

Linux on an *iPAQ* PDA

Thomas Zwinger and Leif Laaksonen

Thomas.Zwinger@csc.fi and Leif.Laaksonen@csc.fi



Figure 1: The *iPAQ* PDA showing the CSC homepage in the web-browser *di11o*.

Whereas Linux has gained popularity as an alternative to commercial operating systems (OS) on PCs as well as workstations or even clusters, the world of handhelds or PDAs (Personal Digital Assistant) is far away from being conquered by open source OS. Nevertheless, there are Linux kernels and even whole distributions available for certain PDA platforms.

This article shall give a short report on the installation process of Linux on a *Compaq iPAQ* PDA. In terms of computer applications the authors of this article are to be called “users” rather than “hackers”. Thus, the explanations will be kept on a very basic level focused on the installation procedure and circumvent technical details. The fact, that a Linux user is able to do this installation, proves that this procedure is less difficult as it in first instance might appear.

This text is meant to be an installation report. By no means the reader should take it as a complete instruction for installation. For this purpose the interested reader might refer to literature, that is found on the Internet (e.g., [1]).

The reader of this article should also keep in mind that addresses in the Internet and names of downloads can change. The references were valid at the time this article had been written. Anytime in the future it easily might be the case, that some of these sites have changed their URLs and binaries for download have been placed somewhere else. In terms of the installation process some of the information also might have become outdated.

The *iPAQ* Handheld and Linux

The installation was done on an *iPAQ* H3630. It is a model of *Compaq*'s earlier *iPAQ* H3600 series (from the year 2000) based on the *StrongARM* SA-1110 RISC processor running at 206 MHz. Our model was shipped with 32 Mbytes of RAM and 16 Mbytes of ROM memory. It further is equipped with a 320 × 240 pixel colour (4096 colours) TFT display integrated in a touch screen, an infrared port and a serial cradle interface. Detailed specifications of this platform are to be found at [2].

Compaq delivered our model with *Microsoft Pocket PC 2000* OS installed. This *Windows* version was to be replaced by a Linux able to be run on the

StrongARM processor provided by the ARM Linux Project. ARM Linux [3] is a port of the successful Linux Operating System to ARM processor based machines.

Once the kernel has been set up, a distribution of applications has to be installed. There are in principle two distributions with entirely different philosophies available:

1. Linux distribution by the *Familiar Project*: This distribution aims to provide a setup of applications that entirely fit into the memory of the PDA. Thus the philosophy of *Familiar* is to provide packages that have a smaller demand for storage capacity. The basic features of the distributions are listed on the homepage of the *Familiar Project* [4]:

- Entirely based on XFree86's/keithp's Tiny-X server, which includes the latest RENDER extension
- OpenSSH's ssh and sshd included by default
- Binary and Library compatible with *Debian*'s ARM distribution [5]
- Full package support based on `ipkg`*

2. Linux distribution by the *Intimate Project*: Based on the *Familiar Project*, *Intimate* is a fully blown *Debian* based Linux distribution for the *Compaq iPAQ*. From the homepage of this project [6] one can read:

... it won't fit in the 16MB Flash but for the lucky few with microdrives then this is the way ahead. The minimum requirements are currently around 140MB of storage for the base image.

The end-configuration of the *iPAQ* PDA tested for this article is the Linux distribution *Familiar* v0.5.1 installed on a system running the ARM Linux kernel version 2.4.16-rmk1.

Prerequisites

The whole installation process follows the instructions to be found from links on the homepage for open source software for use on handhelds [7]. The problem is not, that there would not be enough information on that issue, rather than to find the correct links that guide the user through the installation process.

The needed list of hard- and software for the whole procedure reads as follows:

- *iPAQ* initially running *Pocket PC*
- The serial cradle for the *iPAQ* connected to a PC
- A PC running *ActiveSync* (needs a connected *Windows* desktop) software for installing the new boot software on the *iPAQ* and backing up *Pocket PC*
- A PC running a terminal emulator program, such as *Hyperterm* on *Windows* or *minicom* in Linux (for loading the kernel image)
- A PC connected to the Internet, being able to do a IP forwarding via the PPP-protocol over the serial interface

The steps of the installation are:

1. Backup of old *Pocket PC* image and installation of a bootloader, that is able to boot external images via the serial connection
2. Loading the bootloader, flashing the filesystem and booting a basic Linux (Warning: if this fails, one can screw up the PDA)
3. Configuration for access to Internet
4. Download of distribution via PPP connection from Internet

Although it has been reported that some people did the whole installation inclusively the transfer of the bootloader from a Linux PC, it is recommended at least to do the first steps (backup of *Pocket PC* and installing the bootloader) from a PC running a *Windows* OS.

On Backups, Bootloader and Bricks

The steps of this section are explained on the *Bootloader Installation Instructions* page [8]. It is about the uploading of a program for pre-installation preparations on the *iPAQ*. By means of this program, the old *Pocket PC* system is backed up and a new bootloader, with which one can connect via the serial interface to the handheld during boot-time, is installed. For that purpose one has to download the executable of this program, called *Bootblaster* as well as the binary of the replacement of the bootloader via ftp [9]. In our particular case, the following files have been obtained from the Internet:

*`ipkg` is the package manager of the *Familiar* Linux distribution, similar to the `rpm` (*RedHat* package manager) on *RedHat* Linux.



BootBlaster_1.18.exe
BootBlaster_1.18.exe.md5sum
bootldr-2.18.01.bin
bootldr-2.18.01.bin.md5sum

In general it makes sense to check the md5sum of every binary before uploading it to the handheld. Depending on the ftp software, one has to verify that the correct file transfer mode is engaged for download of binaries. It is also good to verify, whether the downloaded bootloader version is the appropriate one for the specific *iPAQ* model on which the installation process takes place.

The upload of `BootBlaster_1.18.exe` and `bootldr-2.18.01.bin`[†] onto the *iPAQ* is done with the *ActiveSync* software. This program should have been shipped with the *iPAQ* handheld. The latest version can also be downloaded from the specific site provided by *Microsoft* [10].

After upload the user can execute the binary `BootBlaster_1.18.exe` and access the following menu points:

1. Backing up the old bootloader and *Pocket PC*:
At any rate the bootloader image of the previously installed bootloader should be backed up. It is then stored under the filename `saved_bootldr.bin` and also can be transferred to the PC. It is of great importance, that a copy is left on the *iPAQ*. This copy might be needed in order to restore the old bootloader, should the installation of the new one fail.

In order to be able to restore the previously running *Pocket PC*, a backup has to be taken and copied to the desktop PC. `BootBlaster` stores the image of the ROM memory in a GNU-zipped (*.gz) format on the *iPAQ*. The data inflation reduces the original 16 Mbytes to about 9 Mbytes. The transfer of this file via the serial communication port to the PC may take a while.
2. Installing the bootloader: After initiating this menu option, the *iPAQ* does not show any response for a while. But that is nothing to get nervous about. At any rate, should this procedure fail or be interrupted, there is no risk, since `Bootblaster` protects the flash memory where *Pocket PC* resides during this step. The crucial phase is still to come: After protecting the flash, the new bootloader is programmed into the previously erased boot sector.
3. Verifying the bootloader installation: This is done by choosing another menu option *Verify* from `Bootblaster`. **If it now indicates that**

there is no working bootloader one is not supposed to reset the *iPAQ*. Otherwise the handheld computer cannot boot. Since there are no bootable devices (like a floppy drive in a PC) to be found in a PDA, for the user there is no way to bring the *iPAQ* back to functionality. In the installation instructions [8] this illustratively is referred to as a “bricked” *iPAQ*. In case of failure, the user should repeat the second step of this list – if anything fails with the saved old bootloader image – as long, as the verification process indicates a working bootloader.

Apart from the backup of the *Pocket PC 2000* image, all that happened until now is the substitution of the bootloader. Since *Pocket PC 2000* still resides in the *iPAQ*'s flash memory, it also can be launched by simply resetting the PDA.

Start It Up

In order not to boot into *Pocket PC*, one has to take the following steps:

1. Set up the terminal program on the PC (e.g., *Hyperterm* on *Windows*): For the connection one has to choose the correct serial port the *iPAQ* cradle is plugged in. For this port the following parameters have to be set:

baud rate: 115200
flow control: off

2. Launch the terminal emulator program on the PC and reset the *iPAQ*. The reset is done by pressing with the *iPAQ*'s input stylus into the notch at the lower right corner of the PDA. For this purpose the *iPAQ* has to be removed from the cradle. In order to make the bootloader prompt (`boot>`) appear afterwards in the terminal running on the PC, one has to press the joypad[‡] at the same time the reset is done
3. After this, the PDA has to be placed back into the cradle and the connection to the terminal running on the PC is established by either pressing the space key on the PC keyboard or the key with the calendar symbol on the *iPAQ*
4. At the boot-prompt in the terminal emulator program running on the PC one should enter

```
boot> pflash 0x40000 0xffff 0
```

[†]There is no need to upload the md5sum files.

[‡]The navigation key with the integrated speaker at the lower end of the *iPAQ*.

in order to unlock the part of the flash where the image of the OS (now still *Pocket PC 2000*, but later Linux) resides[§]. The part from the addresses below (0 → 0x40000), where the bootloader is to be found, still remains protected

Now the system is ready to receive an upload of the basic Linux kernel image.

The following steps are explained on the installation Howto of the *Familiar* Linux distribution homepage [4]. First, one has to download the image of the basic Linux system, i.e. the root filesystem, from the link provided on the *Familiar Project* homepage [4]. In this particular installation the root image for PPP or USB networking install `task-bootserial.jffs2` of the version v0.5.1[¶] was used. Now the user is only two steps away from a (remotely accessible) working Linux system on the *iPAQ*:

1. For the bootloader configuration, type

```
boot> partition reset
```

followed by a

```
boot> load root
```

in the terminal

2. The *iPAQ* expects the user now to upload the root-image applying the xmodem protocol via the terminal emulator

After having taken these steps, the system is booted by a simple

```
boot> boot
```

After a while, the login-prompt for the user `root` should pop up in the terminal of the PC (unfortunately not on the screen of the *iPAQ*). Now there is a working Linux running on the PDA, but the input/output still depends on a terminal running on a connected PC. The Linux distribution (in this case *Familiar*) providing a driver for the LCD display remains to be installed. Anyhow, things are unfortunately not that straight forward as they seem. Since this distribution in general has to be downloaded directly from the Internet, one first has to set up a working TCP/IP connection for the PDA.

Get Ourselves Connected

This section is about setting up PPP and a forwarded TCP/IP connection between the *iPAQ* and a Linux

desktop PC. Depending on the version of the root-image the user has installed, the basic packages for building up a serial PPP (Point to Point Protocol) connection should already be found on the *iPAQ*. The setup is explained on the PPP Howto page [11].

First, one has to log onto the *iPAQ*. The password for the user `root` by default is `rootme`. In order to prepare the *iPAQ* to run a TCP/IP connection via PPP, the user has to write the domain name and the nameserver, via which the *iPAQ* is able to resolve IP-addresses, to the file `/etc/resolv.conf`:

```
echo "domain csc.fi
nameserver 193.166.1.250
nameserver 193.166.1.249
search csc.fi" > /etc/resolv.conf
```

The input above is just an example, how the settings inside CSC would look like. Of course, one has to change the domain name (`csc.fi`) as well as the nameserver IPs according to the domain the desktop PC belongs to.

Since there is no text editor installed on the *iPAQ* by now, we have to use shell commands for that purpose. This is simply done by use of the `echo`-command in combination with the redirection symbol, `>`, or the append symbol, `>>`, depending if one wants to create a new (or overwrite an existing) file or simply append to a already existing, respectively.

Further the options for the PPP connection from the *iPAQ* side have to be declared. First – if not already existing – the user has to create a directory `/etc/ppp/`, where all the files related to the PPP connection will further reside. This is done by

```
cd /etc
mkdir ppp
```

Then the following lines have to be inserted into the file `/etc/ppp/options`

```
echo "-detach
defaultroute
noauth
nocrtscts
lock
lcp-echo-interval 5
lcp-echo-failure 3
/dev/ttySA0
115200" > /etc/ppp/options
```

Here it is good to place a warning, that one should double-check the typing, since the correction of typing errors in an environment without text editor is a quite tedious task. This is even of higher importance,

[§]The upper address can vary due to different amounts of memory installed on the PDA.

[¶]Meanwhile version v0.5.2 became available.

when there is text to append to an already existing file. For instance, if the user wants to have the PPP related modules being loaded automatically during startup of the PDA, the following lines have to be appended to the module setup-file, `/etc/modules`,

```
echo "  
slhc  
ppp_generic  
ppp_async" >> /etc/modules
```

It is crucial to use the append operator, `>>`, since there are already entries in this file by default. The same warning applies also to the necessary attachment to the module configuration file, `/etc/modules.conf`,

```
echo "alias char-major-108 ppp_generic  
alias /dev/ppp ppp_generic  
alias tty-ldisc-3 ppp_async"  
>> /etc/modules.conf
```

It is also recommended to check settings that should have been done automatically during install of the root image. The file `/etc/passwd` should have the user for the PPP connection, `ppp`, inserted. Thus a line similar to

```
ppp::99:99:PPP Account:/tmp:/sbin/pppd
```

should appear in this file. The contents of an already existing file, for instance `/etc/passwd`, easily can be displayed by the command `cat /etc/passwd` entered at the shell prompt. With this command the user also can check, whether the previously made changes to other files have been implemented correctly.

In order to get the secure shell (`ssh`) connection to work, the user has to enter the following sequences in the shell:

```
ssh-keygen -b 512 -f /etc/ssh/ssh_host_key -N ''  
ssh-keygen -d -b 512 -f  
/etc/ssh/ssh_host_dsa_key -N ''
```

Mind that the last `ssh-keygen` command due to typesetting has been split into two lines but should be executed in a single command-line.

Finally, one should give the permissions for the user `ppp` to start the PPP daemon, `pppd`, and set the `suid` flag for the secure shell daemon,

```
chmod 4755 /usr/sbin/pppd  
chmod u+s /usr/sbin/sshd
```

That is all that is to be done on the *iPAQ* side. After reboot a properly configured PPP daemon as well

as a secure shell daemon, `sshd`, should be launched during boot time. If the user does not want to reboot, the following set of commands will start that services immediately on the running *iPAQ*:

```
insmod slhc  
insmod ppp_generic  
insmod ppp_async  
sshd
```

On the Linux PC, i.e., the desktop machine the *iPAQ* is connected to, the user can create a connection script for the PPP daemon, `/etc/ppp/peers/ipaq`, containing the following lines

```
-detach  
noauth  
nocrtscts  
lock  
user ppp  
connect '/usr/sbin/chat -v -t3 ogin--login: ppp'  
/dev/ttyS0  
115200  
192.168.1.100:10.0.0.0
```

In the sequence above, possibly changes can be necessary, if the *iPAQ* is not connected to the first serial port (`/dev/ttyS0` on Linux – `COM1` in terms of DOS definitions). For instance, if this port already is occupied by another device, the possible choice is the second port (`/dev/ttyS1` – `COM2`). In the example above, the IP for the host machine (i.e., the Linux desktop), `192.168.1.100`, has to be changed accordingly to the one assigned to the desktop PC. The second IP is a private Internet address and can be used without any problems. One also could ask the network manager for a permanent IP for the *iPAQ* or set up the system to dynamically assign the IP (not explained in this article).

The last step now is to take care of the IP forwarding. In other words, to enable IP packages being passed through the desktop PC to the *iPAQ* and vice versa. In most cases this is done by the command

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

entered at the prompt of a root-shell on the desktop PC. If running `ip-chains`, the following lines might be necessary to enter (took one week to find out):

```
/sbin/ipchains -A forward -s 10.0.0.0 -jMASQ  
/sbin/ipchains -A forward -l -jDENY
```

If the *iPAQ* now is rebooted by the command `shutdown -r now` or the needed modules have been loaded manually it will be ready to accept the PPP connection^{||}, which from the desktop PC in most Linux distributions is launched by the user `root` giving the command

^{||}One should not forget to switch of the terminal emulator running on the desktop PC after reboot.

```
pppd call ipaq
```

Depending on the specific model of the *iPAQ* series, a little penguin might pop up after reboot on the LCD display of the PDA, indicating that Linux is up and running.

Getting Familiar

On the Linux desktop now the PPP connection shall be established with the command given in the previous section. Now, the user can log onto the *iPAQ* as user root. This is done by the command,

```
ssh -lroot 10.0.0.0
```

provided that the assigned IP for the PDA is 10.0.0.0.

At any rate it makes sense on a PDA to have the correct time set**. To this end, the correct date has to be fetched from a time server. Entering

```
ntpdate -b c1epsydra.research.compaq.com
```

in the shell running on the *iPAQ* should do the job. This is also a good test to verify that the IP-forwarding works. If the *iPAQ* is not able to resolve the ID, it obviously does not.

The easiest way to load a working system from the net is described in the *postinstall Howto* [12]. After entering the following sequence of commands

```
ipkg update
ipkg install familiar-postinst
/root/postinst
ipkg upgrade
ipkg install task-x
```

This loads the ipkg meta-package for post install, familiar-postinst, which contains the most important features a PDA Linux system needs. The package task-x provides a basic X Window System, i.e., after next reboot the user already will get a grey screen with a mouse-pointer on it. On top of this, there still is a window manager to be installed. On the *iPAQ* installation tested at CSC, the window manager ion (see Fig. 1) was chosen due to its low memory consumption. The command ipkg list provides the user with available packages. A convenient graphical user interface (GUI) running under X to the package manager is pikpak, which depends on the *python*-script language (see Fig. 2).

**Our *iPAQ* here at CSC showed January 1st, 1970, as default date from installation.

††To the knowledge of the authors, there is no free ssh application available for *Pocket PC 2000*.

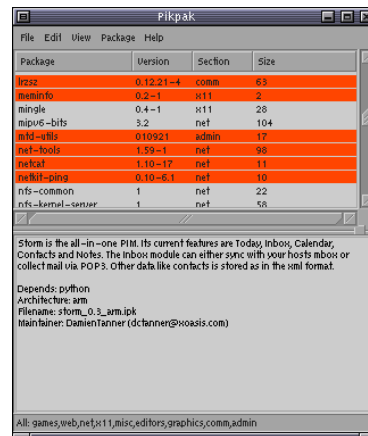


Figure 2: Screen-shot of the GUI to the ipkg package manager, pikpak.

During the first installation process, but also later, it might be comfortable to export the display of the *iPAQ* to the desktop machine (needs X Window also running on the desktop PC). To this end, one has to grant access of the *iPAQ* output on the window manager running on the desktop. The command xhost + 10.0.0.0 entered on a shell of the desktop PC should enable the X-output, again provided that the *iPAQ* is assigned to the IP 10.0.0.0. The command export DISPLAY=192.168.1.100:0.0 given in the shell of the *iPAQ*, redirects X-output to the desktop. Of course the IP 192.168.1.100 has to be replaced by the one assigned to the desktop PC of the user.

Linux vs. Windows

No, there will be no discussion on the philosophy of free software distribution in this article. The heading of this section simply should bring up the question, what the user gains if the pre-installed *Pocket PC* is replaced by Linux.

This question can only be answered by the user himself. In principle, if the *iPAQ*'s purpose is simply to act as a PDA, there might not be a reason for replacement. Especially not, if the user's desktop PC is running *Windows* and if there is a need for exchange of files with established office software provided by *Microsoft*. There are even instructions on how to connect a PDA running *Pocket PC* to a Linux desktop to be found on the Internet [13].

On the other side, some people want to have the advantage of a secure net connection via ssh^{††}. For instance, an administrator who wants to log onto his server via a mobile device might be happy about

ssh being available on the PDA. Additionally, there could be a need of a mobile reception of a X Window output from an application running on a Unix based server. For instance, an application engineer might want to remotely check the convergence of a simulation running on a server somewhere else. Certainly, the reception of X-output demands a fast mobile connection, but with wireless networks gaining popularity, this certainly will be an aspect in the near future. In any case PCMCIA WLAN cards are fully supported by the Linux kernel.

In principle one can get similar functionality from a Linux *iPAQ*, as there is to be found in *Pocket PC 2000*. For instance, besides a virtual keyboard, *xkbd*, the full-screen gesture recognition program *xstroke* is able to be installed. It provides a well working input method, similar to that one offered on the *PalmOS*, under the X Window environment of the PDA (see Fig. 3).

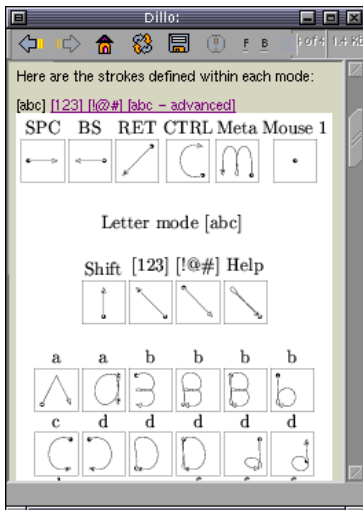


Figure 3: Help page shown in the browser *dillo* of the gesture recognition program *xstroke*.

For people who want to add full PDA functionality to their Linux *iPAQ*, there are two complete PDA environments available:

- *GPE* [14] is a free software GUI environment for handhelds. It is a compound of different programs with emphasis on the implementation of correct data exchange between the components.
- *Opie* [15] is a community fork of the *Qtopia* environment, that comes with some commercial Linux handhelds.

These environments show a higher demand for storage capacity. Thus, an *iPAQ* model with an extended memory capacity, compared to the one that has been tested for this article, is required.

Bibliography

- [1] Jamey Hicks, *How to Run Linux on iPAQ Handhelds*, download from <http://www.handhelds.org/handhelds-faq/handhelds-faq.pdf>
- [2] Technical specifications of *Compaq H3600 Series PDAs*: http://handhelds.org/Compaq/iPAQH3600/iPAQ_H3600.html
- [3] Homepage of the ARM Linux Project: <http://www.arm.linux.org.uk>.
- [4] Homepage of the *Familiar Project*: <http://familiar.handhelds.org>.
Installation Howto: <http://familiar.handhelds.org/familiar/releases/latest/install/H3600/install.html>
- [5] Homepage of *Debian Linux* distribution: <http://www.debian.org>.
The pages for the *Debian ARM Linux* port: <http://www.debian.org/ports/arm>
- [6] Homepage of *Intimate Linux* distribution: <http://intimate.handhelds.org>
- [7] Howto pages for configuring a Linux handheld: <http://www.handhelds.org/minihowto>
- [8] Howto pages for *bootloader* installation: <http://www.handhelds.org/feeds/bootldr/install.html>
- [9] ftp download site of files needed for *bootloader* installation: <ftp://ftp.handhelds.org/pub/linux/compaq/ipaq/v0.30/>
- [10] Download for *ActiveSync* software: <http://www.microsoft.com/mobile/pocketpc/downloads/default.asp>
- [11] Howto pages for configuring PPP connections.
Linux: <http://www.handhelds.org/z/wiki/PPPHowto>
Windows: <http://www.handhelds.org/z/wiki/WindowsPPPHowto>
- [12] Postinstall Howto of *Familiar Linux* distribution: <http://www.handhelds.org/z/wiki/PostInstallHowto>
- [13] Howto pages on linking a *Windows PDA* to a Linux desktop: <http://www.handhelds.org/minihowto/wince-link/index.html>
- [14] *GPE PDA* environment for *iPAQ*: <http://gpe.handhelds.org>
- [15] *Opie PDA* environment for *iPAQ*: <http://opie.handhelds.org>