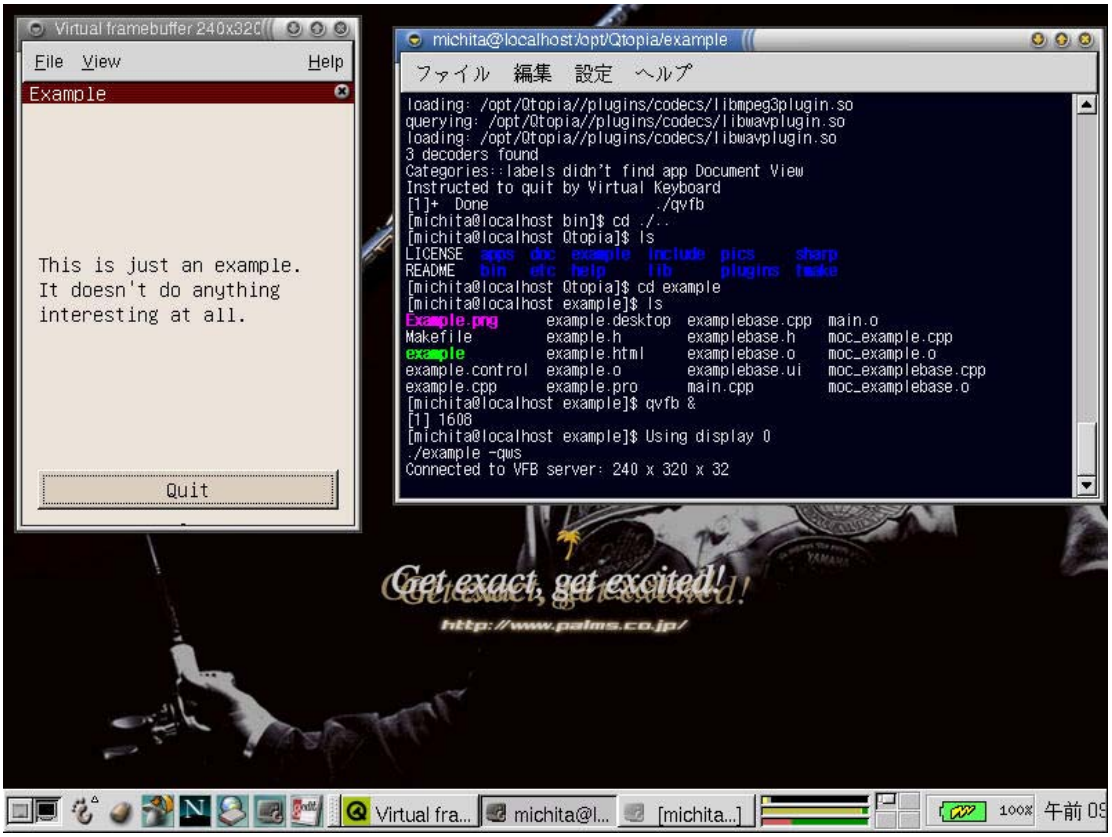
In order to generate a source file by using “moc”, you may want to execute;

```
$ moc mytestclass.h –o moc-mytestclass.cpp
```

By executing the above, you will have the source file so that you can compile link. Alternatively, you may create *.moc file by “moc” tool, and then include the created *.moc file in the source.

2.2.4. qvfb (Qtopia Virtual Frame Buffer)

The "qvfb" is a "Qtopia Virtual Frame Buffer" tool that allows you to simulate your application software on x11. (Also see Section 1). As already described in Section 1, you may test and debug your application to a certain extent without loading your application to the target device (SL-series Zaurus) by using the "qvfb". The following is a screen shot executing the "example" application that came with the Qtopia SDK.



Note that the "qvfb" uses the binary compiled for x86. Also you should note that the device specific part (such as storage device name dev/hdc1) cannot be executed AS-IS on the SL-series Zaurus, and thus you need to make necessary changes by #fdef … #endif.

2.2.5. progen

The "progen" is a tool that generates *.pro file necessary for "tmake" to create a Makefile. The "progen" is located in $QTDIR/tmake/progen (or /opt/Qtopia/tmake/progen).

The *pro file lists *.cpp and *.h files that you wish to make. However, when the project(s) or file(s) are newly added, it is easy to make mistake if you add them manually. The "progen" will help and assist you doing this task, as it scans the necessary files in the directory and add them to the *.pro file. It may also be worth while to create a script that does "progen" and "tmake" in a series, so that you do not have to do them individually.

2.2.6. Tools for multi-language support

The following are the tools used to support multi-languages:

findtr

The “findtr” a tool is to find string(s) enclosed by tr() (such as `QObject::tr(“this is a test string.”)`), and output to *.po file. The following is an example on using this tool:

```
$ findtr test.cpp > test.po
```

This *.po file includes the header part and string setting part. The most important portion in the header part is the line specifying the character code. You will find the following line in the .po file. Make sure that you apply the same encoding to the .po file with the one specified by this `charset`.

```
“ContentType= ~ charset=****¥n”
```

As to the string setting part, you will see a file name, line number, and the original string. You may want to set the translations corresponding to the original string(s). The translation should be edited to `msgstr`.

```
#: test.cpp:36
msgid “test::this is a test”
msgstr “” //translations here
```

Once the .op file is created, you may want to use “msg2qm” tool to convert it to the translation file. (see below for how to use this tool.)

msg2qm

The “msg2qm” is a tool to convert *.po file(s) to *.qm file(s). The following is an example for how to use this tool:

```
$ msg2qm test.po > test.qm
```

In order to use this translation file (*.qm file), you would have to add certain procedure at the beginning of the application start routine. The following is an example for how to add this procedure.

```
int main( int argc, char *argv[] )
{
        QPEApplication app( argc, argv );
        QTranslator    translator( 0 );
        translator.load( “test.qm” );
        app.installTranslator( &translator );
        …
}
```

As shown in the above example, you would first load the translation file (*.qm file) to the Qtranslator class, and then install to the Qapplication class. Note that this “msg2qm” command is not included in the Qtopia SDK, and thus you would have to build it for x86 from the source included in free verion of the Qt/Embedded, or use a tool in Qt/X11.

2.3. Special Recommendations for the SL-series Zaurus Applications

In order to provide unified and a better user experience to the SL-series Zaurus users, Sharp requests and strongly recommends application developers to implement the following features in your application for the SL-series Zaurus.

2.3.1. Operation of “Menu” Key

The default (pre-installed) SL-series Zaurus applications are designed to “open” the “menu” of the application itself, and “close” the opened “menu” of the application, when the “Menu” key is pressed.

In addition, the default applications implement a function allowing users to use cursor keys to change focus of the “menu” if there are multiple menus on the application’s menu bar(e.g. “File” “Edit”, etc.). The following example will allow you to use “menu” button to work as recommended.

main.cpp

```
#include <qpeapplication.h>
#include "appsample.h"
#include <qstring.h>

#include <stdio.h>

int main( int argc, char ** argv )
{
    QPEApplication a( argc, argv );

    AppSample mw;
    a.showMainDocumentWidget( &mw );

    return a.exec();
}
```

appsample.cpp

```
#include "appsample.h"
#include "resource.h"

#include <qmenubar.h>
#include <qwidgetstack.h>
#include <qpetoolbar.h>
#include <qaction.h>
#include <qfiledialog.h>
#include <qmessagebox.h>
#include <qpopupmenu.h>
#include <qlabel.h>
#include <qpainter.h>
#include <qkeycode.h>
#include <qapplication.h>
#include <qclipboard.h>
#include <qtimer.h>
#include <qsizepolicy.h>
#include <qpeapplication.h>
#include <qcopenvelope_qws.h>
#include <qpedecoration_qws.h>
#include <config.h>
#include <qcolor.h>


//======================================================================

AppSample::AppSample( QWidget *parent, const char *name, int wFlags )
    : QMainWindow( parent, name, wFlags )
{
    setCaption( tr("Application Sample") );

    setToolBarsMovable( FALSE );

    // GUI Layout
    QPEToolBar *toolBar = new QPEToolBar(this,"tool");
    toolBar->setHorizontalStretchable( TRUE );
    addToolBar(toolBar,"tool",QMainWindow::Top,TRUE);
    QPEMenuBar *menuBar = new QPEMenuBar( toolBar );
    QPopupMenu *listMenuFile = new QPopupMenu( menuBar );
    QPopupMenu *listMenuOption = new QPopupMenu( menuBar );
    menuBar->insertItem( tr("File"), listMenuFile);
    menuBar->insertItem( tr("Options"), listMenuOption);
```

appsample.cpp (continued)

```
    // File menu
    actMenu1 = new QAction( tr( "Menu1" ), QString::null, 0, this, 0 );
    connect( actMenu1, SIGNAL( activated() ), this, SLOT( slotMenu1() ) );
    actMenu1->addTo(listMenuFile);

    // Option menu
    actMenu2 = new QAction( tr( "Menu2" ), QString::null, 0, this, 0 );
    connect( actMenu2, SIGNAL( activated() ), this, SLOT( slotMenu2() ) );
    actMenu2->addTo(listMenuOption);

}

AppSample::~AppSample()
{
}

void AppSample::setDocument(const QString& fileref)
{

}

void AppSample::slotMenu1()
{
  QMessageBox::warning(this,tr("menu"),tr("Menu1"));
}

void AppSample::slotMenu2()
{
  QMessageBox::warning(this,tr("menu"),tr("Menu2"));
}


// eof
```

appsample.h

```
#ifndef __APPSAMPLE_H__
#define __APPSAMPLE_H__

#include <sys/time.h>
#include <unistd.h>
#include <qwidget.h>
#include <qmainwindow.h>
#include <qimage.h>
#include <qlabel.h>
#include <qlayout.h>
#include <qpetoolbar.h>
#include <qaction.h>
#include <qpemenubar.h>
#include <qslider.h>
#include <qlibrary.h>
#include <qdatetime.h>
#include "applnk.h"

class AppSample : public QMainWindow
{
    Q_OBJECT
public:
    AppSample( QWidget *parent=0, const char *name=0, int wFlags=0 );
    ~AppSample();

private slots:
    void setDocument(const QString& fileref);

    void slotMenu1();
    void slotMenu2();
protected:
    QAction *actMenu1,*actMenu2;
};


#endif // end __APPSAMPLE_H__
```

2.3.2. Operation of “OK” and “Cancel” key

The default (pre-installed) SL-series Zaurus applications implement the same functions on “OK” key on the device and the (OK) button at the top-right corner of the application title bar. Likewise, the same function is implemented with the “Cancel” key on the device and the (X) button on the top-right corner of the application title bar. It is recommended to do so in your applications as well.

2.3.3. Access to the device VRAM

You may want to consider direct access to the device VRAM in order to accelerate the drawing performance of your application. However, VRAM access may vary among the SL-series Zaurus, and may cause your applications to have device dependency.

In order to avoid the device dependency, it is recommended to use the `QDirectPainter` class in order to resolve this problem. Please note that following method is added to Qt 3.0, and cannot be used on the SL-series Zaurus (as it implements Qt/E 2.3.2).

```
void QDirectPainter::setAreaChanged(const QRect &);
```

2.4. Qtopia Development Tutorial

Now that you know how to build the development environment, overall workflow, useful tools, and special conditions for the SL-series Zaurus, you may want to develop your application. The following is a brief tutorial on actually developing a Qtopia application following the overall workflow mentioned above.

Note that following tutorial assumes that you are working in `/home/user1/work-dir`.

2.4.1. Hello World!!

The following is the source that only shows "Hello World!!" on the display. Once you complete editing, save this file as `main.cpp` in the `/home/user1/work-dir`.

```
#include <qpe/qpeapplication.h>
#include <qlabel.h>

int main( int argc, char *argv[] )
{
        QPEApplication app( argc, argv );
        QLabel *label = new QLabel( "Hello World!!" ,
                (Qwidget*)0 );
        label->resize( 100, 50 );
```

2.4.2. Creating the Project File

Use the "progen" tool to create a .pro file. In order to establish a PATH to the "tmake" directory, execute the following commands. Note that following example assumes that you have the `dev-x86-qpe.sh` file in `/home/user1` directory.

```
$ . ../dev-x86-qpe.sh
$ progen -o qpe-test.pro
```

Once you execute the above commands, you have your environment configured to compile for x86, and have the files listed in `qpe-test.pro` file. However, you have to add following the information to the the created `qpe-test.pro` file so that you can actually use this file for further process.

```
DESTDIR = ./
INCLUDEPATH += $(QTDIR)/library
DEPENDPATH += $(QTDIR)/library
TARGET = qpe-test
LIBS += -lqpe
```

If you have a .ui file besides the source file, you may want to add INTERFACE tag in the .pro file, so that you will have the contents of the procedure automatically added to the Makefile.

```
DESTDIR = ./
INCLUDEPATH += $(QTDIR)/library
DEPENDPATH += $(QTDIR)/library
TARGET = qpe-test
LIBS += -lqpe
INTERFACE = qpe-test.ui
```

2.4.3. Creating the Makefile

Use the "tmake" tool to create the Makefile based on the .pro file generated by the "progen":

```
$ tmake -o Makefile qpe-test.pro
```

Once you execute the above command, now you have the Makefile to build the `main.cpp`.

2.4.4. Executing make to build the application

Use "make" command to build `main.cpp` using the created Makefile:

```
$ make
```

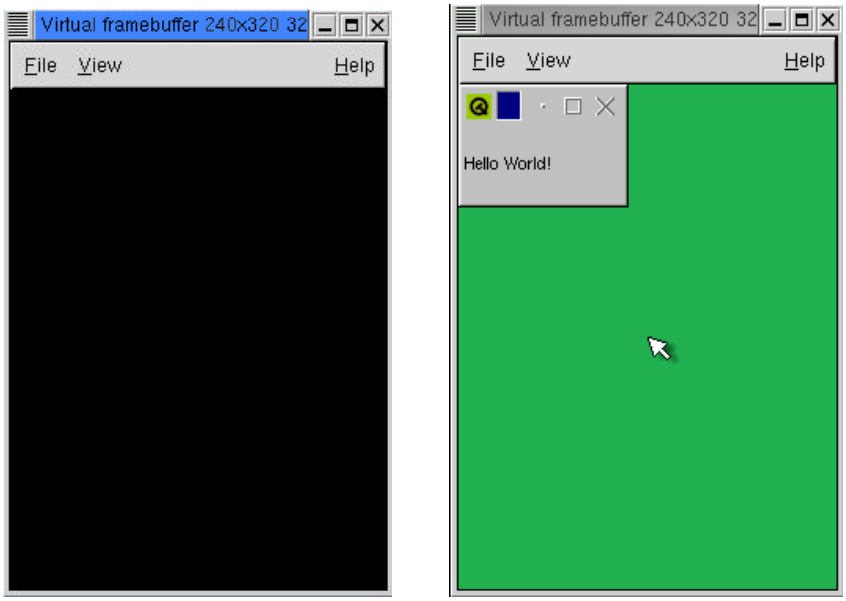
Once you execute, unless you modified the `main.cpp` file or delete the files created by tmake (such as .o file), you cannot re-execute make. In such case, execute the following to initialize, and re-issue make command.

```
$ make clean
```

2.4.5. Running Hello World on the "qvfb"

Once you complete the building of your source file, you should have a `qpe-test` file in the `/home/user1/work-dir`. You may use the "qvfb" tool to see how it runs on your PC. Execute the following commands to see if you get the same of similar screenshot on your PC. Make sure to execute qpe-test after the "qvfb" is booted:

```
$ qvfb &
$ ./qpe-test -qws qpe-test
```



The screenshot on the left is the one you will see after you run "qvfb", and the right is the one when you run qpe-test on the "qvfb".

2.4.6. Running Hello World on Qtopia environment

Qtopia would also run on the "qvfb". If you run your application on the Qtopia running on "qvfb", it would provide a closer image of the application running on the target device. This will also allow you to check and confirm how to writea .desktop file on the installing destination. (See section 3 for the install package details.)

(STEP 1) Layout necessary files

In order to run the applications in the Qtopia running on "qvfb" of PC-Linux, you would have to layout the following files to the following directories. For the details of each file, including what and how to write, see Section 3.

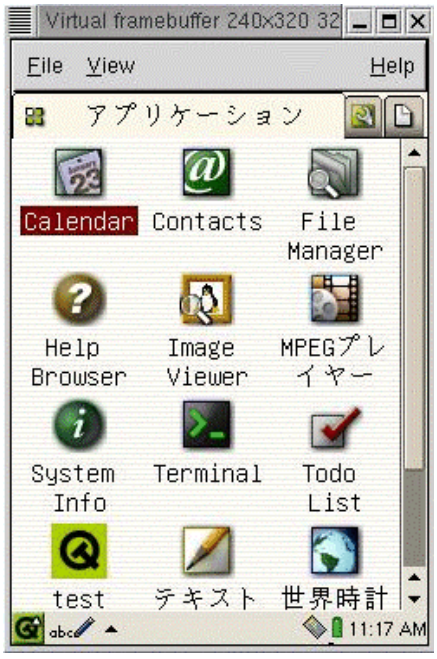
directories		Files to be located
opt/Qtopia	apps/Applications	.desktop file
	bin	executable binary
	pics	icon file
	i18n/en	.qm file (translation file)
	help/en/html	help file (.html file)

(STEP 2) Run both "qvfb" and "qpe"

The Qtopia environment ("qpe") also runs on "qvfb". Execute the following command to run Qtopia:

```
$ qvfb &
$ qpe
```

If you correctly created the files needed, and correctly locate the files, you will see your icon on the Qtopia launcher. The following is an example screenshot. You may actually use the icon to start and you're your application.



2.5. Providing help file with your application

The default (pre-installed) SL-series Zaurus applications have a (?) button on the top-right corner of the application title bar. When you tap this (?) button, the application will show its help file to the users.

This function can be implemented simply by preparing the help file in an html document, and placing them in the correct directory. `/opt/QtPalmtop/help/en/html` (in case of English) is the place to locate the html file(s). You can create the html file(s) just like you do for websites. Make sure to apply the same character encoding to the file as defined in charset tag in the html document.

2.6. Converting character code (for local language support)

The translation file (*.qm file) can be used to support multi-language or local languages. However, you may also directly use the local language characters in the application source. The following is an example of how to do so by using the Japanese characters in the source. Note that multi-language support also requires the corresponding font file as well.

In order to directly use Japanese characters in the application source, the `fromUtf8()` method of the `QString` class is used. This method converts the UTF8 character set to its arguments to Unicode. Thus, the local character (in this case Japanse) provided to the `fromUtf8()` method needs to be written in UTF8 format.

The following is an example on creating a label with the Japanese character “ ラベル ”.

```
QLabel *mylabel= new QLabel( "dummylabel" , this );
mylabel->setGeometry( 10, 10, 100, 30 );

mylabel->setText( QString::fromUtf8( "ラベル" ) );
```

2.7. Event handling: SIGNAL and SLOT

In order for the application to interact with end-users, the application must have an event handling process. The following sections briefly describe how to handle events, namely known as SIGNAL and SLOT.

2.7.1. Using already-defined SIGNAL and SLOT

The Qt/Embedded, or Qtopia uses a framework so called SIGNAL and SLOT to handle events. The following is an example of an event handle process that displays a button, and closes the application when the button is pressed (or tapped).

```
QPushButton *quitbutton = new QPushButton( "quit" );
connect( quitbutton, SIGNAL(clicked()), qApp, SLOT(quit() );
```

The first line is the process to create [quitbutton], and the second line is the process to close the application when the [quitbutton] is pressed.

When the [quitbutton] is clicked, a “clicked” SIGNAL is issued, and the closure process that is set by `quit()` on `qApp` will be executed. The `qApp` is a pointer defined by the Qt that points at `QApplication` object. Because `clicked()` SIGNAL and `quit()` SLOT are already defined by Qt, you simply need to state that in your code.

For the other defined SIGNAL and SLOT, you are encouraged to look up in the Qt reference documentation provided on the Trolltech websites. You will see `pressed`, `released`, `clicked`, `toggled`, `stateChanged` SIGNALs in the `QButton` class, or `setText`, `setPixmap`, and `setPicture` SLOTs for the `QLabel` class.

2.7.2. Creating your own SLOT

If the already-defined SIGNAL or SLOT cannot quite handle the desired process, you can also create your own SIGNAL or SLOT. As it is quite rare that you need your own SIGNAL, this document only describes how to create your own SLOT. If you need to create your own SIGNAL, please refer to the Qt reference documentations.

In order to create your own SLOT, you need to do the following tasks:

- Add SLOT definition to the class definition
- Create the actual SLOT
- Connect SIGNAL and SLOT
- Use “moc” (Meta Object Compiler) to generate a source for the C++ compiler

Add SLOT definition to the class definition

In order to create your own slot, you would have to add a Q_OBJECT statement in the ordinary class definition. In addition, you would also have to add a definition of your own SLOT that you are about to create, by stating the SLOT name to either `public slots:` or `private slots:`.

```
class myMainWindow : public Qwidget
{
     Q_OBJECT
       public:
       myMainWindow();
       public slots:
                 void testSlot();
       private:
                 Qlabel *mylabel;
        };
```

Create the actual SLOT

You do not need any special tricks to implement your own SLOT. Code your SLOT just like you do for the functions and methods you implement.

Connect SIGNAL and SLOT

Use `connect` method to connect the SIGNAL and the SLOT. The following is an example that executes the created `testSlot` when mybutton is clicked.

```
Connect( mybutton, SIGNAL(clicked()), this, SLOT( testSlot()) );
```

Using "moc" (Meta Object Compiler)

The SIGNAL and SLOT framework is Qt specific, and thus you need to do the necessary task for the compilation. As mentioned earlier, you need to use a "moc" tool to generate source file(s) so that the C++ compiler can recognize and handle the code. (See section 2.2.3 for the "moc" tool information.)

Note that the “moc” procedure will automatically be executed when you use Makefile to compile, if you specify the file in the project file (.pro file). The following is an example of .pro file that automatically does the “moc” procedure. With this file, one can use “tmake” tool to create Makefile, and do the compile link.

```
TEMPLATE        = app
CONFIG          = qt warn_on release
HEADERS         = tut1.h
SOURCES         = tut1.cpp
INTERFACES      =
INCLUDEPATH     += $(QPEDIR)/include
DEPENDPATH      += $(QPEDIR)/include
TARGET          = tut1
LIBS            += -lqpe
```

2.8. Sample application source

As a summary of the above sections, the following is a set of sample application source that includes character code conversion as well as own SLOT function.

This sample application changes “Hello Zaurus World” label when the “ ボタン ” labeled button is pressed. It also terminates the application when the “quit” button is pressed. This sample application consists of tut1.h, tut1.cpp, and tut1.pro files.

tut1.h

```
class myMainWindow:public QWidget
{
      Q_OBJECT
              public:
                myMainWindow();

              public slots:
                void changeLabelSlot();

              private:
                QLabel       *mylabel;
                QPushButton   *quitbutton;
                QPushButton   *mybutton;
};
```

tut1.cpp (saved in UTF8 code)

```
#include <qpe/qpeapplication.h>
#include <qlabel.h>
#include <qpushbutton.h>
#include "tut1.h"

    void myMainWindow::changeLabelSlot()
    {
        mylabel->setText( QString::fromUtf8("ボタンが押されました"));
    }

    myMainWindow::myMainWindow()
    {
        setGeometry( 0, 0, 240, 320 );
        mylabel = new QLabel(this, "Hello Zaurus World" );
        mylabel -> setGeometry( 50, 80, 160, 30 );
        mybutton = new QPushButton( this, "button" );
        mybutton -> setGeometry( 70, 140, 100, 30 );
        mybutton -> setText( QString::fromUtf8("ボタン") );
        quitbutton = new QPushButton(this, "quit" );
        quitbutton -> setGeometry( 180, 5, 50, 30 );

        connect( quitbutton, SIGNAL( clicked()), qApp, SLOT( quit()) );
        connect( mybutton, SIGNAL( clicked()), this,
        SLOT( changeLabelSlot()) );
    }

    int main( int argc, char** argv )
    {
        QPEApplication myapp( argc, argv );
        myMainWindow mywidget;
        myapp.setMainWidget( &mywidget );
        mywidget.show();
        return myapp.exec();
}
```

tut1.pro

```
TEMPLATE        = app
CONFIG          = qt warn_on release
HEADERS         = tut1.h
SOURCES         = tut1.cpp
INTERFACES      =
INCLUDEPATH     += $(QPEDIR)/include
DEPENDPATH      += $(QPEDIR)/include
TARGET          = tut1
LIBS            += -lqpe
```

Expected Result (running on the “qvfb” on Linux-PC)

2.9. Development with the QtDesigner

The QtDesigner (also see section 2.2.1) is a useful tool that helps and assists you to design the application GUI as well as to configure the event handling. When you use this tool, you may need to take somewhat different steps than the ones described in the above sections.

When QtDesigner is used, the overall development procedure will be; (i) design application GUI by QtDesigner, (ii) generate the source file and the header file from the QtDesigner file (*.ui), (iii) create subclasses to the generated class and add procedures other than GUI design to the generated class file(s) to compile-link.

The following is an example of the development flow using the QtDesigner for application development.

(STEP 1) Invoking QtDesigner

You would first want to invoke the QtDesigner (/opt/Qtopia/bin/designer). Note that some Linux distributions may have newer version of the QtDesigner (than the one that comes with the Qtopia SDK 1.5.0) depending on the distribution. Please make sure to invoke the QtDesigner installed in the above location.

(STEP 2) Design GUI by the QtDesigner

Once you are done with designing the application GUI, you may want to save the GUI design in a .ui file. (This application development flow assumes that the GUI design is saved in a "form1.ui" file.)

(STEP 3) Create main.cpp

The QtDesigner will generate a class when you design your application GUI by this tool (in case of this example, Form1). The name put into the "name" field of the QtDesigner property editor is in fact the class name.

In the main.cpp file, you need to create an object corresponding to that class (see line #7). You would also have to make an include statement so that it includes the header file of the created class (see line #2):

main.cpp

```
1:      #include <qpe/qpeapplication.h>
2:      #include "Form1.h"
3:
4:      int main( int argc , char **argv )
5:      {
6:              QPEApplication  myapp(argc, argv);
7:              Form1   w;
8:              myapp.showMainWidget( &w );
9:              return myapp.exec();
10:     }
```

(STEP 4) Create project file

You may want to use the "progen" tool to create the project file for your application. Execute the following command file you have form1.ui and main.cpp file in your current working directory.

```
$ progen  -o  demo.pro
```

The generated demo.pro file sets main.cpp file for the SOURCES tag, and form1.ui file for its INTERFACES tag.You would also have to add elements described in the INCLUDEPATH and below:

demo.pro

```
TEMPLATE        = app
CONFIG          = qt warn_on release
HEADERS         =
SOURCES         = main.cpp
INTERFACES      = form1.ui
NCLUDEPATH      += $(QPEDIR)/include
DEPENDPATH      += $(QPEDIR)/include/qpe
TARGET          = demo
LIBS            += -lqpe
```

(STEP 5) Create Makefile by "tmake", and build

Once you have successfully created the project file, you would then want to create Makefile, and then make to generate source and header file for the GUI design file (.ui). Use "tmake" to create Makefile, and the execute "make":

```
$ tmake -o Makefile demo.pro
$ make
```

When you execute "make" while the GUI design file (in this example, "form1.ui") is specified in the INTERFACES tag of the project file (.pro), it will automatically create the source file as well as the header file based on the specified .ui file.

(STEP 6) Add other process(es) to the Form1 class (except for the GUI design)

You may want to add the necessary processes (such as SLOT handling etc.) to the generated source file by executing make.

(STEP 7) Modify the project file

By executing the make in STEP 5 above, you now have form1.h header file and form1.cpp source file from the form1.ui file. Thus, you may want to modify your demo.pro project file so that you can build your application correctly.

Modifications that you need to take care of are; (i) add form1.h to the HEADERS tag, (ii) specify both main.cpp and form1.cpp to the SOURCES tag, (iii) remove form1.ui statement from the INTERFACES tag. Make sure to remove form1.ui statement from the INTERFACES tag, or it will automatically generate the source and header files for form1.ui, and will remove all of the changes you added to the source file in STEP 6.

demo.pro (modified)

```
TEMPLATE        = app
CONFIG          = qt warn_on release
HEADERS         = form1.h
SOURCES         = main.cpp form1.cpp
INTERFACES      =
INCLUDEPATH     += $(QPEDIR)/include
DEPENDPATH      += $(QPEDIR)/include/qpe
TARGET          = demo
LIBS            += -lqpe
```

(STEP 8) Re-create Makefile by "tmake", and build

Once you have modified the project file, you would then want to recreate Makefile, and then make to build your application. Use "tmake" to create Makefile, and the execute "make":

```
$ tmake -o Makefile demo.pro
$ make
```

Note that you need to re-create the project file if you change the GUI design by the QtDesigner tool.

3. Installing Applications to the SL-series Zaurus

Once you have successfully built your application, you need to create the installation package, known as "ipkg" package to actually install your applications on the SL-series Zaurus.

In order to create the ipkg package, application developers must first make the directory structure on the PC-Linux and create some necessary files, and make sure to locate all necessary files in the appropriate directories. This is important to correctly install and run applications on the SL-series Zaurus.

Please also note that the SL-5600, SL-C700 (Japan domestic model), and SL-B500 (Japan domestic model) now adopt "root" and "user" privileges for security reasons. Different from the SL-5500 (and SL-A300), these new SL-series Zaurus run 3rd party applications with the "user" privilege by default. It is also strongly recommended to **incorporate the special considerations for thse models** so that your application will run all of the SL-series Zaurus.

3.1. The ipkg package

The SL-series Zaurus uses the ipk package format. iPKG is a very lightweight package management system. It was designed for Linux installations with severe storage limitations such as handheld computers. This document provides the basic knowledge on how to build an ipk. Advanced features, and more detailed explanations of ipk can be found at http://www.handhelds.org/z/wiki/iPKG.

An .ipk file is basically a gzipped tar archive containing following 3 members:

`./data.tar.gz`: contains the actual files belonging to this package. The contents of this directory will be extracted to "/" (The root directory) when installed by ipkg. So it should contain entries such as ./usr and ./etc as top-level directory entries, if necessary.

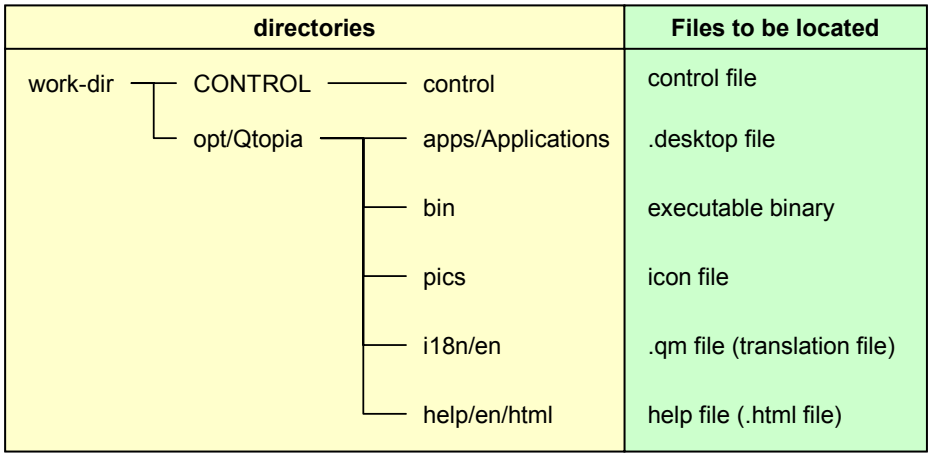
`./control.tar.gz`: contains meta-data and scripts for the package. It must contain a file named `control` (see following sections for the details). It also may contain the following files: `conffiles`, `preinst`, `postinst`, `prerm`, `postrm`.

`./debian-binary`: This file is currently ignored by ipkg. However, in all current ipkgs it is a text file with a single line: 2.0

3.1.1. Making directories for .ipk package

In order to create an .ipk package, you may first want to make the directory structure on your PC-Linux that is the same directory structure as the on on the SL-seires Zaurus. On the SL-series Zaurus, because applications are to be installed in the /opt/QtPalmtop directory, the following illustrates the typical directory struture that you should make on your PC-Linux.

Note that following explanations and example assumes creation of very basic .ipk, and you are encouraged to also refer to http://docs.zaurus.com/ipkg_howto.shtml or http://www.handhelds.org/z/wiki/iPKG for more advanced usage:



3.1.2. control File

The control file is a file that describes and specifies the details and contents of the ipk package. The SL-seires Zaurus installing/uninstalling application (“Add/Remove Software”) uses the information in this file to install the application. This package should contain the following entities. The entity marked with **M** are mandatory:

Entities

M	**`Package:`**	The name of the package and should match the regular expression [a-z0-9.+-]¥+
	`Files:`	Files included in the package, including directry.
	`Priority:`	This entity should be one of: `required`, `standard`, `important`, `optional`, or `extra`. Most programs should use `optional`.
	`Section:`	The catagory that best fits this type of package: - Games - Multimedia (Graphics, video/audio/picture viewer or player) - Communications (Instant messaging, email, etc) - Settings (anything that modifies the system) - Utilities (more often smaller apps) - Applications (Anything that couldn't fit in any of the above)
M	**`Maintainer:`**	This entity should be the name and email address of the person responsible for maintaining the package, (not necessarily the author of the program).
M	**`Architecture:`**	This entity should specify the architecture for which the package is compiled. Valid values currently include "arm" and "all".
M	**`Version:`**	This entity should have at least one digit and should match [a-zA-Z0-9.+]*. Version may also contain an optional trailing revision matching "-fam![0-9]¥+". This revision should be incremented each time the package changes but the version does not, (ie. a packaging tweak). It may be reset, (or simply omitted), each time the version is incremented.

	Depends:	This entity indicates packages which must also be installed in order for this package to work. The packages should be listed on a single line, separated by commas.
M	**Description**	This entity should be a short, (less than 80 characters) description of the program. It may also include a long description on subsequent lines, (each indented by a single space character). Blank lines in the long description may be indicated by a line consisting of a space character followed by a period, ie " ."

control file example

```
Package: foobar
Priority: optional
Section: Misc
Version: 0.1
Architecture: arm
Maintainer: Familiar User famuser@foo.org
Depends: libc6
Description: foo is the ever-present example program -- it does
Everything foo is not a real package. This is simply an example.
```

3.1.3. desktop File

The .desktop file is used to define the icon and information that will be given to the Qtopia desktop (launcher). The following entities should be included in this file.

Entities

[desktop Entry]	
Comment =	Brief explanation of this application
Exec =	Program file name, or executing script file name
Icon =	Application icon file name
Type =	Type of the installing package. Most program should state "Application".
Name =	Application name to be displayed on the desktop
CanFastload =	1 to display the checkbox or 0 to hide the checkbox in the "Details" window displayed by tap-and-hold the application icon
HidePrivilege =	1 to hide the checkbox or 0 to display the checkbox in the "Details" window displayed by tap-and-hold the application icon

Note that you must make sure to state exactly the same file name for the program and the icon for the Exec entity and Icon entity, or the ipk package will either fail to install or fail to execute. The icon file should be created and saved in PNG format. You may use any size of the icon image so long as it is saved in PNG format. The system will automatically adjust the size of the icon image to display it on the Qtopia desktop (launcher).

.desktop file example

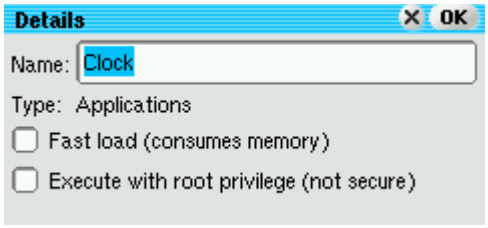
```
[Desktop Entry]
Comment = Qt Sample
Exec = foobar
Icon = foobar
Type = Application
Name = Foobar
CanFastload = 1
HidePrivilege = 0
```

3.1.4. Special Considerations for SL-5600, SL-C700, and SL-B500

(A) QuickExec (Fast Load) entity for .dektop file

The SL-5600, SL-C700, and SL-B500 now incorporate so called "QuickExec" or "Fast Load" capability to minimize the time required to invoke the application.

Enabling the "QuickExec" or "FastLoad" option can be configured by the "Details" window displayed by tap-and-hold the application icon. The `CanFastload =` entity will either show or hide this checkbox. If this function is enabled, the SL-series Zaurus will remain cached even if the application is "closed" by the user, but it consumes memory (and thus the freely available heap area becomes shortened).



(B) Run Applications with "root" Privilege entity for .desktop file

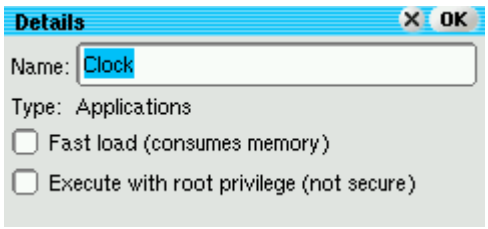
As mentioned earlier, the SL-5600, SL-C700, and SL-B500 now adopt "root" and "user" privilege model for the security reasons. For those security reasons, the SL-5600, the SL-C700, and SL-B500 will not install any 3rd party applications to automatically run with "root" privilege.

However, if necessary, you can make the "Details" window display the checkbox for this configuration, and ask users to change the privilege to run your application as "root". (Just as running the application on the SL-5500). In order to do so, you must add the following statement in the .desktop file to show the checkbox in .desktop file:

```
HidePrivilege = 0
```

Unless the application calls for root privilege to execute and perform expected results, **it is strongly recommended that developers should state "`HidePrivilege = 1`" in its .desktop file so to hide this checkbox from users for security reasons.**

Note that this checkbox for "user" or "root" privilege configuration is set to OFF (meaning execute application by the "user" privilege) by default for user and system security reasons. This default value cannot be configured to ON by default, and thus users must set "root" privilege manually upon their decision.



(C) The SL-5600 (SL-C700 and SL-B500) file system and access privileges

Because the SL-5600 (and SL-C700, SL-B500) incorporates "user" privilege to execute processes or to access files in the system to enhance the security features, you must be careful about permission restrictions when running applications, accessing files already on the device, or creating new files on the Zaurus.

The "Add/Remove Software" on the SL-5600 runs with "root" privilege, and thus all files in the ipk packages can be stored or made in any directories. However, when installed, the system will try to execute the application binaries with "user" privileges, and thus applications will fail to access files and directories if the correct privilege or permissions are not given. Appendix. A provides access permission list for the SL-5600.

You may want to use specific files (ipkg script files) to resolve this permission problem. See section 3.1.6. for the ipkg script file details.

3.1.5. Creating ipk file

Once you've made the directory structure illustrated in section 3.1.1, and located all of the necessary files in the corresponding directory, you are ready to create .ipk file for installation.

There is a useful tool in the developer community called "ipkg-build" to create the .ipk file. You may obtain this tool from the following URL:

IPKG FIND: http://ipkgfind.handhelds.org/result.phtml?section=base

ipkg-build: http://ipkgfind.handhelds.org/details.phtml?package=ipkg-build&official=&format=

Once you've downloaded this tool (this tool is actually a command script), you should then extract this tool to the directory where the PATH is enabled (such as /user/bin). The following is a basic usage of this tool. By executing the above command, you will have the entire directory (`/home/(your user name)/(working directory)`) included in the .ipk file:

```
$ cd /home/(your username)
$ ipkg-build (working directory the directories for ipk file are made)
```

3.1.6. ipkg Script

If needed, the package may include some scripts that will be involved by the package maintenance system. There are four possible times a script will be run: just before the package is installed, just after the package is installed, just before the package is removed, and just after the package is removed.

preinst	A script file that can be executed before ipk installation
postinst	A script file that can be executed after ipk installation
prerm	A script file that can be executed before ipk uninstallation
postrm	A script file that can be executed after ipk uninstallation

These scripts should be located in the CONTROL directory. The scripts should return 0 on success, (a non-zero return value from preinst will prevent the package from being installed -- this can be useful in rare situations). The scripts should not assume a tty is available so they may not prompt the user.

Each script file can be stated just like any other script file. The following is an example for a "`postinst`" file that creates a "`data`" directory accessible by "user" privilege in `/home/`. This postinst file would be included in the application ipk so that the directory would be created along with the application installation. Note that the variable PKG_ROOT is set to the root of the package installation and can be used to refer to the packages contents in their installed locations.

```
#!/bin/sh
mkdir /home/data
chmod +w /home/data
```

3.2. Transferring the ipk package to the SL-series Zaurus

In order to install the application's ipk package to the SL-series Zaurus, there are a couple of ways to do so. In any case, you need to "transfer" the ipk file to the device itself or to a memory media that can be plugged into the device. In order to do so, you may choose one or more of the following ways that best fits to your environment:

1. CF memory card, or SD memory card
2. LAN (by NFS)
3. Synchronization Software (comes with the SL-series Zaurus)

3.2.1. Copying the ipk package to CF memory card, or SL memory card

As the SL-series Zaurus has a CF card type-II slot and a SD card slot, the easiest way of installing the application ipk package is to use these memory cards. You may simply copy the ipk package to its designated directory. Note that these memory cards need to be FAT16 formatted.

The following is an example of the command that you need to execute on your PC-Linux to copy the .ipk package to the memory cards. The following example assumes the device name of the memory card is /dev/hdc1, and its mount point is /mnt/card.

```
$ mount -t vfat /dev/hdc1 /mnt/card
$ cp qpe-test_0.1-1_arm.ipk /mnt/card
```

To eject the memory card after copying the ipk package, you may want to execute the following command. You may also want to execute sync command to completely output the date to the file:

```
$ sync
$ umount /mnt/card
```

NOTE) The file names will need to be recognizable within the SL-series Zaurus file system; make sure that the file name is qualified as follows: characters must be 1 byte, "A - Z" (capital), "0 - 9", and "_"(underscore)

When completed, just insert the memory card to the SL-series Zaurus. The SL-series Zaurus will automatically detect the memory card when inserted and a card icon will appear on the task bar. The memory card will be automatically mounted at the following mount points:

CF Card	`./var/mnt/cf`
SD Card	`./var/mnt/card`

3.2.2. Download ipk package by using NFS

If you have a local area network in your development environment, you may also use NFS to download the ipk package to the SL-series Zaurus from your PC-Linux. In order to do so, you need to have the "terminal" application installed on the SL-series Zaurus. The "terminal" application can be obtained from http://www.myzaurus.com/downloads.asp.

(STEP 1) Enable LAN access on your Zaurus

Insert the network card to the SL-series Zaurus If the network card is successfully recognized upon its insertion to the device, a card icon appears on the task bar. In the case the CF network card is not recognized, pull out the card and re-insert to the SL-series Zaurus. In addition, the following commands from the "terminal" application will also provide the same result:

```
# cardctl eject
# cardctl insert
```

If the device still does not recognize the network even by executing the above above commands, reset the SL-series Zaurus to have cardmgr services available.

(STEP 2) Configure network interface of your Zaurus

Invoke the "Network" application from the "Settings" tab, and enable the TCP/IP networking service on the SL-series Zaurus. See the Operation Manuals for how to configure the network interface of the SL-series Zaurus.

(STEP 3) Mounting file system by NFS

Invoke the "terminal" application on the "Applications" tab. When the application successfully invoked, a console (shell) screen will appear. For instance, if the `/server` directory of the nfs server (whose IP address in SS.SS.SS.SS) is exported for the client, the following command will allow client access to the `/server` directory of the server under `/mnt/net`:

```
$ mount -t nfs SS.SS.SS.SS:/server /mnt/net
```

NOTE) The IP address in Figure 3-8 (SS.SS.SS.SS) must be the IP address of the actual server, and should be entered by dicimal dot expression.

In order to make this command valid, the file system has to be exported on the server side prior to executing the above. For the details, execute the following command to check how to configure to export the file system. The above configuration will allow remote device file system access using NFS.;

```
$ man exports
```

NOTE) nfs daemon must be invoked at the server side prior to execution of above command. Please refer to the manual for "exports".

3.2.3. Download the ipk package by the Synchronization software

You may also use the synchronization software (that comes with the SL-series Zaurus) to download the ipk package to the SL-series Zaurus. You can simply transfer the ipk package to the device by using the file transfer operation of the synchronizing software.

Once you complete the transfer, the ipk package should be stored in the designated location of the SL-series Zaurus. See the Operation Manuals on the synchronization software for the details.

3.3. Install the ipk package on the SL-series Zaurus

In order to install the application's ipk package to the SL-series Zaurus, there ara a couple of ways to do so. You may choose the one that best fits your purpose:

1. Use "Add/Remove Software" application on the SL-series Zaurus
2. Manually install from the "terminal" application

3.3.1. Use "Add/Remove Software" application

In order to install the application from the SL-series Zaurus, you may want to use the "Add/Remove Software" application on the SL-series Zaurus. This application will look for the ipk packages stored on the device, CF memory card, and SD memory card, and gives you a listing of available applications to install.

Please see the Operation Manual that comes with the Zaurus for the details.

3.3.2. Manually install from the command line

In order to install the application from the SL-series Zaurus, you may also use the "terminal" application to manually install the ipk package. The "Add/Remove Software "application is easy enough to install, but does not provide you wih the error log when you encounter an error during installation.

When you are about to release your application ipk, or when your ipk package has problems installing, it is recommended to use this method to analyze the root case, and to resolve the problem. The following command will allow you to manually install the ipk package:

```
$ ipkg install (ipk package name)
```

The following is an example of an error log when it encounters the memory-low error during the installation:

```
$ ipkg install qpe-test_0.1-1_arm.ipk
Unpacking qpe-test...Done.
Configuring qpe-test...tar: ./opt/QtPalmtop/bin/qpe-test: input/output
error -- No space left on device
tar: Bad tar header, skipping
tar: Bad tar header, skipping
tar: Error exit delayed from previous errors
rm: /: is a directory
rm: /opt: is a directory
rm: unable to remove `/opt/QtPalmtop': Read-only file system
rm: /opt/QtPalmtop/bin: is a directory
rm: /opt/QtPalmtop/apps: is a directory
rm: /opt/QtPalmtop/apps/Applications: is a directory
rm: /opt/QtPalmtop/pics: is a directory
$
```

3.3.3. Important notice for "after installation"

The following are some of the important notices for the installation:

A. ipk package remains in the device storage

The ipk package copied to the SL-series Zaurus itself, or memory card remains even if the application is installed. You may want to delete the installed ipk package(s) when your device storage or memory becomes low in the free available space, or if you do not need the ipk package.

B. icon(s) for the applications installed on the memory card(s)

You may also install the applications to the CF memory card or SD memory card aside from the SL-series Zaurus itself. If you install applications on the memory cards, the application icon appears on the Qtopia desktop (launcher) even if the card is not inserted in the device. Note that you need to insert such memory cards to execute the application.

C. Re-installing the applications installed on the memory card(s)

You may also install the applications to the CF memory card or SD memory card aside from the SL-series Zaurus itself. If you install applications on the memory cards, the application is considered to be "already installed" on the SL-series Zaurus even if the card is not inserted in the device. If you want to re-install such application to the device itself, or to the different designation, you would first uninstall the application with the memory card inserted, and then re-install the application.

D. Using the applications installed on the memory card(s)

You may also install the applications to the CF memory card or SD memory card aside from the SL-series Zaurus itself. If you install applications on the memory cards, the application can only be used with the combination of the SL-series Zaurus you executed during the installation.

3.3.4. Uninstalling applications from the SL-series Zaurus

In order to uninstall the application from the SL-series Zaurus, you may want to use the “Add/Remove Software” application on the SL-series Zaurus. Please see the Operation Manual that comes with the Zaurus for the details.

Appendix A. SL-5600 Files and Directories

Access Permission List for SL-5600

```
/
drwxrwsr-x    2 root     root            0 May 20  2002 bin
drwxrwxr-x    2 root     root            0 Feb 10  2003 boot
drwxrwxr-x    5 root     root         7680 Jan  1 05:00 dev
lrwxrwxrwx    1 root     root            9 Feb 10  2003 etc -> /home/etc
drwxrwxr-x   12 root     root            0 Jan  1  1970 home
drwxrwxr-x    3 root     root            0 Feb 10  2003 lib
lrwxrwxrwx    1 root     root            8 Feb 10  2003 mnt -> /var/mnt
drwxrwxr-x    2 root     root            0 Feb 10  2003 opt
dr-xr-xr-x   54 root     root            0 Jan  1  1970 proc
drwxrwxr-x    5 root     root            0 Feb 10  2003 root
drwxrwxr-x    2 root     root            0 Feb 10  2003 sbin
lrwxrwxrwx    1 root     root           12 Feb 10  2003 tmp -> /dev/shm/tmp
drwxrwsr-x   10 root     root            0 Feb 10  2003 usr
lrwxrwxrwx    1 root     root           16 Feb 10  2003 var -> /home/system/var


/bin
lrwxrwxrwx    1 root     root           14 Feb 10  2003 addgroup -> /bin/tinylogin
lrwxrwxrwx    1 root     root           14 Feb 10  2003 adduser -> /bin/tinylogin
-rwxrwxr-x    1 root     root       108744 May 20  2002 ash
-rwxrwxr-x    1 root     root       561156 May 20  2002 bash
-r-xr-xr-x    1 root     root       233452 Oct 26 09:10 busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 cat -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 chgrp -> busybox
-rwxrwxr-x    1 root     root         3600 Jun 27  2002 chkmntsh
lrwxrwxrwx    1 root     root            7 Feb 10  2003 chmod -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 chown -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 cp -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 date -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 dd -> busybox
lrwxrwxrwx    1 root     root           14 Feb 10  2003 delgroup -> /bin/tinylogin
lrwxrwxrwx    1 root     root           14 Feb 10  2003 deluser -> /bin/tinylogin
lrwxrwxrwx    1 root     root            7 Feb 10  2003 df -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 dmesg -> busybox
lrwxrwxrwx    1 root     root            8 Feb 10  2003 dnsdomainname -> hostname
lrwxrwxrwx    1 root     root            7 Feb 10  2003 du -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 dumpkmap -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 echo -> busybox
lrwxrwxrwx    1 root     root            4 Feb 10  2003 egrep -> grep
lrwxrwxrwx    1 root     root            7 Feb 10  2003 false -> busybox
lrwxrwxrwx    1 root     root            7 Feb 10  2003 fdflush -> busybox
lrwxrwxrwx    1 root     root            4 Feb 10  2003 fgrep -> grep
-rwxrwxr-x    1 root     root        15952 May 20  2002 fuser
-rwxrwxr-x    1 root     root        68456 May 20  2002 grep
lrwxrwxrwx    1 root     root            4 Feb 10  2003 gunzip -> zcat
lrwxrwxrwx    1 root     root            4 Feb 10  2003 gzip -> zcat
lrwxrwxrwx    1 root     root            7 Feb 10  2003 head -> busybox
-rwxrwxr-x    1 root     root         9400 May 20  2002 hostname
```

```
lrwxrwxrwx    1 root     root           7 Feb 10  2003 ln -> busybox
lrwxrwxrwx    1 root     root          14 Feb 10  2003 login -> /bin/tinylogin
lrwxrwxrwx    1 root     root           7 Feb 10  2003 ls -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 mkdir -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 mknod -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 mktemp -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 more -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 mount -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 mt -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 mv -> busybox
-rwxrwxr-x    1 root     root       95248 May 20  2002 netstat
lrwxrwxrwx    1 root     root          16 Feb 10  2003 pidof -> ../sbin/killall5
lrwxrwxrwx    1 root     root           8 Feb 10  2003 ping -> sbusybox
-rwxrwxr-x    1 root     root       12428 Feb 10  2003 proxcfg
-rwxrwxr-x    1 root     root        6780 Feb 10  2003 proxnet
-rwxrwxr-x    1 root     root       89312 May 20  2002 ps
lrwxrwxrwx    1 root     root           7 Feb 10  2003 pwd -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 rm -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 rmdir -> busybox
-r-sr-xr-x    1 root     root        7496 Oct 28 05:18 sbusybox
-rwxrwxr-x    1 root     root       47496 May 20  2002 sed
lrwxrwxrwx    1 root     root           3 Feb 10  2003 sh -> ash
lrwxrwxrwx    1 root     root           7 Feb 10  2003 sleep -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 sort -> busybox
-rwxrwxr-x    1 root     root       25960 May 20  2002 stty
lrwxrwxrwx    1 root     root          14 Feb 10  2003 su -> /bin/tinylogin
lrwxrwxrwx    1 root     root           7 Feb 10  2003 sync -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 tar -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 tee -> busybox
-r-sr-xr-x    1 root     root       40108 Nov 16 00:52 tinylogin
lrwxrwxrwx    1 root     root           7 Feb 10  2003 true -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 umount -> busybox
lrwxrwxrwx    1 root     root           7 Feb 10  2003 uname -> busybox
-rwxrwxr-x    1 root     root          26 May 20  2002 uncompress
-rwxrwxr-x    1 root     root       13684 May 20  2002 usleep
-r-sr-xr-x    1 root     root        4700 Nov 16 00:36 utime
-rwxrwxr-x    1 root     root       55120 May 20  2002 zcat
```

/boot

```
drwxrwxr-x    2 root     root           0 Feb 10  2003 .
drwxr-xr-x   13 root     root           0 Jan  1  1970 ..
```

/dev

```
lrwxrwxrwx    1 root     root          13 Jan  1  1970 MAKEDEV -> /sbin/MAKEDEV
crw-------    1 zaurus   root      10, 134 Feb 10  2003 apm_bios
crw-rw----    1 root     root      10,   3 Feb 10  2003 atibm
crw-rw-rw-    1 root     root      14,   4 Feb 10  2003 audio
crw-rw-rw-    1 root     root      14,  20 Feb 10  2003 audio1
crw-rw-rw-    1 root     root      14,  36 Feb 10  2003 audio2
crw-rw-rw-    1 root     root      14,  52 Feb 10  2003 audio3
lrwxrwxrwx    1 root     root           5 Jan  1  1970 audioctl -> mixer
crw-r--r--    1 root     root     254,   0 Feb 10  2003 collie-fl
lrwxrwxrwx    1 root     root           4 Jan  1  1970 console -> tty1
```

```
crw-r--r--    1 root     root     205,   5 Feb 10  2003 cusa0
crw-r--r--    1 root     root     205,   6 Feb 10  2003 cusa1
crw-r--r--    1 root     root     205,   7 Feb 10  2003 cusa2
crw-------    1 zaurus   root      14,   3 Feb 10  2003 dsp
crw-rw-rw-    1 root     root      14,  19 Feb 10  2003 dsp1
crw-rw-rw-    1 root     root      14,  35 Feb 10  2003 dsp2
crw-rw-rw-    1 root     root      14,  51 Feb 10  2003 dsp3
lrwxrwxrwx    1 root     root           3 Jan  1  1970 dspdefault -> dsp
crw-------    1 zaurus   root      29,   0 Feb 10  2003 fb0
crw--w--w-    1 root     tty       29,   1 Feb 10  2003 fb0autodetect
crw--w--w-    1 root     tty       29,   0 Feb 10  2003 fb0current
crw--w--w-    1 root     tty       29,  32 Feb 10  2003 fb1
crw--w--w-    1 root     tty       29,  33 Feb 10  2003 fb1autodetect
crw--w--w-    1 root     tty       29,  32 Feb 10  2003 fb1current
lrwxrwxrwx    1 root     root          13 Jan  1  1970 fd -> /proc/self/fd
lrwxrwxrwx    1 root     root           9 Jan  1  1970 fl -> collie-fl
crw-rw-rw-    1 root     root       1,   7 Feb 10  2003 full
brw-rw----    1 root     operator   3,   0 Feb 10  2003 hda
brw-rw----    1 root     operator   3,   1 Feb 10  2003 hda1
brw-rw----    1 root     operator   3,   2 Feb 10  2003 hda2
brw-rw----    1 root     operator   3,   3 Feb 10  2003 hda3
brw-rw----    1 root     operator   3,   4 Feb 10  2003 hda4
brw-rw----    1 root     operator   3,   5 Feb 10  2003 hda5
brw-rw----    1 root     operator   3,   6 Feb 10  2003 hda6
brw-rw----    1 root     operator   3,   7 Feb 10  2003 hda7
brw-rw----    1 root     operator   3,   8 Feb 10  2003 hda8
brw-rw----    1 root     operator   3,   9 Feb 10  2003 hda9
brw-rw----    1 root     operator   3,  64 Feb 10  2003 hdb
brw-rw----    1 root     operator   3,  65 Feb 10  2003 hdb1
brw-rw----    1 root     operator   3,  66 Feb 10  2003 hdb2
brw-rw----    1 root     operator   3,  67 Feb 10  2003 hdb3
brw-rw----    1 root     operator   3,  68 Feb 10  2003 hdb4
brw-rw----    1 root     operator   3,  69 Feb 10  2003 hdb5
brw-rw----    1 root     operator   3,  70 Feb 10  2003 hdb6
brw-rw----    1 root     operator   3,  71 Feb 10  2003 hdb7
brw-rw----    1 root     operator   3,  72 Feb 10  2003 hdb8
brw-rw----    1 root     operator   3,  73 Feb 10  2003 hdb9
brw-rw----    1 root     operator  22,   0 Feb 10  2003 hdc
brw-rw----    1 root     operator  22,   1 Feb 10  2003 hdc1
brw-rw----    1 root     operator  22,   2 Feb 10  2003 hdc2
brw-rw----    1 root     operator  22,   3 Feb 10  2003 hdc3
brw-rw----    1 root     operator  22,   4 Feb 10  2003 hdc4
brw-rw----    1 root     operator  22,   5 Feb 10  2003 hdc5
brw-rw----    1 root     operator  22,   6 Feb 10  2003 hdc6
brw-rw----    1 root     operator  22,   7 Feb 10  2003 hdc7
brw-rw----    1 root     operator  22,   8 Feb 10  2003 hdc8
brw-rw----    1 root     operator  22,   9 Feb 10  2003 hdc9
prw-------    1 root     root           0 Jan  1 21:27 initctl
crw-rw----    1 root     root      10,   2 Feb 10  2003 inportbm
crw-------    1 zaurus   root     161,   0 Feb 10  2003 ircomm
crw-rw----    1 root     root      10,   4 Feb 10  2003 jbm
crw-r-----    1 root     kmem       1,   2 Feb 10  2003 kmem
crw-rw----    1 root     root      10,   0 Feb 10  2003 logibm
brw-rw----    1 root     operator   7,   0 Feb 10  2003 loop0
brw-rw----    1 root     operator   7,   1 Feb 10  2003 loop1
```

```
crw-rw----    1 root     lp         6,   1 Feb 10  2003 lp1
crw-rw----    1 root     lp         6,   2 Feb 10  2003 lp2
crw-r-----    1 root     kmem       1,   1 Feb 10  2003 mem
lrwxrwxrwx    1 root     root           6 Jan  1  1970 midi -> midi00
crw-rw-rw-    1 root     root      35,   0 Feb 10  2003 midi0
crw-rw-rw-    1 root     root      14,   2 Feb 10  2003 midi00
crw-rw-rw-    1 root     root      14,  18 Feb 10  2003 midi01
crw-rw-rw-    1 root     root      14,  34 Feb 10  2003 midi02
crw-rw-rw-    1 root     root      14,  50 Feb 10  2003 midi03
crw-rw-rw-    1 root     root      35,   1 Feb 10  2003 midi1
crw-rw-rw-    1 root     root      35,   2 Feb 10  2003 midi2
crw-rw-rw-    1 root     root      35,   3 Feb 10  2003 midi3
crw-rw-rw-    1 root     root      14,   0 Feb 10  2003 mixer
crw-rw-rw-    1 root     root      14,  16 Feb 10  2003 mixer1
crw-rw-rw-    1 root     root      14,  32 Feb 10  2003 mixer2
crw-rw-rw-    1 root     root      14,  48 Feb 10  2003 mixer3
brw-r--r--    1 root     root      60,   0 Feb 10  2003 mmcda
brw-r--r--    1 root     root      60,   1 Feb 10  2003 mmcda1
brw-r--r--    1 root     root      60,   2 Feb 10  2003 mmcda2
crw-rw-rw-    1 root     root      31,   0 Feb 10  2003 mpu401data
crw-rw-rw-    1 root     root      31,   1 Feb 10  2003 mpu401stat
drwxrwxr-x    2 root     root         384 Jan  1  1970 msys
crw-r--r--    1 root     root      90,   0 Feb 10  2003 mtd0
crw-r--r--    1 root     root      90,   2 Feb 10  2003 mtd1
crw-r--r--    1 root     root      90,   4 Feb 10  2003 mtd2
crw-r--r--    1 root     root      90,   6 Feb 10  2003 mtd3
brw-r--r--    1 root     root      31,   0 Feb 10  2003 mtdblock0
brw-r--r--    1 root     root      31,   1 Feb 10  2003 mtdblock1
brw-r--r--    1 root     root      31,   2 Feb 10  2003 mtdblock2
brw-r--r--    1 root     root      31,   3 Feb 10  2003 mtdblock3
crw-rw----    1 root     operator   9, 128 Feb 10  2003 nst0
crw-rw----    1 root     operator   9, 224 Feb 10  2003 nst0a
crw-rw----    1 root     operator   9, 160 Feb 10  2003 nst0l
crw-rw----    1 root     operator   9, 192 Feb 10  2003 nst0m
crw-rw----    1 root     operator   9, 129 Feb 10  2003 nst1
crw-rw----    1 root     operator   9, 225 Feb 10  2003 nst1a
crw-rw----    1 root     operator   9, 161 Feb 10  2003 nst1l
crw-rw----    1 root     operator   9, 193 Feb 10  2003 nst1m
crw-rw-rw-    1 root     root       1,   3 Feb 10  2003 null
crw-rw----    1 root     lp         6,   0 Feb 10  2003 par0
crw-rw----    1 root     lp         6,   1 Feb 10  2003 par1
crw-rw----    1 root     lp         6,   2 Feb 10  2003 par2
crw-r-----    1 root     kmem       1,   4 Feb 10  2003 port
crw-r--r--    1 root     root     108,   0 Feb 10  2003 ppp
crw-rw----    1 root     root      10,   1 Feb 10  2003 psaux
crw-rw-rw-    1 root     tty        5,   2 Feb 10  2003 ptmx
drwxr-xr-x    2 root     root           0 Jan  1  1970 pts
crw-rw-rw-    1 root     tty        2, 176 Jan  3 20:56 ptya0
crw-rw-rw-    1 root     tty        2, 177 Feb 10  2003 ptya1
crw-rw-rw-    1 root     tty        2, 178 Feb 10  2003 ptya2
crw-rw-rw-    1 root     tty        2, 179 Feb 10  2003 ptya3
crw-rw-rw-    1 root     tty        2, 180 Feb 10  2003 ptya4
crw-rw-rw-    1 root     tty        2, 181 Feb 10  2003 ptya5
crw-rw-rw-    1 root     tty        2, 182 Feb 10  2003 ptya6
```

```
crw-rw-rw-   1 root     tty       2, 184 Feb 10  2003 ptya8
crw-rw-rw-   1 root     tty       2, 185 Feb 10  2003 ptya9
crw-rw-rw-   1 root     tty       2, 186 Feb 10  2003 ptyaa
crw-rw-rw-   1 root     tty       2, 187 Feb 10  2003 ptyab
crw-rw-rw-   1 root     tty       2, 188 Feb 10  2003 ptyac
crw-rw-rw-   1 root     tty       2, 189 Feb 10  2003 ptyad
crw-rw-rw-   1 root     tty       2, 190 Feb 10  2003 ptyae
crw-rw-rw-   1 root     tty       2, 191 Feb 10  2003 ptyaf
lrwxrwxrwx   1 root     root           4 Jan  1  1970 ram -> ram1
brw-rw----   1 root     operator   1,   0 Feb 10  2003 ram0
brw-rw----   1 root     operator   1,   1 Feb 10  2003 ram1
brw-rw----   1 root     operator   1,   2 Feb 10  2003 ram2
brw-rw----   1 root     operator   1,   3 Feb 10  2003 ram3
cr--r--r--   1 root     root       1,   8 Feb 10  2003 random
crw-rw-rw-   1 root     root      35,  64 Feb 10  2003 rmidi0
crw-rw-rw-   1 root     root      35,  65 Feb 10  2003 rmidi1
crw-rw-rw-   1 root     root      35,  66 Feb 10  2003 rmidi2
crw-rw-rw-   1 root     root      35,  67 Feb 10  2003 rmidi3
lrwxrwxrwx   1 root     root           9 Jan  1  1970 root -> mtdblock2
crw-r--r--   1 root     root      10, 135 Feb 10  2003 rtc
brw-rw----   1 root     operator  11,   0 Feb 10  2003 scd0
brw-rw----   1 root     operator  11,   1 Feb 10  2003 scd1
crw-rw-rw-   1 root     root      10, 240 Feb 10  2003 sd_slotstat
brw-rw----   1 root     operator   8,   0 Feb 10  2003 sda
brw-rw----   1 root     operator   8,   1 Feb 10  2003 sda1
brw-rw----   1 root     operator   8,   2 Feb 10  2003 sda2
brw-rw----   1 root     operator   8,   3 Feb 10  2003 sda3
brw-rw----   1 root     operator   8,   4 Feb 10  2003 sda4
brw-rw----   1 root     operator   8,   5 Feb 10  2003 sda5
brw-rw----   1 root     operator   8,   6 Feb 10  2003 sda6
brw-rw----   1 root     operator   8,   7 Feb 10  2003 sda7
brw-rw----   1 root     operator   8,   8 Feb 10  2003 sda8
brw-rw----   1 root     operator   8,   9 Feb 10  2003 sda9
brw-rw----   1 root     operator   8,  16 Feb 10  2003 sdb
brw-rw----   1 root     operator   8,  17 Feb 10  2003 sdb1
brw-rw----   1 root     operator   8,  18 Feb 10  2003 sdb2
brw-rw----   1 root     operator   8,  19 Feb 10  2003 sdb3
brw-rw----   1 root     operator   8,  20 Feb 10  2003 sdb4
brw-rw----   1 root     operator   8,  21 Feb 10  2003 sdb5
brw-rw----   1 root     operator   8,  22 Feb 10  2003 sdb6
brw-rw----   1 root     operator   8,  23 Feb 10  2003 sdb7
brw-rw----   1 root     operator   8,  24 Feb 10  2003 sdb8
brw-rw----   1 root     operator   8,  25 Feb 10  2003 sdb9
brw-rw----   1 root     operator   8,  32 Feb 10  2003 sdc
brw-rw----   1 root     operator   8,  33 Feb 10  2003 sdc1
brw-rw----   1 root     operator   8,  34 Feb 10  2003 sdc2
brw-rw----   1 root     operator   8,  35 Feb 10  2003 sdc3
brw-rw----   1 root     operator   8,  36 Feb 10  2003 sdc4
brw-rw----   1 root     operator   8,  37 Feb 10  2003 sdc5
brw-rw----   1 root     operator   8,  38 Feb 10  2003 sdc6
brw-rw----   1 root     operator   8,  39 Feb 10  2003 sdc7
brw-rw----   1 root     operator   8,  40 Feb 10  2003 sdc8
brw-rw----   1 root     operator   8,  41 Feb 10  2003 sdc9
crw-rw-rw-   1 root     root      14,   1 Feb 10  2003 sequencer
```

```
crw-rw-rw-   1 root     root      10, 211 Feb 10  2003 sharp_buz
crw-rw-rw-   1 root     root      10, 214 Feb 10  2003 sharp_kbdctl
crw-rw-rw-   1 root     root      10, 210 Feb 10  2003 sharp_led
crw-rw-rw-   1 root     root      11,   0 Feb 10  2003 sharp_ts
drwxrwxrwt   4 root     root           0 Jan  1  1970 shm
crw-rw-rw-   1 root     root      35, 128 Feb 10  2003 smtpe0
crw-rw-rw-   1 root     root      35, 129 Feb 10  2003 smtpe1
crw-rw-rw-   1 root     root      35, 130 Feb 10  2003 smtpe2
crw-rw-rw-   1 root     root      35, 131 Feb 10  2003 smtpe3
crw-rw-rw-   1 root     root      14,   6 Feb 10  2003 sndstat
crw-rw----   1 root     operator   9,   0 Feb 10  2003 st0
crw-rw----   1 root     operator   9,  96 Feb 10  2003 st0a
crw-rw----   1 root     operator   9,  32 Feb 10  2003 st0l
crw-rw----   1 root     operator   9,  64 Feb 10  2003 st0m
crw-rw----   1 root     operator   9,   1 Feb 10  2003 st1
crw-rw----   1 root     operator   9,  97 Feb 10  2003 st1a
crw-rw----   1 root     operator   9,  33 Feb 10  2003 st1l
crw-rw----   1 root     operator   9,  65 Feb 10  2003 st1m
lrwxrwxrwx   1 root     root           4 Jan  1  1970 stderr -> fd/2
lrwxrwxrwx   1 root     root           4 Jan  1  1970 stdin -> fd/0
lrwxrwxrwx   1 root     root           4 Jan  1  1970 stdout -> fd/1
lrwxrwxrwx   1 root     root           8 Jan  1  1970 ts -> sharp_ts
crw-rw-rw-   1 root     tty       5,   0 Feb 10  2003 tty
crw--w--w-   1 root     tty       4,   0 Jan  1 05:00 tty0
crw-rw-rw-   1 root     tty       4,   1 Jan  1 21:27 tty1
crw-rw-rw-   1 root     tty       4,   2 Feb 10  2003 tty2
crw-rw-rw-   1 root     tty       4,   3 Feb 10  2003 tty3
crw-rw-rw-   1 root     tty       4,   4 Feb 10  2003 tty4
crw-rw-rw-   1 root     tty       4,   5 Feb 10  2003 tty5
crw-rw-rw-   1 root     tty       4,   6 Feb 10  2003 tty6
crw-rw-rw-   1 root     tty       4,   7 Feb 10  2003 tty7
crw-rw-rw-   1 root     tty       4,   8 Feb 10  2003 tty8
crw-rw-rw-   1 root     tty       4,   9 Feb 10  2003 tty9
crw-------   1 root     root      4,  64 Jan  3 21:07 ttyS0
crw-------   1 zaurus   root      4,  65 Feb 10  2003 ttyS1
crw-------   1 zaurus   root      4,  66 Feb 10  2003 ttyS2
crw-------   1 zaurus   root      4,  67 Feb 10  2003 ttyS3
crw-r--r--   1 root     root    204,   5 Feb 10  2003 ttySA0
crw-r--r--   1 root     root    204,   6 Feb 10  2003 ttySA1
crw-r--r--   1 root     root    204,   7 Feb 10  2003 ttySA2
crw-r--r--   1 root     root    188,   0 Feb 10  2003 ttyUSB0
crw-r--r--   1 root     root    188,   1 Feb 10  2003 ttyUSB1
crw-rw-rw-   1 root     tty       3, 176 Jan  3 20:56 ttya0
crw-rw-rw-   1 root     tty       3, 177 Feb 10  2003 ttya1
crw-rw-rw-   1 root     tty       3, 178 Feb 10  2003 ttya2
crw-rw-rw-   1 root     tty       3, 179 Feb 10  2003 ttya3
crw-rw-rw-   1 root     tty       3, 180 Feb 10  2003 ttya4
crw-rw-rw-   1 root     tty       3, 181 Feb 10  2003 ttya5
crw-rw-rw-   1 root     tty       3, 182 Feb 10  2003 ttya6
crw-rw-rw-   1 root     tty       3, 183 Feb 10  2003 ttya7
crw-rw-rw-   1 root     tty       3, 184 Feb 10  2003 ttya8
crw-rw-rw-   1 root     tty       3, 185 Feb 10  2003 ttya9
crw-rw-rw-   1 root     tty       3, 186 Feb 10  2003 ttyaa
crw-rw-rw-   1 root     tty       3, 187 Feb 10  2003 ttyab
```

```
crw-rw-rw-    1 root     tty       3, 189 Feb 10  2003 ttyad
crw-rw-rw-    1 root     tty       3, 190 Feb 10  2003 ttyae
crw-rw-rw-    1 root     tty       3, 191 Feb 10  2003 ttyaf
cr--r--r--    1 root     root      1,   9 Feb 10  2003 urandom
crw-rw----    1 root     root     10,  32 Feb 10  2003 usbmouse
crw-rw-rw-    1 root     root      1,   5 Feb 10  2003 zero
```

/etc -> /home/etc

```
drwxrwsr-x   10 root     root          0 Jan  2 13:53 .
drwxrwxr-x   12 root     root          0 Jan  1  1970 ..
-rw-rw-r--    1 root     root          7 Aug 30 05:06 HOSTNAME
-r--------    1 root     root         29 Oct 15 04:15 busybox.conf
-rw-rw-r--    1 root     root        264 Dec  2 02:23 fstab
-rw-rw-r--    1 root     root        254 Sep 27 09:09 group
-rw-rw-r--    1 root     root       1106 Aug 30 05:06 hosts
drwxrwsr-x    2 root     root          0 Jan  1 05:00 hotplug
-rw-rw-r--    1 root     root       3368 May 20  2002 inetd.conf
-rw-rw-r--    1 root     root       1003 Sep 28 00:57 inittab
drwxrwsr-x    2 root     root          0 Jan  1  1970 intent
-rw-------    1 root     root         60 Jan  1 05:00 ioctl.save
lrwxrwxrwx    1 root     root         28 Jan  1  1970 ipkg.conf ->
/opt/QtPalmtop/etc/ipkg.conf
-rw-rw-r--    1 root     root        457 Nov 27 00:00 irda.conf
-rw-rw-r--    1 root     root         42 May 20  2002 issue
-rw-rw-r--    1 root     root         58 May 20  2002 issue.net
-rw-r--r--    1 root     root       4745 Jan  2 13:53 ld.so.cache
-rw-rw-r--    1 root     root         35 Sep 12 05:22 ld.so.conf
lrwxrwxrwx    1 root     root         12 Jan  1  1970 mtab -> /proc/mounts
-rw-rw-r--    1 root     root        132 May 20  2002 network-device
-rw-rw-r--    1 root     root        239 May 20  2002 nsswitch.conf
-rw-rw-r--    1 root     root        444 Sep 27 09:09 passwd
drwxrwxr-x    3 root     root          0 Jan  1  1970 pcmcia
-rw-r--r--    1 zaurus   qpe          39 Jan  1 05:05 pointercal
drwxrwxr-x    3 root     root          0 Jan  1  1970 ppp
-rw-rw-r--    1 root     root        894 May 20  2002 protocols
drwxrwsr-x   10 root     root          0 Jan  1  1970 rc.d
-rw-rw-r--    1 root     root         99 Jun 28  2002 resolv.conf
-rwxrwxr-x    1 root     root       2649 Oct 29 11:22 sdcontrol
-rw-rw-r--    1 root     root        114 May 20  2002 securetty
-rw-rw-r--    1 root     root       8377 May 20  2002 services
-rw-rw-r--    1 root     root        320 Sep 27 09:09 shadow
drwxrwsr-x    2 root     root          0 Jan  1  1970 sync
drwxrwsr-x    2 root     root          0 Jan  1  1970 sysconfig
-rw-rw-r--    1 root     root        903 May 20  2002 syslog.conf
-rw-rw-r--    1 root     root     288451 May 20  2002 termcap
-r--------    1 root     root         50 Oct  1 12:07 tinylogin.conf
-rwxrwxr-x    1 root     root        682 Jun 12  2002 usbcontrol
drwxrwsr-x    2 root     root          0 Jan  1  1970 wlan
```

/home

```
drwxrwxr-x   12 root    root          0 Jan  1  1970 .
drwxr-xr-x   13 root    root          0 Jan  1  1970 ..
drwxr-x---   16 root    qpe           0 Jan  2 13:53 QtPalmtop
drwxrwsr-x   10 root    root          0 Jan  2 13:53 etc
drwxrwxr-x    5 root    root          0 Jan  1 05:00 root
drwxrwxr-x    2 root    root          0 Jan  3 21:05 samba
drwxrwxr-x    7 root    root          0 Jan  1  1970 sharp
drwxrwxr-x    3 root    root          0 Jan  1  1970 system
drwxrwxrwt    3 root    root          0 Jan  2 13:53 tmp
drwxrwxr-x    2 root    root          0 Feb 10  2003 userdata
drwx------    6 zaurus  qpe           0 Jan  1  1970 zaurus
```

/lib

```
-rwxrwxr-x    1 root    root      105908 May 20  2002 ld-2.2.2.so
lrwxrwxrwx    1 root    root          11 Feb 10  2003 ld-linux.so.2 -> ld-2.2.2.so
lrwxrwxrwx    1 root    root          11 Feb 10  2003 ld.so.2 -> ld-2.2.2.so
-rwxrwxr-x    1 root    root     1152532 Jan 15  2003 libc-2.2.2.so
lrwxrwxrwx    1 root    root          13 Feb 10  2003 libc.so.6 -> libc-2.2.2.so
lrwxrwxrwx    1 root    root          17 Feb 10  2003 libcom_err.so.2 -> libcom_err.so.2.0
-rwxrwxr-x    1 root    root        6184 May 20  2002 libcom_err.so.2.0
-rwxrwxr-x    1 root    root       22624 May 20  2002 libcrypt-2.2.2.so
lrwxrwxrwx    1 root    root          17 Feb 10  2003 libcrypt.so.1 -> libcrypt-2.2.2.so
-rwxrwxr-x    1 root    root       10176 May 20  2002 libdl-2.2.2.so
lrwxrwxrwx    1 root    root          14 Feb 10  2003 libdl.so.2 -> libdl-2.2.2.so
lrwxrwxrwx    1 root    root          13 Feb 10  2003 libe2p.so.2 -> libe2p.so.2.3
-rwxrwxr-x    1 root    root       15052 May 20  2002 libe2p.so.2.3
lrwxrwxrwx    1 root    root          16 Feb 10  2003 libext2fs.so.2 -> libext2fs.so.2.4
-rwxrwxr-x    1 root    root       74720 May 20  2002 libext2fs.so.2.4
-rwxrwxr-x    1 root    root       18832 Feb 10  2003 libiw.so.25
-rwxrwxr-x    1 root    root      163740 May 20  2002 libm-2.2.2.so
lrwxrwxrwx    1 root    root          13 Feb 10  2003 libm.so.6 -> libm-2.2.2.so
lrwxrwxrwx    1 root    root          17 Feb 10  2003 libncurses.so.4 -> libncurses.so.4.2
-rw-rw-r--    1 root    root      358620 May 20  2002 libncurses.so.4.2
-rwxrwxr-x    1 root    root       75332 May 20  2002 libnsl-2.2.2.so
lrwxrwxrwx    1 root    root          15 Feb 10  2003 libnsl.so.1 -> libnsl-2.2.2.so
-rwxrwxr-x    1 root    root       44300 May 20  2002 libnss_compat-2.2.2.so
lrwxrwxrwx    1 root    root          22 Feb 10  2003 libnss_compat.so.2 ->
libnss_compat-2.2.2.so
-rwxrwxr-x    1 root    root       12040 May 20  2002 libnss_dns-2.2.2.so
lrwxrwxrwx    1 root    root          19 Feb 10  2003 libnss_dns.so.2 -> libnss_dns-2.2.2.so
-rwxrwxr-x    1 root    root       40916 May 20  2002 libnss_files-2.2.2.so
lrwxrwxrwx    1 root    root          21 Feb 10  2003 libnss_files.so.2 ->
libnss_files-2.2.2.so
-rwxrwxr-x    1 root    root       41412 May 20  2002 libnss_nis-2.2.2.so
lrwxrwxrwx    1 root    root          19 Feb 10  2003 libnss_nis.so.2 -> libnss_nis-2.2.2.so
-rwxrwxr-x    1 root    root       90520 May 20  2002 libpthread-0.9.so
lrwxrwxrwx    1 root    root          17 Feb 10  2003 libpthread.so.0 -> libpthread-0.9.so
-rwxrwxr-x    1 root    root       60384 Jan 15  2003 libresolv-2.2.2.so
lrwxrwxrwx    1 root    root          18 Feb 10  2003 libresolv.so.2 -> libresolv-2.2.2.so
lrwxrwxrwx    1 root    root          12 Feb 10  2003 libss.so.2 -> libss.so.2.0
-rwxrwxr-x    1 root    root       16496 May 20  2002 libss.so.2.0
-rwxrwxr-x    1 root    root        7940 May 20  2002 libutil-2.2.2.so
```

```
lrwxrwxrwx    1 root     root           14 Feb 10  2003 libuuid.so.1 -> libuuid.so.1.2
-rwxrwxr-x    1 root     root        12724 May 20  2002 libuuid.so.1.2
lrwxrwxrwx    1 root     root           18 Feb 10  2003 modules -> /home/root/modules
drwxrwxr-x    3 root     root            0 Feb 10  2003 modules.rom
```

/mnt -> /var/mnt

```
lrwxrwxrwx    1 root     root           17 Jan  1  1970 card -> /usr/mnt.rom/card
lrwxrwxrwx    1 root     root           15 Jan  1  1970 cf -> /usr/mnt.rom/cf
drwxrwxr-x    2 root     root            0 Feb 10  2003 ide
lrwxrwxrwx    1 root     root           16 Jan  1  1970 net -> /usr/mnt.rom/net
```

/opt

```
lrwxrwxrwx    1 root     root           15 Feb 10  2003 QtPalmtop -> /home/QtPalmtop
lrwxrwxrwx    1 root     root           15 Feb 10  2003 Qtopia -> /home/QtPalmtop
```

/proc

```
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 1
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 10
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 103
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 1050
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 1051
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 11
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 12
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 13
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 1327
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 134
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 15
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 153
dr-xr-xr-x    3 bin      root            0 Jan  3 21:08 163
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 174
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 2
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 204
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 205
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 206
dr-xr-xr-x    3 root     qpe             0 Jan  3 21:08 226
dr-xr-xr-x    3 root     qpe             0 Jan  3 21:08 227
dr-xr-xr-x    3 root     qpe             0 Jan  3 21:08 228
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 244
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 245
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 246
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 247
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 248
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 249
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 250
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 251
dr-xr-xr-x    3 zaurus   qpe             0 Jan  3 21:08 252
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 275
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 3
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 4
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 44
dr-xr-xr-x    3 root     root            0 Jan  3 21:08 5
```

```
dr-xr-xr-x    3 root     root           0 Jan  3 21:08 7
dr-xr-xr-x    3 root     root           0 Jan  3 21:08 8
dr-xr-xr-x    3 root     root           0 Jan  3 21:08 9
dr-xr-xr-x    3 zaurus   qpe            0 Jan  3 21:08 996
dr-xr-xr-x    3 zaurus   qpe            0 Jan  3 21:08 997
dr-xr-xr-x    3 root     root           0 Jan  3 21:08 999
-r--r--r--    1 root     root           0 Jan  3 21:08 apm
dr-xr-xr-x    3 root     root           0 Jan  3 21:08 bus
-r--r--r--    1 root     root           0 Jan  3 21:08 cmdline
dr-xr-xr-x    2 root     root           0 Jan  3 21:08 cpu
-r--r--r--    1 root     root           0 Jan  3 21:08 cpuinfo
dr-xr-xr-x    2 root     root           0 Jan  3 21:08 deviceinfo
-r--r--r--    1 root     root           0 Jan  3 21:08 devices
dr-xr-xr-x    5 root     root           0 Jan  1 05:00 driver
-r--r--r--    1 root     root           0 Jan  3 21:08 execdomains
-r--r--r--    1 root     root           0 Jan  3 21:08 fb
-r--r--r--    1 root     root           0 Jan  3 21:08 filesystems
dr-xr-xr-x    3 root     root           0 Jan  3 21:08 fs
dr-xr-xr-x    2 root     root           0 Jan  3 21:08 hermes
dr-xr-xr-x    3 root     root           0 Jan  3 21:08 ide
-r--r--r--    1 root     root           0 Jan  3 21:08 interrupts
-r--r--r--    1 root     root           0 Jan  3 21:08 iomem
-r--r--r--    1 root     root           0 Jan  3 21:08 ioports
-r--------    1 root     root    33558528 Jan  3 21:08 kcore
-r--------    1 root     root           0 Jan  3 21:08 kmsg
-r--r--r--    1 root     root           0 Jan  3 21:08 ksyms
-r--r--r--    1 root     root           0 Jan  3 21:08 loadavg
-r--r--r--    1 root     root           0 Jan  1 05:00 lock_fcs
-r--r--r--    1 root     root           0 Jan  3 21:08 locks
-r--r--r--    1 root     root           0 Jan  3 21:08 meminfo
-r--r--r--    1 root     root           0 Jan  3 21:08 misc
-r--r--r--    1 root     root           0 Jan  3 21:08 modules
-r--r--r--    1 root     root           0 Jan  3 21:08 mounts
-r--r--r--    1 root     root           0 Jan  3 21:08 mtd
dr-xr-xr-x    4 root     root           0 Jan  3 21:08 net
-r--r--r--    1 root     root           0 Jan  3 21:08 partitions
-r--r--r--    1 root     root           0 Jan  3 21:08 power_mode
lrwxrwxrwx    1 root     root          64 Jan  3 20:50 self -> 1327
-rw-r--r--    1 root     root           0 Jan  3 21:08 slabinfo
-r--r--r--    1 root     root           0 Jan  3 21:08 stat
-r--r--r--    1 root     root           0 Jan  3 21:08 swaps
dr-xr-xr-x   11 root     root           0 Jan  3 21:08 sys
dr-xr-xr-x    2 root     root           0 Jan  3 21:08 sysvipc
dr-xr-xr-x    4 root     root           0 Jan  3 21:08 tty
-r--r--r--    1 root     root           0 Jan  3 21:08 uptime
-r--r--r--    1 root     root           0 Jan  3 21:08 usb-condition
-r--r--r--    1 root     root           0 Jan  3 21:08 usb-devices
-r--r--r--    1 root     root           0 Jan  3 21:08 usb-functions
-r--r--r--    1 root     root           0 Jan  3 21:08 usb-monitor
-r--r--r--    1 root     root           0 Jan  3 21:08 usbd
-r--r--r--    1 root     root           0 Jan  3 21:08 version
```

/root

```
-rw-rw-r--    1 root     root      133120 Feb 10  2003 .dev_default.tar
-rw-rw-r--    1 root     root     1597440 Feb 10  2003 .home_default.tar
-rw-rw-r--    1 root     root         313 Aug 30 05:06 .profile
-rw-rw-r--    1 root     root       10240 Feb 10  2003 .var_default.tar
drwxrwxr-x    2 root     root           0 Feb 10  2003 bin
drwxrwsr-x    4 root     root           0 May 20  2002 etc
drwxrwxr-x    2 root     root           0 Feb 10  2003 samba
```

/sbin

```
-rwxrwxr-x    1 root     root       36956 May 20  2002 MAKEDEV
-rwxrwxr-x    1 root     root       44876 Feb 10  2003 arp
-rwxrwxr-x    1 root     root       14508 May 20  2002 badblocks
-rwsrwxr-x    1 root     root       12056 Nov 12 11:48 cardctl
-rwxrwxr-x    1 root     root       38892 May 20  2002 cardmgr
lrwxrwxrwx    1 root     root          12 Feb 10  2003 chroot -> /bin/busybox
-rwxrwxr-x    1 root     root        3088 May 20  2002 consoletype
-rwxrwxr-x    1 root     root       40016 May 20  2002 debugfs
-rwxrwxr-x    1 root     root       58644 May 20  2002 depmod
-rwxrwxr-x    1 root     root       42460 Aug 11 05:12 dhcpcd
lrwxrwxrwx    1 root     root           9 Feb 10  2003 dosfsck -> fsck.vfat
-rwxrwxr-x    1 root     root       19300 May 20  2002 dump_cis
-rwxrwxr-x    1 root     root        7160 May 20  2002 dumpe2fs
lrwxrwxrwx    1 root     root           9 Feb 10  2003 e2fsck -> fsck.ext3
-rwxrwxr-x    1 root     root        4260 May 20  2002 e2label
-rwxrwxr-x    1 root     root        5260 Aug 29 08:28 eraseall
lrwxrwxrwx    1 root     root          12 Feb 10  2003 freeramdisk -> /bin/busybox
-rwxrwxr-x    1 root     root       14264 May 20  2002 fsck
lrwxrwxrwx    1 root     root           9 Feb 10  2003 fsck.ext2 -> fsck.ext3
-rwxrwxr-x    1 root     root      484072 May 20  2002 fsck.ext3
lrwxrwxrwx    1 root     root          12 Feb 10  2003 fsck.minix -> /bin/busybox
lrwxrwxrwx    1 root     root           9 Feb 10  2003 fsck.msdos -> fsck.vfat
-rwxrwxr-x    1 root     root       45216 Feb 10  2003 fsck.vfat
-rwxrwxr-x    1 root     root        5700 May 20  2002 ftl_check
-rwxrwxr-x    1 root     root        7336 May 20  2002 ftl_format
-rwxrwxr-x    1 root     root       36848 May 20  2002 genksyms
-rwxrwxr-x    1 root     root        4672 May 20  2002 getkey
lrwxrwxrwx    1 root     root          14 Feb 10  2003 getty -> /bin/tinylogin
lrwxrwxrwx    1 root     root           6 Feb 10  2003 halt -> reboot
-rwxrwxr-x    1 root     root        1836 May 20  2002 hotplug
-rwxrwxr-x    1 root     root       30368 May 20  2002 hwclock
-rwxrwxr-x    1 root     root        4356 May 20  2002 ide_info
-rwxrwxr-x    1 root     root       59816 Feb 10  2003 ifconfig
-rwxrwxr-x    1 root     root        3952 May 20  2002 ifport
-rwxrwxr-x    1 root     root        4840 May 20  2002 ifuser
lrwxrwxrwx    1 root     root           7 Feb 10  2003 init -> telinit
-rwxrwxr-x    1 root     root       27648 May 20  2002 initlog
-rwxrwxr-x    1 root     root       98152 May 20  2002 insmod
-rwxrwxr-x    1 root     root         359 May 20  2002 insmod_ksymoops_clean
-rwxrwxr-x    1 root     root       11064 Feb 10  2003 ipmaddr
-rwxrwxr-x    1 root     root       15404 Feb 10  2003 iptunnel
-rwxrwxr-x    1 root     root       18912 Sep 18 07:01 iwconfig
-rwxrwxr-x    1 root     root        5712 Sep 18 07:01 iwevent
```

```
-rwxrwxr-x    1 root     root        13088 Sep 18 07:01 iwlist
-rwxrwxr-x    1 root     root        11452 Sep 18 07:01 iwpriv
-rwxrwxr-x    1 root     root         6504 Sep 18 07:01 iwspy
lrwxrwxrwx    1 root     root            6 Feb 10  2003 kallsyms -> insmod
-rwxrwxr-x    1 root     root          451 May 20  2002 kernelversion
-rwxrwxr-x    1 root     root         8664 May 20  2002 killall5
-rwxrwxr-x    1 root     root        26164 May 20  2002 klogd
lrwxrwxrwx    1 root     root            6 Feb 10  2003 ksyms -> insmod
-rwxrwxr-x    1 root     root        10232 Nov  6 02:32 launch
-rwxrwxr-x    1 root     root       404748 Feb 10  2003 ldconfig
lrwxrwxrwx    1 root     root           12 Feb 10  2003 loadkmap -> /bin/busybox
lrwxrwxrwx    1 root     root            6 Feb 10  2003 lsmod -> insmod
lrwxrwxrwx    1 root     root           12 Feb 10  2003 makedevs -> /bin/busybox
-rwxrwxr-x    1 root     root        10480 Feb 10  2003 mii-tool
lrwxrwxrwx    1 root     root            9 Feb 10  2003 mkdosfs -> mkfs.vfat
lrwxrwxrwx    1 root     root            9 Feb 10  2003 mke2fs -> mkfs.ext2
-rwxrwxr-x    1 root     root        20100 May 20  2002 mkfs.ext2
lrwxrwxrwx    1 root     root           12 Feb 10  2003 mkfs.minix -> /bin/busybox
lrwxrwxrwx    1 root     root            9 Feb 10  2003 mkfs.msdos -> mkfs.vfat
-rwxrwxr-x    1 root     root        25004 Feb 10  2003 mkfs.vfat
lrwxrwxrwx    1 root     root           12 Feb 10  2003 mkswap -> /bin/busybox
-rwxrwxr-x    1 root     root        44876 May 20  2002 modinfo
lrwxrwxrwx    1 root     root            6 Feb 10  2003 modprobe -> insmod
-rwxrwxr-x    1 root     root         7676 Feb 10  2003 nameif
-rwxrwxr-x    1 root     root         3492 Feb 10  2003 nwepgen
-rwxrwxr-x    1 root     root         9376 Feb 10  2003 oncheck
-rwxrwxr-x    1 root     root        30352 May 20  2002 pack_cis
-rwxrwxr-x    1 root     root         4831 May 20  2002 pcinitrd
-rwxrwxr-x    1 root     root         4836 Feb 10  2003 plipconfig
lrwxrwxrwx    1 root     root            6 Feb 10  2003 poweroff -> reboot
-rwxrwxr-x    1 root     root         6444 May 20  2002 probe
-rwxrwxr-x    1 root     root           56 May 20  2002 qt
-rwxrwxr-x    1 root     root        21288 Feb 10  2003 rarp
-rwxrwxr-x    1 root     root         7628 May 20  2002 reboot
-rwxrwxr-x    1 root     root        21192 May 20  2002 resize2fs
lrwxrwxrwx    1 root     root            6 Feb 10  2003 rmmod -> insmod
-rwxrwxr-x    1 root     root        45832 Feb 10  2003 route
-rwxrwxr-x    1 root     root         3044 May 20  2002 runlevel
-rwxrwxr-x    1 root     root           46 May 20  2002 scshotcf
-rwxrwxr-x    1 root     root          532 Nov  2 12:37 scshotram
-rwxrwxr-x    1 root     root           48 May 20  2002 scshotsd
-rwxrwxr-x    1 root     root         4704 May 20  2002 scsi_info
-rwxrwxr-x    1 root     root         6316 May 20  2002 sdmgr
-rwxrwxr-x    1 root     root        14304 May 20  2002 setserial
-rwxrwxr-x    1 root     root         2688 Feb 10  2003 shsync
-rwxrwxr-x    1 root     root        14732 May 20  2002 shutdown
-rwxrwxr-x    1 root     root        26436 Feb 10  2003 slattach
-rwxrwxr-x    1 root     root        10156 Nov 27 06:08 sltime
-rwxrwxr-x    1 root     root        11196 May 20  2002 spm
-rwxrwxr-x    1 root     root         8644 May 20  2002 sulogin
-rwxrwxr-x    1 root     root         4476 May 20  2002 survive
lrwxrwxrwx    1 root     root           12 Feb 10  2003 swapoff -> /bin/busybox
lrwxrwxrwx    1 root     root           12 Feb 10  2003 swapon -> /bin/busybox
-rwxrwxr-x    1 root     root        32268 May 20  2002 syslogd
```

```
-rwxrwxr-x    1 root     root        10064 May 20  2002 tune2fs
lrwxrwxrwx    1 root     root           12 Feb 10  2003 update -> /bin/busybox
-rwxrwxr-x    1 root     root        61536 Feb 10  2003 wlancfg
-rwxrwxr-x    1 root     root        61176 Feb 10  2003 wlanctl-ng
-rwxrwxr-x    1 root     root        59524 Feb 10  2003 wland
-rwxrwxr-x    1 root     root         5152 Feb  5  2003 writerominfo
```

/tmp -> /dev/shm/tmp

```
srw-------    1 zaurus   qpe            0 Jan  1 05:00 .quickexec
drwx------    2 root     qpe            0 Jan  1 05:00 qtembedded-root
drwx------    2 zaurus   qpe            0 Jan  1 05:00 qtembedded-zaurus
```

/usr

```
drwxr-x---   15 root     qpe            0 Feb 10  2003 QtPalmtop.rom
lrwxrwxrwx    1 root     root          18 Feb 10  2003 bin -> /home/root/usr/bin
drwxrwxr-x    2 root     root           0 May 20  2002 bin.rom
lrwxrwxrwx    1 root     root          18 Feb 10  2003 etc -> /home/root/usr/etc
lrwxrwxrwx    1 root     root          18 Feb 10  2003 lib -> /home/root/usr/lib
drwxrwxr-x    3 root     root           0 Feb 10  2003 lib.rom
lrwxrwxrwx    1 root     root          20 Feb 10  2003 local -> /home/root/usr/local
drwxrwxr-x    3 root     root           0 Feb 10  2003 local.rom
drwxrwxr-x    5 root     root           0 Feb 10  2003 mnt.rom
lrwxrwxrwx    1 root     root          19 Feb 10  2003 sbin -> /home/root/usr/sbin
drwxrwxr-x    2 root     root           0 Feb 10  2003 sbin.rom
lrwxrwxrwx    1 root     root          20 Feb 10  2003 share -> /home/root/usr/share
drwxrwxr-x    5 root     root           0 Feb 10  2003 share.rom
lrwxrwxrwx    1 root     root          11 Feb 10  2003 sharp -> /home/sharp
drwxrwxr-x    7 root     root           0 Feb 10  2003 sharp.rom
```

/var -> /home/system/var

```
drwxrwxr-x    2 root     root           0 Feb 10  2003 home
drwxrwxr-x    3 root     root           0 Jan  1  1970 lib
drwxrwxr-x    4 root     uucp           0 Jan  1  1970 lock
drwxrwxr-x    2 root     root           0 Jan  1  1970 log
drwxrwxr-x    3 root     root           0 Jan  1  1970 mnt
lrwxrwxrwx    1 root     root          12 Jan  1  1970 run -> /dev/shm/run
drwxrwxr-x    2 root     root           0 Feb 10  2003 smb
drwxrwxr-x    3 root     root           0 Jan  1 05:00 spool
```