UNC20 Developer's Kit with CompactFlash

Getting Started Guide

® Copyright 2003:

PO Box 292528, 43229
☎+1 888 546 9741
info@es-usa.com

- Columbus OH (USA)
- Fax +1 888 546 9741
- www.es-usa.com

Kueferstrasse 8
☎+49 7667 908-0
sales@fsforth.de

- Breisach (Germany)
- Fax +49 7667 908-200
- www.fsforth.de

Calvo Sotelo 1, 1º - Dcha
☎+34 941 270 060
sitemas@embebidos.com

- Logroño (Spain)
- Fax +34 941 237 770
- www.embebidos.com

Release of document: December 04, 2003
Filename: UNCBASCF_GS.doc
Author: Héctor Palacios, Nigel James
Version: 1.1

Table of Contents

1. Documentation History

Date	Version	Responsible	Description
2003-12-04	1.0	Héctor Palacios	Release
2003-10-24	0.2	Pedro Pérez de Heredia	Revision
2003-10-23	0.1	Héctor Palacios	Initial version

2. Introduction

The UNC20 Developer's Kit with CompactFlash is a member of a family of application-specific Developer's Kits based on the popular UNC20 module and provides a very cost-effective platform for low to medium volume embedded solutions with a need for CompactFlash expansion. It consists of a UNC20 module; a base board (UNCBASCF) which is assembled with a type-II CompactFlash slot; and LxNETES, FS Forth-Systeme's uClinux distribution.

The UNC20 Developer's Kit for LxNETES runs the well-known and proven uClinux operating system and comes with all the necessary cross-development tools. Any programmer with a Linux background will feel familiar with the environment in just a few hours. For further details, please consult the LxNETES User's Manual on the LxNETES CD (P/N 999).

This manual guides the user in installing the CompactFlash project, which is provided on the documentation CD (P/N 9024), on LxNETES. The manual guides the user in the creation of a kernel with support for a CompactFlash storage card and a CompactFlash 802.11b wireless LAN card, and explains how to use them.

This document assumes that the user has basic knowledge of Linux. In addition, it is recommended that the reader has experience with compiling a standard Linux kernel on the host.

3. What's on the CD (P/N 9024)?

The CompactFlash Documentation CD contains all the software and documentation which are specific to the UNC20 Developer's Kit with CompactFlash and has the following sub-directories:

3.1. doc

This Getting Started Guide as a PDF file, the schematics and BOM of the baseboard, plus Hardware User's Manual for the baseboard, UNC20 module and NS7520 processor. Also Linux man files for the wireless tools.

3.2. images_uncbascf

All pre-built images, for example "linux.bin" and "jffs2.img".

3.3. logic_uncbascf

The JTAG files for the CPLD used as the interface to the CompactFlash.

3.4. software

All software related specifically to the UNC20 Developer's Kit with CompactFlash. This software must be installed on top of the standard LxNETES distribution, which is available on a separate CD.

3.5. uClinux

The PPJ software for programming the Flash via the JTAG interface.

3.6. WinNT

The Kithara software required to access the PC's parallel port in User mode.

4. Requirements

A storage CompactFlash card, formatted in MS-DOS mode, is necessary for the demo of the storage card. This is provided with the Developer's Kit.

A CompactFlash 802.11b wireless LAN card and a wireless LAN access point are necessary for the demo of wireless Ethernet. A CompactFlash 802.11b wireless LAN card is provided with the Developer's Kit. The wireless access point is not part of the Developer's Kit.

Note: UNC20 Developer's Kit with CompactFlash requires LxNETES v2.3 or later.

4.1. Caveats

4.1.1. PORTC0 Conflict with bas_test Demo

Among the sample applications delivered with LxNETES, there is one called ***bas_test***. This application shows how to control the two user buttons and LEDs and therefore configures PORTC0 as an input.

The CompactFlash interface uses PORTC0 pin as an interrupt pin, therefore the *bas_test* application should not be run while the CompactFlash modules are loaded. Be sure to remove BASTEST environment variable from your EEPROM so that *bas_test* is not automatically launched. Also, the user should avoid pressing the PORTC0 button to prevent undesired interrupts. These unserviced interrupts will delay access to the CompactFlash.

4.1.2. Rescue Kernel

Per default the kernel is configured to include the 'rescue kernel' feature. This feature allows the user to boot a rescue kernel (saved in Flash partition 5) by pressing PORTC0 button while powering up the target.

The 'rescue kernel' feature makes use of the PORTC0 line (push button) and some CompactFlash cards may leave this line active during operation. In that case, the target would boot the rescue kernel after pressing the reset button even though the push button PORTC0 is not active. You can detect that the 'rescue kernel' is booting if the serial output begins showing this message:

```
LxNETES Bootloader $Revision: 1.19 $
ABCDEFGH
*** Using Rescue Image ***
```

This effect can be solved by a cold reset or a software call to 'reset' program, which will boot the normal kernel. Or you can choose to disable the 'rescue kernel' feature when configuring the kernel.

5. Installation

It is assumed that you have previously successfully installed LxNETES according to the instructions given in the LxNETES User's Manual.

Insert the CompactFlash Documentation CD (P/N 9024) into your CD-ROM drive and mount it. Then change to your CD-ROM drive folder and run the script *install.tcl*:

```
[user@ezcaray /]# mount cdrom
[user@ezcaray /]# cd /cdrom
[user@ezcaray cdrom]# ./install.tcl
```

You will be asked whether you want to install apps sources. If you answer 'y', then the source files of the wireless tools will be installed to your /targets/LxNETES/ewp/apps/ folder.

The installation process installs a template project called **project_uncbascf** in /targets/LxNETES/. Also some kernel patches for Compact Flash support are installed locally within the linux folder of this project. Additionally, pre-built binaries of the wireless tools ported to LxNETES are copied to your LxNETES distribution apps/bin folder.

6. uncbascf Project

The **uncbascf** project includes all configuration and application source files to quickly develop a uClinux kernel for the uncbascf platform, with support for Compact Flash storage and wireless LAN cards.

The project directory is **/targets/LxNETES/project_uncbascf**

6.1. Directory Structure

The **uncbascf** project contains the following files and directories:

LxNET.sh	Shell script to configure environment variables
LxNET.csh	As above, but for csh users
Makefile	Rules to build the kernel, the sample applications, the rootfs and install it to the destination folders
bin/	The resulting binary image with the kernel and the rootfs will be placed here
build.sh	Script to build the rootfs
etc/	The etc/ folder of the rootfs
linux/	Kernel configuration and object files
prepare.sh	Script to update symbolic links to the LxNETES kernel sources
rootfs/	The rootfs is temporarily created in this folder
apps/	Folder for user applications
kernel_patches/	Folder that contains kernel patches to support the

	CompactFlash storage and WLAN.

6.2. Testing the CompactFlash Storage Card

The factory-default kernel is pre-configured to support the CompactFlash storage card which is delivered with the Developer's Kit.

Insert an MS-DOS FAT16 formatted CompactFlash storage card into the socket on the UNCBASCF board while it is powered off (note that hot plugging of cards is not supported).

Now power on the target and wait for the kernel to start up.

IMPORTANT: make sure that jumper J1 pins 1-2 is not inserted, otherwise the CompactFlash card will not be recognized.

Since we have configured the drivers of ATA and FAT to be compiled as modules, these services will not be available until we load them explicitly.

You have two methods of loading these modules. The easiest one consists of adding a variable to the EEPROM that will load the modules automatically after startup. This is accomplished by running the following command in your target console:

```
# eeprom -a MSDOSMOUNT=yes
New EEPROM contents:
  Checksum: 12
  EEPROM Version: 0
  Use DHCP: yes
  MAC Address: 00:04:f3:00:0d:5c
  IP Address: 192.168.50.45
  Subnet Mask: 255.255.255.0
  Gateway: 192.168.50.5
  DNS: 212.163.200.2
  Auto Negotiation: enabled
  Environment variables:
     JFFSMOUNT=yes
     MSDOSMOUNT=yes
#
```

After the EEPROM has been updated you must reset your target and the modules will be automatically loaded. Additionally, the CompactFlash card will be mounted in the folder /cf.

The other method consists of manually loading all the required modules:

```
# insmod ide-core.o options="ide0=0;0;0"
Using /lib/modules/2.4.22-uc0-fs1/kernel/drivers/ide/ide-core.o
ide_setup: ide0=0;0;0

Uniform Multi-Platform E-IDE driver Revision: 7.00beta4-2.4
ide: Assuming 50MHz system bus speed for PIO modes; override with
   idebus=xx
# insmod ide-detect.o
Using /lib/modules/2.4.22-uc0-fs1/kernel/drivers/ide/ide-detect.o
hda: Flash Card, CFA DISK drive
ide0 at 0x6000008-0x600000f,0x600000e on irq 0
# insmod ide-disk.o
Using /lib/modules/2.4.22-uc0-fs1/kernel/drivers/ide/ide-disk.o
hda: attached ide-disk driver.
hda: 62720 sectors (32 MB) w/1KiB Cache, CHS=490/4/32
Partition check:
 /dev/ide/host0/bus0/target0/lun0: p1
# insmod fat.o
Using /lib/modules/2.4.22-uc0-fs1/kernel/fs/fat/fat.o
# insmod msdos.o
Using /lib/modules/2.4.22-uc0-fs1/kernel/fs/msdos/msdos.o
#
```

And finally mounting the CompactFlash card manually with this command:

```
# mount -t msdos /dev/ide/host0/bus0/target0/lun0/part1 /cf
```

Now you can change to this folder /cf and copy or delete files to you card.

IMPORTANT NOTE: As is the case with other removable devices, Linux may delay the synchronization of files on the CompactFlash card, therefore, after doing an I/O operation on the CompactFlash, you must wait some time before powering off (or resetting) the target, or else unmount the CompactFlash folder with the command 'umount /cf' to force synchronization of files. If you power off (or reset) the target immediately after an I/O operation to the CompactFlash, synchronization may not occur and the changes might not be there the next time.

```
# umount /cf
```

6.3. Testing the CompactFlash Wireless LAN Card

The factory-default kernel is pre-configured to support the CompactFlash WLAN card which is delivered with the Developer's Kit.

This demo assumes that you have a wireless LAN access point configured to work in your Ethernet LAN.

Insert the CompactFlash 802.11b wireless LAN adapter card into the socket of your UNCBASCF board while it is powered off (note that hot plugging of cards is not supported).

Now power on the target and wait for the kernel to start up.

IMPORTANT: make sure that jumper J1 pins 1-2 is not inserted, otherwise the CompactFlash card will not be recognized.

Since the wireless LAN drivers are loadable modules, these services will not be available until we load them explicitly.

You have two methods of loading these modules. The easiest one consists of adding a variable to the EEPROM that will load the modules automatically after startup. This is accomplished by running the following command in your target console:

```
# eeprom -a WIRELESS=yes
```

```
New EEPROM contents:
  Checksum: 12
  EEPROM Version: 0
  Use DHCP: yes
  MAC Address: 00:04:f3:00:0d:5c
  IP Address: 192.168.50.45
  Subnet Mask: 255.255.255.0
  Gateway: 192.168.50.5
  DNS: 212.163.200.2
  Auto Negotiation: enabled
  Environment variables:
     JFFSMOUNT=yes
     WIRELESS=yes
#
```

After the EEPROM has been updated you must reset your target and the modules will be automatically loaded.

The other method consists of manually loading all the required modules:

```
# insmod hermes.o
Using /lib/modules/2.4.22-uc0-
    fs1/kernel/drivers/net/wireless/hermes.o
# insmod orinoco.o
Using /lib/modules/2.4.22-uc0-
    fs1/kernel/drivers/net/wireless/orinoco.o
# insmod orinoco_unc20.o
Using /lib/modules/2.4.22-uc0-
    fs1/kernel/drivers/net/wireless/orinoco_unc20.o
orinoco_unc20: CF wireless card initialized
#
```

Normally the built-in Ethernet of UNC20 the will be configured as interface eth0 and the IP address from the EEPROM will be assigned to it, while the wireless interface will be named as eth1 and won't be assigned an IP address. In this case, you must first assign a second IP address to the wireless interface by running the command:

```
# ifconfig eth1 192.168.50.46
eth1: New link status: Connected (0001)
```

where 192.168.50.46 is a free address in your LAN different to the one assigned to interface eth0. It is recommended to use an address in a different subnet to that of the eth0 interface. Note that the wireless access point must be up and running, otherwise you will get the message:

```
eth1: New link status: Disconnected (0002)
```

You can display more information with:

```
# ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 00:00:CB:10:04:14
          inet addr:192.168.50.46  Bcast:192.168.50.255
   Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
#
```

Now you can test the CompactFlash wireless LAN adapter by doing a ping from the target to any IP address in your new subnet, e.g. ping 192.168.50.1. Likewise, you should be able to ping the target from any host in this subnet – ping 192.168.50.46.

If you have decided to use an IP address for your WLAN card in the same subnet of the built-in Ethernet interface eth0, the Ethernet traffic will go through your default interface eth0. In order to redirect the traffic to your Wireless interface, you must configure the routing table of your target with the following commands:

```
# route del default
# route del –net 192.168.50.0 netmask 255.255.255.0 dev eth0
# route add default gw 192.168.50.5 dev eth1
```

where 192.168.50.5 is the address of the gateway of your network.

Your routing table should finally look like this:

```
# route
Kernel IP routing table
Destination    Gateway       Genmask         Flags Metric Ref Use Iface
192.168.50.0   *             255.255.255.0   U     0      0   0    eth1
127.0.0.0      *             255.255.255.0   U     0      0   0    lo
default        192.168.50.5  0.0.0.0         UG    0      0   0    eth1
#
```

If you have disabled the UNC20's built-in Ethernet when configuring the kernel, the CompactFlash wireless LAN adapter card will be configured as interface eth0 and the default IP address from the EEPROM will be assigned to it.

Once the wireless interface is up and running, it should behave just like the built-in Ethernet interface of the UNC20. Therefore, you can mount remote units through nfs, access the board with telnet and do ftp transfers.

6.4. Wireless Tools

The standard Linux wireless tools (version 26) have been ported to the UNC20 and are delivered (pre-built) with the project under the folder $LXNETES_APPS_PATH/bin/.

If, during the installation process, the user has selected to install the apps sources, the source files for these tools are stored under /targets/LxNETES/ewp/apps/wireless_tools.

By default, these tools are not included into the root file system because they use a lot of space. Only the iwconfig tool is included in the root file system.

In the following paragraphs a short description of each tool is given. For more information, the user can consult the man pages of these tools, included on the CD under subfolder doc/wireless_tools/man_pages/

6.4.1. iwconfig

This tool is similar to ifconfig but dedicated to the wireless interfaces. It is used to set the parameters of the network interface which are specific to the wireless operation (frequency, channel, encryption,...).

6.4.2. iwevent

This tool displays wireless events received through the RTNetlink socket.

6.4.3. iwgetid

This tool is used to retrieve the NWID, ESSID or AP/Cell address of the wireless network that is currently used.

6.4.4. iwlist

This tool lists wireless statistics (frequency, access points in range, bit-rates) of the specified node.

6.4.5. iwpriv

This is the companion tool to iwconfig. iwpriv configures parameters and settings specific to each driver (as opposed to iwconfig which deals with generic ones).

6.4.6. iwredir

This tool is used to redirect the use of multiple versions of the wireless tools according to the WE (Wireless Extension) of the current kernel.

6.4.7. iwspy

This tool is used to retrieve quality of link information of a wireless device.

6.4.8. macaddr

This tool is used to retrieve the MAC address of an interface.

```
# ./macaddr eth1
00:00:CB:10:04:14
```

```
#
```

6.5. Encrypting the WLAN network

If we want to secure our wireless network, the Access Point must be configured to encrypt the network with a key. This key can be given with several hex numbers (or with the corresponding ASCII string).

To configure you WLAN card to connect to the encrypted network, you must execute the command 'iwconfig' with these command line:

```
# iwconfig eth1 key s:MyKey
```

Where 'MyKey' is the key used to encrypt the network given as an ASCII string. For more information see the man page of 'iwconfig' tool.

7. Building the uncbascf Project

You probably started by using the default kernel image stored in Flash on the UNC20 (also available on the CD in the images_uncbascf directory). The next step is to rebuild the kernel on the host to familiarize yourself with the build process.

Change to the project directory, i.e. /targets/LxNETES/project_uncbascf and run the script 'LxNET.sh' to set up the environment variables.

Now the project is ready to compile. Just enter “make”!

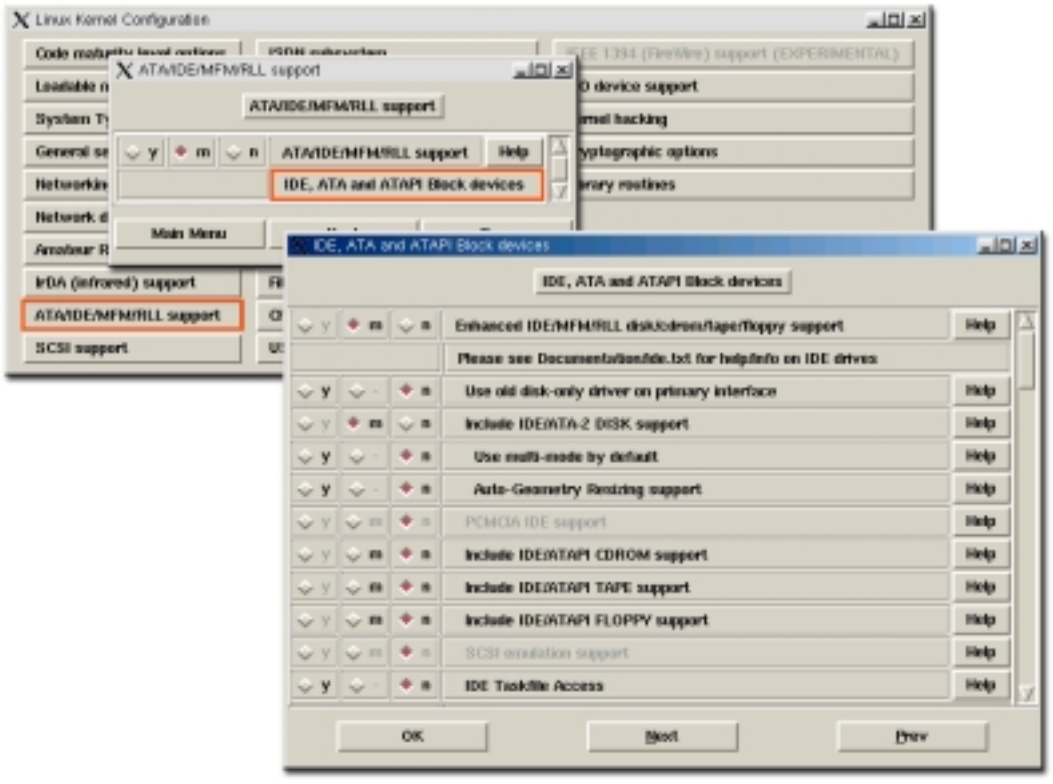
This will rebuild the kernel using the default kernel configuration which is setup to support the CompactFlash cards. The following sections are for reference and show which settings are needed to include CompactFlash support for the storage card and WLAN card in the kernel.

7.1. Adding Support for CompactFlash Storage Cards

In order to configure the kernel to support CompactFlash storage cards, change into the subfolder linux/, i.e. /targets/LxNETES/project_uncbascf/linux, and run *'make xconfig'*.

7.1.1. Include ATA/IDE support

Click on the *ATA/IDE/MFM/RLL support* button. Another window will appear. Check the 'm' option to include support of ATA/IDE/MFM/RLL as a module. Then click on the *IDE, ATA and ATAPI Block devices* button.



Another window will appear. Check the 'm' option for *Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support* and for *Include IDE/ATA-2 DISK support* to include these as modules. Finally, click OK and go back to the Main Menu.

7.1.2. Include MS-DOS FAT File System Support

Click on the *File systems* button. Another window will appear. Check the 'm' option for *DOS FAT fs support* and for *MSDOS fs support* to include these as modules.

7.1.3. Include Partition Support

Still within the *File systems* window, click on the *Partition Types* button. A new window will appear. Check the 'y' option for *Advanced Partition selection* and for *PC BIOS (MSDOS partition tables) support* to have direct support on the kernel to handle MS-DOS partitions.

Click OK and return to the Main Menu. Then save the changes.

You can now compile a kernel with support for CompactFlash storage cards formatted in MS-DOS FAT16. Go to the project folder and run 'make'. The compilation process will take several minutes. It compiles the kernel, the applications, the Root File System and compresses all into a binary CRAMFS image called 'linux.bin' which is stored under the subfolder bin/.

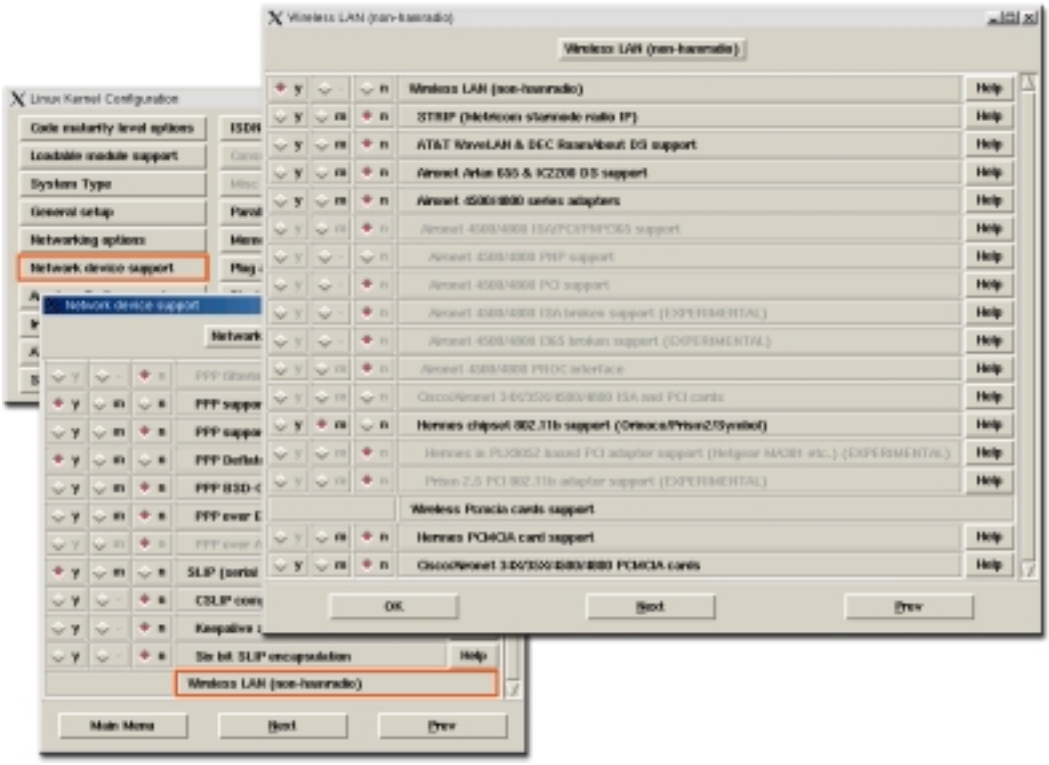
Refer to the LxNETES User's Manual for instructions about updating your target with the new image.

7.2. Adding support for CompactFlash Wireless LAN

In order to configure the kernel for CompactFlash 802.11b wireless LAN adapter cards support, change to the subfolder linux/ and run *'make xconfig'*.

7.2.1. Include Wireless LAN support

Click on the *Network device support* button. A new window will appear. Now click on the *Wireless LAN (non-hamradio)* button. A new window will appear.



Check the 'y' option for *Wireless LAN (non-hamradio)* and check the 'm' option for *Hermes chipset 802.11b support (Orinoco/Prism2/Symbol)* to support this as a module.

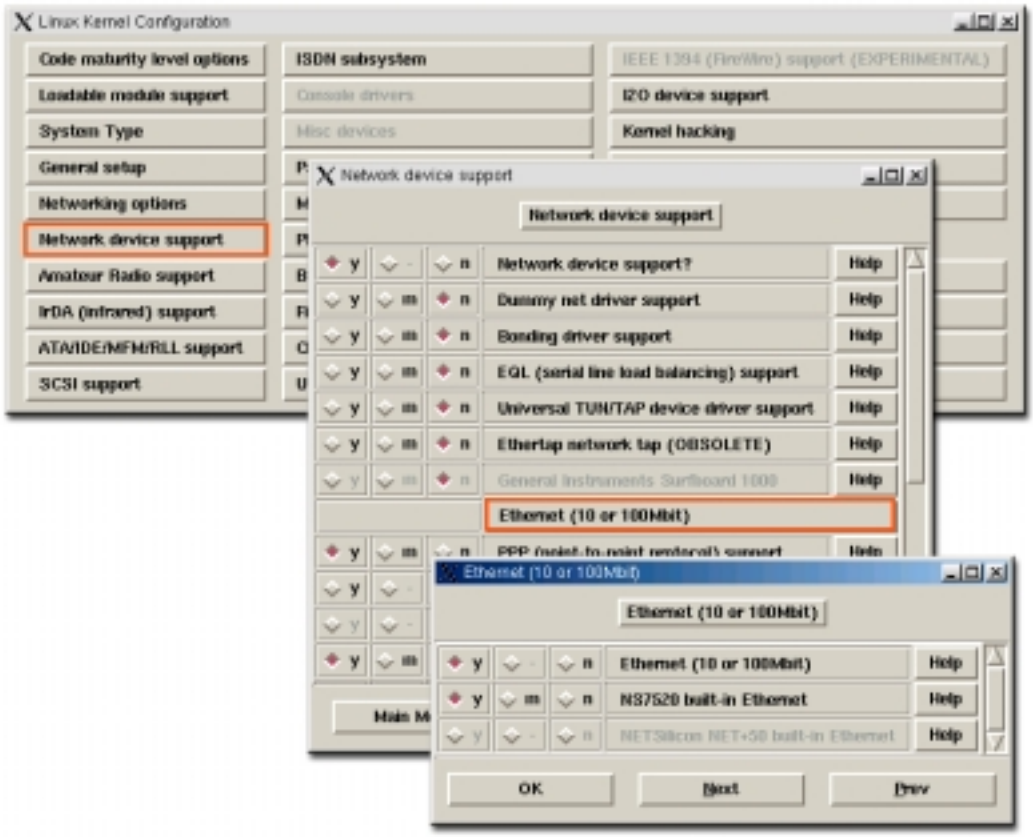
Click OK and return to the Main menu. Then save the changes.

You can now compile a kernel with support for CompactFlash 802.11b Wireless LAN Adapter cards. The compilation process will take several minutes. It compiles the kernel, the applications, the root file system and compresses all into a binary CRAMFS image called 'linux.bin' which is stored under the subfolder bin/.

7.3. Enabling / Disabling UNC20's built-in Ethernet interface

Since the UNC20 has a built-in Ethernet interface, you can configure the kernel to support it or not. If you decide to support it, you will finally have two Ethernet interfaces: the NS7520 built-in controller (which will appear as the kernel's default Ethernet driver 'eth0') and the CompactFlash wireless LAN adapter (which will appear as 'eth1').

Click on the *Network device support* button. A new window will appear. Click on the *Ethernet (10 or 100 Mbit)* button. A new window will appear. Here you can choose whether to enable support for the built-in Ethernet interface or not.


Linux Kernel Configuration
Code maturity level options
Loadable module support
System Type
General setup
Networking options
Network device support
Amateur Radio support
IrDA (infrared) support
ATA/IDE/MFM/RLL support
SCSI support
ISDN subsystem
Console drivers
Misc devices
IEEE 1394 (FireWire) support (EXPERIMENTAL)
I2O device support
Kernel hacking
Network device support
Network device support
Network device support?
Dummy net driver support
Bonding driver support
EQL (serial line load balancing) support
Universal TUN/TAP device driver support
Ethertap network tap (OBSOLETE)
General Instruments Surfboard 1000
Ethernet (10 or 100Mbit)
Help
Ethernet (10 or 100Mbit)
Ethernet (10 or 100Mbit)
Ethernet (10 or 100Mbit)
NS7520 built-in Ethernet
OK
Next
Prev