Linux on the Psion netBook HOWTO

Last modified 9 July 2006. Changes.

This document contains information on how to get a Psion netBook to boot up the Linux operating system, how to subsequently configure the system, and how to obtain, install and use applications suitable for the netBook's limited resources. This document also contains notes on the remaining problems at the moment, and instructions on such things as compiling a custom kernel and assembling a custom OS.IMG file. Framebuffer support is implemented, and X windows is supported. Compactflash, PCMCIA, and touch screen are rudimentarily supported. Many wireless PCMCIA cards will work fine; indeed, in general most PCMCIA devices (16-bit PC Card only!) will work fine. The touchscreen patch is relatively new. The only hold up to a fully functional, operational linbook seems to be that no power management is available (you can't turn the linBook off!). There is no support for sound yet.

Other useful hints may be found on the Linux on Psion 5MX HOWTO (http://linux-7110.sourceforge.net/howtos/series5mx_new/index.htm). The netBook and 5MX systems have considerable similarity.

Offer (May 2006): An infrared modem is offered for free (shipped anywhere in the world) to anyone who posts a patch for the netbook kernel that enables infrared. However, a patch for proper PCMCIA support is considered more critical and will take precedence if posted prior to, or within a few days of, the infrared patch. Please send inquiries or patches to the OpenPsion mail list.

PDF Version of this HOWTO (netHOWTO.pdf) (1.4 MB)

Please post information, questions, FAQ (with answers?) to the openpsion mail list (mailto:linux-7110-psion@lists.sourceforge.net).

Table of Contents

1. Disclaimer **6**
2. Credits **6**
3. Introduction **7**
3.1. A Note to EPOC Users Thinking of Migrating to Linux 7
3.2. What Works, Generally 7
3.3. What Doesn't Work, Specifically 8
3.4. The netBook Specifications 8
3.5. The Booting Problem 10
3.6. Malaysian netBooks 10
4. Installation to Initrd **10**
4.1. Getting the things you need 11
4.2. Backup EPOC 11
4.3. Configure ArLo 11
4.4. Booting and Rebooting 12
5. Installation to Compactflash **13**
5.1. Getting the things you need 14
5.2. Configure ArLo 14
5.3. Repartition your Compactflash Card 15
5.4. Swap Space Comments 15
5.5. Install the new system on the Compactflash Card 16
5.6. Alternate Installation: An OS.IMG for linux 16
5.7. Appendix: The /linuxrc script 16
6. Using Bookboot - the Linux OS.img **17**
6.1. Making your own OS.img 18
6.2. Memory Configuration 18
7. Using ArLo - Booting Linux from EPOC **18**
7.1. Introduction/Example Configuration File 19
7.2. ArLo Kernel Memory Configuration Options 20
7.3. Howto Boot ArLo First in EPOC 20
7.4. ArLo Memory Limitations 21
8. X Windows **21**
8.1. Introduction 21
8.2. A Working X Server 21
8.3. Touch Screen 22
8.4. Using Keylaunch 24
8.5. X modules 25
8.6. Keys 25
8.7. X Window Managers 25
8.8. Alternate Window Systems - Microwindows, Opie, Picogui, Minigui 27
8.9. Debugging Notes 27
8.10. Alternate (not altogether working) X Servers 27

9. Optimization and Adjustments 28
9.1. Settings in /proc/ 28
9.1.1. /proc/sys/vm/laptop_mode 28
9.1.2. /proc/sys/cpu/0/ 28
9.1.3. /proc/netbook/ 28
9.2. IDE Disk Performance 29
9.3. System and Hardware Clocks 29
10. Applications 29
10.1. Installing Packages 29
10.2. Editors and Word Processors 30
10.3. Spreadsheets 30
10.4. Web Browsers 30
10.5. Calculators - RPN and otherwise 32
10.6. GPS on the netBook 32
10.7. Dictionary 33
10.8. Astronomical netBook 33
10.9. Games 35
10.10. Movie/Video Players 37
10.11. Compiling the Kernel or Applications with GCC 37
10.12. Web Servers 37
10.13. Misc Notes - Man Pages 37
10.14. Java 37
11. Serial Ports 38
11.1. Onboard Serial Port 38
11.2. Infrared Serial Port (Doesn't Work) 38
11.3. PCMCIA Serial Port 39
12. Using a Mouse 39
12.1. A WARNING on using the netBook's onboard serial port with a mouse. 39
12.2. Using the onboard serial port with a mouse 39
12.3. Using a PCMCIA serial port with a mouse 40
12.4. Using a mouse with X windows 40
12.5. Mouse types to consider 40
12.6. Mouse troubleshooting 40
12.7. Using a USB or bluetooth mouse 41
13. PPP from the netBook 41
13.1. Connecting to the desktop with PPP 41
13.2. Telnet/rsh/ssh 43
13.3. Connecting to the internet using an external modem 43
14. Making Presentations Using a VGA-out PCMCIA Card 44
14.1. Starting the X server 44
14.2. Window managers 45
14.3. Slide presentations 45
15. External IDE Devices using a PCMCIA Adapter 46
15.1. PCMCIA IDE Hard Disk 46
15.2. PCMCIA CD-ROM 47

16. Resources/Links 48
16.1. The OpenPsion Website 48
16.2. netBook Linux Distributions 48
16.2.1. Openembedded Linux on the netBook (IPKG based) 48
16.2.2. Debian Linux on the netBook (DPKG based) 49
16.3. IPKG and DEB Feeds 49
16.4. The PCMCIA HOWTO 50
16.5. Plp Tools 50
16.6. ArLo 50
16.7. Bookboot 50
16.8. Proboot 51
16.9. Precompiled Kernels 51
16.10. Initial Ramdisks (initrd's) 51
16.11. The Mail List 51
16.12. netBook Stripped! 51
16.13. The "simputer", from India 51
16.14. RISC OS on a netBook 52
17. Poor Man's Linux's: XTM and ELKS, or EPOCemx 52
17.1. XTM and ELKS 52
17.2. EPOCEMX 53
18. Kernel Development 53
18.1. Wish List 54
18.2. Kernel Development Needs 54
19. Recipe for Compiling the Kernel 56
19.1. Preferred Cross-compiler: Scratchbox 56
19.2. Qemu and Debian Linux - a Complete Emulated System 57
19.3. Cross-compiler Alternative: The emdebian cross compiler for Debian Woody 57
19.4. Get kernel files and patches 57
19.5. Compile 58
19.6. Compiling and Installing Modules 58
20. Developer's Notes 59
20.1. Some Documentation 59
20.2. PCMCIA 59
20.3. Compactflash 59
20.4. PCMCIA/CF pinouts 60
20.5. LCD/Framebuffer Driver 60
20.6. IRDA 60
20.7. Ramdisks/ramfs/tmpfs 61
20.8. Power Control 61
20.9. Keyboard 61
20.10. Arlo Needs 61
20.11. Sound 62
20.12. SanDisk 128MB + Wi-Fi CompactFlash ideal wireless solution? 62
20.13. Initial Ramdisks (initrd's) 63

21. FAQs 63
21.1. Why do the URL's of the HOWTO pages keep disappearing/changing? 64
21.2. How can I type the pipe (|) character? 64
21.3. Is there any size limit to the compactflash or microdrives that are supported? 64
21.4. Compare and contrast compactflash or microdrives for me please? 64
21.5. How can I get the kernel to boot directly without having to boot up EPOC? 64
21.6. Are there any EPOC backup solutions that don't require the desktop software? 65
21.7. Can I plug in a serial mouse to the serial port and use a mouse that way? [Yes, but be a little wary.] 65
21.8. Can you tell me the specifications of the netBook's serial port? 66
21.9. Are There Any Issues with the Malaysian netBooks? 67
21.10. What Are the Kernel Differences Between Series 7/netBook? 67
21.11. What Effect Does a Memory Upgrade Have/Is All Memory Recognized? 67
21.12. Is it safe to send my netBook through an airport X-ray machine? 67
21.13. Is possible to rebuild the netBook's Lithium-Ion battery pack with new batteries? 68
21.14. How can I manually unpack an *.ipk package? 68
21.15. How can I manually unpack a *.deb file? 68
22. Floating Point Emulation Notes 69
22.1. 3. What's the difference between gcc soft-float support and the kernel NWFPE/FastFPE support? 69
22.2. 4. Can I use both hard & soft float at the same time, or must I choose one of them? 69
23. Extracting a Zaurus ROM (MTD and JFFS2) filesystem 70
23.1. MTD support on your system 70
23.2. Extracting the Zaurus filesystem 70
24. Changes 71

1. Disclaimer

This is the method generally used for installing and configuring Linux for the Psion netBook. It may not yet work for you. It may make your netBook blow up and your house fall down, for all I know. It might even steal your girl (or boy) friend and leave you in the gutter begging for beer money with only bent floppy disks to your name. Don't blame anybody if something goes wrong, and your tears will fall on deaf ears, so to speak. That said, if you have any problems you can always post a query on the mailing list (send e-mail to: linux-7110-psion@lists.sourceforge.net). Somebody there will likely be able to, but nobody is obligated to, assist you. Make sure you backup, backup, backup!!!

In short, we are not responsible.

This is what might happen:

so you are duly warned.

2. Credits

This document is based on the Series 5MX HOWTO (/howtos/series5mx/5MXHOWTO.htm). The bulk of the credit for OpenPsion goes to those dedicated hackers who got the working kernel up and running and the various bits of hardware working. The credits list for the present document is as follows:

1. Anonymous

Please see the openpsion website (http://www.openpsion.org/) (or more directly linux-7110.sourceforge.net (http://linux-7110.sourceforge.net/) for the more excellent people involved in the port.

3. Introduction

The OpenPsion project is an effort to port the widely used Linux operating system to the various Psion hand held computers. Considerable success has been achieved with the Series 5MX/5MXPRO pocket computers. For the netBook, familiarity with linux is essential at this point. The kernel now offers rudimentary support for compactflash, PCMCIA, and X windows, and touch screen. There are at two prepackaged distributions (one based on dpkg and one based on ipkg) that offer are fairly complete and ready to go systems.

3.1. A Note to EPOC Users Thinking of Migrating to Linux

If you have never used linux before, this system is probably not for you. We are still at a phase where the system requires considerable technical fussing by the user to get things like networking, applications, etc. to work. This is not (yet) a user-friendly system like EPOC is; the system is essentially a linux workstation. In addition, the very nice applications in EPOC (Word, Spreadsheets, Calendars, Contacts, Time/Alarm , etc.) do not really have lightweight equivalents in linux. Linux does offer better browsers, networking, and many other things, however, and the development of linux on ARM PDA's in general is quite active, unlike EPOC.

In short, unless you are familiar with linux and know that it might work for you and you don't mind spending some time fussing with technical system stuff, migrating to linux from EPOC on your netBook is not recommended. Try linux on a desktop PC (or on one of the "live" bootable CD's) first to get some familiarity with it, if you think linux might be a way to go.

On the other hand, linux is free and it will cost you nothing to try it out (unless, of course, you pay someone to install it and fuss with it for you...) With adequate backups, the EPOC system can easily be restored like it was (I've done this many, many times without apparent harm). We may eventually get to a distribution that will be user friendly with GUI's for everything, but as of 4/05 that is likely a few years away...(and where will your netBook be then?)

A table listing linux equivalents to EPOC applications can be found on the former openpsion wiki page (http://linux-7110.sourceforge.net/oldwiki/EpocLinux.shtml).

3.2. What Works, Generally

The team has ported the Linux 2.4.26 kernel to the netBook/Series 7, created ArLo (the boot loader), created several initial ramdisks (initrd), and created at least two preliminary systems for compactflash as the root disk. An effort, called "bookboot" based on proboot, now allows the netBook to boot directly to linux through an linux-customized OS.img file. Through the efforts for linux on the 5MX, a larger prepackaged distribution based on Debian Woody/Sarge has been ported to the netBook for installation on compactflash disks. In addition, an openembedded IPKG-based installation is being tailored to work on the netBook.

For the netBook, the kernel will boot up o.k. and the system will reboot back into EPOC. The compactflash support can be used to mount an e.g., ext2 partition onto which a system can be installed. Other systems and applications can be crammed into an initrd.gz file (ram disk filesystem), if one wants to avoid using a compactflash card. The kernel does have framebuffer support, but it is clear the the kernel framebuffer driver is not quite complete. However,

X-windows (or PicoGUI) will work, and there is now alpha support for the touchscreen (and an external mouse using the onboard serial port (but see the mouse section in this HOWTO!) or a PCMCIA serial card works.)

Now that compactflash support is available, distributions containing all the basic elements of any linux system: bash, vi (vim), groff, less, man, perl, nawk, bc, dc, lynx (a web browser), ppp, telnet, ftp (ncftp), rsh, ssh, minicom, emacs; ("zile", a smaller emacs-type editor is probably preferred), etc. can be assembled. Starting with a basic system, it would be easy, though it would take a little time, to assemble your own custom system. All known systems are based on Debian Linux or ipk systems (Sharp zaurus, Compaq ipaq - see e.g., handhelds.org (http://handhelds.org), or myzaurus.com (http://myzaurus.com)) since these systems have the best support for the ARM cpus. New debian arm packages (*_arm.deb, or a slimmer set of packages, *_arm.ipkg) can be downloaded and installed directly using *"dpkg -i *_arm.deb"*, similarly for *ipkg*. Most people have been pleasantly surprised at how far along the whole OpenPsion project has come. OpenPsion is a fully functional, basic linux system. Slim windowing systems such as PicoGUI are being developed, and the X11 packages now work "out of the box." Most of the larger distributions, or most useful systems, are fairly sizeable and so require a compactflash card of 256MB or greater in size (a system without X could be squeezed onto a 64 MB card, and compactflash cards of up to 1GB are available and relatively inexpensive). Smaller systems based on initrd's can also be found, and they can be quite functional.

You can see a list of PCMCIA cards know to work (or not work) with the netBook at the former netBook hardware Wiki (http://linux-7110.sourceforge.net/oldwiki/SupportedHardware.shtml). Please feel free to add to this Wiki page if you find other hardware that works/doesn't work in your linBook.

3.3. What Doesn't Work, Specifically

The following things don't yet work on the linux netBook:

Touchscreen. A patch for touchscreen has recently been made available (10/04); it seems to work o.k., but perhaps should be considered alpha status. It is known to stop working at irregular times, requiring a reboot to reset it.

Power Control. There is at present no way to turn the linBook off, and as yet no attention has been given to power/energy management. Adequate power control will take a bit of work. For now, when you want to turn the machine off, reboot and turn off the machine at the netBook/One-Ed startup screen. Turned back on again, the machine will continue to boot to linux.

Battery Levels There is at present no way to read the energy level of the battery or backup battery, so there is no way of knowing how when your netBook will run out of power.

Sound. Someone was meant to be working on sound support, and close to posting a patch, but sound out of the linBook has never been reported.

PCMCIA. PCMCIA support is still rather crude. Many PC Card (16 bit) Cards have been reported to work, but PCMCIA support is certainly not 100%. Combined modem/ethernet, or modem cards seem to be the most problematic, e.g., ironically, Psion's own DACOM 56K/ethernet card.

LCD/X Windows. X Windows is working at the moment, but there have been to be unspecific problems with the LCD controler and X Windows. At times this manifests itself as an off color or blue screen when X windows is started, although this can be worked around by ssh'ing to the netBook from the netBook to put the terminal onto a pseudo terminal.

Infrared Serial. Infrared serial communications are not yet supported, although it seems to be almost working. This hardware is meant to be one of the easiest things to get working on the sa1100 cpu, but it still eludes us.

3.4. The netBook Specifications

Download the A4 netBook Specifications Sheet (netBook_A4-2.pdf).

1. Processor - 32 bit, Intel SA-1100 StrongARM 190 Mhz. This CPU has a 16 KB instruction cache, and an 8 KB data cache.
2. Internal Memory - 32 MB DRAM (Series 7: 16 MB) (32 MB DIMM expansion option for 64 MB max. memory.) (Series 7: 16 MB DIMM expansion for 32 MB max. memory.)
3. Internal ROM - none (in EPOC, about 14 MB of RAM acts as the Z: drive ROM)
4. Flash ROM - boot loader. (All this unconfirmed but probable). The netBook has a 16 Mb (2 MB) rewritable flash chip that stores the bootloader. The chip is a Sharp LH28F 160S 3HT-L10A (also known as: LHF16KA7). All netBooks have their eeprom flashed with the Bootloader version 11, and all MBooks with version 12; this is likely the difference between the various models (Series 7, netBook, mBook). Programs apparently exist to reflash this ROM, but few have dared...see the mail list at psilinux.org.
5. Expandable Memory - Type I/II 3.3V. CF Card and IBM micro drive support (disks of up to 512 MB have been tested; larger disks (to 1GB!) are untested)
6. PCMCIA - Type I/II 3.3V. One slot for serial, modem, GSM, Ethernet and Flash ATA support [This specification is 3.3V, but 5V cards also seem to work, except perhaps in a 7Book.]
7. Display Size, Resolution, Type - 7.7" 640x480 back-lit STN LCD Panel (VGA). (I believe the display is a Hantronix HDM6448ATSC-7)
8. Display Type - Touch-screen, 256 colors
9. Default OS - EPOC (32 bit, multitasking)
10. Serial Ports - Standard RS232 and SIR Infrared; up to 115200 baud; (rumoured: IrDA 1.1, 115kbit/sec and 4Mbit/sec) The standard rs232 port is provided by a Maxim 78266 - and from what has been discovered, this chip was created specifically for Psion. However, it appears to be identical to a Maxim MAX3243-RS chip. A pdf file of the MAX3243 specifications can be found HERE (MAX3221-MAX3243.pdf). The document explicitly states that the chip supports a mouse, but there has as yet been only one verification that this works (and the serial chip survived; no serial port burn outs have ever been reported). What we know about the 78266 chip is HERE (NetBookSerial.pdf). The Psion serial cable is a null modem cable, so that it allows direct connection to the PC. This means that to use any external serial devices (e.g., modem, GPS), a null-modem adapter (nullmodem.jpg) is needed to "un-null" the Psion cable.
11. Power - Li-Ion battery pack, up to 10 hours, backed up by a CR2032 lithium battery Charge status and alerts LED’s Approx. 8 hrs of operation. Optional 15.5V, 1.5A external power (RadioShack Adaptaplug "C", 4.75mm OD, 1.75mm ID, 1cm long) Battery is a rechargeable 11.1 volt, 1500mA pack containing three CGR18650H Li-Ion cells.
12. Sound - 12 bit, built in speaker
13. Microphone - yes
14. Keyboard - 84 key, QWERTY layout
15. Size - 235x 182 x 37 mm
16. Weight - System incl. battery pack 1150g
17. Operating Temperature - 0 to 50 C

18. Shock - Designed to withstand a 1m drop onto a wooden surface
19. Electronic Emissions - EN55022 Class B. FCC Part 15 Class B.
20. IP Rating - IP30

3.5. The Booting Problem

There appear to be two basic approaches to booting a linux kernel. One is through ArLo (http://www.yipton.demon.co.uk/content.html#ArLo) which boots linux from a running EPOC system. The other is by making an image file similar to Psion's OS.img. A program called "bookboot" has been developed for the netBook to assemble OS.img files for booting linux; bookboot was based on proboot (http://www.muru.com/linux/psion/proboot/), developed for the 5MX-Pro. There are sections on both of these approaches in this HOWTO. Bookboot is used for the netBook to create a OS.img file from a working linux kernel and small initrd. This file is then put on a compactflash disk, where it is loaded in by the EEPROM on the netBook at initial boot up. That is, it loads in this file rather than the usual EPOC kernel. Note that the DOS partition on which this file resides has to be set to be bootable. Bookboot for the netbook could still use some polishing; but it looks like most of the main difficulties have been resolved. Bookboot seems to be quite functional; you can find the latest version of bookboot at bookboot (http://linux-7110.sourceforge.net/files/People/Klaasjan/netbook/).

3.6. Malaysian netBooks

A crop of a few thousand netBook's became available in late 2002 after an education company (OneEd.com) went bankrupt. These were (are?) cheap netBooks that apparently have the identical hardware capabilities of an ordinary netBook. They appear to have a slightly different bootloader on them so that booting up upgrades to the EPOC operating system has not worked. Presumably, replacing the bootloader, which is located as a small bit of ROM on the 16 MB DIMM (also known as a "personality module") with a normal netBook DIMM would make these machines identical to a normal netBook. See a description of these at E. Hasbrouck's Malaybook Site (http://hasbrouck.org/netbook/).

Insofar as linux booting from ArLo or bookboot, however, these malayBooks (or mbooks or Mnetbooks or etc.) are no different than any other netBook. Linux works on them just as well as ordinary netBooks, which they are except for the small software change (properties unknown) to the EPOC bootloader. OS.img files made using bookboot do not have to be tailored for the several variety of netBooks.

4. Installation to Initrd

This installation applies equally to all netbooks, including the malaybooks, as far as we know. Linux does not know any difference between these flavours of machines. This section limits itself to an initrd system. The next section describes the installation of a larger, more complete linux system onto a compactflash disk.

This page describes a linux installation that uses the Arlo boot loader to boot to linux from a running EPOC system. Using Arlo is optional; the section on "Bookboot" in this HOWTO describes the construction of an "OS.IMG" file that boots to linux directly; this is the preferred method by far these days. The OS.IMG has the advantage of a quicker, less involved boot to linux. An "OS.IMG" based system just boots the kernel and initrd that have been

already included in the "OS.IMG" file, so installation in this case consists of merely putting the "OS.IMG" file onto the first FAT partition of the compactflash card, just as for the original netBook's "OS.IMG" file.

4.1. Getting the things you need

You will need three things to complete an installation of linux on the netBook: (1) a kernel image, (2) a ram disk root filesystem (initrd.gz), and (3) the EPOC application ArLo.

(1) You can obtain kernel images from Psilinux Downloads (https://sourceforge.net/project/showfiles.php?group_id=8846). A single linux kernel supports the 16 MB, 32MB and 64MB systems (depending on if you have a Series 7, a netBook or the 32MB memory expansion card in the netBook, etc.). You will find a legacy 16MB kernel there as well. Put the kernel on your netBook as, e.g., *D:\linux.image*. You can also check out openpsion (http://staff.washington.edu/dushaw/psion/openpsion/) for "testing" kernels [well...everything is "testing" at this stage!] for the netBook.

(2) You can obtain a ram disk root filesystem from the Psilinux Downloads (https://sourceforge.net/project/showfiles.php?group_id=8846) page. Just about any initrd.gz system will work (maybe), but here is a place to get started. Put this on your netbook as, e.g., *D:\initrd.gz*. For now you can also check out openpsion (http://staff.washington.edu/dushaw/psion/openpsion/) for initrd's developed specifically for the netBook (32 or 64 MB ram). In general, a Series 7 has too little memory to warrant an initrd-only system.

(3) Finally, get and install Arlo (https://sourceforge.net/project/showfiles.php?group_id=8846) which is used to boot linux from a running EPOC system. (The original Arlo webpage is ArLo (http://www.yipton.demon.co.uk/content.html#ArLo).) ArLo can now be installed as a *.zip installation. Unzip the zip file (preserving the folder structure) on either drive C: or D:. ArLo has a nice GUI frontend. Details on how to use ArLo can be found at that web site, or the manual (http://www.yipton.demon.co.uk/arlo/latest/readme.html). This HOWTO also has a section Using ArLo - Booting Linux from EPOC.

4.2. Backup EPOC

When you boot linux everything on your C: drive will be wiped clean, so you will need to backup your C: drive. Linux will use all the memory of the Psion as system memory (except for the ram disk, of course). I've found that the best way to do this is to copy everything from the C: drive, including the System directory, onto a backup directory on the compactflash (I have a big compactflash, and I recommend that you get the largest one that you can afford). To restore the backup after rebooting into EPOC, I first delete everything on the C: drive, including the System directory. Then, copy paste your back up files (e.g., Documents and System) from the D: drive back onto the (empty) C: drive. This preserves most of your settings, e.g., modem or ethernet. (I have a malaybook that does not allow overwriting of files, hence I have to delete everything on the C: drive to restore the backup.)

When rebooting, it is helpful to have the netBook's OS.img file on the compactflash as well. With present systems, from linux a <Ctrl><Menu><Delete> will cause the system to shutdown and reboot into EPOC if the OS.img file is present on on the compactflash.

You may want to get and install the free "sysback" package which will restore many of your system settings (e.g., home city) after a hard reboot.

4.3. Configure ArLo

The linux kernel will presently boot up using an initrd. We had troubles originally, but the issue was resolved by putting a space between "series" and "7" in "machine = series 7" of the arlo.cfg (/howtos/netbook/netHOWTO/arlo.cfg) file. You can use this file as the configuration file, or as a guide of how to configure ArLo. The key is to use the line "*machine = series 7*" to tell ArLo what type of machine it is on. More about ArLo on the Series 7/netBook can be found at Peter van Sebille's Place (http://www.yipton.demon.co.uk/content.html#ArLo) and a manual (http://www.yipton.demon.co.uk/arlo/latest/readme.html) for ArLo is available. Note that the Arlo GUI configuration save will not save the "machine = series 7" option, so the arlo.cfg file has to be edited manually.

4.4. Booting and Rebooting

With the above ArLo configuration, you should be able to select ArLo from the tools and just hit a return to see linux boot up. Check out the /proc directory, etc. Things are limited to basic networking, ssh, etc. without X windows at the moment (July 2005). To reboot, just type *reboot* at the shell prompt which will boot you back into EPOC (I've got EPOC to boot up by cycling the compactflash card and pressing the reset button, etc.) So, you need not remove all the batteries of the netBook for a complete reset. Some distributions have enabled reboot through *<Ctrl><Menu><Delete>* .

Figure 1. A netBook with linux booted up

5. Installation to Compactflash

A compactflash (CF) system first starts by using an initrd to boot up and enable the CF and mount its system partition. Then "pivot_root" is executed to switch from the initrd root filesystem to the CF root filesystem. The trick is to have a suitable, minimal initrd system to get things started, a decent distribution installed on your compactflash card, and various scripts set up to orchestrate the transfer between the initrd and the compactflash systems. The installation is therefore similar to the previous section (Installation to Initrd), with the addition of a system on compactflash. *That section is a prerequisite to this section.*

The netBook will support microdrives (IBM, Hitachi, etc.), which behave identically to CF cards (or IDE devices, in general). Microdrives of size 4 GB or larger are supported; there does not seem to be a particular limit to disk size for either CF or microdrive.

Linux can now be booted using an "OS.IMG" file, roughly equivalent to the netBook's original "OS.IMG" file, and this is the preferred method of booting. The linux OS.IMG file contains and starts the kernel and a small initrd to transfer the system to a compactflash-based system disk. With a suitable OS.IMG file, the details of how the system is booted can be largely ignored by those wishing to get on to the linux system. The advantage of booting to linux from EPOC using Arlo, rather than a replacement OS.IMG file, is that Arlo gives you essentially a dual-boot system - you get the best of both EPOC and Linux. (But the OS.IMG approach starts Linux much faster.)

5.1. Getting the things you need

For a compactflash system that boots from EPOC you will need **four** things to complete an installation of linux on the netBook: (1) a kernel image, (2) a ram disk root filesystem (initrd.gz), (3) the EPOC application ArLo, and (4) a decent system installed onto an ext2 partition on the compactflash. **N.B. You cannot install a linux root filesystem onto the FAT filesystem that EPOC uses. That filesystem does not support the linux device filenames.**

(1) You can obtain kernel images from Psilinux Downloads (https://sourceforge.net/project/showfiles.php?group_id=8846) - a kernel that supports compactflash has now been formally released. The linux kernel will support either the 32MB or 64MB version, depending on if you have the 32MB memory expansion card or not. Separate kernels are no longer required for the various memory configurations. You might find a legacy 16MB kernel there as well. Put the appropriate kernel on your netBook as, e.g., *D:\linux.image*. For kernels that support compactflash, also check out openpsion (http://staff.washington.edu/dushaw/psion/openpsion/) for "testing" kernels [well...everything is "testing" at this stage!] for the netBook. Also try Leppä's Blog (http://sektori.com/~rosmo/netbook/index.php), the origin of the most cutting edge kernels at the moment.

(2) The ram disk root filesystem for a compactflash system is a small system tailored to mounting the compactflash drive and transferring the root filesystem to the compactflash ext2 partition. You can obtain a ram disk root filesystem for this purpose from the Psilinux Downloads (https://sourceforge.net/project/showfiles.php?group_id=8846) page. Put this file on your netbook as, e.g., *D:\initrd.gz*. For now, Debian on netBook (http://staff.washington.edu/dushaw/psion/openpsion/) has the only initrd's developed specifically for the netBook and compactflash support.

(3) Next, get and install Arlo (https://sourceforge.net/project/showfiles.php?group_id=8846) which is used to boot linux from a running EPOC system. (The original Arlo webpage is ArLo (http://www.yipton.demon.co.uk/content.html#ArLo).) ArLo can now be installed as a *.zip installation. Unzip the zip file (preserving the folder structure) on either drive C: or D:. ArLo has a nice GUI frontend. Details on how to use ArLo can be found at that web site, or the manual (http://www.yipton.demon.co.uk/arlo/latest/readme.html).

(4) There are several root filesystems that you can now install onto the ext2 or ext3 filesystem of your compactflash card. An ipkg-based system can be found at www.netbook-linux.org (http://www.netbook-linux.org). Debian linux based systems can be found at Debian linux on netbook (http://staff.washington.edu/dushaw/psion/openpsion/) (download from Psilinux Downloads (https://sourceforge.net/project/showfiles.php?group_id=8846)) and possibly here (http://linbook.risible.org/). Install one of these systems on the second partition of your compactflash card.

5.2. Configure ArLo

The ArLo configuration is identical to that described in the previous section with one important difference; see the section on ArLo in this HOWTO Using ArLo - Booting Linux from EPOC for details. The entry for booting to the linux initrd in the arlo.cfg file needs to have another line giving the option "boot = init=/linuxrc". This line will cause linux to first run "/linuxrc" on the initrd, and this script takes care of mounting the compactflash disk and transferring the root filesystem to it.

5.3. Repartition your Compactflash Card

To use a compactflash-based system, you'll need to repartition your compactflash drive to have a small FAT16 partition and a larger ext2 or ext3 partition. For a 128 MB Compactflash drive, I have a 28 MB FAT16/DOS partition for EPOC and 100 MB ext2 (or ext3) for linux. The EPOC partition has Arlo installed on it. The netBook needs the 13 MB OS.img file to boot back to EPOC, so about 20 MB is probably the minimum size.

In one case, my CF disk would no longer boot up the OS.img file after repartitioning. This may have to do with nuances of the DOS filesystem. See the linux man pages for fdisk. I followed the advice from these man pages and zeroed the first 512 bytes of the new DOS filesystem with "dd if=/dev/zero of=/dev/sda1 bs=512 count=1", and then formatted the new 28 MB DOS partition using either linux or EPOC. The EPOC system would then boot the OS.img file again.

Interestingly, if you make two or more DOS partitions on your CF, EPOC will apparently recognize all of them. Some people claim to have better performance from their CF this way.

If you do not have a linux system handy to repartition and format the compactflash card as described above, you can always download any of the "Live" CD/DVD's that various systems have made available. These disks allow your computer to boot off the CD and operate entirely off the CD; your hard disks are untouched. And you have a complete, portable linux system at your disposal. I can recommend Ubuntu Linux (http://www.ubuntulinux.org/), which is a spinoff of Debian linux (see the Downloads section - it is free, of course).

5.4. Swap Space Comments

There has been an on-going dialog concerning the use of swap space on a netBook. Generally, swap space is a good idea - this is an area of the disk, the compactflash card in this case, that is defined as a memory extension. When the system needs more memory than is available, memory can be made available from the disk; linux will "swap" the less used applications to the disk on swap space, freeing up system memory for more important needs. Given the netBook's limited 64 MB (max), swap space can enable compilation of programs or larger applications that would otherwise not be feasible. One person commented on the mail list that with swap space, the netBook seemed to be a bit happier; the exact effect of swap space on a system that does not have an obvious need for such space is unclear, however. Note that swap space is VERY slow; your application may finish, but it will likely take a while.

Sluggish performance is something that is commonly associated with low memory because desktop systems start swapping when memory is low. A swapless system won't exhibit this sluggish behaviour. When a memory limit is reached, an application will, if you're lucky, give you an error, or it may just crash. It shouldn't crash the entire system; the kernel should intelligent enough to make a user-space application crash, rather than refusing to allocate memory to a kernel process. What a particular application does when it hits the RAM limit depends on application, however. Some exit nicely, some crash in nasty ways. (This author is not altogether convinced that the netBook's kernel is completely intelligent in this matter...)

The main problem is that compactflash cards have only a limited number of writes. If swapspace is formally defined on the compactflash card, it will be used extensively by the linux kernel and may well cause that area of the compactflash card to fail at the hardware level. Formally defining a partition on the compactflash card to be swap space is not recommended for this reason. Note that the number of writes is ca. 100,000, and the disk controller is smart enough to keep rotating where the writes occur, so that it usually takes a long, long time for the card to "wear out". Microdrives are not subject to this issue.

There are, however, various strategies for enabling swap space. One approach is to make a file of size desired for swap space and then format it for swap. This file can then be mounted and used from time-to-time as necessary. The procedures to do this are:

1. Create a contiguous file of zeros by: "dd if=/dev/zero of=/somewhere/swapfilename bs=512 count=16800" which makes a ca. 8 MB file of zeros (adjust the size as needed; the real required size is defined by the most memory you'll need for the applications you want to run. The size of your RAM + Swap should be slightly more than this.)
2. Format the swap file to be used: "mkswap /somewhere/swapfilename"
3. Mount the new swap space: "swapon /somewhere/swapfilename"; your system now has 8 MB of swapspace available to it. Do a "free" to verify the system is accessing the swap space.
4. And after you are done using the swap space, turn it off: "swapoff /somewhere/swapfilename"

Another suggestion is to format an old, smaller compactflash card for swap, and use it in the pcmcia slot as needed. We all have various old cards lying around! This is a sacraficial use of an old, small card that would otherwise not be used.

Similarly, and perhaps optimally, an external laptop hard drive can be plugged into the pcmcia slot, and swap space, or even a swap partition on it, can then be used on it. In this case there is effectively all the swap space you want and no danger of wearing out the external disk.

5.5. Install the new system on the Compactflash Card

The Debian linux system is just a tar ball that should be unpacked onto the ext2 or ext3 filesystem of the second partition. First mount the compactflash partition on the desktop. Then unpack the system tar ball to it, "tar xfz system.tgz -C /mnt/disk2", if the CF ext2 partition is mounted to /mnt/disk2. That should be all that is required, although you can then customize whatever aspects you'd like on the CF system from the desktop, before booting it on the netBook.

5.6. Alternate Installation: An OS.IMG for linux

It is now possible to install a custom OS.IMG file that will boot the netbook directly to linux. For this system, you just need the OS.img file copied to the first fat16 partition of your compactflash card, and the linux system installed on the second ext2 partition of the compactflash card. For now, you can obtain an OS.IMG file from Psilinux Downloads (https://sourceforge.net/project/showfiles.php?group_id=8846) (originally from Debian linux on netbook (http://staff.washington.edu/dushaw/psion/openpsion/)). Repartition the CF and install the linux filesystem as described above. Then remove the power and batteries from your netBook for a moment. When you power up again, the new OS.IMG should boot, mount your linux filesystem, and start linux...no fuss, no muss! The kernel, initrd and Arlo can be dispensed with using this approach.

5.7. Appendix: The /linuxrc script

I have used the following minimal script (on the initrd filesytem) to mount the CF partition, transfer root to it, and then start /sbin/init from the new root:

```
#!/bin/sh
#
# /linuxrc - run once at boot time

echo "Set the path" > /dev/console
# Set the path
PATH=/bin:/sbin:/usr/bin:/usr/sbin
export PATH

# mount proc filesystem
mount -t proc /proc /proc

depmod -a

# the -o option causes cardmgr to run in the foreground
# and then exit when it is done configuring the disk.
cardmgr -o

echo "Mounting /dev/hda2 to /mnt/disk" > /dev/console
mount /dev/hda2 /mnt/disk

/bin/mount -o remount,ro /mnt/disk

echo "Executing pivot_root" > /dev/console
cd /mnt/disk
/sbin/pivot_root . old

exec /usr/sbin/chroot . /sbin/init <dev/console >dev/console

# The new system is now mounted as "/", with the old root system mounted
# to /old.  To free up the initrd memory, the startup has to umount /old/proc,
# and then umount /old.
```

6. Using Bookboot - the Linux OS.img

"Bookboot" can be used to construct OS.img files for the netBook so that linux can be booted directly, avoiding Arlo and EPOC altogether. This approach seems to work fine, and it is the preferred way to boot linux these days. Bookboot works so well, that no additional development seems to be required. Book boot can be downloaded from here: bookboot (http://linux-7110.sourceforge.net/files/People/Klaasjan/netbook/). There is no web page specifically describing bookboot yet (perhaps it is the one you are reading now), but you can certainly download it. Test OS.img's are now available for downloading; they can boot to an initrd, and even start up the compactflash drive. Bookboot has now been tested on both malaybook and netbook with various memory configurations, and seems to perform quite well in all cases.

There may be a display issue when booting through bookboot: only linux-2.4.26-vrs1-kvd2 and later set the proper registers for the LCD to work.

One advantage of bookboot is that the vfat partition size required for its OS.img file can be rather small at about 2-3 MB (perhaps larger depending on the size of the initrd system within the OS.img file). This means that more of your compactflash disk can be devoted to the linux system disk and the system boots up faster. The OS.img approach does not suffer the same 4 MB limit that Arlo does.

6.1. Making your own OS.img

Bookboot requires as input a working linux kernel (zImage) and (optionally) an initrd.gz. If you want bookboot to boot to a compactflash drive, an initrd.gz with the cardmgr utility and ide-cs module is essential at the moment. Bookboot is used by first editing a file "cfg.pl" to set your own parameters, putting the kernel image to "zImage" and the initrd to "initrd.gz", and then running "make". This runs the perl script "bookglue.pl" that puts all the pieces together and creates the OS.img file.

You need to set the memory parameters. The preferred method is to just use "$memdetect = "auto";", which should autodetect the available memory and send that information to the kernel (see more below). For this to work, your kernel must have been compiled with the default memory option (16MB). You can also set the memory manually by

```
$memsize = 32; # memory size in MB  (possible values: 32, 48 and 64)
```

Another configuration is:

```
$cmdline = "console=ttySA0 video=sa1100 root=/dev/ram0 init=/linuxrc rw"
```

where the init=/linuxrc will start the linuxrc script on the initrd root filesystem which will transfer the root system over to the compactflash root system. The "console=ttySA0" will make the serial port be the system console - this is optional, and perhaps not desirable for an operational system.

You can also add the option "ramdisk_size=16384" to change the default ramdisk size to 16MB (or what ever size you desire).

The netBook's firmware loads the os.img data into the second 16MB bank, which is probably why 32MB is the smallest amount of memory for any netbook. The os.img file should always be less than 16MB because of the 16MB bank size. Bookboot relies on this behavior by the firmware.

6.2. Memory Configuration

The bookbook + kernel memory configuraton uses the standard taglist kernel interface: it appends ATAG_MEMs to to the existing list. So you shouldn't need a patch, but the kernel needs to be willing to accept the memory info from the taglist. For this, the memory config should _not_ be hardwired into the kernel. When compling a new kernel, the 'default' memory configuration (giving you 16MB unless additional information is supplied) should be selected. The '32MB' and '64MB' kernels have their memory configuration hardwired, and so any bookboot options regarding memory will be ignored.

7. Using ArLo - Booting Linux from EPOC

7.1. Introduction/Example Configuration File

ArLo (http://www.yipton.demon.co.uk/content.html#ArLo) is used to boot to linux after EPOC has been started. Note, however, that Bookboot described in the previous section seems to be the preferred booting approach. ArLo is installed by unzipping its distribution files onto the D:\ drive of the compactflash card, the EPOC partition. The ArLo distribution can be obtained from the Openpsion file releases (https://sourceforge.net/project/showfiles.php?group_id=8846). ArLo comes with a variety of elements, the one requiring user configuration is the arlo.cfg file. For the netBook, the option "*machine = series 7*" is used to tell ArLo the machine type; note the space between "series" and "7". You can use the arlo.cfg file below as a guide of how to configure ArLo. More about ArLo on the Series 7/netBook can be found at Peter van Sebille's Place (http://www.yipton.demon.co.uk/content.html#ArLo) and a manual (http://www.yipton.demon.co.uk/arlo/latest/readme.html) for ArLo is available. Note that the Arlo GUI configuration save will not save the "machine = series 7" option, so the arlo.cfg file MUST be edited manually with a text editor.

The arlo.cfg file has an option "boot = init=/linuxrc" which will cause linux to first run "/linuxrc" on the initrd, and this script is normally used to take care of mounting the compactflash disk and transferring the root filesystem to it. A failsafe option is "boot = init=/bin/ash" (if you have /bin/ash on your system) which will merely start an ash shell (that you can't exit from without a system crash...) and do nothing else.

There are a number of other options described in the example arlo.cfg file.

Here is an example ARLO.CFG file:

```
# This configuration file will work for the Psion netBook.
# You must edit this file with a text editor to have "series 7" option saved properly.
#

[EPOC]
# The text within brackets "[XXX]" just defines a name.
# This "null" option boots back to ordinary EPOC.

[linux]
  image = D:\linux.image
  boot = root=/dev/hda2
# This option will not work at present, because we have no way of mounting /dev/hda2
# without using the PCMCIA modules.  So an initrd must first be used to mount the compactflash
# system and transfer root to it.

[linux initrd]
  image = D:\linux.image
  initrd = D:\initrd.gz
# Tells ArLo where to find the initrd.gz filesystem in EPOC.
# This will be the root filesystem, kept in memory.
  boot = init=/bin/ash
# * The "init=/bin/ash" option is for initrd's that do not have an init/sysvinit set up.
#     It starts an ash shell as the initial program, and does nothing else.
#     (change to /bin/bash, etc. if you like)
#     Omit this if your initrd has a proper /sbin/init.
# * For compactflash support, use the boot option "init=/linuxrc" , or something similar,
#    where /linuxrc is a script that will initialize and mount the compactflash partition.
```

```
# * To have the system console be dumped down the serial line, add the boot option "console=ttySA0".
# * To enable ramdisk size's to be XXXXX KB in size, add the boot option "ramdisk_size=XXXXX".
#    This option seems to work sometimes, and not others.
  machine = series 7
# note the space between "series" and "7"!!! :)

#
# General options
#
default = linux initrd
# Tells ArLo that the default option is "linux initrd" - the
#  option with this name will start by default.
#
```

7.2. ArLo Kernel Memory Configuration Options

It is possible (through any boot method, including ArLo) to specify the memory config via options to the "boot=" commandline.

1. For 32MB this should be: *mem=16MB@0xc0000000 mem=16M@0xc8000000*
2. For 64MB this should be: *mem=16MB@0xc0000000 mem=16M@0xc8000000 mem=16MB@0xd0000000 mem=16M@0xd8000000*

Once again, the kernel will only use this information if the memory config is not hardwired; the 16MB kernels fulfill this criterion, while the 32MB and 64MB do not. Kernels booted through ArLo should use the 'default' memory config [16 MB] at compile time, and supply one of the above commandline items in the arlo.cfg file, depending on if you have a 32MB or 64MB system.

So for example a boot option to start a /linuxrc script on an initrd, configure 32 MB of system memory, and start the system console on the serial line, would be:

```
boot = init=/linuxrc mem=16MB@0xc0000000 mem=16M@0xc8000000 console=ttySA0
```

7.3. Howto Boot ArLo First in EPOC

To boot up ArLo as the first thing to start after resetting your netBook, you can define a plain text file called D:\System\Data\wsini.ini containing the following lines (modified from the default Z:\SYSTEM\DATA\WSINI.INI):

```
BASERGB 255,255,255
PALETTE 0,85,150,255
AUTOCLEAR 1
SHELLCMD \System\Apps\Shell\Shell.APP
STARTUP D:\ARLO\ARLOSH.EXE
WINDOWMODE COLOR256
SCR_WIDTH1 640
```

```
SCR_HEIGHT1 480
SCR_WIDTH2 480
SCR_HEIGHT2 640
```

This will cause EPOC to startup ARLOSH.EXE at boot up first, giving the option of continuing to boot up EPOC or boot to linux. [This looks very much like the LILO prompt in an ordinary linuxbox.] The advantage here is that the start up time is much faster, since all of the EPOC programs don't have to startup before you get to ArLo.

7.4. ArLo Memory Limitations

Arlo has a design limitation that prevents initrd files that are larger than about 3.5-4 MB from loading properly. The size of the initrd.gz file must be less than this to work, even though the filesystem itself (once loaded in linux) can be any size. The number of applications that can be put on a working initrd filesystem is therefore greatly constrained. There has not been much progress on reworking Arlo (and boot2nd) to allow for larger initrd's; with compactflash support the need is not really there (though it would be nice to have larger initrd's). Technical details from Klaasjan:

```
Arlo and boot2nd assume that the memory_map is limited to one page (4096
bytes). This memory_map contains word-size (4 bytes) entries of each page
that needs to be relocated.  So the memory_map can contain 1024 entries,
each pointing to a 4kB page, allowing for 4MB total space for everything
needed in the second stage of the boot process: bootn2d, its parameters,
kernel, and initrd.  Since the compressed kernel is about 0.5MB (while
boot2nd and parameters only take a few pages), you're left with about
3.5MB for the initrd.
```

So redesigning Arlo and boot 2nd for larger initrd's will take a bit of doing. I suspect that this is not likely to be developed any further for the forseeable future.

8. X Windows

8.1. Introduction

X Windows seems to be working fine. The touch screen is also now supported, but it probably needs a bit more development. Touch screen will sometimes just stop working for no apparent reason, requiring a reboot to fix, and it will behave erratically all too often. You can use a mouse with the netBook using the onboard or PCMCIA serial port. There is a Using a Mouse section in this HOWTO about how to do that.

See the section on Making Presentations Using a VGA-out PCMCIA Card in this HOWTO for an alternate X server that can drive a PCMCIA VGA out cards (Colographic Voyager; Hewlet-Packard F1252A). The PCMCIA VGA out cards are supported by only the server xserver-tiny-h3600_4.2.20030126-14_arm.ipk from handhelds.org, although that server does not support touch screen.

8.2. A Working X Server

At this point, for overall usefulness, including touch screen support, the ordinary XFree86 server (framebuffer) from Debian seems to be the most functional. In addition to the VGA out X server mentioned above, the chapter at the end of this section describes other X servers that do not work so well.

For installing X, if you have a Debian system you can just install the appropriate *.deb packages for XFree86. If you have a handhelds.org system (ipk/Xipaq), you may have to kludge the X server installation. I kept installing required libraries from handhelds.org until it worked. Giving me a potentially awkward mixed handhelds/Debian X server system.

Only the XFree86 X server from Debian uses the /etc/X11/XF86Config file; all the others get the information they need from the command line or by auto probing. The touch screen support therefore works only for the Debian XFree86 X server at present.

XFree86 from Debian.org is the the ordinary X server with fbdev driver. One nice thing about this server, is that it has built in keyboard extensions let you use your keyboard to control the mouse; see the section on "Keys" below. Of the available X server binaries, the Debian server has the largest memory footprint at about 4 MB; the others are around 2 MB. With the Debian server, you can rotate the X windows screen by 90 degrees by adding

```
Option          "Rotate" "CW"
```

to the "Device" Section of the XF86Config (XF86Config-4) file and restarting X windows. A rotated screen can be useful for some applications, e.g., some games. The XF86Config configuration file can be set up to support a serial mouse on a PCMCIA serial card, as well as touchscreen; the XF86Config example given here is set up for a mouse.

The Voyager VGA-out card is not supported at all by the Debian X server.

8.3. Touch Screen

To get the touch screen to work, you first need a kernel with the latest patch to support the touch screen. For touchscreen and X windows, you will have to use the ordinary XFree86 fbdev X server. The touchscreen driver is known to have a number of lingering problems, e.g., erratic pointer behaviour is altogther too common; so sorry.

The touchscreen driver uses the Summa protocol in absolute mode (or enough of the protocol to get gpm to work), and dynamically allocates a character device major number (probably 254). This number is given at boot up time; this major number can be found by:

```
dmesg | grep touchscreen
```

The major number will likely be 254. Adjust the following if you don’t get 254.

You’ll need to make a character device in your filesystem that connects to the driver. I’ll call it "tscreen" here to be more descriptive.

```
mknod /dev/tscreen c 254 0
```

Then GPM or the X server should be able to "connect" to this device.

In the /etc/X11/XF86Config file you’ll need to have sections like:

```
Section "InputDevice"
    Identifier  "Touchscreen"
    Driver      "summa"
```

```
    Option      "Device" "/dev/tscreen" # or whatever
    Option      "Mode" "absolute"
# Other options that might be useful:
#    Option      "Cursor" "stylus"
#    Compatible (driver is now compatible enough to not need this),
#    Resolution, XSize, YSize, XOffset, YOffset
EndSection
```

And an addition to "ServerLayout":

```
Section "ServerLayout"
    # Other things here
    # .
    # .
    InputDevice    "Touchscreen" "CorePointer"
EndSection
```

To use the touchscreen, with gpm:

```
gpm -t summa -m /dev/tscreen
```

The position of the pointer on the touch screen can be calibrated by adjusting the values in the files in /proc/netbook/touchscreen. The values of min-x, max-x, min-y, and max-y can be adjusted prior to starting X11:

```
min-x - calibration along the left edge of the screen. (default 375).
If the mouse pointer is being draw too far to the right when near the
left edge of the screen, then you should increase this value.

min-y - calibration along the top edge of the screen (default 255).
If the mouse pointer is being draw too low here, then you should
increase this value.

max-x - calibration along the right edge of the screen (default 3575).
If the mouse pointer is being draw too far to the right here, then
you should increase this value.

max-y - calibration along the bottom edge of the screen (default 3750).
I'll leave you to work this one out ;-)

A hint that might save you some time: The two min- values can be
changed while X is running - note that doing this will throw the
calibration off for the rest of the screen until you restart X, but
when you restart, the top/left edge calibration will be as you have
just set them, and the X-server will re-read the bottom/right
calibration from the driver, so these will be restored to what they
were like before you started fiddling.  Unfortunately to get the max
values to have an effect you will need to restart the X server.

/proc/netbook/touchscreen/offset-x was added to expand the active
touchscreen to cover the icons down the left side of the screen.
```

```
offset-y was then added for orthogonality.

Unless you have some purpose in mind, it's probably best to leave both
of these offsets as zero (which means that touching the icons down the
left of the screen will send a pen event on the left edge of the LCD,
and hence to the right of the icons).  If you do change these offsets,
then it's better to already have a set of good calibration values,
since calibrating with nonzero offsets is rather fiddly.

To use, you'll need to set the XOffset value in the Touchscreen
section in XF86Config-4 - I did this very roughly with

   Option "XOffset" "0.5"

then setting the value with

   echo 260 > /proc/netbook/touchscreen/offset-x
```

On my malaybook I needed: min-x = 375, max-x = 3490, min-y = 295, and max-y = 3750 for a reasonable calibration. The values can be changed with commands like

```
"echo 375 > /proc/netbook/touchscreen/min-x"
```

Once you've calibrated your netBook, you can add these commands to a small script in the /etc/rc2.d, so that your netBook will be properly calibrated after each boot up.

The value in the file /proc/netbook/touchscreen/buttons can also be set to 1, 2, 3 (or 4) in a similar way, which will set the default mouse button that is clicked when the screen is tapped.

The keylaunch program, described below can be used to toggle the pointer button when the screen is tapped.

You can use a nice GUI application called XRmouse http://projects.gnome.hu/xrmouse/index.en.html to control and set the default mouse button. This application is based on GTK+. The ARM binary of this application is located at http://projects.gnome.hu/xrmouse/XRmouse.

8.4. Using Keylaunch

You can use the "keylaunch" program, available as a Debian package, to launch applications or commands by keystrokes. In particular, set mouse clicks 1, 2, or 3 to be the screen touch. The X server or summa driver seems to support four pointer keys (I forget the details of this). A .keylaunchrc file with lines that looks like

```
key=.*.1:xmodmap -e "pointer = 1 2 3 4"
key=.*.2:xmodmap -e "pointer = 2 3 4 1"
key=.*.3:xmodmap -e "pointer = 3 4 1 2"
```

will work to make Ctrl-1 set the screen tap to be mouse button 1; Ctrl-2 set the screen tap to be mouse button 2; and Ctrl-3 set the screen tap to be mouse button 3. [See also "Keys" below"]

Add a line like

```
key=..*s:xwd -out screendump -root
```

and <Alt>-<s> will take a snapshot of the root X window and save it to the file screendump. Any other applications or commands can be implemented; you get the idea.

8.5. X modules

X comes with a number of "bells and whistles" modules. You can exclude a number of these by appropriate entries in the X configuration which seems to save considerable memory. Which modules to leave out, I could not say, however - I am not altogether sure which ones are very useful and which ones can be left out.

8.6. Keys

The netBook's keys are not entirely defined according to any standard keyboard, so some care is needed to define the various keys properly. In addition with the ordinary XFree86 framebuffer X server from Debian, the "X Keyboard (XKB) Extension" can be used to control mouse functions. TinyX does not enable this extension, as far as I know. To set up all the keys and define the mouse functions, I have for my .Xmodmap file:

```
keycode 9 = Escape Escape KP_5
keycode 10 = 1 exclam KP_Divide
keycode 11 = 2 quotedbl KP_Multiply
keycode 12 = 3 sterling KP_Subtract
keycode 13 = 4 dollar EuroSign
keycode 14 = 5 percent grave
keycode 15 = 6 asciicircum apostrophe
keycode 16 = 7 ampersand braceleft
keycode 17 = 8 asterisk braceright
keycode 18 = 9 parenleft bracketleft
keycode 19 = 0 parenright bracketright
keycode 20 = minus underscore numbersign
keycode 28 = t T bar
keycode 39 = s
keycode 33 = p P Pointer_EnableKeys
keycode 55 = v
keycode 65 = space
keycode 66 = Alt_L
keycode 67 = Mode_switch Num_Lock
keycode 80 = Up KP_Up Prior
keycode 88 = Down KP_Down Next
keycode 85 = Right KP_Right End
keycode 83 = Left KP_Left Home
```

so that Fn-p will "Pointer_EnableKeys". This means that, for example, KP_5 (keypad 5, Fn-Esc here) will act as a mouse click. So with this .Xmodmap file (after Fn-p is pressed), Fn-Escape will act as a mouse click, Shift-arrows will act to move the mouse, and In addition, Shift-Menu will enable NumLock so that just the arrows will move the mouse around, and Shift-arrows will resize the windows. You get the idea.

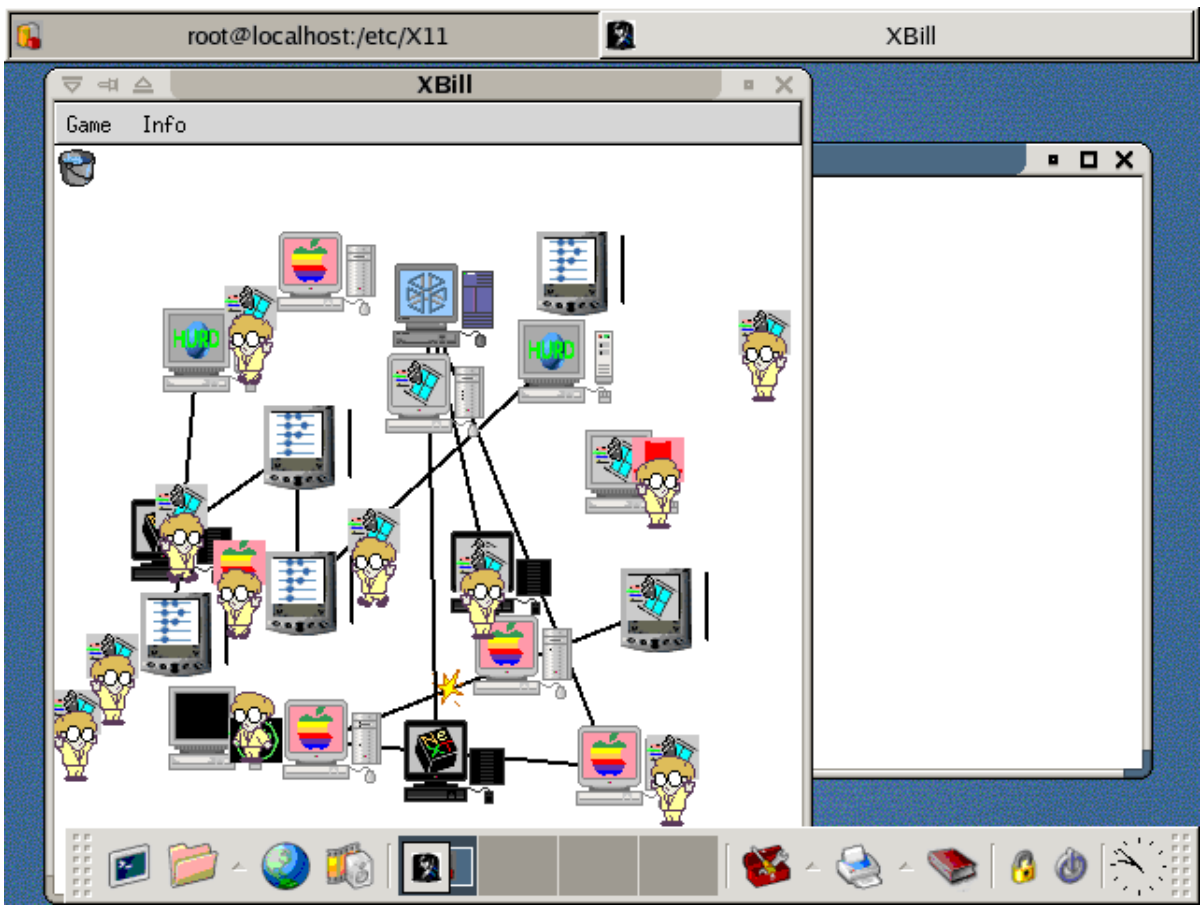
Incidentally, in console mode the exterior button (we'll call it Ext) of the netBook (in front) is defined as F1. Then Shift-Ext is F2, Ctrl-Ext is F3, Menu-Ext is F4, and Fn-Ext is F5. These are probably not defined in X windows without some configuration to the .Xmodmap. The power switch just gives "p" for now (for "power", get it?) until we figure out how to turn the netBook off.

8.7. X Window Managers

There are several possible lightweight window managers to try out. Blackbox, Afterstep, FVWM, WindowMaker, ICEWM, and XFCE4, are all possibilities. Which one works best on the netBook with 8-bit/256 colors is not certain yet. You can see a set of performance benchmarks for various window managers here (http://www.windowmaker.org/features-performance.html). I like XFCE4, but it appears to be a little too demanding of the netBook's resources for comfort. FVWM has special keys inherent in it that allow the mouse to be controled by the keyboard, even if you don't have the "X Keyboard (XKB) Extension" described above. All in all, the WindowMaker manager seems to work pretty well. Some people like the simpler Blackbox (or Fluxbox) manager because it uses fewer colors and applications look better on it; it also uses a little less cpu resources than WindowMaker.

The autorepeat key problem in WindowMaker can be solved by adding "xset r rate 500 50" to your .xinitrc file. The delay (500) and interval (50) rates, usually set in the XF86Config file, don't seem to get through to WindowMaker until you execute this command.

Figure 2. XFCE4 (http://xfce.org/index.php) on a 640x480 screen, 256 colors



8.8. Alternate Window Systems - Microwindows, Opie, Picogui, Minigui

There are a number of interesting light-weight alternatives to X windows, some may work o.k., some not so o.k.. I've not tried any of these.

Microwindows. Packages available from Debian.org I don't know how this system works, but I think it is meant to be an X windows compatible windowing system.

Qtopia or Opie. Opie, e.g., from OpenEmbedded, may have to be modified and recompiled for 8-bit (256 colors) color. No success yet with Opie, although people are poking around with this, e.g., see: this link (http://wickedpsyched.net/opie.html). This is probably the most advanced windowing system alternative suitable for PDAs. Opie may be much more suited to the end user market, seems to me.

Picogui. Picogui at http://picogui.org/ has been developing slowly. It has a limited number of applications that run on it.

Minigui. One interesting possibility is minigui at http://www.minigui.org/ This package is being actively developed in Beijing, China.

8.9. Debugging Notes

One potential problem, which perhaps still lingers for some X servers, although not the ordinary X server from Debian, has to do with the LCD controller and the fact that X windows will sometimes start up with very odd, often unworkable, colors (generally dark blue). If this happens, you can probably start X with the correct colors by ssh or rsh'ing to localhost before starting the X server. Apparently being on a pseudoterminal rather than a virtual terminal can resolve some issues. It may be that the kernel LCD driver is not quite 100% with X windows.

Sometimes the netBook gets stuck with its blank screen and unresponsive keyboard when X is started. The screen/keyboard can be restored by ssh'ing into the netBook from the desktop and starting X windows again - a hard reboot is not required. Similarly, if X windows crashes, you'll still see what appears to be X windows, but you'll have an unresponsive keyboard. Log in by ssh and restart X.

8.10. Alternate (not altogether working) X Servers

There are several other X servers that you can try with your netBook. All of them seem to behave a little differently, some work better than others, some have very useful features, but other drawbacks. TinyX/kdrive find the mouse by following where the /dev/mouse link takes them (if it exists) and then auto probing the mouse type. This means, among other things, that touchscreen is not supported by these drivers since the summa touchscreen is not set up properly.

In some cases, the only servers that worked with proper colors used Taneli's fb patch that sets cmap_static=1 in the kernel framebuffer driver. In the case of Debian's ordinary XFree86 server, the cmap_static flag is not required, which is a good thing.

If you have a handhelds.org system, you may have to kludge the X server installation. I kept installing required libraries from handhelds.org until it worked. Giving me a mixed handhelds/Debian X server system.

Here is a list of other servers I have tried, and some comments on what happened for me:

X server #1: TinyX/kdrive from http://ipkgfind.handhelds.org/ extracted from xserver-kdrive-fbdev_6.6.1-14_arm.ipk It does not recognize or drive the Voyager VGA card, although it has the "-card pcmcia" option.

X server #2: The kdrive server from http://www.netbook-linux.org/packages/latest/ipk/ extracted from xserver-kdrive-fbdev_0.1cvs20040706-r0_arm.ipk This server also worked, but inexplicitly gave a completely black screen (black screen, black font?) for the rxvt terminal; the terminal was working however, it just didn't show anything. This server seemed to offer a nicer color rendering, however. But because of the rxvt issue, I can't use this X server at the moment. It does not recognize or drive the Voyager VGA card, although it has the "-card pcmcia" option.

X server #3: Xipaq from http://www.fairlite.demon.co.uk/handhelds.html Download Xipaq.gz Does not work for X windows on the netBook - a blank screen. BUT: It gives scrambled output to my monitor using the Voyager VGA card. It is obviously trying to do something.

9. Optimization and Adjustments

There are a number of ajustments that may or may not be possible to the netBook to optimize performance of one sort or another. This section discusses the various possibilities, although many of them do not work quite yet.

9.1. Settings in /proc/

The /proc directory contains files or directories that allow various settings to be made for the netBook.

9.1.1. /proc/sys/vm/laptop_mode

If set to 1 (Set it by merely: "echo 1 > /proc/sys/vm/laptop_mode") the disks will be written to less often, apparently. What this really means for the netBook is not quite understood, but it seems like it might be a good thing to set.

9.1.2. /proc/sys/cpu/0/

Has speed, speed-max, and speed-min which represent a possible control of cpu clock speed. However, these options do not seem to be able to be adjusted at the moment. Apparently this option in /proc/... is deprecated now. In the 2.6 kernel /sysfs is used for this option, so perhaps it only works in the 2.6 kernels. The idea here is that the cpu clock speed can be adjusted on the fly to conserve energy or perhaps to overclock to maximize performance.

9.1.3. /proc/netbook/

Has many of the settings specific to the netBook, including touch screen settings as described in the section on X Windows. Those settings that can be changed, are changed by "echo VALUE > /proc/netbook/foo".

1. battery: tells whether a battery is present or not.
2. brightness: sets the screen brightness.

3. contrast: sets the screen contrast.
4. external-power: tell whether external power is present or not.
5. flash: ???
6. recharge: ???
7. touchscreen: this directory has adjustments for the touch screen, as described in the X windows section.

9.2. IDE Disk Performance

The utility "hdparm" can be used to adjust IDE performance, although at present no adjustments seem to be of much help. Check out hdparm and ide performance (http://gentoo-wiki.com/HOWTO_Use_hdparm_to_improve_IDE_device_performance) for a complete discussion. Note that the disk access is 16 bit IO only (-c0 for hdparm) because pcmcia is 16-bit pc card. No PIO or DMA modes seem to be possible at present; this seems to be a kernel development issue. Further, pcmcia disk access (which includes the compactflash IDE card) at present uses a slower polling method, rather than IRQ's. Timing disk access using, e.g., "hdparm -tT /dev/hda", the quickest reads I've been able to achieve are around 2 MB/s. "hdparm -u1 /dev/hda" might be helpful.

You can check the current various settings for your disk by executing "cat /proc/ide/hda/settings."

9.3. System and Hardware Clocks

Apparently these clocks are different, and the EPOC system could potentially access the hardware clock. However, reboots seem to set the hardware clock back to 1971. "hwclock --systohc" will set the system clock to the hardware clock.

10. Applications

Any number of applications of perhaps 100's of packages can be installed on the netBook to run in either a console or X-windows. Examples of these include octave (plotting with gnuplot; there are several versions: the 2.0.17 ipk, and the 2.0 and 2.1 versions as Debian packages. The 2.1 version takes up 20 MB of memory but is the most modern option. The 2.0.17 ipk version is more lightweight), midnight commander, gcc, perl, emacs, etc. Any of the *.ipk applications for the zaurus (downloaded from www.killefiz.de/zaurus/ (http://www.killefiz.de/zaurus/), or www.openzaurus.org (http://www.openzaurus.org), or ipkgFind at handhelds.org (http://ipkgfind.handhelds.org/), or Zaurus Software Index (http://zsi2.stonekeep.com/)) can be readily installed and should run. Any of the arm versions of the Debian *.deb packages (http://www.debian.org/distrib/packages) (ARM cpu version) should also work - either install apt-get, or more basically, dpkg, or even manually unpack the *.deb file and install the package by hand.

A table listing linux equivalents to EPOC applications, with links to some of those applications, can be found on the former openpsion wiki page (no longer a wiki) (http://linux-7110.sourceforge.net/oldwiki/EpocLinux.shtml).

A number of hard-to-find applications designed for minimal use of resources that have been compiled for the netBook can be found on the netbook applications wiki page (http://linux-7110.sourceforge.net/oldwiki/NetBookCompiledApplicatons.shtml).

10.1. Installing Packages

zaurus *.ipk applications can be easily installed by "ipkg install newpackage.ipk". The Debian packages are meant to be fully compatible with ipkg, so *.deb packages should be able to be installed with ipkg like anyother *.ipk... an e-mail list suggested that, anyways, though I did not find that to be true.

With a Debian-based system, dpkg can be used to install individual packages, but you will have to sort out and download all the package dependencies manually. dpkg is a low-level package management tool. A better tool is "apt-get", which will sort out and download all the dependencies (including the upgrades) for you. So install apt-get and don't worry and be happy.

10.2. Editors and Word Processors

Abiword is a fairly advanced word processor that will work on the netBook for simple documents. It is a little sluggish, but can work well with a methodical attitude. For more complicated documents, it rather overexerted the netBook's resources. Graphics can be inserted (png files worked for me.) Abiword has an RSS memory footprint of 11 MB, with 8.9 MB shared memory. Abiword has a plugin package that has, among other things, support for Psion Word file format.

SIAG (http://siag.nu/) has a writer that works o.k. called "Pathetic Writer". You can get binaries of the SIAG suite of programs compiled for the netBook here (http://staff.washington.edu/dushaw/psion/openpsion/#SIAG).

Lyx is another possible word processor, but Lyx requires the rather large tetex package and all that it requires.

gvim is also a nice option for a vi/text editor. After a 10 second startup, gvim works rather well for editing text files. gvim has an RSS memory footprint of 8.3 MB and a shared memory footprint of 6.0 MB. gvim has style formats that recognize gcc, fortran, octave, html, etc text files and gives them nice color styles with some code error debugging assistance.

10.3. Spreadsheets

There seem to be a dearth of reasonably functional spreadsheets in linux, alas. The standard linux spreadsheet is gnumeric, which will run fine on the netBook apparently. The downside of gnumeric is that it is a gnome application, so to install gnumeric takes a bit of disk space with all the gnome dependencies. When I installed it, it required some 90 MB. Gnumeric takes 25 s to startup, so it is a little slow, but it is a full fledged spreadsheet. Gnumeric is a little cramped on the netBook's 640x480 screen - there is no PDA option for gnumeric, as far as I know.

SIAG (http://siag.nu/) has a spreadsheet that works o.k. called "siag" ("Scheme In A Grid"). You can get binaries of the SIAG suite of programs compiled for the netBook here (http://staff.washington.edu/dushaw/psion/openpsion/#SIAG).

10.4. Web Browsers

In console mode, the "links"/"elinks" browsers work quite well. There are ipks and Debian packages for these that can be readily installed. This can be quite useful for searching for and downloading software packages for your linbook. Newer versions of this browser have full graphics support (although these cannot deal with the netbook's framebuffer so graphics don't work in console mode).

There are a number of web browsers for X windows that might be suitable for the linbook. The three most promising are minimo (http://www.mozilla.org/projects/minimo/), dillo (http://www.dillo.org/), and firefox. IPKG packages of dillo and minimo for ARM are available, and both are under active development. There is also a standard Debian package for dillo. The minimo browser requires about 25 MB to run, but an incarnation suitable for the netBook does not seem to exist yet.

Dillo is not so fully functional as minimo, but it is much smaller and faster and is under active development. I use "dillo -geometry 580x440-10-10" to start dillo to work best on the netBook's screen, and I've set the panel_size to be tiny and small_icons to be YES in my ~/.dillo/dillorc file (see /usr/local/etc/dillorc). Dillo can also be compiled from source code on the linbook, if the various development packages are installed - get a tarball of a newer verson of dillo binary here (http://staff.washington.edu/dushaw/psion/openpsion/).

Figure 3. dillo (http://www.dillo.org) on the netBook's 640x480 screen, 256 colors

Most promisingly, the firefox browser, a standard Debian package, will also work on the linbook - a full fledged, modern browser. It is a little slow (55 seconds to startup), but fully functional. Firefox has a memory footprint of 31

MB RSS. Just install the Debian firefox package. Other browsers are available such as a PDA version of konqueror. Firefox version 1.5 has been compiled and optimized specifically for the netBook - configured for the more lightweight gtk version 1. This version is quicker and has smaller memory requirements than the standard firefox. Download a tarball of the binary from sourceforge Files (https://sourceforge.net/project/showfiles.php?group_id=8846). Note that when loading a webpage, the firefox browser has a "throbber" that is basically an animated gif file. This animated gif/throbber takes up cpu cycles on the netBook, so changing the throbber to be something other than an animated gif would seem to be helpful. See Customizing the Firefox Throbber (http://urbanmainframe.com/folders/blog/20041202/) for more information. I have set my throbber [no comments please] to be a green dot (netHOWTO/green.gif) when done, and a red dot (netHOWTO/red.gif) when loading a page.

Opie comes with a version of the Opera web browser that might work on a netBook running Opie. There are ipks of Opera that are available, having been extracted from the original Opie-based ROMs; See the links under Resources. Opie doesn't yet work on the netBook, however.

10.5. Calculators - RPN and otherwise

There are any number of calculators that will run on the netBook, some better some worse. They range from the full-blown mathematics package octave to the ordinary xcalc.

There are three reverse polish notation calculators to consider: "xcalc -rpn" will present an RPN calculator, calcoo is an RPN calculator, and gdcalc (http://bhepple.freeshell.org/dcalc/unix/). Of these I prefer gdcalc, which is based on the original Psion EPOC dcalc. xcalc -rpn is limited in features, while calcoo is a little better but takes quite a while to start up (and it is rather slow). Both xcalc and calcoo can be installed by Debian packages. I compiled gdcalc for the ARM cpu from the source code; you can download it, from here (http://staff.washington.edu/dushaw/psion/openpsion/) (0.3 MB). One disadvantage of gdcalc is that it requires a few of the older gnome libraries (which packages??). There is also a console verion of gdcalc, called dcalc (again) that works quite well on the netbook.

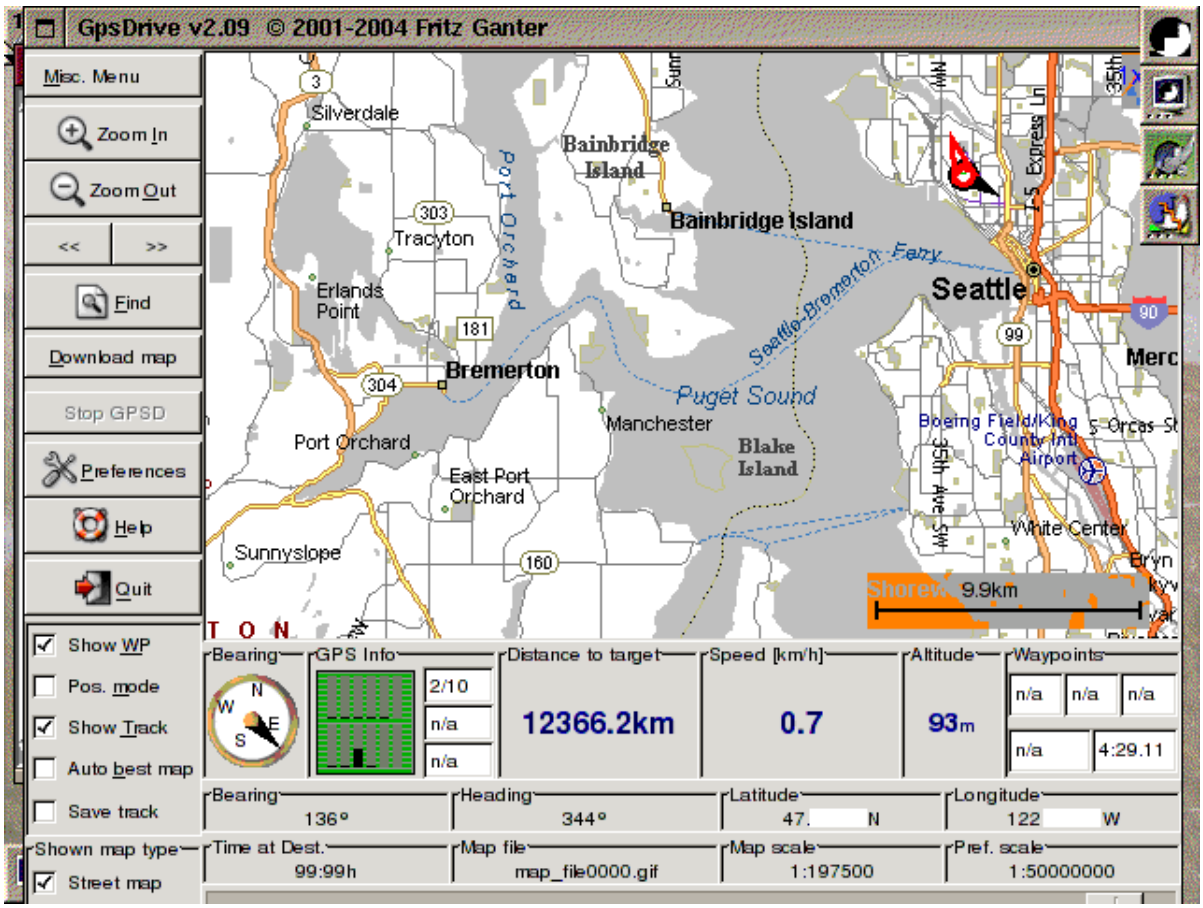
xcalc can be configured to present a nice color interface; see man xcalc. I am sure there are other "regular" calculators (I am mainly interested in the RPN.)

10.6. GPS on the netBook

A common GPS application for linux PDA's is gpsdrive at http://www.kraftvoll.at/software/. This web page provides ARM binaries (v. 1.32) that you can download, although there are more uptodate packages (v. 2.09) from Debian that are likely to work better. gpsdrive requires the GTK+ graphic toolkit. gpsdrive uses a small daemon "gpsd" that logs the GPS NMEA strings from the serial port and forwards them to gpsdrive, although the newer version is meant to read the serial port directly.

GPS can be set up using either an external handheld device such as the Magellan 315 connected to the serial port, or using a compactflash/PCMCIA GPS card such as the GlobalSat BC-307.

Figure 4. gpsdrive (http://www.kraftvoll.at/software/) on the netBook's 640x480 screen, 256 colors



10.7. Dictionary

The application "stardict" is a gtk-GUI-based dictionary that works fine on the netBook. You'll need to install the Debian packages, and then download the dictionaries your interested in from the stardict webpage (http://stardict.sourceforge.net/), where you can see screenshots as well. This application has translation dictionaries available as well.

10.8. Astronomical netBook

Xephem (http://www.clearskyinstitute.com/xephem/) is a complete, professional planetarium application that will run fine on the netBook. The binary for this, optimized and configured (more or less) for the netBook, can be found on the former applications wiki (http://www.openpsion.org/oldwiki/NetBookCompiledApplicatons.shtml). This

program can also be used to control telescopes through the serial port; there is no reason why the netbook would not also be capable of this.

Figure 5. xephem (http://www.clearskyinstitute.com/xephem/) on the netBook - Saturn's moons.

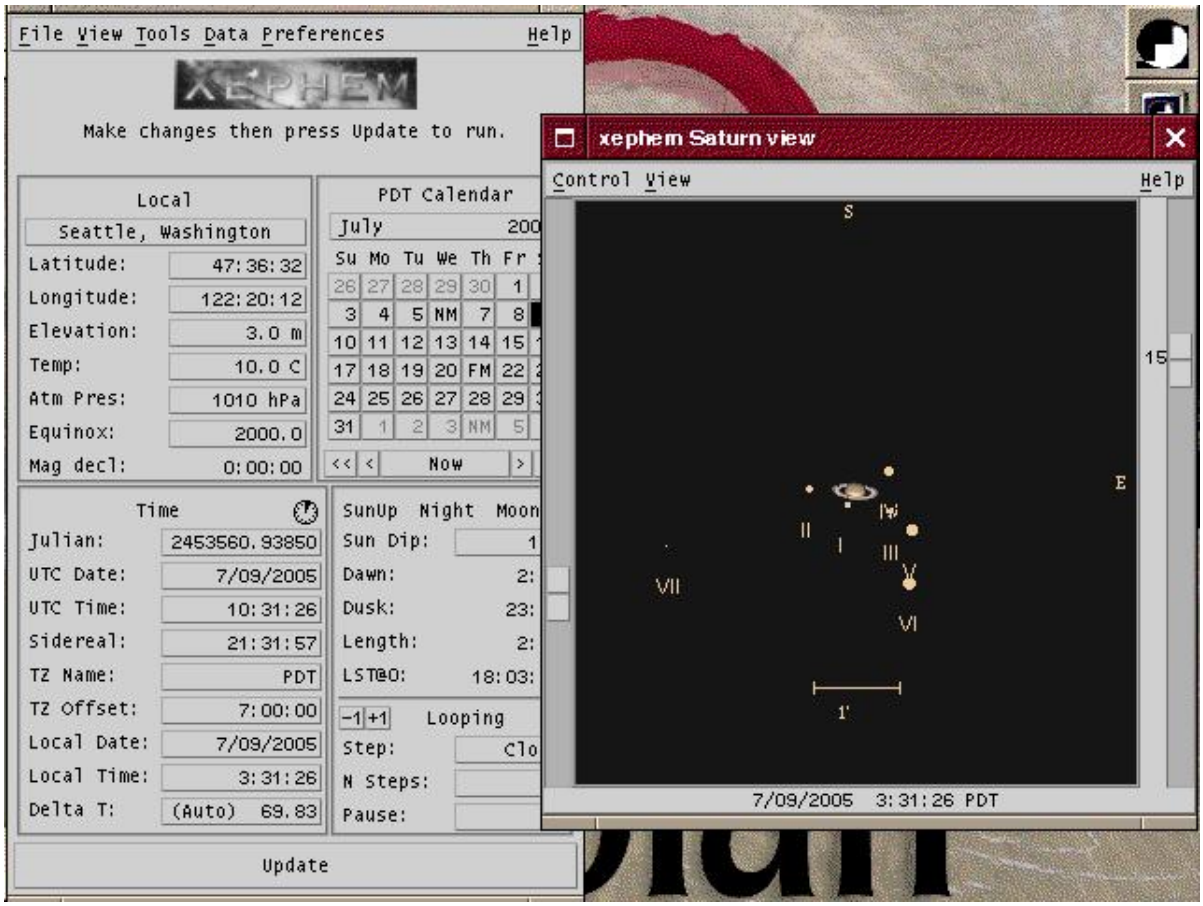
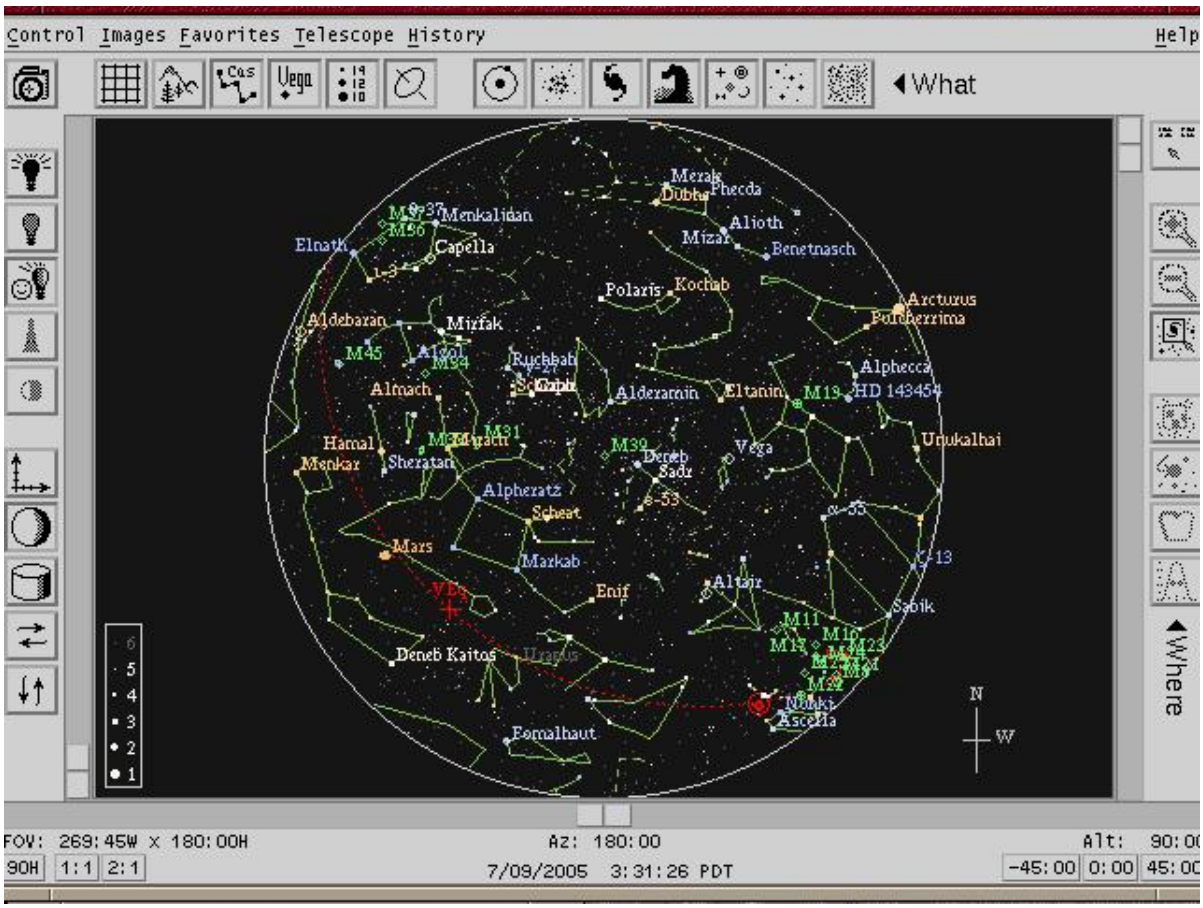


Figure 6. xephem () on the netBook - Skymap.



10.9. Games

There are many games that have been developed for linux, but unfortunately only a few of them will work on the netBook. Many of these games require 3D graphics, which the netBook lacks. Others require a screen larger than the netBook provides. The handhelds.org GPE package "gpe-games" comes with a number of games that will run on the netBook. xbill is one game that will work, of course; see the screenshot on the previous page. The classic arcade game xinvaders can be easily compiled and runs fine on the netBook - see the applications link in the introductory paragraph of this page.

One game that seems to work o.k. is xboing, that is available as a Debian package. xboing is rather too tall for the netBook's screen, however, but you can rotate the X windows screen by adding

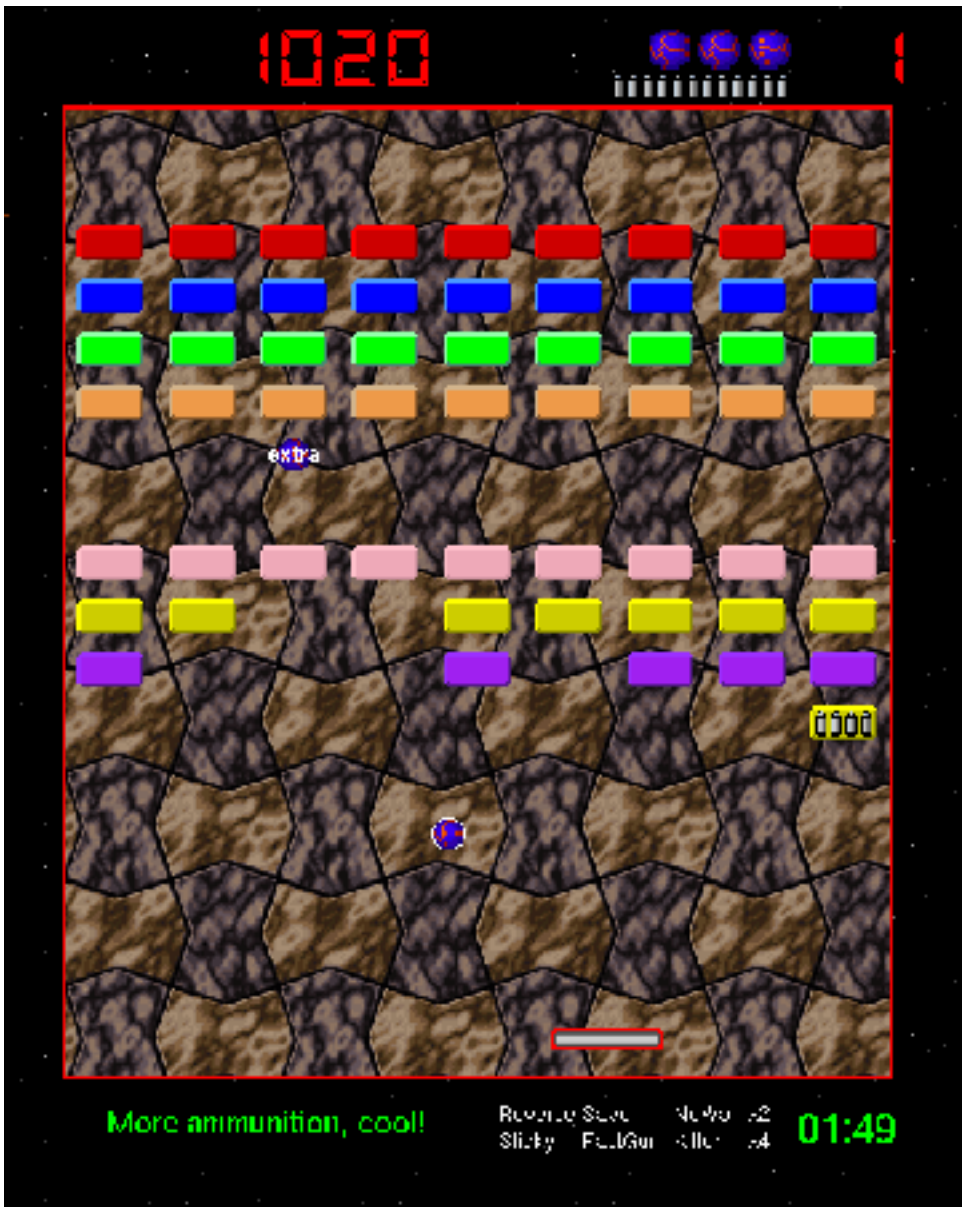
```
Option          "Rotate" "CW"
```

In the "Device" Section of the XF86Config file and restarting X windows. xboing then fits on the netBook's screen (barely); you then play the game by turning the netBook sideways. I found I could get it going and it will work fine with:

```
xboing -keys -no-sound -nosfx -speed 3 -no-usedefcmap -no-grab
```

xboing works best with a mouse, although it can be played with keys as well. This game requires 256 colors, apparently, so it won't work with, e.g., dillo running at the same time.

Figure 7. xboing screenshot - Level 1



10.10. Movie/Video Players

Apparently xine can be installed from the Debian packages. Only small mgp and mov files have been tried. aaxine and xine for x11 work, but fbxine does not. Of course, there is no sound yet on the netBook, but "xine test.mpg#noaudio" will start without complaining.

10.11. Compiling the Kernel or Applications with GCC

gcc can be installed on the netBook, but it takes considerable space (20 MB). gcc is verified to work, though you probably need a 64MB netBook because of its size and memory requirements. The option "-mtune=strongarm1100" seems to be a good one for optimizing for your sa1100 cpu. Try "-Os -mcpu=strongarm1100 -mtune=strongarm1100 -fomit-frame-pointer" for good optimization. The option "-Os" is essentially -O2, but avoids optimization that increase instruction size. This option may be better for the SA1100's small 16KB instruction cache. With -mtune=strongarm1100, the option -mcpu=strongarm1100 must also be given, or a "segmentation fault" may result. (Anyone know any other options that would make things go a little faster?)

I have installed development packages on my netBook and then used it to compile such packages as dillo, gpsdrive, and xzgv. There doesn't seem to be much to limit the ability of gcc to compile things, although it can be slow (gpsdrive takes 20 min to compile). The development packages require 100-200 MB of free space on your CF drive.

We could obviously use a better floating point emulator! (The "-msoft-float" option in gcc, and associated libraries installed, will likely yield executables that will do floating point calculations much faster. Soft-float, however, seems to require system libraries that are also compiled with soft-float - it is a tangled web of difficulty...) Benchmarks here do not scale well at all with the BogoMIPS number. See the section on floating point emulation towards the end of this HOWTO.

10.12. Web Servers

The Boa webserver (http://www.boa.org/) seems to be ideal for a small, efficient webserver. It is designed for embedded processors. There is a Debian package for it. It supports CGI.

Any of the webservers will run on the netBook; upto and including apache.

10.13. Misc Notes - Man Pages

"Man" pages do not usually come with *.ipk packages; the "man" system is rather large at over 1 MB. However, there is a package called "zman" at http://mulesex.port5.com/zman.shtml that uses a more compact system of compressed text files combined with "less". The groff pages of "man" are easily converted to "zman" text files.

10.14. Java

Four versions of java VM that have a port to linux/ARM are (in no particular order) Wonka (http://www.acunia.com/wonka/), Kaffe (http://www.kaffe.org/), SableVM (http://sablevm.org/news.html), and Blackdown (http://www.blackdown.org). Then there is apparently Sun's java for the Zaurus (http://developers.sun.com/techtopics/mobility/personal/articles/ztutorial/), apparently called Jeode. There are versions for ARM for all of these available, and ipk's are available for kaffe, Jeode and blackdown. deb's are available for kaffe and sablevm. See also JAVA Apps for (Linux) PDAs (http://tuxmobil.org/pda_linux_apps_java.html) at Tuxmobil for other java hints. Which one works best on the netBook? I dunno; none of them seem to work very well, if at all, at the moment. If you have some experiences, please post them on the openpsion mail list. Wonka segmentation faulted on my "testing" Debian system. Blackdown may eventually work (version 1.3.1 is compiled for ARM; I got a segmentation fault), but it is large (18 MB) and slow see an e-mail comment (http://www.mail-archive.com/java-linux@java.blackdown.org/msg15236.html) and another e-mail comment (http://grids.dbgrandi.org/java_issues/java_131_on_the_zaurus.html). SableVM started on my netbook but then never seemed to start anything; after using up cpu cycles for 10 min I killed it. Kaffe seems to work, but it will not run jar files. Jeode seems to be an Opie-only VM, while blackdown is X11 based.

We'd be happy to hear of any progress with java; please report to the openpsion mail list!

11. Serial Ports

The netBook's serial devices are /dev/ttySA0 and /dev/ttySA1; you can see these at boot up time. ttySA0 is wire, ttySA1 is irda. irda doesn't quite work yet, except I've not tried powering up irda in EPOC to see if a subsequent quick boot to linux won't give irda. PPP will work over the serial ports - see the chapter on PPP from the netBook in this HOWTO. Mice can also be used over the serial ports - see the chapter on Using a Mouse in this HOWTO.

Minicom can be used to poke and prod a modem or GPS serial port, such as might be on a PCMCIA card that you've just inserted. The modem should respond with OK to mincom's initialization sequence; raw NMEA strings should be seen from a GPS device.

This section is new and under construction.

11.1. Onboard Serial Port

You can install minicom to check that serial port is working and communicate with your desktop.

The netBook's serial port is meant to be a maxim MAX3243-RS chip, which is a complete, robust serial port.

11.2. Infrared Serial Port (Doesn't Work)

(DOESNT WORK YET) To enable the infrared connection, one executes irattach /dev/ttySA1 -s 1 on the netBook, and irattach /dev/ttyS1 -s 1 on the other machine (if ttyS1 was recognized at boot up of the notebook computer as the IR device). There is also likely to be a /etc/init.d/irda startup script on the notebook computer that you could use. The -s 1 flag enables system logging. Even though ttySA1 or ttyS1 are used to make the IR attachment, /dev/ircomm0 is still the serial port to use on either notebook computer or netBook. The command irdadump, which is a standard tool, can help in debugging the connection - it will show, at the lowest level, when the remote IR signal is picked up or

not. NOTE that the use of IR is meant to consume a considerable amount of power; turn it off by killing the irattach process when it is not in use.

11.3. PCMCIA Serial Port

You can buy PCMCIA serial ports (check e-bay; ca. $40). Such a serial port can be used with a mouse to add mouse capability to your netBook - see the section on the mouse in this HOWTO.

12. Using a Mouse

You can use a serial port mouse on your netBook in either the netBook's integrated serial port, or in the PCMCIA slot using a PCMCIA serial card. A mouse on the netBook has a much more definite, accurate feel to it than the touch screen. Bear in mind that you need a real serial port compatible mouse. In general, a ps/2 mouse with a ps/2 to serial port adapater will not work, unless the mouse specifically supports both ps/2 and serial modes. Generally, mice that support both modes come with the ps/2 to serial port adapter in the box.

12.1. A WARNING on using the netBook's onboard serial port with a mouse.

`CAUTION:`

The XTM people reported that they were able to use the onboard serial port to use a mouse in emulated windows 3.0 and their XTM machine in EPOC for a while...BUT then the mouse burned out the Psion's serial port. They were using a Series 5 or Series 5MX machine. So using a mouse on the netBook's serial port may be dangerous. A mouse would seem to draw more current than the onboard serial port can supply.

On the other hand, the netBook's serial chip is a maxim MAX3243-RS (MAX3221-MAX3243.pdf) (same as on the 5MX), which explicitly supports a serial mouse. Further, there seems to be a great variation among the mice in the current that they draw, and the exact mouse the XTM people used is unknown. My ps/2 optical mouse says on the bottom that it draws 100 mA at 5V, while my trackball mouse says it draws 10 mA at 5V. So it may be possible to choose a mouse carefully for its low power requirements and use it on the netBook's serial port. Then, perhaps the XTM people were using an older Series 5 and it may be that the netBook's serial port is a more robust. The exact limits are unknown, however.

So a bit of caution is in order when attempting to use a serial mouse, at least until we get more reports back from the frontlines.

12.2. Using the onboard serial port with a mouse

I have used a Logitech serial mouse in the netBook's onboard (/dev/ttySA0) port for sometime now without apparent harm. At the moment that is the sum total of the extent of our experience with a mouse in the onboard port.

The main issue with the onboard port is that a null modem adapter is required for the mouse to work properly. I have had success with both a handmade null modem adapter and an adapter I bought at Radio Shack. One issue is that the

mouse does not always seem to be detected, or does not always seem to communicate properly with whatever program is trying to access it. I've found that booting the netBook up, after a complete powerdown, with the mouse already attached will work reliably, or disconnecting and reconnecting the serial connection - at the end of the Psion's serial cable (rather than the end of the serial mouse cable) would sometimes get the mouse working. Both of these fixes suggest that the various null modem issues may be keeping a working serial connection from forming - "ring indicator", DCD, etc... In any case, be aware that a bit of fiddling may be in order - see Troubleshooting below.

12.3. Using a PCMCIA serial port with a mouse

At least one PCMCIA serial card has been shown to work in supporting a mouse on the netBook. A Socket (http://www.socketcom.com/product/serial.asp?Type=Single) R-I/O PCMCIA serial card (ruggedized) (ebay.com, about $30) is recognized as a 16C950 serial port at ttyS0. This card is advertised as being a very low-power card (5-15 mA) so it would seem to be a good choice for the netBook. (But, no doubt there are other brands equally suitable.) This is the current the card itself draws, not neccesarily the current the card can supply to a mouse. N.B.: One needs to set

```
SERIAL_OPTS="baud_base 460800"
```

in the /etc/pcmcia/serial.opts file to get linux to be able to set the correct baud rate for these particular PCMCIA serial cards.

The downside of using a mouse in the PCMCIA slot is that the slot is in use, so one cannot browse on wireless networking while using a mouse; it's one or the other. On the other hand, if you have an external modem, you can attach the modem to ttySA0 and then use your mouse in the PCMCIA slot - internet browsing with a mouse!

It is likely that the gpm daemon will operate the mouse in console mode, but this is at present untested.

12.4. Using a mouse with X windows

If you use the XFree86 X server, you'll have to edit your /etc/X11/XF86Config file to set up your mouse type and device (/dev/ttyS0 or /dev/ttySA0). If you use the TinyX or Xfbdev X servers, all you have to do is make a link from /dev/mouse to /dev/ttyS0 or /dev/ttySA0. These latter servers will automatically probe for a mouse at /dev/mouse and determine what type it is (use the -2bottons option to obtain 3 button emulation).

Once your X configuration is set up properly, plug your serial mouse into the PCMCIA card serial port, or the onboard serial port, and start X windows. You should be able to use your mouse then. Emulate 3 buttons is supported. The mouse should be plugged in before you start X windows, of course.

12.5. Mouse types to consider

The mouse that worked for me was a Logitech serial mouse, of undetermined model number.

I also used a Cirque glidepoint serial mouse that I got for $7 at the local used computer store. It's rather hard to find decent serial mice anymore! And I hate the traditional ball mouse. But, I'm not sure I like the glidepoint mouse. It's a small touch pad that operates as an ordinary serial "microsoft" mouse. In any case, you should try to find a mouse that draws a minimal amount of current to preserve your netBook's battery power. Then, some old serial mice are comparable in size to the netBook itself...

12.6. Mouse troubleshooting

Have you verified that your mouse is indeed an honest serial mouse? Most PS/2 mice will not work, even if you have an adapter. It is hard to find real serial mice these days - try e-bay or your local used computer store. In the latter, you will likely have to sort through a dusty box of jumbled mice to find one with the serial connector.

Try plugging your serial mouse into the serial port of a desktop computer to see if you can get it to work there.

If in X windows the mouse is erratic and bounces around, then either you've set the wrong mouse protocol, or the baud rate for the PCMCIA serial card is wrong (see the hint above about "baud_base").

With your mouse plugged in, start up minicom and set the port to the correct device and the baud rate to 1200 (and 8N1, which worked for me). Then, moving the mouse should show a stream of binary characters. Nothing will happen until you get stuff coming out of your serial port.

You need to have the correct XF86Config file in /etc/X11 for the mouse to work properly. This XF86Config (XF86Config-4) file worked for me.

The onboard serial port needs a null modem adapter, and as mentioned above the serial mouse can have a bit of difficulty getting a communication set up to the software. Try plugging the mouse into the serial port before you boot the linBook up, after a complete powerdown (reboot, and at the bootloader flash, do a Fn-Esc to turn the netbook off for a moment); that seems to work reliably for me anyways. Then, while in console mode (no X windows), the sequence of: (1) start minicom and check for a connection, (2) stop minicom, (3) unplug the mouse at the end of the Psion's serial cable (between the cable and the the null modem adapter) and replug it (i.e. break the connection for a moment), and (4) start minicom and check for the binary stream with mouse movement again. This sometimes seemed to work for me.

Once you see a stream of binary characters with mouse movement in minicom, the mouse should work fine when X windows is started.

12.7. Using a USB or bluetooth mouse

It may be possible to use a USB or bluetooth PCMCIA card together with a USB or bluetooth mouse. These configurations have not been tested or tried to the best of my knowledge, however. A 16-bit USB or bluetooth card would be required, which might be difficult to find; I know nothing of bluetooth. But there are apparently self-powered, external bluetooth devices that can plug into the serial port, to obtain a bluetooth connection through the serial port. I don't think 16-bit USB PCMCIA cards even exist.

13. PPP from the netBook

Read the Linux PPP howto: http://www.linux.org/docs/ldp/howto/PPP-HOWTO/index.html. When setting up PPP on the netBook, be sure that the ppp kernel modules are loaded and that any desktop firewalls are set up (or disabled) to allow the ppp connection. The ppp kernel modules are: ppp_generic, ppp_async, ppp_deflate, zlib_inflate, zlib_deflate.

The netBook's possible devices are /dev/ttySA0 (ordinary on-board serial), /dev/ttySA1 (on-board infrared serial), and /dev/ttyS0 (likely any serial port from PCMCIA, either modem, dedicated serial card, or GPS). Any one of which can be connected to a modem (except infrared for now, since that doesn't work yet).

13.1. Connecting to the desktop with PPP

Before beginning, note that most modern desktops have a networking firewall set up. Such a firewall needs to be either configured to allow the desktop to netBook connection, or disabled altogether (temporarily when it is necessary to connect the desktop to the netBook - a firewall is generally a good thing.) The symptoms of a firewall issue are being able to ping between netbook and desktop - an apparent healthy network connection -, but not able do make any kind of connection by tools such as ssh, rsh, or ftp.

To set up PPP, first configure something like this for the /etc/ppp/options file on the netBook:

```
-detach
defaultroute
noauth
nocrtscts
lock
lcp-echo-interval 5
lcp-echo-failure 3
/dev/ttySA0
115200
```

(or use /dev/ircomm0 for IRDA)

Then add the ppp user to your Psion's /etc/passwd file: echo "ppp:*:101:101:PPP User:/etc/ppp:/usr/sbin/pppd" >> /etc/passwd

And make sure pppd is executable for user ppp and executes as root: "chmod a+x /usr/sbin/pppd" and then suid root: "chmod +s /usr/sbin/pppd".

You will need to run a getty, e.g., on /dev/ttySA0, to enable logins to the Psion by ppp. The getty gives you the login prompt when you connect over the serial port. The command "getty 115200 /dev/ttySA0" - or /dev/ircomm0 - will start a getty on the serial port. It may be preferrable to start the getty on the serial port in the /etc/inittab file. After editting /etc/inittab, "init q" will restart init and get any getty going. You can test that the getty is working o.k. using minicom from the desktop.

On your desktop computer have something like this for a /etc/ppp/peers/psion file:

```
-detach
noauth
nocrtscts
lock
local
connect '/usr/sbin/chat -v -t3 ogin--ogin: ppp'
/dev/ttyS0
115200
192.168.1.100:192.168.1.101
```

Then just type "pppd call psion" on your computer to connect to the Psion. [You may prefer to start up the ppp connection from the psion, rather than from the desktop computer - but that might not be a security risk you want to take.] You should then be able to ping the desktop from the netBook, and vice-versa, and then use your favorite internet tools over this network.

Edit the /etc/hosts files on the desktop and netBook to include lines such as

```
192.168.1.101     openpsion
192.168.1.100     desktop
```

so the IP addresses don't have to be remembered.

To access the internet from your Psion through the desktop computer, you need to add routing, masquerading, and NAT [I don't know what these are either...but it works, so who am I to complain?] to your desktop:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o eth0 -j MASQUERADE
```

then you can browse the internet from your psion using dillo or lynx. Be aware that your desktop kernel must be compiled to support these things; default RedHat or SuSE kernels support this. Any firewalls must be disabled or otherwise set up to allow this forwarding. You will have to make an entry to /etc/resolv.conf to enter a proper domainname nameserver (you can use "nameserver 63.240.76.4" if you like, but you should really use an IP address provided by your internet provider).

13.2. Telnet/rsh/ssh

PPP can used with telnet, rsh, ssh, etc. rsh may be preferable to telnet, because then you can efficiently copy things between the psion and desktop using rcp. Install the rsh client and servers if you want rsh. rsh is preferable to ssh for psion-computer connections because it does not have the overhead of ssh encryption. You will likely have to edit /etc/hosts.allow to let in telnet, rsh or ssh connections.

NOTE: installing packages with shells, daemons and so on with dpkg, will sometimes make a directory /etc/pam.d; you are to delete this directory, because if it exists you will not be able to login!! Boot to single user mode if you forget, and delete the directory.

To get telnetd running, add the required entry to the /etc/passwd file: echo "telnetd:*:101:101::/usr/lib/telnetd:/bin/false" >> /etc/passwd

Then add the telnetd entry to the /etc/inetd.conf file:

```
echo "telnet stream tcp nowait telnetd.telnetd /usr/sbin/tcpd \
      /usr/sbin/in.telnetd" >> /etc/inetd.conf
```

Finally add your computer's PPP IP address to /etc/hosts on the Psion to make telnet login happen faster:

```
echo "# This makes the telnet login to psion faster
192.168.1.100 gateway" >> /etc/hosts
```

13.3. Connecting to the internet using an external modem

Either a standard external modem or most PCMCIA modems will work in the netBook. (Psion's own Dacom 56K+ethernet PCMCIA modems DON'T seem to work!) Bear in mind that if you use an external modem on the netBook's internal serial port, you will need to also use a null modem adapter (nullmodem.jpg), because the Psion's cable is a null-modem cable. You can find commercial versions of null modem adapters on-line (e.g., expansys.com), or at such places as Radio Shack.

AT PRESENT THE INFRARED SERIAL PORT IS NOT SUPPORTED - information below relating to infrared is kept here for future purposes. An external infrared modem (e.g., Psion's nifty external travel modem - the Diamond Mako travel modems are apparently identical to the Psion travel modems) will work for using your psion to connect to the internet over the telephone lines. You might be able to pick up an infrared Psion (or Diamond Mako) travel modem from ebay.com for US$55-US$70); it works just fine wit the linux netBook (probably) with very little fuss - relink /dev/modem to /dev/ircomm0 and use baud rate 115200.

With its installation via dpkg (or ipkg?), PPP is pretty much all set up and ready to go with the modem. First try setting up PPP with just the "pppconfig" utility. Alternatively and manually, edit the files /etc/chatscripts/provider (requires chat to be installed), /etc/ppp/peers/provider, and /etc/ppp/pap-secrets to include your own ppp information, or run the script pppconfig, which should come with PPP. PPP is started and stopped using pon and poff. (I made a script "ppp-on" that starts pon, dumps its messages to /dev/null, and puts it in the background.) You may also need to modify the /etc/resolv.conf file, to give your nameserver IP numbers.

14. Making Presentations Using a VGA-out PCMCIA Card

This section is experimental - there are many things I am ignorant of here, including how to deal with multiple screens in X windows. PCMCIA VGA cards have nothing to do with cardmgr - cardmgr will more or less ignore them, which is the correct thing for cardmgr to do. A page similar to this chapter on giving presentations on the iPAQ can be found HERE (http://www.iptel-now.de/HOWTO/IPAQ_PRESENTER/ipaq_presenter.html), with a tcl programs to give slideshows.

The compactflash VGA out cards are not supported, nor are they likely to be supported anytime soon. There is, however, a card from IO Data (http://www.iodata.com/usa/products/products.php?cat=GRPH&ts=2&tsc=15&sc=DISP-OUT) that is meant to work on the linux zaurus. However, that card is designed for Qtopia. But the source code for the linux driver can be downloaded from the iodata site...hint, hint, to an energetic developer...

14.1. Starting the X server

The Xipaq server from ipkgfind xserver-tiny-h3600_4.2.20030126-14_arm.ipk will work to drive a Hewlet-Packard F1252A VGA-out PCMCIA card. These cards are hard to find these days, long out of production, but they seem to appear on e-bay from time-to-time. The Colorgraphic Voyager card works too, although so poorly as to be almost unuseable; probably a server issue. Both cards have only 512K of videoram, which limits the display resolutions. The HP card has a trident chipset, while the Voyager contains a standard VGA controller, the Cirrus Logic GD5422. A description of how to compile the X server can be found Here (http://www.handhelds.org/minihowto/building-x.html). The XFree86 CVS source code for this driver can be found at the kdrive CVS (http://cvs.sourceforge.net/viewcvs.py/dmx/dmx/xc/programs/Xserver/hw/kdrive/). The presently available server binary seems to be rather limiting in its functions; more development is needed to take advantage of the video cards capabilities.

The most uptodate server binary seems to be at Xipaq (http://www.fairlite.demon.co.uk/handhelds.html), however this binary does not seem to work with the netBook. It expects the touchscreen device from the hp3600, /dev/ts, or /dev/h3600_ts, which do not exist (and don't seem to be able to be fooled into existing) so the server crashes.

If you start the X server with

```
Xipaq -dpi 75 -nolisten tcp -screen 640x480x8x60 -card pcmcia -screen 800x600x8x75
```

you will get two screens - one on the netBook's LCD and one on the external monitor. I don't know how to clone the netBook's screen to the external monitor screen. These two screens are separate - you will need to find a way to toggle between them. For example, in WindowMaker you can set an option "ScreenNextSwitchKey = XX" in the file ~/GNUstep/Defaults to set a key to toggle between screens. This could be F5 for example, if F5 were a defined key. The server also has a "-zaphod" option, which I suspect gives one the option of clicking on the X background to toggle screens. Otherwise I think that moving the mouse to the screen edge will toggle the screen, if the mouse could be made to move (see below regarding serial mouse). This driver seems to limit the number of colors to the external screen to 16 colors (not 16 bit colors, but 16 colors), however.

When you start this X server, you might get a very blue, off-color screen on your netBook. If this happens, first stop the X server, then ssh to localhost to get onto a pseudo-terminal, and then restart the X server.

You can also start the X server with

```
Xipaq -dpi 75 -nolisten tcp -card pcmcia -screen 800x600x8x75
```

(in /usr/X11R6/lib/X11/xinit/xserverrc) in which case you'll just get a single screen to the external monitor.

The F1252A VGA-out card is meant to support 1024x768 resolution, but I think the Xipaq server will only go to 800x600 - all modes in what appears to be 16 colors.

If you add the option

```
-mouse /dev/ttySA0
```

to the lines above, the X server will find and configure a serial mouse, which can be quite handy. See the section in this HOWTO on using a serial mouse, however. The mouse function for this X server appears to be a little sluggish compared to the XFree86 X server.

This Xipaq X server does not support the "Pointer_EnableKeys" extension for X windows. So if the serial mouse is not used, some other way to control the mouse must be used (FVWM apparently has mouse control through the keyboard), or a mouse free application must be used. However, once this server is running you can use xmodmap and assign the mouse buttons:

```
keycode 68 = Pointer_Button1
keycode 69 = Pointer_Button2
keycode XX = Pointer_Button3
```

This assignment does not work with the ordinary X server.

14.2. Window managers

The various window managers behave differently when there are two screens present. IceWM gives only an empty second screen, while WindowMaker gives a separate complete WindowMaker desktop, as does AfterStep. The trick is to start applications on the 2nd screen, while one is in the first screen. Obviously a careful selection of the window manager for your purposes would make working with the screens easier.

Apparently, export DISPLAY=":0.1" (rather than ":0.0") will cause applications to start on the second screen.

14.3. Slide presentations

The trick then is to find an application that will show slides (jpg, gif, png, etc.). I've had a hard time finding a useable application. xview seems to badly distort the colors, perhaps because of the 16 color limitation, and it will show only the first gif image in an animated gif file (for movies). Dillo could be used to show slides, but there is no mouse function to manage Dillo. Touch screen will be of no use here, even if it were working. But certainly with an external serial mouse pugged in there would be no problems controling the slides. Happy to hear ideas - post on the mail list.

However, the package "xzgv" seems to work rather well at displaying images, hence this application will likely work for making presentations of slides. In addition, xzgv is designed to navigate the a set of images using just the keyboard. Debian xzgv doesn't seem to display gif images (let alone animated gif imgages), however. And the 16 color limit is, well, rather limiting.

15. External IDE Devices using a PCMCIA Adapter

It is apparent that a variety of external IDE devices can be used for enhanced data storage for the netBook. Cheap data storage for the netBook can be found by using many PCMCIA IDE devices. Iomega clik! drives (very cheap 40Mb disks that slot into a reader that sits in the pcmcia socket) are one approach. Another is to use a PCMCIA to IDE converter (ca US$40 on ebay) to connect a normal or laptop CDROM drive or normal (3.5") or laptop (2.5") hard disk to the PCMCIA socket. Some of these devices will power directly from the PCMCIA socket (laptop hard drives?), but most will need some form of power supply in order to work. So it appears that with a PCMCIA IDE card (of which there are several varieties), all manner of external IDE devices can be plugged in. These cards seem to frequently have an odd 36 pin arrangement that doesn't plug directly into the normal 40 or 44 pin IDE devices; an adapter or housing is needed for the IDE device.

WARNING: Some of these cards/devices will only work with an external power supply to drive them; the netBook does not provide sufficient power on its own to the device. **Do not plug the card in with the external 5 V power supply attached to the pcmcia device when the netBook is not started up/powered up already.** The external device will power up the netBook the wrong way and it will cause a hardware failure. In my case, it blew a small fuse that was fairly easily mended, but it could also burn out the netBook's 5V supply, etc. - your netBook will fail to boot! Nothing on the screen, etc. i.e., disaster. (Some lessons are learned the hard way.) The fuse is near the power plug insert on the netBook mainboard. It is a tiny clear plastic or glass item that might have letters on it (e.g., "FG"), with a thin copper (or gold) filament running through it. If blown, the filament will have a gap. One quick fix is to short the fuse by wrapping this item with a thin, very thin, wire; it is work in a tiny space to be sure. A more conservative fix is to replace the fuse; in my case I am so terrible at soldering that the conservative fix is to send the netBook in for servicing.

Once the netBook has started to boot, the externally-powered pcmcia card can be safely plugged in - don't forget and push the pcmcia card in before the netBook is powered up: tie a ribbon on your finger, etc.; DON'T FORGET!!

At this stage it is unclear which types of IDE devices will work and which won't. Experience is limited.

15.1. PCMCIA IDE Hard Disk

It is possible to run a e.g., many-GB IDE disk on the netBook. This allows, among other things, a complete development environment to be set up, independent of the CF card, and also allows large, efficient swap space to be used. In addition, such disks can be booted by the netBook as the root device. The PCMCIA slot will recognized as

/dev/hda (rather than the CF), if there is a disk there at boot up. An external linux system on a disk plugged into the PCMCIA slot can replace the system on CF by just being there when the system boots.

Once such PCMCIA card that has been verified to work is the "i88990" PCMCIA to IDE adapter frequently advertised on e-bay these days (July 2005) for $42 after shipping (from Hong Kong, worldwide) They appear to be no-brand custom cards. It allows connection of 2.5" hdd, but also have an adapter to allow 3.5" drives to be connected. One netBook (which has a rev 10 motherboard) was able to power laptop hard disks and laptop cd/dvd drives without external power. The laptop cd drives can be connected by using the 3.5" connector, and an adapter on the drive to allow connection to standard power/ide (3.5") cables for use in a desktop. One can of course also use standard cd/dvd drives with a power supply unit.

Another card that has been verified to work is the "IBM Travelstar E" PC Card. This is a PCMCIA card and a housing for the 2.5" disk. The housing is nice and offers lots of padding for the hard drive. These apparently originally came with a hard drive in them; the one I bought had had the disk removed (easily opened and straightforward install to be sure, but not exactly user friendly either!). With the disk plugged in and placed in the housing this card and a 12 GB Travelstar hard disk were both recognized and worked fine - but only with an external 5 V, 1A power supply. The exterior connection on the housing is the same as for the Noteworthy CD ROM described below, but the two PC Cards did not seem to be interchangable.

With a linux system installed on the hard drive (e.g. Sargebook installed on /dev/hda2), the system will be recognized as /dev/hda at boot up if the card is present, and so the netBook's linux system will boot off the hard drive, even if a CF disk is present. I partitioned my hard drive with 5.5 GB for ordinary i386 SuSE linux on /dev/hda1 and 5.5 GB for SargeBook on /dev/hda2; the SargeBook system booted fine - 5 GB free disk space!.

15.2. PCMCIA CD-ROM

The Port Noteworthy PCMCIA 24X CD-ROM (Targus, NW24XCD) (http://www.targus.com/us/Downloads/NW24XCD_FAQ.htm) works fine on the netbook, and I am sure that many other types will as well. The NW24XCD seems to sell on e-bay for US$10-40 (7/05). This particular device requires an external 5V, 1A power supply to run properly in the netBook (otherwise the CDROM just goes clickity-clack and no CDROM device is recognized; try leaving the CDROM drawer open when the PCMCIA card is plugged in if you lack the power supply and want to test it), and the kernel must be compiled with ide-cd and iso9660 support. Other than that, there is little to the configuration; this is mostly handled by pcmcia services. If you attach the CDROM and then put in a CD, you may have to load the ide-cd module by hand. Mount the CD with the usual "mount -t iso9660 /dev/hdc /mnt/disk". Don't forget to unmount the CD and stop PCMCIA services before unplugging the PCMCIA card or your system will crash. Finally, it is probably best not to have the CDROM plugged in a boot up time, or your system may fail to start - the CDROM will be identfied as /dev/hda (unless you wanted to boot a system off CDROM!).

Figure 8. Port Noteworthy (aka Targus) CD-ROM plugged in the netBook

Whether the netBook is capable of running a CD-Writer or DVD remains to be seen.

16. Resources/Links

16.1. The OpenPsion Website

The main website for the OpenPsion project is: openpsion.org (http://www.openpsion.org), or more directly, http://linux-7110.sourceforge.net (http://linux-7110.sourceforge.net/). In particular, check out the Linux on the Psion 5MX/5MX-PRO HOWTO (http://linux-7110.sourceforge.net/howtos/series5mx/index.htm) for general usage information, e.g., how to set up a desktop<->netBook PPP connection.

16.2. netBook Linux Distributions

There are now two distributions available for installation on the linBook. Our aim is to eventually migrate and merge a distribution with openembedded (http://www.openembedded.org). The original "distribution" for the netBook consisted of a root filesystem extracted and modified from the Sharp Zaurus ROM images (openzaurus). The Zaurus ROM's approach still remains an option (see the section on how to extract the contents of the ROM's in this HOWTO).

16.2.1. Openembedded Linux on the netBook (IPKG based)

Taneli Leppä has developed a kernel that has rudimentary compactflash and PCMCIA support. He is keeping a Blog of his efforts at Taneli Leppä's Blog (http://sektori.com/~rosmo/netbook/index.php). There are hints that LCD, touchscreen, irda, sound developments are on the way....

Taneli has also compiled the latest openembedded (http://www.openembedded.org) distribution for the netBook. You can find this ipkg distribution, being tailored to the netBook, at www.netbook-linux.org (http://www.netbook-linux.org/). The feed of the latest ipks is there as well.

16.2.2. Debian Linux on the netBook (DPKG based)

There is a prototype Debian linux distribution for the netbook developed at Debian Linux on netBook (http://staff.washington.edu/dushaw/psion/openpsion/). This distribution is presently Debian Sarge ("stable"). Tarballs of this system are available directly from Sourceforge Files/Distributions (https://sourceforge.net/project/showfiles.php?group_id=8846). The X server that seems to work best is the ordinary Debian framebuffer server. Other X servers can also be installed, for example TinyX from IPKGfind at Handhelds.org (http://ipkgfind.handhelds.org/). *.deb's from Debian "stable" should install on this system. Installation of Debian linux on the netBook requires a compactflash card of 256 MB or greater, in reality. (If you wanted only console and minimal applications, and didn't want the netBook's EPOC OS.img file, you might squeeze Debian onto a 64 MB card. My 128 MB card has Debian on it with X windows, but there is no more disk space left.)

An alternate Debian system might be found at linBook: Debian Woody Linux on the Psion netBook (http://linbook.risible.org/). This has a cleaner Debian installation, but it requires 170 MB to install. This distribution includes the graphical links browser using DirectFB. This distribution does not include X windows, the author preferring to wait until touchscreen is supported.

16.3. IPKG and DEB Feeds

The links below offer small root filesystems and a selection of IPKG or DEB packages. In general, any of these packages can be installed on any of the systems; feel free to experiment. Of the distributions below, "openzaurus", now migrated to "openembedded", is probably the most uptodate and developed.

The pdaXrom system does not work on the netBook. The issue is that pdaXrom is aggressively compiled for Xscale cpus; binaries from pdaXrom won't run on the netBook's sa1100 cpu (they merely give a segmentation fault). The lesson is when you scavange for ipks, avoid those compiled with aggressive Xscale optimization.

There are two levels of Xscale optimization. There is a mild optimization (-march=armv4 -mtune=xscale) and a more aggressive optimization (-mcpu=xscale -mtune=xscale). With the mild optimization, xscale-binaries also run on strongarm. Agressive optimized xscale-binaries don't run on strongarm - they just segfault.

1. Debian Packages (http://www.debian.org/distrib/packages). Generic ARM binaries.
2. IPKGfind at Handhelds.org (http://ipkgfind.handhelds.org/). Perhaps the most advanced distribution for linux on IPAQ's.
3. Openembedded (http://openembedded.org/). OpenEmbedded ... That's the successor of the great OpenZaurus project. Openpsion has made motions to merge with this group, although the motion has stalled out of late. Openembedded uses the "soft-float" floating point emulation (see floating point) by default these days.
4. openzaurus (http://www.openzaurus.org/). The original openzaurus system. A grass-roots alternative to "myzaurus".
5. Zaurus Software (http://www.killefiz.de/zaurus/).
6. myzaurus.com (http://www.myzaurus.com/). The official Zaurus update page.

7. pdaXrom (http://www.pdaxrom.org/). A newer X-windows-based feed for newer zauruses (zaurii?). This system will not work on the netBook because it is highly optimized for the Xscale cpu. They claim that releases for other, older cpus may be made available "soon" (as of 7/04) so it may be worth keeping an eye on this site.
8. PTXdist - Userland Configuration Tool (http://www.pengutronix.de/software/ptxdist_en.html). PTXdist is a collection of Makefiles under the GNU GPL which are used at Pengutronix for generating userlands.
9. Emdebian (http://www.emdebian.org/) Embedded Debian is a project to make Debian GNU/Linux a mainstream choice for embedded projects.
10. Gentoo for Zaurus (http://gentooforzaurus.opensistemas.com/) Apparently ipkg compatible, although the packages seem to have the *.tbz2 extension (whatever that is...). Looks problematic at the moment, for a number of reasons.
11. Pocket Workstation - Debian on Handhelds (http://pocketworkstation.org/) Complete Debian + X windows ported to the Zaurus. Installs in a single directory. Requires a large compactflash card (256 MB preferred).
12. Crow ROM (http://www.schwag.org/~crow/) For Zaurus. This ROM is an enhancement of Sharp's 3.10 release. (Perhaps an ipk for Opera is available in the applications zip file).
13. Zaurus usergroup downloads (http://downloads.zaurususergroup.com/downloads/) Might be useful. See also Zaurus User Group (http://zaurususergroup.com/); ipks for the Opera browser can be found here.

16.4. The PCMCIA HOWTO

The HOWTO for PCMCIA on linux is at http://pcmcia-cs.sourceforge.net/ftp/doc/PCMCIA-HOWTO.html. This HOWTO is of great help in figuring out how to configure and run your PCMCIA card.

16.5. Plp Tools

This might be the easiest program to transfer files from you linux desktop to the psion over the serial line, pretty much plug-n-play. PLP Tools at Source Forge (http://plptools.sourceforge.net/). (These days I just transfer data using a compactflash card and a USB reader on my desktop, aka sneakernet. It's much faster.)

16.6. ArLo

Zipped source and binary files can now be downloaded from the OpenPsion web page downloads (https://sourceforge.net/project/showfiles.php?group_id=8846).

The official ArLo page is located at Peter van Sebille's webpage (http://www.yipton.demon.co.uk/content.html#ArLo) Peter has moved on from OpenPsion work, alas.

The ArLo HOWTO is HERE (http://www.yipton.demon.co.uk/arlo/latest/readme.html)

16.7. Bookboot

"Bookboot" is a project to construct OS.img files for the netBook to allow booting up directly into linux and avoiding EPOC altogether. This approach seems to be working o.k., although it is not well tested. Check out bookboot

(http://linux-7110.sourceforge.net/files/People/Klaasjan/netbook/). The web page for it is contained within this HOWTO, and you can certainly download it. OS.img's are available for downloading.

16.8. Proboot

Tony Lindgren's Proboot page (http://www.muru.com/linux/psion/proboot/) Also some very useful information in the README about how the Psion's load the OS.img files and boot them up.

16.9. Precompiled Kernels

Try the openpsion downloads page at Source Forge (https://sourceforge.net/project/showfiles.php?group_id=8846).

You can also try the kernels at openpsion (http://staff.washington.edu/dushaw/psion/openpsion/), although this site is likely to be temporary.

The original (now long out of date) kernel for the Series 7 is at Peter van Sebille's webpage (http://www.yipton.demon.co.uk/content.html#PsiLinuxS7); this will work on the netBook, but it supports only 16 MB of ram. Peter did a lot of the startup work with the Series 7, but has more or less dropped out of the picture now.

16.10. Initial Ramdisks (initrd's)

Initrd's can be downloaded from the openpsion downloads page at Source Forge (https://sourceforge.net/project/showfiles.php?group_id=8846). A newer initrd, still underdevelopment, can be found at Debian on netBook (http://staff.washington.edu/dushaw/psion/openpsion/), a system derived from openzaurus (http://www.openzaurus.org). openpsion supports ipkg, networking, PPP, and ssh, among other things. Also check out Peter's site.

16.11. The Mail List

You can post queries at the OpenPsion mail list by sending an e-mail to linux-7110-psion@lists.sourceforge.net (mailto:linux-7110-psion@lists.sourceforge.net). You might even get a reply either directly, or appearing on the mail list.

You can browse the mail list archives, including the e-mail you just sent, by going to http://sourceforge.net/mailarchive/forum.php?forum_id=7163

16.12. netBook Stripped!

www.symbcity.com/ (http://www.symbcity.com/featureshow.php3?featureid=100015) has pictures of a netBook taken apart.

16.13. The "simputer", from India

A firm in India apparently developed a PDA thing based on the sa1100 cpu: "Aims at developing low cost access device that can pervade the rural landscape, especially in third world countries." (The linBook is a better device, by far!) They post some of their applications on line, which may be of use on the netBook www.simputer.org/ (http://www.simputer.org) gives a description of the device and some of its software. Alas, the simputer seems to have failed to catch on in rural India.

16.14. RISC OS on a netBook

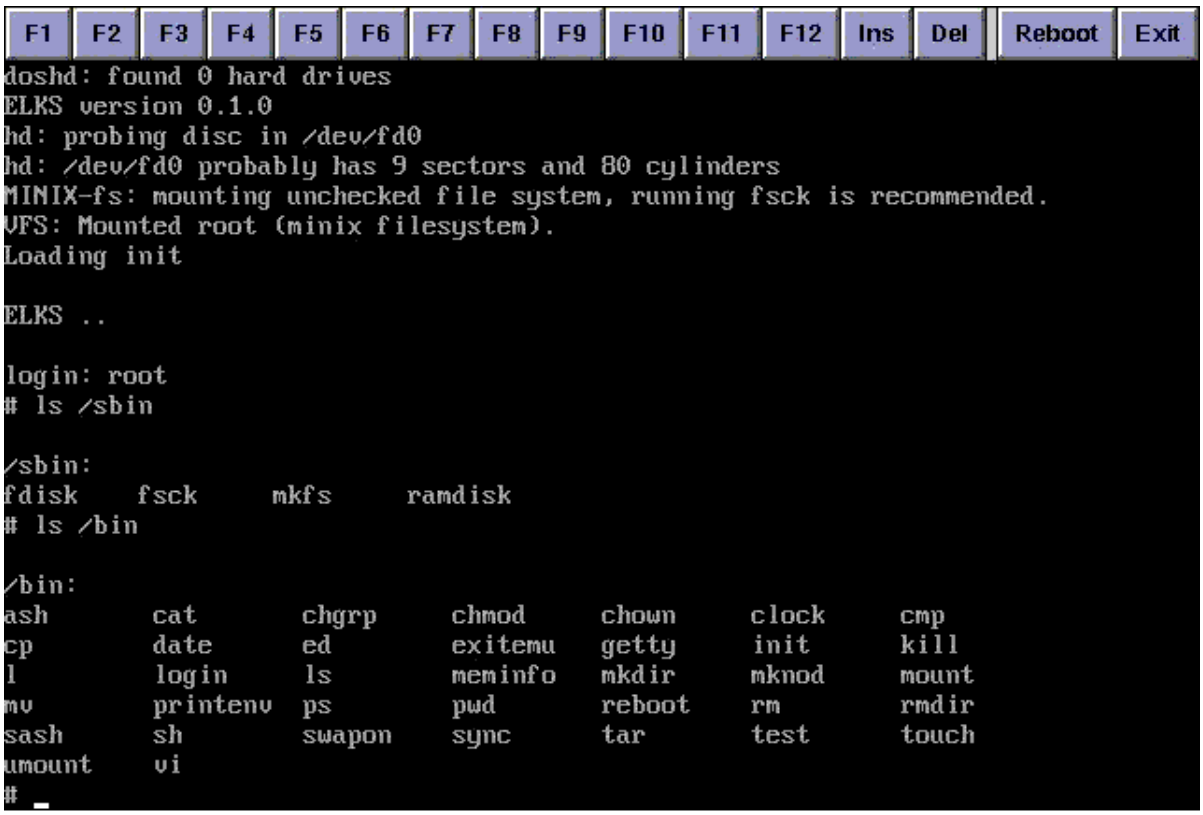
There apparently was a move by RISC OS and Psion to produce an alternate OS for the netBook called RON (http://www.zen7228.zen.co.uk/editor/ron.htm). I don't believe anything came of it... A Press Release (http://www.riscos.com/news/news_items/PR201001.htm) A quote from www.psionplace.com "I e-mailed Risc, when I wanted a netBook running Risc OS and they said that the Risc on netbook was put on the long term hold." I sent and e-mail asking if the (abandoned) work that was done on RON might be used for linux development - I received no reply.

17. Poor Man's Linux's: XTM and ELKS, or EPOCemx

17.1. XTM and ELKS

A poor substitute for linux on the netBook is to use the XTM 80188 emulator together with ELKS linux. The XTM (http://www.nb-info.co.uk/index.htm) emulator is an EPOC program that emulates the 80188 instruction set with MCGA graphics and 640KB of base RAM. This emulator is not free, but you can install and try out a trial version of it. ELKS (http://elks.sourceforge.net/) is a linux-based operating system that is designed to run on old cpu's with limited resources. It uses the minix filesystem. You can download a boot + root floppy image from the ELKS website and use it to boot up XTM. The XTM emulator is, of course, aimed at the DOS OS and all its applications (I installed DR DOS/Caldera DOS (http://www.drdos.com/).) Unfortunately, there are as yet few (if any) applications that will run in ELKS. Still, it is a possibility, and development of ELKS continues, albeit slowly.

Figure 9. XTM with ELKS booted up



17.2. EPOCEMX

The epocemx (http://epocemx.sourceforge.net/) project has had nice success in porting the unix/linux environment to EPOC. This has the best shell for EPOC of all, to my knowledge. It works nicely on the netBook. This project was aimed at getting gcc to work for compiling EPOC applications and they seem to have done it. The gcc compiler is meant to work on the netBook (although slowly), because the netBook has enough memory. (EPOCemx was orignally designed to run on the EPOC emulator on the desktop, just for the purpose of compiling applications.) There are a set of *.sis files now that let you install this package. Highly recommended. You can also install vim (http://www.starship.freeserve.co.uk/) which has been ported to EPOC and combine it with emx. Then you will be stylin'! :)

18. Kernel Development

The following items are desired for kernel development or compilation. Some are easy, some are hard. Since most people have not set up a cross-compilation environment, the idea is for kernel binaries to be provided with these features for posting. The kernel patches also have to be posted.

18.1. Wish List

1. Keyboard: There is one more key that needs defining: the latch to close the netBook - that should give a keystroke that might eventually be used to power down or up the netBook when it is closed or opened. And I need not say that the power-up or down keys still need to be implemented.
2. Power control: power off after inactivity, screen lighten/darken, powerdown on shutdown, etc.
3. Compactflash support: Could still probably use development to maturity.
4. PCMCIA support: Although wireless networking works, PCMCIA support is still quite crude. PCMCIA would let us use things like ethernet cards, modem cards, wireless ethernet, and second compactflash cards. At boot up, a 2nd compactflash card presently causes the first compactflash to be non-funtional, although later on the 2nd compactflash card can be inserted without a problem.
5. Sound support - at least for system beeps! This is also under development; some initial patches exist, but not to a functioning sound system yet. The netBook's sound is 12 bit, which is not supported by linux. So we will have an 8 bit sound system.

18.2. Kernel Development Needs

Peter wrote the following concerning information etc needed for kernel development. Peter has had some success in getting a kernel booting on the Series 7.

```
Date: Sun, 23 Feb 2003 17:58:14 +0000
From: Peter van Sebille <peter@yipton.net>
To: Marco Carando <marco.carando@psion.com>
Cc: linux-7110-psion@lists.sourceforge.net, jim.george@blueyonder.co.uk,
nsc@qsf.demon.co.uk, wookey@aleph1.co.uk
Subject: Re: [Linux-7110-psion] Linux on Psion netpad & netbook

Hi Marco,

I'm sorry for the late response, but I have overlooked your mail on the
linux-7110-psion mailing list.

I'd still be interested in continuing my efforts in porting Linux to the
Series 7 and the Psion netbook. As you can see on http://www.yipton.net,
I managed to get Linux up and running (to some degree) on the Psion
Series 7.

The bits that work are essentially the standard SA-1100 hardware
features which are documented in StrongArm's datasheet. In order to get
a fully functional port of Linux on the Series 7 & Netbook, we need more
```

hardware info. The latter includes the following:
* List of hardware configurations of the different Series 7 / Netbook
  models (physical memory layout etc).
* Info describing register layout of hardware features not present/used
  in the SA-1100 (things like touchscreen, compact flash etc)
* For the Netbook it would be great to get the info on how the "boot from
  compact flash card" solution works. This allows Linux to be booted on
  startup without the help of EPOC.

Thing that would be helpful as well:
* Access to the Series 7 / Netbook ER5 sw drivers and or kernel source.
  No matter how well the hardware is described, being able to look at
  production quality drivers is a *huge* benefit.
* Maybe we could also get email support from a Psion hw/sw engineer.
  I don't think you have to be afraid for lots of mail discussions,
  but I think it would be helpful if we have a channel to ask very specific
  technical questions.
* When things are moving forward, maybe Psion could donate/lend a couple
  of S7 and Netbook machines for people to test and verify.

In the past, Psion has allowed us to distribute information on the Psion
5mx ASIC under the conditions stipulated below (with the one difference
that we were allowed to replace "LINUX" by "OpenSource"). I've acted as an
intermediate in distributing these documents. People could mail me asking
for them; I would then send them a mail Psion's conditions asking them
whether they agree. If yes, I'd mail them the document.
If Psion prefers this method over distributing information on the Series
7 / Netbook themselves, I'm more than willing to do this again for the
Series 7 / Netbook stuff.

I'm looking forward to your reply on getting some new momentum in the Linux
on the Psion Series 7 & Netbook project!

cheers,
        Peter

***** Original Psion condition for Psion Series 5mx hardware info ********

1) Any information I give you is for your own personal use or for use by the
genuine LINUX community.
2) You will not give this information to your employer or deliberately allow
it to be used for any commercial enterprise.
3) You will not publicly distribute this information. This includes but is
not limited to posting it to a web site.
4) You will only share this information with genuine LINUX developers who
also agree to abide by these rules.
5) You are a genuine LINUX developer.
6) You fully understand that any information I send to you is the
confidential property of Psion Computers PLC and we reserve the right to
limit it's use.

Marco Carando wrote:

```
> Hi all
> there is someone interested to port Linux on netpad and netbook?
>
> if you are interested please contact us by mail or phone , we are really
>  interested  to analyse a proposal
>
> best regards
>
> Marco Carando
```

19. Recipe for Compiling the Kernel

Here, in brief, are the steps to assemble the pieces to compile your own version of the linux kernel for the netBook. The cross-compiling environment can be set up in less than an hour or so (and you have the marginal benefit of notes polished in response to my experience!).

For those of you with lots of disk space on your netBook, note that you can compile the kernel and its modules directly on the netBook. This does not take THAT long, all things considered - 1-2 hours. To do this, you will of course need to install make and gcc on your netBook, in addition to the kernel source. But it saves you having to install and set up the cross-compiling environment. When compiling the kernel, it is recommended to stop all other processes, including X windows. The main potential problem is lack of memory during the compile process.

19.1. Preferred Cross-compiler: Scratchbox

The cross compiler that seems to be preferred by openpsion developers is scratchbox at www.scratchbox.org (http://www.scratchbox.org/), which is a complete x-compiling environment. This has a fairly easy installation, with a set of rpm or deb packages, depending on the system you want. Scratch box now has a fairly uptodate compiler (gcc version 3.3.4) which has been verified compile a functioning the kernel binary for kernel version 2.4.27. Scratchbox is contained within a large virtual file system, and I've found you have to login to the virtual system to use the compiler on kernel source. You may need a fast internet connection (or a lot of patience) to get it and loads of free diskspace to unpack it. I installed the following packages:

- binutils-arm-linux_2.15-5_i386.deb
- cpp-3.4-arm-linux_3.4.2-3_i386.deb
- g++-3.4-arm-linux_3.4.2-3_i386.deb
- gcc-3.4-arm-linux_3.4.2-3_i386.deb
- libc6-dev-arm-cross_2.3.2.ds1-18_all.deb
- libgcc1-arm-cross_3.4.2-3_all.deb
- libstdc++6-0-arm-cross_3.4.2-3_all.deb
- libstdc++6-0-dev-arm-cross_3.4.2-3_all.deb

After installing, get an ARM cross compiling system set up (I am a little confused by all this too, but I managed to muddle through to something that worked).

19.2. Qemu and Debian Linux - a Complete Emulated System

You can now use the Qemu emulator to emulate a complete ARM system. The emulated system does not support PCI or hard disks, so a network/NFS boot has to be employed. You can see instructions for how to set up the network system boot, or download a complete debian system from emulated Debian ARM system (http://909ers.apl.washington.edu/~dushaw/ARM/). This is a complete Debian Sarge system for ARM processors emulated using qemu. This is not technically a cross-compiler, but a native compiler. This system is easy to install and easy to use, but has the one downside that it runs a slower than, e.g., scratchbox. It has the advantage that since it really is Debian Sarge, compiled binaries can be tested, and are guaranteed to work, on a Debian Sarge system used on the Psion - there will be no library issues.

Qemu could also be used to compile a system for the netbook from scratch, e.g., to optimize for floating point.

19.3. Cross-compiler Alternative: The emdebian cross compiler for Debian Woody

- follow instructions on emdebian website (www.emdebian.org (http://www.emdebian.org/)) (entry in /etc/apt/sources.list)

- apt-get install task-cross-arm

this leads to one error message "couldn't find /usr/share/info/iostream.info.gz". Fix this error using: "ln -s iostream-295.info.gz /usr/share/info/iostream.info.gz"

These tools can also be set up on rpm-based systems. Get the rpm's from http://www.emdebian.org/tools/crossdev.html. You'll need something these (now deprecated, no doubt):

1. binutils-arm-2.9.5.0.37-1e3.i386.rpm
2. cpp-arm-2.95.2-12e4.i386.rpm
3. g++-arm-2.95.2-12e4.i386.rpm
4. gcc-arm-2.95.2-12e4.i386.rpm
5. libc6-dev-arm-2.1.3-8e4.noarch.rpm
6. libstdc++2.10-arm-2.95.2-12e4.i386.rpm
7. libstdc++2.10-dev-arm-2.95.2-12e4.i386.rpm

These rpm's did not install properly with rpm on my RedHat 9.0 system, but it was fairly easy to unpack and install them by hand. Put the rpm's in their own directory and for each file (a) unpack it with "rpm2cpio filename.rpm > temp", (b) then unpack the cpio "temp" file with "cpio -i < temp". This will make a directory "usr" with the stuff in it. Then tar the usr directory over to the system files " tar cf - usr | (cd / ; tar xf -)". (This has not noticeably cause my RedHat system to become non-functional...it's o.k....) The cross compiler (x-compiler) is then available.

19.4. Get kernel files and patches

The versions of all of these files changes fairly regularly; get the most recent ones!

1. linux-2.4.27.tar.bz2 from www.kernel.org (Here (ftp://ftp.kernel.org/pub/linux/kernel/v2.4/))
2. patch-2.4.27-vrs1.bz2 from www.arm.linux.org.uk (Here (ftp://ftp.arm.linux.org.uk/pub/armlinux/kernel/v2.4/))
3. patch-2.4.27-vrs1-kvd1 from www.psilinux.org (Here (http://linux-7110.sourceforge.net/files/People/Klaasjan/netbook/))

   (This patch gives the options of 16, 32, or 64 MB of memory; but use the default 16 MB.)

Put all these files in a directory "some/sourcedir/with/files/"

19.5. Compile

Unpack the kernel source, apply the patches, configure the kernel:

1. cd some/sourcedir/with/files/
2. tar -xjvf linux-2.4.27.tar.bz2
3. cd linux
4. zcat ../patch-2.4.27-vrs1.gz | patch -p1
5. cat ../patch-2.4.27-vrs1-kvd1 | patch -p1
6. make mrproper
7. make netbook_config
8. make oldconfig (experts can then do "make config", or "make xconfig" to customize the options)
9. make dep; make clean

Now log in to the scratchbox system and compile:

1. make zImage

Copy the arch/arm/boot/zImage file to the compactflash card and it should boot up your netBook.

19.6. Compiling and Installing Modules

You will likely configure the kernel to use kernel modules. [From http://www.arm.linux.org.uk/docs/kerncomp.shtml#crossinstall]:

After *make modules*, install the modules into /usr/src/arm/ [or some other directory] as follows:

bash$ make modules_install INSTALL_MOD_PATH=/usr/src/arm/

If your kernel version is x.y.z, this command will place the modules into the /usr/src/arm/lib/modules/x.y.z directory on the host, which can then be placed into an suitable filesystem, or transferred to the target machine. (note that

/usr/src/arm/lib/modules/x.y.z should become /lib/modules/x.y.z on the target machine). Please also note that you should not install these kernel modules into the hosts root filesystem, since they are incompatible with your host kernel.

However, within the scratchbox virtual environment, a "make modules_install" will just safely put the modules in the virtual /lib/modules directory of the virtual environment.

You will also need the module utilities programs (modutils) in your filesystem to manage the modules.

20. Developer's Notes

These are a collection of notes for developers. This section is intended to provide information, specifications, descriptions of things that have been tried and not tried, etc. to facilitate new development on the netBook kernel and filesystem. Our present most uptodate kernel is 2.4.21. My own view is that a great deal can be accomplished in kernel development by just poking around the kernel source - I've made a few advancements myself, and if I can do it anyone can! (I don't even know the C programming language!)

We continue to collect any and all information relevant to kernel development, however minor that information may be. We request that should you know of, or develop, information on present on these pages, that you make a post to the psilinux maillists (linux-7110-psion@lists.sourceforge.net (mailto:linux-7110-psion@lists.sourceforge.net)). Please, please, don't try to make things perfect before posting something. Others may be able to provide feedback, and help improve things.

20.1. Some Documentation

Discussion of linux on ARM processors, and some documentation, can be found at The ARM Linux Project (http://www.arm.linux.org.uk/). There is a lot of information for developers at that site.

Documentation on the StrongARM SA-1100 processor (the basis of the Netbook), including GPIOs, serial ports etc can be found in Intel StrongARM SA-1100 Microprocessor Developer's Manual, Order Number: 278088-004, available from the LART website: www.lart.tudelft.nl/doc.php3 (http://www.lart.tudelft.nl/doc.php3)

Extensive information on the ARM architecture is also available from ARM Ltd (fill in a web form, and they'll mail you a cd full of pdf documentation). www.arm.com/documentation/cd_request.html (http://www.arm.com/documentation/cd_request.html) (useful for the entire psilinux project, not just the netbook)

There is an article by Wookey and Tak-Shing entitled "Porting the Linux Kernel to a New ARM Platform" that provides an overview of the kernel for ARM. You can obtain it locally here: porting2arm_aleph.pdf.

20.2. PCMCIA

There is a great deal of existing kernel source code for PCMCIA on the SA1100 cpu. Rudimentary development of PCMCIA support on the netBook has occurred so that wireless networking works, but it is still fairly crude.

20.3. Compactflash

Debian Woody or openpsion (modified from openzaurus) can now be installed on compactflash. At times, people have taken a multimeter to the netBook to identify the pin functions. The remaining problem is that we haven't yet figured out how to properly configure PCMCIA interrupts.

20.4. PCMCIA/CF pinouts

A few people have taken a multimeter to the netBook's PCMCIA pins to extract information needed for kernel development. This information is...(a link?)

20.5. LCD/Framebuffer Driver

The problems with the X windows driver (the system responds with a blank screen and unresponsive keyboard), together with the bookboot problems, suggest that the framebuffer driver is not quite 100% all working. The framebuffer driver doesn't seem to be able to reset itself - perhaps the reason it works at all is that it inherits a certain amount of functionality from EPOC through the Arlo boot process. It would seem that the kernel framebuffer driver still has a few issues to resolve, in any case. Peter van S. comments regarding his 2.4.2 patch: "frame buffer: changed the sa1100 frame buffer driver to remap the video memory as setup by EPOC; it doesn't (yet) reprogram the LCD controller." However, in the 2.4.21 patch, video memory is _not_ remapped, and the LCD controller is programmed by the kernel (unlike Peter's 2.4.2 patch, which avoided touching any lcd settings) See netbook_series7_info in drivers/video/sa1100fb.c of linux-2.4.21-rmk2-kvd1.

20.6. IRDA

IRDA does not seem to work on the netBook for any of the 2.4.2, 2.4.19, or 2.4.21 kernels. IRDA seems to be healthy and working insofar as the kernel is concerned, but I've not got any indication on my laptop that IR signals are being sent out. With the present kernel the /dev/ttySA1 (2.4.21) device, which might be expected to be associated with IRDA SIR, is never recognized - this can be fixed in "arch/arm/mach-sa1100/netbook_series7.c". (The 5MX recognizes the equivalent /dev/ttyAM0 and /dev/ttyAM1.) From the looks of the serial port and irda drivers, it looks as if development is needed to customize these a little bit for the Series 7/netBook hardware. Ordinarily, SIR would be set up with "irattach /dev/ttySA1 -s 1", and FIR with "irattach irda0 -s 1". Alternatively, "ifconfig irda0 up" followed by "echo 1 > /proc/sys/net/irda/discovery" is meant to start up FIR. Communication is then through /dev/ircomm0, e.g. by minicom or PPP. Linux's "irdadump" will show irda activity if it is present; irdadump seems to work fine on the netBook. SIR is a low-level generic driver for IRDA and the netBook's SA1100 is meant to support this - it is meant to be straight forward and hardly more complicated than the normal serial port...

It may be that IRDA requires additional configuration by installing it using modules and appropriate options. My own suspicion is that everything is working o.k., except the sa1100 irda routine lacks the small bits of code to just turn the irda on for the netBook. I also suspect that IRDA would work if it were first started in EPOC, and then linux was quickly booted. Linux would inherit the IRDA powered up.

From the documentation: The Infrared Communications Port (ICP) is serial port 2 with IRQ 16. Both SIR and FIR cannot be enabled at the same time. Hewlett-Packard Serial Infrared (SIR) modulation is used for low-speed transmission up to 115.2 Kbps. The HP-SIR enable (HSE) bit controls whether the HP-SIR bit modulation logic is enabled or disabled. When HSE=0, HP-SIR modulation is disabled, and if UART operation is enabled (ITR=0), it is

used for normal serial transmission (NRZ encoding only) rather than IrDA communication. When HSE=1, HP-SIR modulation is enabled for low-speed IrDA communication; zeros are represented by pulses that are 3/16 of the programmed bit width, while ones are represented by no pulses.

The netBooks regular serial port is serial port 3 with IRQ 17.

The netBook is rumoured to support 4Mb/s FIR IRDA, but this has not been substantiated.

See http://www.hpl.hp.com/personal/Jean_Tourrilhes/IrDA/IrDA.html for as complete a description of IRDA and configuration as I have ever seen.

20.7. Ramdisks/ramfs/tmpfs

There seems to be some peculiarities with ramdisks. It seems as if mounting a second ramdisk causes memory problems, which is strange (e.g., mke2fs /dev/ram1 ; mount /dev/ram1 /mnt/disk"). Basically, the problem I've had is that copying large amounts of data to the ramdisk (e.g., the X windows system derived from the 5MX package) gradually fills up memory, and is perhaps not written to disk, leading eventually to a system freeze. The latest kernels behaved a little better, but were not trouble free.

One solution to this is to use ramfs or, better yet, tmpfs which seem to work better. However, for a ramfs filesystem, the newer kernels report nothing back using "df". There is meant to be a patch for this (to allow for accounting in a ramfs) in Alan C's package. Philosophically, ramfs is mean to be as simple as possible, apparently, hence the lack of accounting. So data can be stored and used on the ramfs disk, but no accounting of it is possible, other than a loss of memory shown with "free". For this reason the tmpfs is better, since it does keep accounting and "df" shows what's going on.

ramfs are mounted with "mount -t ramfs ramfs /mnt/disk", with no need to format the filesystem before doing this. ramfs are meant to expand in size as needed, so they are more flexible than ramdisks.

tmpfs are mounted with "mount -t tmpfs -o size=20M tmpfs /mnt/disk", with no need to format the filesystem before doing this. Like ramfs, tmpfs are meant to expand in size as needed, so they are more flexible than ramdisks. The "-o size=20M" option makes the disk size 20 MB, rather than the default ramdisk size. It may be that a default size of 4 MB could be used, omit the "-o size=20M" option for th tmpfs, and have the tmpfs grow as needed beyond 4 MB would work (but I've not tested this).

20.8. Power Control

There presently does not exist any means to power off the netBook, set an automatic shutoff, or to lighten or darken the screen. I don't think anyone has yet worked on the power management issues. There is source code in the patch for the Series 5MX that may prove relevant to these issues, if not an easy port. There are also options for changing the frequency of the sa1100 - lower for power savings, higher for greater performance.

20.9. Keyboard

It appears as if we've managed to at least define keycodes for all the keys on the netBook, including the power switch and the external button at the front of the netBook. The key definitions we have are for the malaybook only, so there is a need to define keys for other versions (UK? DE?). The power keys and screen lighten/darken are not yet associated with those functions, however.

20.10. Arlo Needs

Arlo has a design limitation that prevents initrd.gz files of larger than 3.5-4 MB from being loaded. The initrd filesystem can be any size, so long as the compressed filesystem size plus kernel size is less than 4 MB. This is presently limiting the amount of data that can be placed on the initrd - the tools must be chosen carefully. Arlo is being redesigned and worked on to resolve the issue.

A list of requested Arlo developments:

```
        - pass memory layout in kernel boot params (ATAG_MEM),
        - possibly detect memory layout at runtime in boot2nd.
        - pass boot parameter address also in r2 to the kernel,
              compliant with Documentation/arm/Booting.
        - Allow run-time adjustment of boot parameter address (e.g. in arlo.cfg)
        - allow more than 4MB of total stuff to be passed to boot2nd
        - save the "machine= Series 7" option
```

20.11. Sound

In the kernel tree directory "devices/sound", there are a couple of sound routines for the sa1100: sa1100-audio.c, sa1100ssp.c. These look like they are ready to go for the netBook; they are presently set up for "lart" (another sa1100 system). I think the GPIO's that are used need merely be adjusted for the netBook, but I know nothing, really...

20.12. SanDisk 128MB + Wi-Fi CompactFlash ideal wireless solution?

SanDisk have a CF card called "ConnectPlus" that combines 128MB CF and Wi-Fi. The 5MX people are proceding with development that may eventually be of use for the netBook. Output of dump_cis:

```
Socket 0:
  dev_info
    fn_specific 700ns, 2kb
  mfc {
    vers_1 7.0, "SanDisk", "ConnectPlus"
    manfid 0xd601, 0x0101
    funcid network_adapter
    lan_technology wireless
    lan_speed 34 mb/sec
    lan_speed 35 mb/sec
    config base 0x03e0 mask 0x0001 last_index 0x01
    cftable_entry 0x01 [default]
      Vcc Vnom 3300mV Vmin 3V Vmax 3600mV Iavg 300mA
      Ipeak 300mA Idown 10mA
      io 0x0000-0x007f [lines=7] [16bit]
      irq mask 0xffff [level] [pulse] [shared]
  }, {
    common_jedec 0xdf 0x01
    funcid fixed_disk [post]
```

```
  disk_interface [ide]
  disk_features [silicon] [unique] [single]
    [sleep] [standby] [idle] [low power]
  config base 0x0200 mask 0x000f last_index 0x07
  cftable_entry 0x00 [default]
    [rdybsy] [mwait] [pwrdown]
    Vcc Vnom 5V Vmin 4500mV Vmax 5500mV Ipeak 80mA
    memory 0x0000-0x07ff @ 0x0000
  cftable_entry 0x00
    Vcc Vnom 3300mV Ipeak 45mA
  cftable_entry 0x01 [default]
    [rdybsy] [pwrdown]
    Vcc Vnom 5V Vmin 4500mV Vmax 5500mV Ipeak 80mA
    io 0x0000-0x000f [lines=4] [8bit] [16bit]
    irq mask 0xffff [level] [pulse] [shared]
  cftable_entry 0x01
    Vcc Vnom 3300mV Ipeak 45mA
  cftable_entry 0x02 [default]
    [rdybsy] [pwrdown]
    Vcc Vnom 5V Vmin 4500mV Vmax 5500mV Ipeak 80mA
    io 0x01f0-0x01f7, 0x03f6-0x03f7 [lines=10] [8bit] [16bit] [range]
    irq 14 [level] [pulse] [shared]
  cftable_entry 0x02
    Vcc Vnom 3300mV Ipeak 45mA
  cftable_entry 0x03 [default]
    [rdybsy] [pwrdown]
    Vcc Vnom 5V Vmin 4500mV Vmax 5500mV Ipeak 80mA
    io 0x0170-0x0177, 0x0376-0x0377 [lines=10] [8bit] [16bit] [range]
    irq 14 [level] [pulse] [shared]
  cftable_entry 0x03
    Vcc Vnom 3300mV Ipeak 45mA
  cftable_entry 0x07
}
```

20.13. Initial Ramdisks (initrd's)

Although compactflash is now working, a decent initrd on a 10-20 MB, say, RAM disk might still be useful. This, together with a reasonable kernel, will make for a minimally functional system, perhaps even with X windows. See, e.g., openpsion (http://staff.washington.edu/dushaw/psion/openpsion/).

The development of a complete initrd is hampered by the Arlo design limiting (size of initrd.gz)+(size of kernel) < 4 MB. This means that things have to be transferred in to the RAM filesystem over the serial port after the minimal system is booted up. The X-windows system is about 17 MB in size, so this takes a while at 115kb/s, but is certainly doable. Transfer is faster using wireless networking, of course!

21. FAQs

Be sure to also check the FAQ at the main www.openpsion.org (http://linux-7110.sourceforge.net/faqs.shtml) site. Also the old FAQS (http://linux-7110.sourceforge.net/faqs/) (archaic).

21.1. Why do the URL's of the HOWTO pages keep disappearing/changing?

This HOWTO is written in SGML and processed to make the html pages. The processing in docbook automatically makes the page names (and all the links between them), which change as the document evolves. So linking to a page like: http://linux-7110.sourceforge.net/howtos/netbook_new/x196.htm will not generally work, since x196.htm may not be there next time, or perhaps worse will be a link to an old, perhaps erroneous, page. Set your bookmarks to http://linux-7110.sourceforge.net/howtos/netbook_new/index.htm and hit "refresh" frequently and you'll be o.k.

21.2. How can I type the pipe (|) character?

The all-important pipe character, "|", is <Ctrl><t>.

21.3. Is there any size limit to the compactflash or microdrives that are supported?

Large compactflash and microdrive drives are becoming more affordable all the time. There does not seem to be any size limit inherent in the netbook's hardware (and certainly not in the linux kernel). Disks of size 4 GB have been verified to work fine.

21.4. Compare and contrast compactflash or microdrives for me please?

In terms of how these drives behave or interact with the linux kernel (or EPOC kernel) there is no difference. They are just IDE disks. Compactflash disks have a limited number of writes that they can make, although they are intelligent about how they write data. It would take extensive, repeated disk writes to come close to wearing out a compactflash drive. Microdrives have mini platters in them and don't have any writing limitations. Microdrives may be more suitable for using swap space than compactflash cards. However, microdrives are a mechanical drive, and hence will draw more power than compactflash cards - your netbook will run out of battery power a little sooner (how much sooner? dunno.) Microdrives make a very slight, barely noticeable, whirring sound, while compactflash cards are silent solid state devices..

The relative speeds of compactflash and microdrives is unknown, but likely to be highly variable depending on brand of compactflash and microdrive. Speed is largely irrelevant for the netbook at the moment because of the crude ("polling") PCMCIA support - everything is slow.

21.5. How can I get the kernel to boot directly without having to boot up EPOC?

A system of generating OS.img files, similar to those from psion, has been put together; it is called bookboot. This will allow a kernel image and initrd.gz file to be stored in this file, to be directly loaded by the netBook's bootloader. This procedure works just fine; see the section on bookboot in this HOWTO.

Alternatively, you can define a plain text file called D:\System\Data\wsini.ini containing the following lines (modified from the default Z:\SYSTEM\DATA\WSINI.INI):

```
BASERGB 255,255,255
PALETTE 0,85,150,255
AUTOCLEAR 1
SHELLCMD \System\Apps\Shell\Shell.APP
STARTUP D:\ARLO\ARLOSH.EXE
WINDOWMODE COLOR256
SCR_WIDTH1 640
SCR_HEIGHT1 480
SCR_WIDTH2 480
SCR_HEIGHT2 640
```

This will cause EPOC to startup ARLOSH.EXE at boot up first, giving the option of continuing to boot up EPOC or boot to linux. [This looks very much like the LILO prompt.]

21.6. Are there any EPOC backup solutions that don't require the desktop software?

My backup approach is to copy the contents of the entire C:\ drive, including the System directory, to a subdirectory on the compactfash D:\ drive before booting linux (select everything, then ^C to copy, ^V to paste). To restore the system on reboot, I first delete the entire contents of the C:\ drive (including the System directory - C:\ is completely empty; select everything, then ^D to delete), then copy the original contents of C:\ saved on compactflash back to the empty C:\ drive (^C to copy, ^V to paste). This has worked quite well for some time. The copy-paste between C: and D: drives puts things back exactly as they were, and doesn't take but a few minutes. The annoyances are that one has to stop (^E) all the applications that start at reboot (or you can't delete everything) and reset all the time, home city, sound, show hidden system directory, etc. preferences. I've started using Sysback recently which helped some. (And from time-to-time I'll burn a CD of all the EPOC contents!)

21.7. Can I plug in a serial mouse to the serial port and use a mouse that way? [Yes, but be a little wary.]

An honest serial mouse appears to work o.k. on the netBook's onboard serial port. See the section in this HOWTO on using a mouse. There was one report, perhaps using an older series 5 or 5MX, that the mouse would work for a while, but then the serial port was burned out. I had the following reply from the XTM (http://www.nb-info.co.uk/index.htm) people:

```
One thing we tried, and you must NOT try, was plugging a serial
mouse into the RS232 port.  Since the serial mouse draws power
```

```
from the RS232 port, it pulls a LOT of current from the plug,
and we burned out the serial port on a Psion [a series 5 or 5MX]
playing with it... :-( The good news is, it actually worked, right
up to the bit where it burned out!
```

This naturally raises the issue of whether this would work o.k. on a netBook, or one could use a mouse that draws less current, or that perhaps is powered by some external power source. I have used a Logitech serial mouse without apparent harm (so far...). However, trying out a mouse on the netBook may be risky. For more information on power and serial ports, see: www.hut.fi/Misc/Electronics/circuits/rspower.html (http://www.hut.fi/Misc/Electronics/circuits/rspower.html)

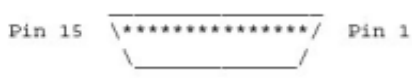
21.8. Can you tell me the specifications of the netBook's serial port?

This is the specification for a standard (as supplied by Psion) serial cable, including the equivalent 25-pin D pinout in case you want that instead/as well. This has RX/TX and RTS/CTS 'swapped' as both devices expect to be 'in charge'. This is what's called a 'null modem' cable. Note that DTR and DSR seem to be straightthrough, but my working null modem adapter swaps these, in addition to swapping RX/TX and RTS/CTS. The pin assignments and wire colors have been verified by me, my multimeter, and a sacraficial Psion cable [which I made into a non-null-modem cable]. Whether there is or isn't a DCD signal out of the Psion itself (e.g., pin 15 is the likely candidate), I couldn't say, but there is not even a wire in the Psion cable for the DCD signal. The Psion connector type seems to be the same as that on a somewhat standard 15-Pin PCMCIA Modem Cable Dongle, although the two cables are "keyed" differently (the plastic notches don't match).

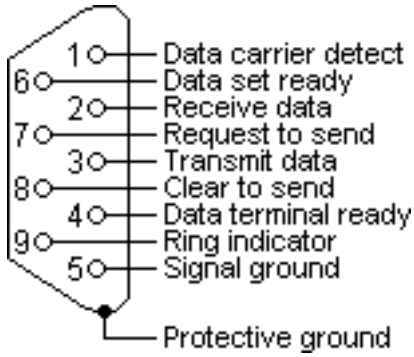
```
(25-pin D) 9-pin D   Signal    Psion       Wire Color
8-----------  1 ------DCD--- not connected   (not connected to 9-pin or Psion pin)
3-----------  2 --RX-- /--TX---  12          green
2-----------  3 --TX--/ --RX---   8         yellow
20----------  4 ----- DTR -----  11         orange
7-----------  5 ----- Gnd -----   1          black
6-----------  6 ----- DSR -----  13            red
4-----------  7 -RTS-- /--CTS--   9         purple
5-----------  8 -CTS--/ --RTS--  14           blue
22----------  9 ------ RI -----  10          brown

Cable ground ---------------- bare wire
```

With the Psion's cable pins defined as:



And the 9-pin D pins defined



See also: www.lammertbies.nl/comm/cable/RS-232.html (http://www.lammertbies.nl/comm/cable/RS-232.html) for general information on serial pins.

21.9. Are There Any Issues with the Malaysian netBooks?

Kernels that work on a netBook will work on a Malaysian netBook. They appear to have identical hardware to the netBook, but have some small software changes to their OS.img loader which is irrelevant for our purposes. I don't think we are faced with the problem of myriad hardware configurations, which would be a problem for kernel development.

21.10. What Are the Kernel Differences Between Series 7/netBook?

Series 7 kernels work for the netBook. The hardware is essentially the same, except for the memory and perhaps the power supplied to PCMCIA. We've had no one report what happens when a netBook kernel (e.g., configured for 64 MB of RAM) is booted on a Series 7 (with 16 MB of RAM).

21.11. What Effect Does a Memory Upgrade Have/Is All Memory Recognized?

Kernels are now available that will recognize either 32 or 64 MB of RAM. If you have upgraded your netBook to have 64 MB of RAM, a linux kernel is available that will recognize this. If a Series 7 kernel is used, then only 16 MB of memory is available. At present, separate kernels are required to recognize these various memory configurations; please get the kernel appropriate to your netBook (the effect of a 64MB kernel on a 32MB netBook is unknown, and you probably don't want to find out).

Unfortunately, it appears that the netBook or kernel ramdisk has problems - attempting to transfer large amounts of data (10-15 MB) to the netBook (to a separate ramdisk, say) will gradually consume all the memory and cause the system to freeze up. I think this may be a bug in the ramdisk driver or netBook's memory patch. Using a ramfs or tmpfs (the preferred filesystem), rather than ramdisk, seems to work o.k.

21.12. Is it safe to send my netBook through an airport X-ray machine?

There have never been any reports (www.psionplace.com/ (http://www.pdastreet.com/forums/) search for X-ray - the discussion here is a paraphrase of the discussions there.) that netBook's sent through the carry-on security X-rays experience any problems because of it. I have sent my netBook through the X-rays many times. I'd never check any of my 'gadget' equipment in any case, because the threat of theft or damage outweighs everything else...

It is true, however, that bags *checked* at the airport go through very high intensity X-rays. Films and memory and CF are likely to be erased. One film cameraman told me that the Xray on checked luggages is 10 to 20 times more intense than on hand-carry items. He said that now all film cameramen have been advised to hand carry films and insist on visual inspection. The advisory came from the US authorities. I have two sticks of 512 Meg DRR ram which were placed in luggage. After travelling from Singapore to Boston via London, the rams were dead. Hand-carry laptops and PDAs were ok.

Technically, X-rays will hurt any type of electronic eqiupment, eventually. Chances are that nothing will happen, but every time you expose electronic equipment to radiation there is a small chance that electronics will be damaged, the more exposure the greater the chance of damage. People who work with all types of radiation in the medical field are required to change out many of the electronics (boards) at a regular interval because of damage due to exposure. However, there is a pretty significant difference between hardened application-specific equipment used in high-radiation environments (X-rays, gamma rays, etc.) with a specified change date to guarantee performance to standard and a PDA. The amount of radiation a PDA would be exposed to during an airport scan should not have any effect on the equipment. Even a frequent flyer need not worry.

In short, carry on your netBook?...O.K., but check your netBook?...NO!

21.13. Is possible to rebuild the netBook's Lithium-Ion battery pack with new batteries?

Yes it is! See Rebuilding a netBook's Lithium-Ion Battery (http://linux-7110.sourceforge.net/howtos/netbook_new/LIbattery/LIbattery.html). Interestingly, you can rebuild it with batteries of a higher amp-hour rating (2200 mAh or larger) and get 12 or so hours of battery time.

21.14. How can I manually unpack an *.ipk package?

These packages seem to be packed up in a few different ways. Within each package is a control.tar.gz tarball which describes the package and how it is to be configured and a data.tar.gz tarball which contains the actual binary and data files of the package. To unpack an ipk and get at the binaries (only the data.tar.gz file is needed for the binaries), try any of: "ar -x newpackage.ipk", or "tar xfz newpackage.ipk", or "tar xf newpackage.ipk". With the data.tar.gz file extracted, this can be installed with "tar xfz data.tar.gz -C /", where "-C /" extracts the file contents to the root directory. Sometimes only a single binary is needed, however, so a local unpacking will work fine, just "tar xfz data.tar.gz" and copy the binary file of interest to your favorite place. Manual unpack and installation of ipk files to a Debian system in this way frequently works without any problems (although there is no management of the installation).

21.15. How can I manually unpack a *.deb file?

You can do this a variety of ways, but perhaps the most straightforward is "ar -x packagename.deb". This will give two tarballs control.tar.gz and data.tar.gz. The first is the information dpkg needs to do a proper installation and configuration of the package, the second is contains the binaries and data files.

22. Floating Point Emulation Notes

The netBook's ARM cpu does not have hardware floating point capabilities, so floating point must be emulated. This emulation can be done in a number of ways. These are some enlightening notes about this subject I gleened off the web. The conclusion is that we would be better off with software, rather than kernel, floating point emulation, but it is difficult to assemble such a system. In the answers, "np" is Nicolas Pitre who is one of the experts on ARM floating point emulation. NWFPE is "Netwinder Floating Point Emulation", and "FastFPE" is Peter Teichmann's Fast Floating Point Emulation. The latter is notorious for rather crude results for floating point, and has caused Debian's dpkg/perl to fail, though it is 3-5 times faster than NWFPE. Both NWFPE and FastFPE are options that are selected when the kernel is configured - one must choose one or the other.

22.1. 3. What's the difference between gcc soft-float support and the kernel NWFPE/FastFPE support?

[31 January 2004 - np] Scenario 1 (hard-float): The compiler emits opcodes designed to be used with a hardware floating point coprocessor (FPU). The FPU usually has a set of extra registers for its use, and the compiler may as well pass floating point argument to functions through those registers. This is of course the best performing solution when a real hardware FPU is used.

Scenario 2 (soft-float): the compiler converts floating point operations into function calls and a special library is used to provide all functions performing the required operations, all in software with no FPU instructions at all. There is obviously no extra floating point registers available in this case, therefore all FP arguments to functions have to be passed through standard registers or on the stack. This is of course the best performing solution when no hardware FPU is available, given that the library implementing the FP operations is optimally coded.

Now, unfortunately, the default on ARM Linux has traditionally been set to have the compiler use hard-float, even if ARM Linux never ran on any ARM CPU with a real hardware FPU. The CPU is therefore raising the invalid instruction exception each time some FPU opcode is encountered. Then the kernel traps that exception, looks at the given FPU instruction and emulates it in software. But here not only must the kernel perform the FP operation, it must also emulate the whole hardware FPU as well. That's what NWFPE or FastFPE are doing. This is obviously the worst performing arrangement that can be due to the exception trap and emulation overhead.

22.2. 4. Can I use both hard & soft float at the same time, or must I choose one of them?

[31 January 2004 - np] Depends. You can use both at the same time, but not in the same application. The problem has to do with the ABI incompatibility between soft-float and hard-float due to the different floating point argument passing conventions. So, if you decide to switch to using soft-float for some application, you MUST recompile ALL the libraries that application is going to use, including system libraries like the C library. If those libraries are

dynamically linked libraries, you then must also recompile all the applications that share those same libraries. And then your application will run correctly only on systems with soft-float environments, unless you link it statically in which case it will run anywhere (even on a kernel with NWFPE configured in -- it will simply not be invoked). That's the main reason why mainstream ARM distributions are still reluctant to switch to soft-float because of the associated compatibility pain.

23. Extracting a Zaurus ROM (MTD and JFFS2) filesystem

There are a number of ROM images that have been put together for the Sharp Zaurus. With some modification, these images make suitable disk images for the netBook. This chapter shows how to mount these ROM images and convert them to the usual ramdisk image.

23.1. MTD support on your system

The Zaurus ROM images use the jffs2 filesystem on MTD disks. I'm not quite sure what that means, but I think MTD is Memory Technology Device and jffs2 is a filesystem designed to work well and efficiently on MTD disks. In any case you will need mtd/jffs2 support on your desktop to access the Zaurus ROM image.

I started by reconfiguring my RedHat linux kernel to have the mtdcore, jffs2, mtdram, mtdchar and mtdblock modules. I recompiled and installed these modules. Later, I had a Suse 9.1 system, and all these things were available by default - nothing to do! Install these modules by:

1. modprobe mtdcore
2. modprobe jffs2
3. modprobe mtdram total_size=32768 erase_size=256
4. modprobe mtdchar
5. modprobe mtdblock
6. mknod /dev/mtdblock0 b 31 0

The /dev/mtdblock0 device was already available on my Suse 9.1 system; you may not need to make it.

The next two steps are easy:

1. dd if=zaurus.initrd.bin of=/dev/mtdblock0
2. mount -t jffs2 /dev/mtdblock0 /mnt/disk

where "zaurus.initrd.bin" is the Zaurus ROM image and "/mnt/disk" can be any convenient mount point. The ROM filesystem is now available in /mnt/disk.

23.2. Extracting the Zaurus filesystem

You now need to extract the stuff from /mnt/disk and put it into an ext2 initrd file. From "/mnt/disk" do:

1. dd if=/dev/zero of=/tmp/boot.img bs=1024 count=57600

2. mke2fs -m 3 /tmp/boot.img
3. mount -t ext2 -o loop /tmp/boot.img /mnt/disk2
4. tar cf - * | (cd /mnt/disk2 ; tar xf -)

where "/mnt/disk2" is another convenient mount point.

Now the Zaurus filesystem has been transfered to "/mnt/disk2" a ramdisk. You can then customize it for your own purposes there. Don't forget to unmount the mtd disk "/mnt/disk"; you are done with it.

When you are happy with your new ram disk, you should first repeat the steps above for making another ext2 ram disk ("/tmp/boot2.img" on "/mnt/disk3" ?) and tar the stuff from /mnt/disk2 to it. This will efficiently pack the filesystem; your modifications in /mnt/disk2 will have left a bunch of holes in the filesystem which will make the compressed initrd file unnecessarily large.

When you are ready, first change directory to where you want to save the new initrd. And then:

1. umount /mnt/disk3
2. cp /tmp/boot2.img initrd
3. gzip -9 initrd

And you have your new initrd.gz file!

24. Changes

- July 9, 2006: Added link to the Qemu emulated ARM system. An alternative to cross compiling.
- May 25, 2006: Minor additions.
- January 20, 2006: Added info on SA1100 cache size. Optimal gcc compile options corrected. Browser discussion update. Added an "Optimizations and Adjustments" section. Added information on the "stardict" dictionary.
- October 24, 2005: Fixed links to old Wiki pages. The wiki pages are saved, but they are no longer wikis. Added comments on using swap space under the CF install page. Revised the warning on burning out your netBook's motherboard using an external PCMCIA device with external power supply. Included a short description on fixing the fuse after this problem occurs.
- July 25-27, 2005: More on IDE devices; CD-ROM and external hard disks are supported.
- July 10-13, 2005: Minor changes throughout; revision of Applications section (addition of xephem); revision of X windows section; start of Serial Ports section; start of PCMCIA IDE section.
- May 18, 2005: SIAG added; small corrections.
- April 22, 2005: Deprecated the phpwiki page links; added a hardware.txt file that lists supported hardware.
- April 9, 2005: Updated using a mouse with a VGA out card; VGA out card driver allows only 16 colors at the moment.
- April 5, 2005: "Changes" reorganization, mouse corrections, gnumeric works o.k.
- March 26, 2005: Updated the comments on the use of a serial mouse on the netBook's serial port. It seems to work just fine, but warnings are probably still warranted. Added a link for the "simputer" from India.

- January 13: Updated kernel compiling information. More on internal serial port.
- January 1: Added a section on ArLo; more minor clean ups/updates.
- October 23 2004: More on touch screen; some minor clean ups.
- October 3: Minor development for touch screen support; addition to FAQ for manually unpacking ipk packages.
- September 9: Added a link for information on rebuilding the LI-ion battery; added a list of things that are not yet working.
- August 9: Using the netBook for VGA-out presentations on an HP F1252A PCMCIA card.
- August 6: Added a chapter on using PPP from the netBook; comments on using java (the situation looks bleak...); screenshots of dillo and gpsdrive under Applications.
- August 1: Continued minor development, GPS on the netBook, brief comments on alternate window systems.
- July 17: Added "Using a Mouse on a netBook" and "Using Bookboot - the Linux OS.img" sections.
- July 14: More development - X windows, OpenPsion name change.
- June 29-July 1 2004: Lots of development directed toward installing a compactflash-based system.
- June 28 2004: Added a recipe for extracting the filesystem from the Zaurus ROM images.
- June 27 2004: News and links on the new compactflash and PCMCIA support. Updated links to IPKG feeds from the IPAQ and Zaurus people. General preliminary overhaul to start to adapt this document to compactflash usage.
- January 31 2004: Minor corrections, additions; added an article "Porting the Linux Kernel to a New ARM Platform" by Wookey and Tak-Shing (see "Developer's Notes", under "Some Documentation").
- January 20 2004: Minor corrections, additions; see screenshot under Applications!
- January 12 2004: Added an Applications section.
- January 10, 2004: Added a FAQ section; added a "Developer's Notes" section; kernel instructions updated for 2.4.21 kernel.
- December 27, 2003: A general update; fixing and updating links
- June 21 2003: Added a recipe for compiling the kernel.
- June 19 2003: Added a kernel wishlist.
- May 4 2003: Minor development of HOWTO, inclusion of a chapter on "Installation". Kernels for 32 and 64MB memory are available.
- May 1 2003: Announcement that linux WILL boot after all - a small tweek to the arlo.cfg file was needed.
- Feb. 24 2003: Addition of XTM and Elks. List of kernel development needs.
- Feb. 8 2003: Initial webpage set up. Full of optimism.