AN010
Software Manual

Version 1.0.2

February 11, 2005

Atmark Techno, Inc.
http://www.atmark-techno.com/

Armadillo Official Site
http://armadillo.atmark-techno.com/

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1. About This Manual

This document covers the following areas of information necessary for using the Armadillo-9.

- Rewriting Flash Memory
- Basic Usage
- Kernel and Userland Builds
- Application Development

We hope the information in this document will help you get the best functionality out of the Armadillo-9.

1.2. Typographical Conventions

The following typographical conventions are used in this manual.

Table 1-1 Fonts

Fonts	Description
Fonts in text	Text
[PC ~]$ ls	Prompt and user input character strings

1.3. Command Input Example Conventions

The command input examples contained in this manual are based on the assumed execution environment associated with the respective display prompt. The directory part "/" will differ depending on the current directory. The home directory of each user is represented by "~".

Table 1-2 Relationship between Display Prompt and Execution Environment

Prompt	Command Execution Environment
[PC /]#	To be executed by a privileged user on Work PC
[PC /]$	To be executed by a general user on Work PC
[armadillo9 /]#	To be executed by a privileged user on Armadillo-9
[armadillo9 /]$	To be executed by a general user on Armadillo-9

1.4. A Word of Thanks

The software used in the Armadillo-9 is composed from Free Software / Open Source Software. This Free Software / Open Source Software is the result of efforts from developers from all over the world. We would like to take this opportunity to express our gratitude.

1.5. Precautions

The software and documentation contained in this product is provided “AS IS” without warranty of any kind including warranty of merchantability or fitness for a particular purpose, reliability, correctness or accuracy. Furthermore, we do not guarantee any outcomes resulting from the use of this product.

2. Before Getting Started

2.1. Preparation

Please ready the following before using the Armadillo-9.

- Work PC
  A PC that runs either Linux or Windows and has at least one serial port.

- Serial cross cable
  A D-Sub9-pin (female-to-female) cable for cross connections.

- Supplied CD-ROM
  This CD-ROM contains various manuals and source code related to the Armadillo-9.

- Serial console software
  Please install serial console software such as “minicom” or “Tera Term” on the work PC. (Software for Linux is included in the “tools” directory in the supplied CD-ROM.)

2.2. Connections

Connect the serial cross cable, the power cable, and the LAN cable to the Armadillo-9 referring to the figure shown below.

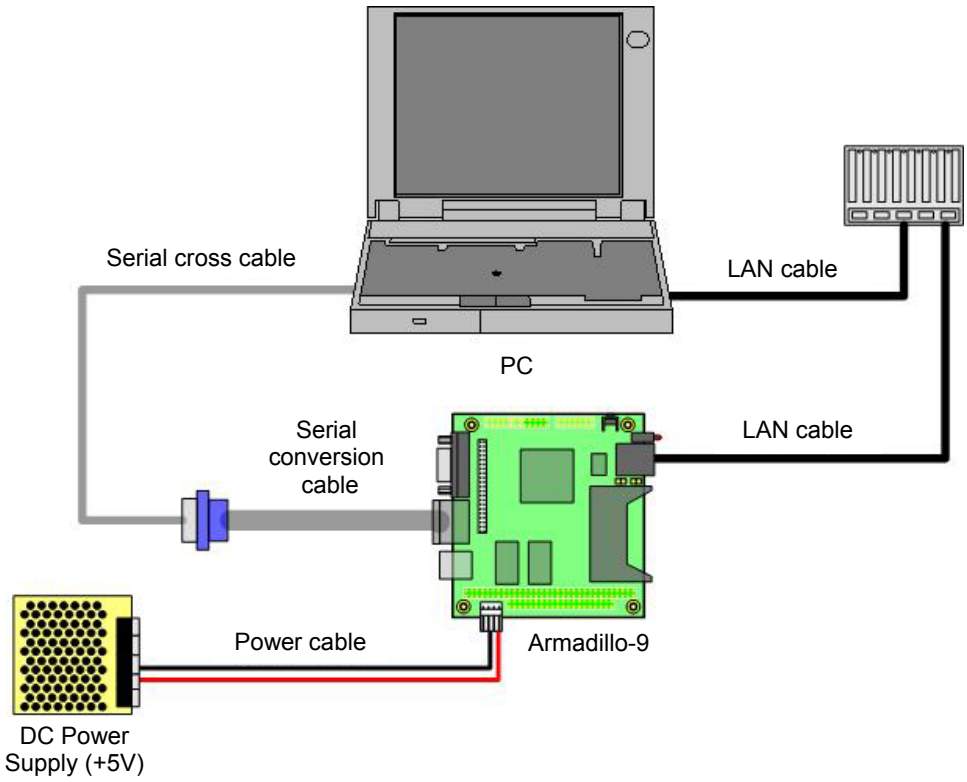


Figure 2-1 Armadillo-9 Cable Connections

2.3. Jumper Pin Settings

The boot sequence can be set by adjusting the jumper settings on the Armadillo-9. The jumper settings and their effect are described in the following table.

Table 2-1 Jumper Settings and Boot Sequence

JP1	JP2	Compact Flash Socket Status	Operation at Boot-Time
OFF	OFF		Linux kernel in the onboard flash memory is booted
OFF	ON	A Compact Flash card that holds a bootable Linux kernel is inserted.	Linux kernel on Compact Flash is booted
		No Compact Flash inserted, or there is no bootable Linux kernel in the inserted Compact Flash.	Hermit command prompt is activated
ON			EP9315 On-chip Boot ROM is activated Note: Used when restoring the bootloader with Shoehorn etc.

Note: • Booting from a Linux kernel stored in an IDE connected disk drive is not currently supported. This support is planned for the next release of the boot loader.
• The use of an IDE connected disk drive as a Linux root device can be achieved through the combination of an onboard flash memory kernel and the specification of the root device using the Hermit setenv command (root=/dev/hda1noinitrd).

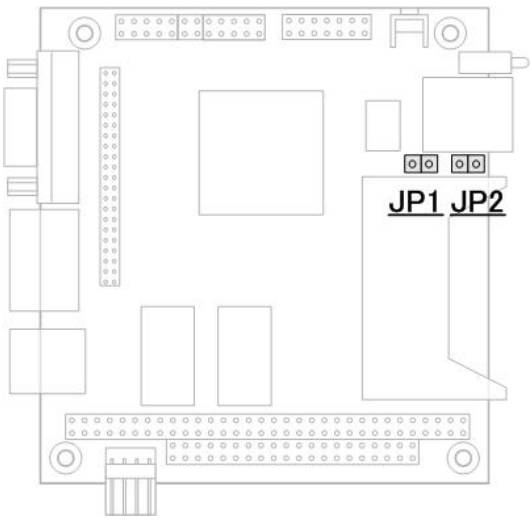


Figure 2-2 Jumper Position

3. Rewriting Flash Memory

The functionality of the Armadillo-9 can be modified by rewriting the content of the flash memory. This chapter covers how to carry out the rewriting.

Caution

If for some reason the process of transferring the rewriting image fails, the Armadillo-9 may become unbootable. Please be careful of the followings points when rewriting.

- Do not power off the Armadillo-9.
- Do not disconnect the serial cable connecting the Armadillo-9 to the development PC.

3.1. Installing the Downloader

Install the downloader (hermit) to the work PC. This downloader is used for rewriting the Armadillo-9's Flash Memory.

1) For Linux

Install the package files from the supplied CD-ROM (hereinafter, called the "supplied CD"). This should be done with root privileges.

The package files are in the following formats: "deb" (for Debian distributions), "rpm" (for Red Hat distributions) and "tgz" (non-installer). Select the format appropriate for the operating system in use.

```
[PC ~]# dpkg -i hermit_1.3-armadillo9-1_i386.deb ← when the deb package is used
[PC ~]# rpm -i hermit-1.3_armadillo9-1.rpm       ← when the rpm package is used
[PC ~]# tar zxf hermit-1.3-armadillo9.tgz -C /   ← when "tgz" is used
```

Figure 3-1 Install Command Examples

2) For Windows

Extract Hermit host for Win32 (tools/hermit-1.3_win32.zip) from the supplied CD to an appropriate folder.

3.2. Specifying Memory Region

The destination writing address of the flash memory can be specified by region name. The three region names and their associated images are explained below.

- Bootloader
  This area holds the bootloader software image executed first after the power is turned on. The bootloader's functions include rewriting flash memory when using a serial downloader and booting the operating system.

- Kernel
  This area stores the Linux kernel image. The kernel in this area is booted by the bootloader.

- Userland
  This area holds the system image which includes all applications. Applications such as telnet, ftp and web servers, various configuration files and user data are stored here.

The image file for each region is contained in the supplied CD.

Table 3-1 Image File Name for Each Region

Region Name	File Name
Bootloader	loader-a9-1.img
Kernel	linux.bin.gz
Userland	romfs-a9-1.img.gz

For the flash memory map, refer to Chapter 8, "Memory Maps."

3.3. Rewriting Procedure

Rewriting of the flash memory is done according to the following procedure.

3.3.1. Jumper Pin Settings

Set the jumper pins as follows before turning the Armadillo-9 on.

- JP1 : Open
- JP2 : Short

Do not insert anything into the Compact Flash socket.

For more details on jumper pin settings, refer to section 2.3, “Jumper Pin Settings.”

3.3.2. Transferring Rewrite Image

First, turn the power supply of the Armadillo-9 on.

1) For Linux

Open a terminal on the work PC, and enter the hermit command line specifying the kernel image and region. In the following example, the kernel image file linux.bin.gz is specified as the file name. Choose bootloader, kernel or userland for the region setting.

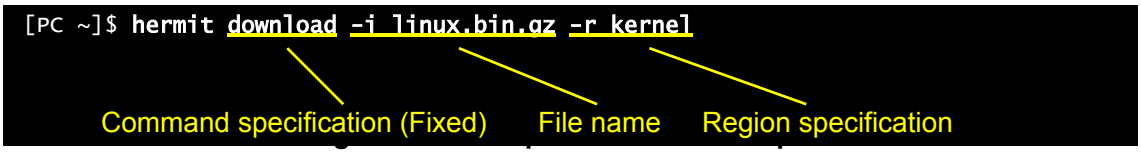


Figure 3-2 Command Line Example

Add option --port ‘port name’ if the serial port being used on the work PC is not ttyS0.

Tips

The option “--force-locked” must be added when rewriting the bootloader area (region:bootloader / address:0x60000000-0x6000ffff). If it is not, the writing will not take place and a warning message will be displayed.

Caution

It will not be possible to boot from onboard flash memory if an incompatible image is written to the bootloader area. If this does occur, restore the bootloader by referring to section 6.4.1, “Returning Boot Loader to Factory Default.”

Once the rewriting has completed, set JP2 to open and reboot the Armadillo-9 to load the newly written image.

2) For Windows
Activate Hermit host for Win32 from where it was extracted to in section 3.1, “Installing the Downloader.”

Set each parameter as shown in the following table.

Table 3-2 Parameter Examples When Using Hermit host for WIN32

Item name	Description	Value
Port	Port name connecting the Armadillo-9	COM1 (When COM1 is used)
Input file	File name to be written	File name (dialog selection possible)
Region	Region to be written to	bootloader, kernel, userland

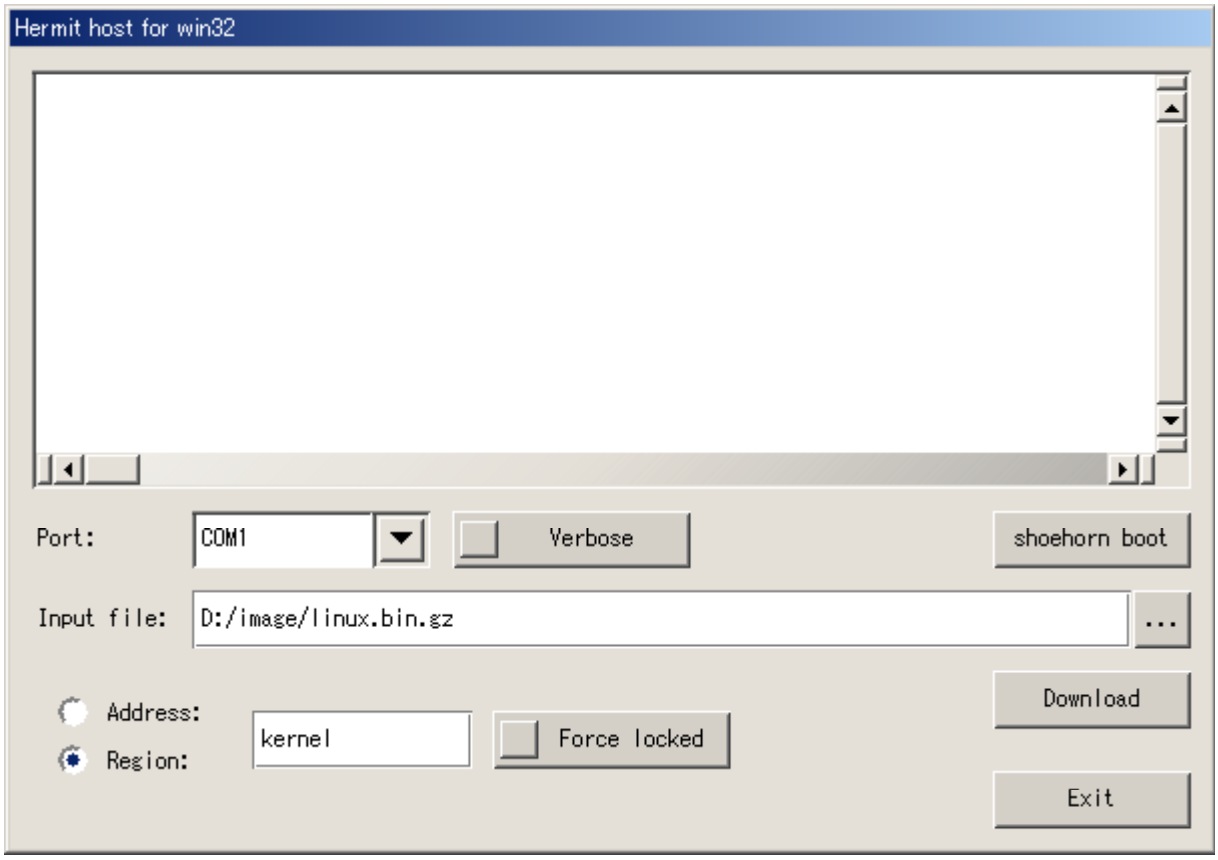


Figure 3-3 Rewriting Image Transfer Example

Tips

The option “--force-locked” must be added when rewriting the bootloader area (region:bootloader / address:0x60000000-0x6000ffff). If it is not, the writing will not take place and a warning message will be displayed.

Caution

It will not be possible to boot from onboard flash memory if an incompatible image is written to the bootloader area. If this does occur, restore the bootloader by referring to section 6.4.1, “Returning Boot Loader to Factory Default.”

Once the rewriting has completed, set JP2 to open and reboot the Armadillo-9 to load the newly written image.

3.4. Linux Boot Option Configuration

With the Armadillo-9, the boot options of the automatically booted Linux can be configured. Settings are stored in the flash memory and used the next time Linux is started.

The Linux boot options are set from the Hermit command prompt.

Tips

It is necessary to have knowledge of the Linux kernel being used in order to configure the boot options. For details on the options and their effect, refer to related documents on the Linux kernel or the documents attached to the source files.

3.4.1. Activating Hermit Command Prompt

1) Activating serial console software

Connect the Armadillo-9 to the work PC with a serial cable, and activate the serial console software. The communication settings are as follows.

Table 3-3 Serial Communication Settings

Item	Setting
Transfer Rate	115,200bps
Data Length	8bit
Stop Bit	1bit
Parity	None
Flow Control	None

2) Jumper pin settings

Set the jumper pins as follows before turning on the Armadillo-9's power supply.

- JP1：Open
- JP2：Short

Do not insert anything into the Compact Flash socket.

For details on jumper pin settings, refer to section 2.3 "Jumper Pin Settings."

3) Booting the Armadillo-9

After turning on the Armadillo-9's power supply, the Hermit command prompt will be displayed

```
Hermit v1.3-armadillo9-1 compiled at 06:45:05, Dec 18 2004
hermit>
```

3.4.2. Linux Boot Option Configuration

Use the setenv command from the Hermit command prompt to set the Linux boot option. Enter setenv, followed by the Linux boot option to be set.

```
hermit> setenv console=ttyAM0,115200 root=/dev/hda1
```

Caution

When the Linux boot options are unset (at their defaults), the bootloader automatically uses the option console=ttyAM0,115200 and sets the serial port to console. However, when setenv is used to set a boot option, this option will not automatically be used. To continue to use the serial console even when using setenv, make sure to add the console=ttyAM0,115200 option.

To start Linux with the newly set boot options, once power off the Armadillo-9 and then reboot it after having set the jumpers to their appropriate positions.

3.4.3. Linux Boot Option Confirmation

To display the Linux boot options, enter the setenv command without any parameters.

```
hermit> setenv
1: console=ttyAM0,115200
2: root=/dev/hda1
3: noinitrd
```

3.4.4. Clearing Linux Boot Options

Enter the clearenv command to clear the currently set boot options and return them to their defaults.

```
hermit> clearenv
```

3.4.5. Linux Boot Option Example

The following is an example of setting a Linux boot option.

Example 1)Using the serial console and setting the first partition of an IDE disk drive as the root device.
(It is assumed that both JP1 and JP2 are set to open and the Linux kernel is in onboard flash)

```
hermit> setenv console=ttyAM0,115200 root=/dev/hda1
```

Example 2)Using a VGA console

```
hermit> setenv video
```

Tips

A USB keyboard must be connected to input to the VGA console.

4. Usage

This chapter covers basic usage of the Armadillo-9.

4.1. Before Power-On

Connect the Armadillo-9 and the work PC with a serial cable, and activate the serial console software. Enter the connection settings as follows.

Table 4-1 Serial Communication Settings

Item	Setting
Transfer Rate	115,200bps
Data Length	8bit
Stop Bit	1bit
Parity	None
Flow Control	None

4.2. Booting

The Armadillo-9 will boot when JP2 is set to open and the power turned on. The following log will be displayed during the normal boot process.

```
Uncompressing  kernel.....................................................done.
Copying       ramdisk.done.
Doing console=ttyAM0,115200
Doing mtdparts=armadillo9-nor:0x10000(bootloader)ro,0x170000(kernel),0x670000(userland),-(config)
Linux version 2.4.27-a9-1 (atmark@build) (gcc version 3.4.1 (Debian 3.4.1-4sarge1)) #2 Sat Dec 18
11:57:29 JST 2004
CPU: Arm920Tid(wb) revision 0
Machine: Armadillo-9
alloc_bootmem_low
memtable_init
On node 0 totalpages: 16384
zone(0): 24576 pages.
zone(1): 0 pages.
zone(2): 0 pages.
Kernel command line: console=ttyAM0,115200 mtdparts=armadillo9-
nor:0x10000(bootloader)ro,0x170000(kernel),0x670000(user
land),-(config)
Console: colour dummy device 80x30
Calibrating delay loop... 99.73 BogoMIPS
Memory: 32MB 32MB = 64MB total
Memory: 55880KB available (1827K code, 351K data, 68K init)
Dentry cache hash table entries: 8192 (order: 4, 65536 bytes)
Inode cache hash table entries: 4096 (order: 3, 32768 bytes)
Mount cache hash table entries: 512 (order: 0, 4096 bytes)
Buffer cache hash table entries: 4096 (order: 2, 16384 bytes)
Page-cache hash table entries: 16384 (order: 4, 65536 bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.4
Based upon Swansea University Computer Society NET3.039
Initializing RT netlink socket
Starting kswapd
Journalled Block Device driver loaded
i2c-core.o: i2c core module version 2.6.1 (20010830)
i2c-algo-bit.o: i2c bit algorithm module
i2c-armadillo9: i2c Armadillo9 driver, (C) 2004 Atmark Techno, Inc.
i2c-s3531a: i2c S-3531A driver, (C) 2001-2004 Atmark Techno, Inc.
i2c-at24cxx: i2c at24cxx eeprom driver, (C) 2004 Atmark Techno, Inc.
Console: switching to colour frame buffer device 80x60
ttyAM0 at MMIO 0xff8c0000 (irq = 52) is a AMBA
ttyAM1 at MMIO 0xff8d0000 (irq = 54) is a AMBA
ttyAM2 at MMIO 0xff8e0000 (irq = 55) is a AMBA
ep93xx_eth() version: ep93xx_eth.c: V1.0 09/04/2003 Cirrus Logic
RAMDISK driver initialized: 16 RAM disks of 12288K size 1024 blocksize
loop: loaded (max 8 devices)
Uniform Multi-Platform E-IDE driver Revision: 7.00beta4-2.4
ide: Assuming 50MHz system bus speed for PIO modes; override with idebus=xx
No card in slot: PFDR=000000ff
SCSI subsystem driver Revision: 1.00
ARMADILLO9-NOR:0x00800000 at 0x60000000
 Amd/Fujitsu Extended Query Table v1.3 at 0x0040
number of CFI chips: 1
cfi_cmdset_0002: Disabling fast programming due to code brokenness.
ARMADILLO9-NOR:using command line partition definition
Creating 4 MTD partitions on "NOR flash on ARMADILLO9":
0x00000000-0x00010000 : "bootloader"
0x00010000-0x00180000 : "kernel"
0x00180000-0x007f0000 : "userland"
0x007f0000-0x00800000 : "config"
usb.c: registered new driver usbdevfs
usb.c: registered new driver hub
host/usb-ohci.c: USB OHCI at membase 0xff020000, IRQ 56
host/usb-ohci.c: usb-, ep93xx
usb.c: new USB bus registered, assigned bus number 1
hub.c: USB hub found
hub.c: 3 ports detected
usb.c: registered new driver hid
hid-core.c: v1.8.1 Andreas Gal, Vojtech Pavlik <vojtech@suse.cz>
hid-core.c: USB HID support drivers
usb.c: registered new driver audio
audio.c: v1.0.0:USB Audio Class driver
Initializing USB Mass Storage driver...
usb.c: registered new driver usb-storage
USB Mass Storage support registered.
mice: PS/2 mouse device common for all mice
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP, IGMP
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP: Hash tables configured (established 4096 bind 8192)
```

```
ip_tables: (C) 2000-2002 Netfilter core team
NET4: Unix domain sockets 1.0/SMP for Linux NET4.0.
NetWinder Floating Point Emulator V0.97 (double precision)
RAMDISK: ext2 filesystem found at block 0
RAMDISK: Loading 6592 blocks [1 disk] into ram disk... done.
Freeing initrd memory: 6656K
VFS: Mounted root (ext2 filesystem).
Freeing init memory: 68K
init started:  BusyBox v1.00 (2004.12.01-11:04+0000) multi-call binary
Mounting proc:                                             done
Starting fsck for root filesystem.
mount: /etc/mtab: Read-only file system
fsck 1.25 (20-Sep-2001)
/dev/ram0: clean, 596/824 files, 5830/6592 blocks
Checking root filesystem:                                  done
Remounting root rw:                                        done
Setting hostname:                                          done
Cleaning up system:                                        done
Running local start scripts.
Changing file permissions:                                 done
Starting syslogd:                                          done
Starting klogd:                                            done
Starting basic firewall:                                   done
Configuring network interfaces...                          done
Starting thttpd:                                           done
Starting inetd:                                            done

Armadillo Linux V2.0.0
Linux 2.4.27-a9-1 [armv4l arch]

armadillo9 login:
```

Figure 4-1 Boot Log

The login prompt will be displayed on the VGA in addition to the serial port ttyAM0(CON1) in the default userland.

Note: • A USB keyboard must be connected in order to login from the VGA console.
• To set the VGA as the standard console where the Linux kernel boot log is displayed, refer to section 3.4 “Linux Boot Option Configuration.”

The following two login users have been created.

Table 4-2 User Name and Password for Console Login

User name	Password	Privilege
Root	root	Super user
Guest	(None)	General user

4.3. Directory Structure

The directory structure is as follows.

Table 4-3 Directory Structure Overview

Directory name	Description
/bin	Applications
/dev	Device nodes
/etc	System settings
/etc/network	Network settings
/lib	Common libraries
/mnt	Mount points
/proc	Process information
/root	root home directory
/sbin	System management commands
/usr	Common user data
/home	user home directories
/home/ftp/pub	ftp data transfer
/tmp	Temporary backup
/var	Modified data

4.4. Shutdown

The Armadillo-9 can be shutdown by turning the power supply off.

However, doing so may damage stored data when a Compact Flash card is mounted. If a card is mounted, either dismount it before turning off the power, or shutdown the system by executing the halt command and then turning the power off.

4.5. Network Settings

Network settings can be configured by editing the /etc/network/interfaces file in the Armadillo-9.

4.5.1. Using a Fixed IP Address

The following is an example of network settings when a fixed IP address is used.

Table 4-4 Network Settings

Item	Value
IP Address	192.168.10.10
Net Mask	255.255.255.0
Broadcast Address	192.168.10.255
Default Gateway	192.168.10.1

```
# /etc/network/interfaces - configuration file for ifup(8), ifdown(8)

auto lo eth0

iface lo inet loopback

iface eth0 inet static
        address 192.168.10.10
        netmask 255.255.255.0
        network 192.168.10.0
        broadcast 192.168.10.255
        gateway 192.168.10.1
```

Figure 4-2 Network Settings Example (Using Fixed IP Address)

Delete or comment out the following line when a gateway is not being used.

```
#       gateway 192.168.10.1 ← Comment out by adding "#"
```

Figure 4-3 Network Settings Example (Deselecting Gateway)

4.5.2. Using DHCP

Below is an example where DHCP is used to obtain an IP address.

```
# /etc/network/interfaces - configuration file for ifup(8), ifdown(8)
auto lo eth0
iface lo inet loopback
iface eth0 inet dhcp
```

Figure 4-4 Network Settings Example (Using DHCP)

4.5.3. Applying Network Settings

To connect to a network after having made changes to the network settings, the networking script must be executed. Dependant on the environment, this script exists in either /etc/rc.d/rc.start or /etc/init.d.

If already connected to a network, the connection must be closed first. It can be closed with ifconfig for a fixed IP connection, or with the dhcpcd command and -k option for DHCP connections.

```
[armadillo9 /]# ifconfig eth0 down ← Closing a fixed IP network connection
[armadillo9 /]# dhcpcd -k          ← Closing a DHCP network connection
```

Figure 4-5 Closing Network Connection

```
[armadillo9 /]# /etc/init.d/networking
```

Figure 4-6 Making a New Network Connection

4.6. telnet Login

It is possible to login via telnet with the following user name and password. It is not possible to login as the root user. Root privileges can be obtained when required by using the su command after logging in as guest.

Table 4-5 User Name and Password for telnet Login

User name	Password
Guest	None

4.7. File Transfer

It is possible to transfer files with ftp by logging in with the following user name and password. The home directory is /home/ftp. Data can be uploaded by moving to the home/ftp/pub directory.

Table 4-6 User Name and Password for ftp

User name	Password
ftp	None

4.8. Web Server

As small HTTP server named thttpd is run by default, the Armadillo-9 can be accessed from a Web browser. The data directory is /var/www. The URL is http:///(Armadillo-9 IP address), for example: http://192.168.10.10/

5. Preparation of Development Environment

Development for the Armadillo-9 can be carried out on Linux.

5.1. Installing Cross Development Environment Packages

Install the cross development environment packages from the cross-dev directory on the supplied CD. Installation should be done under root privileges. The following packages have been readied.

Table 5-1 List of Cross Development Environment Packages

Package name	Version	Description
binutils-arm-linux	2.14.90.0.7-8	Binary utilities
cpp-arm-linux	3.4.1-4sarge1	The GNU C preprocessor
g++-arm-linux	3.4.1-4sarge1	The GNU C++ compiler
gcc-arm-linux	3.4.1-4sarge1	The GNU C compiler
libc6-arm-cross	2.3.2.ds1-13	GNU C Library: Shared libraries and Time zone data
libc6-dev-arm-cross	2.3.2.ds1-13	GNU C Library: Development Libraries and Header Files
libc6-pic-arm-cross	2.3.2.ds1-13	GNU C Library: PIC archive library
libc6-prof-arm-cross	2.3.2.ds1-13	GNU C Library: Profiling Libraries
libdb1-compat-arm-cross	2.1.3-7	The Berkeley database routines
libgcc1-arm-cross	3.4.1-4sarge1	GCC support library
libstdc++6-0-arm-cross	3.4.1-4sarge1	The GNU Standard C++ Library v3
libstdc++6-0-dbg-arm-cross	3.4.1-4sarge1	The GNU Standard C++ Library v3 (debugging files)
libstdc++6-0-dev-arm-cross	3.4.1-4sarge1	The GNU Standard C++ Library v3 (development files)
libstdc++6-0-pic-arm-cross	3.4.1-4sarge1	The GNU Standard C++ Library v3 (shared library subset kit)
linux-kernel-headers-arm-cross	2.5.999-test7-bk-16	Linux Kernel Headers for development

Included package formats are, “deb” (for Debian distributions), “rpm” (for Red Hat distributions) and “tgz” (non-installer). Select the one appropriate for the operating system in use.

```
[PC ~]# dpkg -i hermit_1.3-armadillo9-1_i386.deb ← when “deb” packages are used
[PC ~]# rpm -i hermit-1.3_armadillo9-1.rpm       ← when “rpm” packages are used
[PC ~]# tar zxf hermit-1.3-armadillo9.tgz -C /   ← when “tgz” is used
```

Figure 5-1 Example of Extracting Tool Chain for Development

5.2. Setting Environment Variables

The directory that contains the executable files of the development tool chain is added to the PATH environment variable to make them easier to use. As the method differs dependant on the shell, refer to the manual of the shell in use for details.

The following example shows how to set the PATH in bash.

```
[PC ~]$ export PATH="$PATH:/usr/local/bin"
[PC ~]$ echo $PATH
/usr/bin:/bin:/usr/bin/X11:/usr/sbin:/sbin:/usr/local/bin
[PC ~]$
```

Figure 5-2 Setting of PATH Environment Variable (bash)

6. Bootloader

This chapter explains about the bootloader of the Armadillo-9.

6.1. Preparing Packages

Copy the following packages from the hermit directory in the supplied CD to the work PC.

Table 6-1 List of Bootloader Related Packages

Package name	Description
hermit-1.3-armadillo9	A downloader that operates with the Armadillo-9 boot program. (Includes the Armadillo-9 boot program itself)
shoehorn-3.4-armadillo9	A downloader that operates with the CPU on-chip boot ROM

For the installation method of the packages, refer to section 5.1, “Installing Cross Development Environment Packages.”

6.2. Bootloader Types

The bootloaders readied for the Armadillo-9 are shown below.

Table 6-2 Bootloader List

Bootloader name	Description
loader-armadillo9	The standard bootloader written to Flash at shipment. Uses COM1.
loader-armadillo9-ttyAM1	A bootloader that uses COM2.
loader-armadillo9-notty	A bootloader that does not use the console.

6.3. Creating a Bootloader

While the bootloaders have been included on the supplied CD, it is also possible to build an original boot loader from source code.

6.3.1. Readying the source code

Copy and extract hermit_1.3-armadillo9-1.tar.gz from the source directory on the supplied CD to the work PC.

```
[PC ~]$ tar zxf hermit_1.3-armadillo9-1.tar.gz
```

6.3.2. Build

Move to the directory created extracting the file above and enter the make command.

```
[PC ~]$ cd hermit-1.3-armadillo9-1
[PC ~]$ make target=armadillo9
```

Once the make has completed, the bootloader will have been created in the hermit-1.3-armadillo9-1/src/target/armadillo9 directory.

6.4. CPU On-Chip Boot ROM

The following covers how to rewrite the Armadillo-9's bootloader containing loader-armadillo9-notty, and what to do when the Armadillo-9 will no longer boot because an incompatible bootloader has been installed.
The Armadillo-9 includes a CPU on-chip ROM. By expanding the data from this ROM to RAM, the bootloader can be returned to its shipped state by following the procedure below.

6.4.1. Returning the bootloader to its default state

1) For Linux
   1. Make sure the Armadillo-9 is turned off, and connect COM1 on the Armadillo-9 to the serial port on the work PC with a cross (reverse) serial cable.
   2. Short the JP1 jumper on the Armadillo-9.
   3. Activate shoehorn on the work PC.

```
[PC ~]$ shoehorn --armadillo9 --boot --terminal
         --kernel /usr/lib/hermit/loader-armadillo9-boot.bin --initrd /dev/null
```

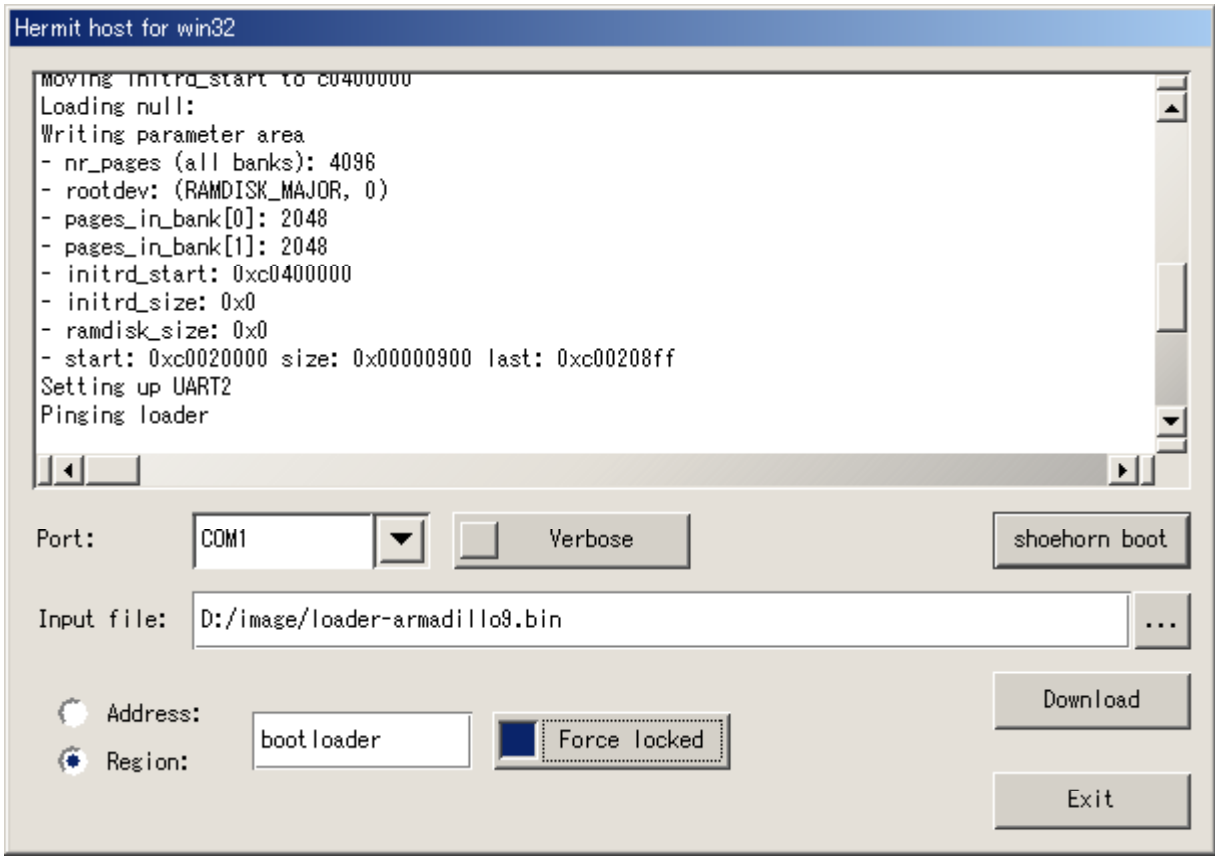
   Note: • The above example shows when the Armadillo-9 is connected to the serial port /dev/ttyS0 on the work PC. Add the following option to the shoehorn command line when the Armadillo-9 is connected to another serial port.
   --port [Serial port name]
   • The command must be entered as one line.

   4. Turn the power supply of the Armadillo-9 on.
   Note: • Display of system messages should soon begin. If they do not appear, turn the Armadillo-9 off and make sure the serial cable connections and the jumper (JP1) are set correctly.

   5. Enter Ctrl+C once the hermit> prompt is displayed.

This completes the required preparation for downloading the bootloader to the Armadillo-9 from the work PC using hermit. For instructions on rewriting the image in flash memory, refer to Chapter 3, "Rewriting Flash Memory."

2) For Windows
   1. Make sure the Armadillo-9 is turned off, and connect COM1 on the Armadillo-9 to the serial port on the work PC with a cross (reverse) serial cable.
   2. Short the JP1 jumper on the Armadillo-9.
   3. Activate Hermit host for Win32 on the work PC.
   4. Push the shoehorn boot button.



Tips

This example shows when the Armadillo-9 is connected to serial port COM1 on the work PC. Change the Port selection if it is connected to another serial port.

   5. Turn the power supply of the Armadillo-9 on.

Display of system messages should soon begin. If they do not appear, turn the Armadillo-9 off and make sure the serial cable connections and the jumper (JP1) are set correctly.

This completes the required preparation for downloading the bootloader to the Armadillo-9 from the work PC using hermit. For instructions on rewriting the image in flash memory, refer to Chapter 3, “Rewriting Flash Memory.”

7. Image Creation With uClinux-dist

This chapter shows how to create kernel and userland images using uClinux-dist. For detailed usage of uClinux-dist, refer to the uClinux-dist Developers Guide.

Caution

Development with uClinux-dist involves the creation and handling of basic libraries, applications and system configuration files. While all files are dealt with under the uClinux directory, please carry out all operations under a general user and not the root user to avoid any damage to the operating system of the development PC resulting from input errors.

7.1. Extracting Source Code Archive

There is a source code archive named uClinux-dist.tar.gz in the dist directory of the supplied CD. Extract this file to an appropriate directory. Here, it is extracted to the user's home directory (~/).

```
[PC ~]$ tar zxvf uClinux-dist-test.tar.gz
```

Figure 7-1 Example of Source Code Extraction

Next, extract the Linux kernel source code and create a symbolic link in the uClinux-dist directory with the name linux-2.4.x. There is a kernel source code file under the name linux-2.4.27-a9-1.tar.gz in the source directory of the supplied CD.

```
[PC ~]$ tar zxvf linux-2.4.27-a9-1.tar.gz
  :
  :
[PC ~]$ cd uClinux-dist
[PC ~/uClinux-dist]$ ln -s ../linux-2.4.27-a9-1 ./linux-2.4.x
```

7.2. Configuration

First, carry out a configuration of the dist for the target board. Start configuration by entering the command as shown in the following example.

```
[PC ~/uClinux-dist]$ make config
```

You will be asked to choose the vendor name of the board. Enter AtmarkTechno.

```
[PC ~/uClinux-dist]$ make config
config/mkconfig > config.in
#
# No defaults found
#
*
* Vendor/Product Selection
*
*
* Select the Vendor you wish to target
*
Vendor (3com, ADI, Akizuki, Apple, Arcturus, Arnewsh, AtmarkTechno, Atmel,
Avnet, Cirrus, Cogent, Conexant, Cwlinux, CyberGuard, Cytek, Exys, Feith,
Future, GDB, Hitachi, Imt, Insight, Intel, KendinMicrel, LEOX, Mecel, Midas,
Motorola, NEC, NetSilicon, Netburner, Nintendo, OPENcores, Promise, SNEHA, SSV,
SWARM, Samsung, SecureEdge, Signal, SnapGear, Soekris, Sony, StrawberryLinux,
TI, TeleIP, Triscend, Via, Weiss, Xilinx, senTec) [SnapGear] (NEW) AtmarkTechno
```

You will then be asked to choose the board name. Enter Armadillo-9.

```
*
* Select the Product you wish to target
*
AtmarkTechno Products (Armadillo-9, SUZAKU) [Armadillo-9] (NEW) Armadillo-9
```

The C library to be used is then specified. Supported libraries depend on the board being used. For the Armadillo-9, select None.

```
*
* Kernel/Library/Defaults Selection
*
*
* Kernel is linux-2.4.x
*
Libc Version (None, glibc, uC-libc, uClibc) [uClibc] (NEW) None
```

You will be asked whether or not to default all settings. Select Yes.

```
Default all settings (lose changes) (CONFIG_DEFAULTS_OVERRIDE) [N/y/?] (NEW) y
```

Select No for the last three questions.

```
Customize Kernel Settings (CONFIG_DEFAULTS_KERNEL) [N/y/?] n
Customize Vendor/User Settings (CONFIG_DEFAULTS_VENDOR) [N/y/?] n
Update Default Vendor Settings (CONFIG_DEFAULTS_VENDOR_UPDATE) [N/y/?] n
```

Configuration of the build system will begin after all questions have been completed. You will return to the prompt once the configuration has finished.

7.3. Build

Enter the following command line to execute the build.

```
[PC ~/uClinux-dist]$ make dep all
```

Dependant on the dist version, the make process may pause at times to ask for confirmation of certain settings. Generally, as default settings are fine as they are, just hit the return key to continue the process.

The kernel image linux.bin.gz, and the userland image romfs.img are created in the uClinux-dist/images directory once the build completes. For information on writing the created images to the Armadillo-9, refer to Chapter 3, “Rewriting Flash Memory.”

8. Memory Maps

Table 8-1 Memory Map (Flash Memory)

Address	Region	Size	Description
0x60000000 0x6000ffff	bootloader	64KB	Hermit bootloader image “loader-armadillo9.bin”
0x60010000 0x6017ffff	kernel	approx. 1.44MB	Linux Kernel image “linux.bin.gz” (Uncompressed and gz compressed images supported)
0x60180000 0x607effff	userland	approx. 6.44MB	Userland image “romfs.img” (Uncompressed and gz compressed images supported)
0x607f0000 0x60800000	config	64KB	Configuration area

Note: Only the kernel and userland are extracted/copied to RAM before Linux is booted.

Table 8-2 Memory Map (RAM)

Address	Content	File System	Description
0xc0018000	kernel		Extracted and copied from flash memory before Linux is booted
0xc0800000	userland	EXT2	Extracted and copied from flash memory before Linux is booted

Table 8-3 Memory Map (PC/104)

Linux Logical Address	Physical Address	Description
0xf2000000 0xf200ffff	0x12000000 0x1200ffff	PC/104 I/O Space (8bit)
0xf3000000 0xf3ffffff	0x13000000 0x13ffffff	PC/104 Memory Space (8bit)
0xf6000000 0xf600ffff	0x22000000 0x2200ffff	PC/104 I/O Space (16bit)
0xf7000000 0xf7ffffff	0x17000000 0x17ffffff	PC/104 Memory Space (16bit)

9. Interrupts (IRQ)

Table 9-1 List of Interrupts (IRQ)

IRQ # In Linux	Interrupt name	Interrupt source	Description
2	IRQ_COMMRX	VIC #2	COMMRX
3	IRQ_COMMTX	VIC #3	COMMTX
4	IRQ_TIMER1	VIC #4	TIMER1
5	IRQ_TIMER2	VIC #5	TIMER2
6	IRQ_AAC	VIC #6	AAC
7	IRQ_DMAM2P0	VIC #7	DMAM2P0
8	IRQ_DMAM2P1	VIC #8	DMAM2P1
9	IRQ_DMAM2P2	VIC #9	DMAM2P2
10	IRQ_DMAM2P3	VIC #10	DMAM2P3
11	IRQ_DMAM2P4	VIC #11	DMAM2P4
12	IRQ_DMAM2P5	VIC #12	DMAM2P5
13	IRQ_DMAM2P6	VIC #13	DMAM2P6
14	IRQ_DMAM2P7	VIC #14	DMAM2P7
15	IRQ_DMAM2P8	VIC #15	DMAM2P8
16	IRQ_DMAM2P9	VIC #16	DMAM2P9
17	IRQ_DMAM2M0	VIC #17	DMAM2M0
18	IRQ_DMAM2M1	VIC #18	DMAM2M1
19	IRQ_GPIO0	VIC #19	GPIO0
20	IRQ_GPIO1	VIC #20	GPIO1
21	IRQ_GPIO2	VIC #21	GPIO2
22	IRQ_GPIO3	VIC #22	GPIO3
23	IRQ_UARTRX1	VIC #23	UARTRX1
24	IRQ_UARTTX1	VIC #24	UARTTX1
25	IRQ_UARTRX2	VIC #25	UARTRX2
26	IRQ_UARTTX2	VIC #26	UARTTX2
27	IRQ_UARTRX3	VIC #27	UARTRX3
28	IRQ_UARTTX3	VIC #28	UARTTX3
29	IRQ_KEY	VIC #29	KEY
30	IRQ_TOUCH	VIC #30	TOUCH
32	IRQ_EXT0	VIC #32	EXT0
33	IRQ_EXT1	VIC #33	EXT1
34	IRQ_EXT2	VIC #34	EXT2
35	IRQ_64HZ	VIC #35	64HZ
36	IRQ_WEIN	VIC #36	WEIN
37	IRQ_RTC	VIC #37	RTC
38	IRQ_IRDA	VIC #38	IRDA
39	IRQ_MAC	VIC #39	MAC
40	IRQ_EXT3 (IRQ_EIDE)	VIC #40	EXT3 (EIDE)
41	IRQ_PROG	VIC #41	PROG
42	IRQ_1HZ	VIC #42	1HZ
43	IRQ_VSYNC	VIC #43	VSYNC
44	IRQ_VIDEOFIFO	VIC #44	VIDEOFIFO
45	IRQ_SSPRX	VIC #45	SSPRX
46	IRQ_SSPTX	VIC #46	SSPTX
47	IRQ_GPIO4	VIC #47	GPIO4
48	IRQ_GPIO5	VIC #48	GPIO5
49	IRQ_GPIO6	VIC #49	GPIO6
50	IRQ_GPIO7	VIC #50	GPIO7
51	IRQ_TIMER	VIC #51	TIMER3
52	IRQ_UART1	VIC #52	UART1

53	IRQ_SSP	VIC #53	SSP
54	IRQ_UART2	VIC #54	UART2
55	IRQ_UART3	VIC #55	UART3
56	IRQ_USH	VIC #56	USH
57	IRQ_PME	VIC #57	PME
58	IRQ_DSP	VIC #58	DSP
59	IRQ_GPIO	VIC #59	GPIO
60	IRQ_SAI	VIC #60	SAI
64	IRQ_ISA3	PC/104 #3	
65	IRQ_ISA4	PC/104 #4	
66	IRQ_ISA5	PC/104 #5	
67	IRQ_ISA6	PC/104 #6	
68	IRQ_ISA7	PC/104 #7	
69	IRQ_ISA9	PC/104 #9	
70	IRQ_ISA10	PC/104 #10	
71	IRQ_ISA11	PC/104 #11	
72	IRQ_ISA12	PC/104 #12	
73	IRQ_ISA14	PC/104 #14	
74	IRQ_ISA15	PC/104 #15	

Table 9-2 PC/104 IRQ Support Functions

Usage	Function definition	Use example

Conversion from Linux IRQ # to PC/104 IRQ #	static __inline__
	unsigned int
	convirq_to_isa
	(unsigned int irq);
		// Convert Linux IRQ # IRQ_ISA3
		// to PC/104 IRQ #
		const unsigned
		linux_irq = IRQ_ISA3;
		unsigned int isa_irq;
		isa_irq =
		convirq_to_isa(linux_irq);

Conversion from PC/104 IRQ # to Linux IRQ #	static __inline__
	unsigned int
	convirq_from_isa
	(unsigned int irq);
		// Convert PC/104 IRQ #3
		// to Linux IRQ #
		const unsigned int isa_irq = 3;
		unsigned linux_irq;
		linux_irq =
		convirq_from_isa(isa_irq);

Note: Defined in (Kernel source)/include/asm-arm/arch-ep93xx/irqs.h

10. Original Device Driver Specifications

10.1. GPIO Port

Parameters of the device node corresponding to the GPIO port are as follows.

Table 10-1 GPIO Node

Type	Major number	Minor number	Node name (/dev/xxx)	Device name
Character Device	10	185	gpio	EP93XX_GPIO_PADR EP93XX_GPIO_PBDR EP93XX_GPIO_PCDR EP93XX_GPIO_PDDR EP93XX_GPIO_PEDR EP93XX_GPIO_PFDR EP93XX_GPIO_PGDR EP93XX_GPIO_PHDR EP93XX_GPIO_PADDR EP93XX_GPIO_PBDDR EP93XX_GPIO_PCDDR EP93XX_GPIO_PDDDR EP93XX_GPIO_PEDDR EP93XX_GPIO_PFDDR EP93XX_GPIO_PGDDR EP93XX_GPIO_PHDDR

The GPIO register of EP9315 can be directly operated via access with ioctl.

The “struct ep93xx_gpio_ioctl_data” structure defined in (Kernel source)/include/ep93xx_gpio.h and each macro are used for the third argument.

For details on registers, refer to Chapter 28, “GPIO Interface” in “EP9351 User’s Guide” issued by Cirrus Logic, Inc.

Example 10-1 Structure of ep93xx_gpio.h and macro definition

```
        ↓ Data structure (Define structure, specify the pointer for third argument of
              "ioctl")
struct ep93xx_gpio_ioctl_data {
        __u32 device;     ← Substitute register specifying macro
        __u32 mask;       ← Specify bit position to be obtained/used
        __u32 data;       ← Read/Write data variable
};

#define EP93XX_GPIO_IOCTL_BASE 'N'
        ↓ command specifying macro (used for second argument of "ioctl")
#define EP93XX_GPIO_IN     _IOWR(EP93XX_GPIO_IOCTL_BASE, 0, struct
ep93xx_gpio_ioctl_data)
#define EP93XX_GPIO_OUT    _IOW (EP93XX_GPIO_IOCTL_BASE, 1, struct
ep93xx_gpio_ioctl_data)

enum {    ↓ register specifying macro
  EP93XX_GPIO_PADR = 0,
  EP93XX_GPIO_PBDR,
  EP93XX_GPIO_PCDR,
  EP93XX_GPIO_PDDR,
  EP93XX_GPIO_PADDR,
  EP93XX_GPIO_PBDDR,
  EP93XX_GPIO_PCDDR,
  EP93XX_GPIO_PDDDR,
  EP93XX_GPIO_PEDR,
  EP93XX_GPIO_PEDDR,
  EP93XX_GPIO_PFDR,
  EP93XX_GPIO_PFDDR,
  EP93XX_GPIO_PGDR,
  EP93XX_GPIO_PGDDR,
  EP93XX_GPIO_PHDR,
  EP93XX_GPIO_PHDDR,
  EP93XX_GPIO_NUM,
};
```

Example 10-2 Sample program for GPIO operation

```
#include <fcntl.h>
#include <stdio.h>

#include "ep93xx_gpio.h"

int main (void)
{
        int fd;
        ep93xx_gpio_ioctl_data d;

        // Open GPIO with readable/writable mode
        fd = open ("/dev/gpio", O_RDWR);
        if (fd < 0) {
                fprintf (stderr, "Open error.\n");
                return -1;
        }

        //Change pin 0 of Port B to input, pin 1 of Port B to output
        d.device = EP93XX_GPIO_PBDDR;
        d.mask = 0x00000003;
        d.data = 0x00000002;
        ioctl (fd, EP93XX_GPIO_OUT, &d);

        //Display input value of pin 0 of Port B
        d.device = EP93XX_GPIO_PBDR;
        d.mask = 0x00000001;
        ioctl (fd, EP93XX_GPIO_IN, &d);
        printf ("Port B[0]: %d\n", d.data & 0x1);

        //Output "High" to pin1 of Port B
        d.device = EP93XX_GPIO_PBDR;
        d.mask = 0x00000002;
        d.data = 0x00000002;
        ioctl (fd, EP93XX_GPIO_OUT, &d);

        close (fd);

        return 0;
}
```

10.2. Real Time Clock

The parameters of the device node corresponding to the real time clock are as follows.

Table 10-2 Real Time Clock Node

Type	Major number	Minor number	Node name (/dev/xxx)	Device name
Character device	10	135	rtc	Real Time Clock

10.3. Onboard Flash Memory

The onboard flash memory is handled in region units as a Memory Technology Device (MTD). For information on the regions of the onboard flash memory, refer to Chapter 8, “Memory Maps.”
The parameters of the device nodes for Linux are as follows.

Table 10-3 MTD Nodes

Type	Major number	Minor number	Node name (/dev/xxx)	Device name
Character device	90	0	mtd0	bootloader
		1	mtdr0	bootloader (read only)
		2	mtd1	kernel
		3	mtdr1	kernel (read only)
		4	mtd2	usernald
		5	mtdr2	userland (read only)
		6	mtd3	config
		7	mtdr4	config (read only)
Block device	31	0	mtdblock0	bootloader
		1	mtdblock1	kernel
		2	mtdblock2	userland
		3	mtdblock3	config

10.4. USB Host

The EP9315 has OHCI compatible USB host functionality. The drivers of some devices have made available in the default kernel, and therefore can be used immediately after they are connected.

10.4.1. USB Audio

USB audio devices are supported. The can be handled as a standard sound device from /dev/dsp (character device, major number: 14, minor number: 3) etc.

10.4.2. USB Storage

USB memory, disk drives, and memory card readers etc. are supported. They are recognized by Linux in the same way as general SCSI equipment, and can be handled from /dev/sda (block device, major number: 8, minor number: 0) and /dev/sda1 (block device, major number: 8, minor number: 1) and so on.

10.4.3. USB Human Interface Device (HID)

Various input devices such as USB keyboards, mice and so on are supported. In particular, combined with VGA output USB keyboards can be used for console input as dev/tty0.

10.5. VGA Output

The flame buffer driver has been readied for the VGA output, and therefore can be used as a console screen.

While it is set to 16 bit color and VGA (640x480) resolution by default, SVGA (800x600) and XGA (1024x768) resolutions and 8 bit color are also supported.

Tips

In the current kernel, a recompile is necessary in order to change the resolution or color settings.

10.6. IDE and Compact Flash

Disk drives connected via IDE can be handled from /dev/hda (block device, major number: 3, minor number: 0) and /dev/hda1 (block device, major number: 3, minor number: 1) and so on.

Storage devices inserted in the Compact Flash socket can be handled from /dev/hdc (block device, major number: 22, minor number: 0) and /dev/hdc1 (block device, major number: 22, minor number: 1) and so on. Compact flash cards cannot be inserted or removed while the Armadillo-9 is running when using the default kernel Compact Flash recognition. To hot swap Compact Flash cards, use PCMCIA-CS instead of the kernel Compact Flash recognition.

Caution

The kernel Compact Flash driver is activated by default at ATA/ATAPI/MFM/RLL support → IDE, ATA and ATAPI Block devices → EP93xx PCMCIA IDE Support in the kernel configuration. As this driver conflicts with PCMCIA-CS, please use a kernel with EP93xx PCMCIA IDE Support deactivated when using PCMCIA-CS

11. Building a Compact Flash System

11.1. Creating an Armadillo-9 Bootable Compact Flash Card

The Armadillo-9 can be booted from a Linux system stored in a Compact Flash card. The following is the procedure for creating a bootable Compact Flash card on the Armadillo-9.

1) Booting the Armadillo-9
   Insert a Compact Flash card in the Armadillo-9, set jumpers JP1 and JP2 to OFF and boot the Linux in onboard flash memory.

2) Partition configuration
   Create a partition on the Compact Flash that can be booted on the Armadillo-9 by using fdisk. The partition type of the boot partition must set to 83 (Linux).

```
[armadillo9 ~]# fdisk /dev/hdc
 hdc: hdc1
Command (m for help):
```

Tips

The following are fdisk command examples.
- Delete an existing partition with the d command
- Create a partition with the n command
- Set the partition type to 83 (Linux) with the t command
- Write the settings and end fdisk with the w command

3) Partition format
   Format the created partition as an EXT2 file system. The -O none option must be added when formatting an Armadillo-9 boot partition with mke2fs.

```
[armadillo9 ~]# mke2fs -O none /dev/hdc1
```

Tips

mke2fs is not included in the default Linux system in the onboard flash memory. This can be corrected by either of the following methods.
1) Use uClinux-dist to rebuild userland adding mke2fs (Select mke2fs from Customize Vendor/User Settings, Filesystem Applications).
2) Work from Linux on a PC that can use Compact Flash.

(mke2fs is planned to be included by default in the next release of the userland.)

4) Mounting the Compact Flash
Mount the Compact Flash to /mnt

```
[armadillo9 ~]# mount /dev/hda1 /mnt
```

5) Building the system on Compact Flash
Build the system by copying the files from a prepared system image to the Compact Flash mounted to /mnt.

```
[armadillo9 ~]# (cd /tmp; tar cf - *) | (cd /mnt; tar xf -)
```

Note: The above is an example when the directory tree for a Compact Flash system has been created in /tmp in advance.

6) Copying the Linux kernel
In order to boot the Armadillo-9 from the Compact Flash, a kernel must be stored in the /boot directory of the Compact Flash as a uncompressed image named “image”, or a compressed gz image named “image.gz”. Copy a kernel image under one of these file names.

```
[armadillo9 ~]# cp linux.bin.gz /mnt/boot/Image.gz
```

Note: The above is an example when the kernel image linux.bin.gz was created in the current directory in advance.

7) Unmounting the Compact Flash
Unmount the Compact Flash after the writing is complete.

```
[armadillo9 ~]# umount /mnt
```

This completes the construction of a Compact Flash system. Shut the Armadillo-9 down, set the jumpers by referring to Chapter 2.3, “Jumper Pin Settings,” and then restart the Armarillo-9 to boot from the Compact Flash system.

11.2. Installing Debian/GNU Linux onto Compact Flash

The following shows how to install Debian/GNU Linux on a Compact Flash card. The Debian image readied has been divided into four, debian/arm-sarge1.tar.gz to debian/arm-sarge4.tar.gz.

The method of creating the Armadillo-9 bootable Compact Flash is the same as that described in section 11.1, “Creating a Armadillo-9 Bootable Compact Flash Card.” The procedure below expands on step five to show how to make use of the readied Debian/GNU Linux images.

Tips

Installing the Debian/GNU Linux images requires 300MB or more of free space on the install partition of the Compact Flash.

5) -a Mounting the RAM file system

Mount a RAM File System to /home/ftp/pub, and give the write permission.

```
[armadillo9 ~]# mount -t ramfs ramfs /home/ftp/pub
[armadillo9 ~]# chmod  777 /home/ftp/pub
```

5) -b File transfer with ftp

Transfer arm-sarge1.tar.gz from the PC via ftp.

```
[PC ~]$ ftp xxx.xxx.xxx.xxx ← IP address of the Armadillo-9
Password:
ftp> cd pub
ftp> bin
ftp> put arm-sarge1.tar.gz
```

5) -c Extraction to Compact Flash

Extract the compressed file arm-sarge1.tar.gz to Compact Flash. Delete the compressed file from the RAM File System after extraction has completed.

```
[armadillo9 ~]# gzip -cd /home/ftp/pub/arm-sarge1.tar.gz | (cd /mnt; tar xf -)
[armadillo9 ~]# rm /home/ftp/pub/arm-sarge1.tar.gz
```

5) -d Transferring and extracting all divided files

Repeat steps b and c for all remaining files, arm-sarge2.tar.gz to arm-sarge4.tar.gz.

That completes the extraction of the Debian/GNU Linux files. As the Linux kernel is not included in the Debian image, make sure to copy it separately referring section 11.1, “Creating an Armadillo-9 Bootable Compact Flash Card.”

12. Userland With PCMCIA-CS Support

12.1. Kernel and Userland Images

PCMCIA-CS is a card service package that enables hot swapping for Compact Flash and supports Compact Flash cards other than storage cards (I/O device cards). The userland that includes PCMCIA-CS, and the kernel to be used in combination with this userland, have been prepared for use with the Armadillo-9.

- romfs_pcmcia.img.gz PCMCIA-CS supported userland
- linux-pcmcia.bin.gz Linux kernel which can be used with the PCMCIA-CS supported userland

Caution

Please do not use the standard kernel without PCMCIA-CS support as it contains a driver that conflicts with PCMCIA-CS,.

Write each image to the onboard flash memory. For the procedure on writing an image to flash memory, refer to Chapter 3, “Rewriting Flash Memory.”

12.2. Activating PCMCIA-CS

Enter the following command to activate PCMCIA-CS.

```
[armadillo9 ~]# /etc/rc.d/rc.pcmcia start
```

The Compact Flash card will be recognized as soon as it is inserted, and the driver loaded automatically if it is confirmed to be a supported device.

Note: If the device is not included on the PCMCIA-CS compatible list, it may not be recognized automatically. While it may be possible to have it recognized by rewriting the appropriate configuration file, please refer to the PCMCIA-CS documentation for details.

12.3. Other Packages Included only in the PCMCIA-CS Supported Userland

The following are included in the PCMCIA-CS supported userland as support packages.

- linux-wlan-ng Driver for Prism2 chip wireless LAN cards
  (Some Compact Flash wireless LAN cards are supported.)
- wireless-tools Wireless LAN control tools group

For details, refer to the documentation attached to the source code of each package.

13. Appendix

13.1. Building a Development Environment in Windows

A cross development environment for the Armadillo-9 can be built in Windows by using coLinux (http://www.colinux.org/). Windows XP and Windows 2000 are supported by coLinux.

13.1.1. Installing coLinux

1) Activate coLinux-0.6.1.exe from the colinux directory on the supplied CD.
2) Specify c:\colinux as the installation directory and work through the installation leaving all other settings at their defaults.

Tips

If the installation directory is specified as anything other than that described above, the directory name in the default.colinux.xml file must be changed appropriately.

13.1.2. Readying Files

Expand the following files from the coLinux directory on the supplied CD to the coLinux installation folder (c:\colinux).

- root_fs.lzh (root file system)
- swap_device_256M.lzh (swap file system)
- home_fs_2G.lzh (file system mounted to /home)
- default.colinux.xml.lzh (device information configuration file)

Tips

The numbers in the file names swap_device_… and home_fs_… etc. represent the file size after extraction. As other file sizes are also available, extract the file with the size deemed to be most appropriate.

Extraction of these files may fail when using some particular extraction programs. Extraction has been confirmed to work properly using LHA Utility 32 Ver1.46 (http://www.lhut32.com/index.shtml).

13.1.3. Starting coLinux

1) Open a DOS prompt window and move to the installation folder (c:\colinux).
2) Enter colinux-daemon.exe –c default.colinux.xml at the command line.
3) When “colinux login:” appears after the boot log, login as root.

13.1.4. Network Settings

As coLinux has a different IP address from that of Windows and accesses the network through Windows, changes must be made to the Windows network settings.

Configuration methods include “router connections” and “bridge connections.” The following explains the procedure for creating a “router connection.”

(For WindowsXP)

1) Open Network Connections from the Control Panel.
2) Right-click the externally connected network and open Properties.
3) Choose the Advanced tab, and enable internet connection sharing.

(For Windows2000)

1) Open Network and Dial-up Connections from the Control Panel.
2) Right-click the externally connected network and open Properties
3) Choose the Sharing tab, and enable internet connection sharing.

Next, execute the following command to have the network settings take effective on coLinux.

Example 13-1 Network configuration command

```
colinux:~# /etc/init.d/networking restart
Reconfiguring network interfaces: done.
colinux:~#
```

As the network address 192.168.0.0/24 is automatically used for router connections, these settings will fail when the external network connection uses the same address. If this occurs, change the network address of external connection.

Refer to section 13.1.8, “Windows Network Settings under Special Circumstances” when the network address of the external connection cannot be changed.

13.1.5. Creating a coLinux User

Enter the following command at the coLinux prompt to create a new user. Specify a password if necessary.

Example 13-2 Adding user “somebody”

```
colinux:~# adduser somebody
Adding user somebody...
Adding new group somebody (1000).
Adding new user somebody (1000) with group somebody.
Creating home directory /home/somebody.
Copying files from /etc/skel
Enter new UNIX password:
```

13.1.6. File Sharing between Windows and coLinux

This method makes it possible to exchange files between coLinux and Windows using the Windows shared folder. Enter the smbmount command at the coLinux prompt as follows, and then enter the password for the shared folder.

Example 13-3 Windows IP address: 192.168.0.100, Shared folder name: shared

```
colinux:~# mkdir /mnt/smb
colinux:~# smbmount //192.168.0.100/shared /mnt/smb
212: session request to 192.168.0.100 failed (Called name not present)
212: session request to 192 failed (Called name not present)
Password:
```

If the coLinux user name differs to that on the Windows side, specify the user name as a command option. For details, enter the man smbmount command and refer to the help.

Thereafter, the data in the shared folder “shared” in Windows will be the same as in the /mnt/smb directory in coLinux.

13.1.7. Installing the Cross Development Environment

Install the cross development environment in coLinux by referring to Chapter 5, “Preparation of Development Environment.”

The necessary files for building the environment can be obtained in coLinux through the shared folder setup in the previous section.

Development of the Armadillo-9 can now be carried out from Windows. All following instructions are for special circumstances only.

13.1.8. Windows Network Settings under Special Circumstances

The following method is to be used when the network address of the external connection is 192.168.0.0/24

(For Windows XP)

Here, the “bridge connection” method is used.

1) Open Network Connections from the Control Panel.
2) Select both the externally connected network and the network that has the device name of “TAP-Win32 adapter.”
3) Select Bridge connections from the Advanced menu.

(For Windows2000)

In Windows2000, a network address other than 192.168.0.0/24 is used for the private network. Here, 192.168.1.0/24 is used.

1) Open Network and Dial-up Connections from the Control Panel.
2) Right-click the externally connected network and disable it.
3) Right-click the externally connected network and open Properties.
4) Select Internet Protocol (TCP/IP) from the General tab, and click the Properties button.
5) Select Use the following IP address and set to 192.168.100.100.
6) Open the Sharing tab, and enable internet connection sharing.
7) Right-click the network connection that has the “TAP-Win32 adapter” device name and open Properties.
8) Select Internet Protocol (TCP/IP) in the General tab, and click the Properties button.
9) Select Use the following IP address and set to 192.168.1.1.
10) Right-click the externally connected network and open Properties.
11) Select Internet Protocol (TCP/IP) from the General tab, and click the Properties button.
12) Set the IP address back to its original setting.
13) Right-click the externally connected network and enable it.

13.1.9. coLinux Network Settings

While DHCP is selected by default at time of installation, a fixed IP address must be set in an environment where there is no DHCP server operating.

The network settings can be displayed with the ifconfig command.

Example 13-4 ifconfig command execution

```
colinux:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:43:4F:4E:45:30
          inet addr:192.168.0.151  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:189 errors:0 dropped:0 overruns:0 frame:0
          TX packets:115 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:24472 (23.8 KiB)  TX bytes:9776 (9.5 KiB)
          Interrupt:2

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  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:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

colinux:~#
```

A fixed IP address must be set when the IP address of the eth0 device is not displayed. The IP address to be set should match that of the “TAP-Win32 adapter” network when the “router connection” method is being used, or match that of the externally connected network when the “bridge connection” method is being used.

The following shows how to implement the settings listed in the table below.

Table 13-1 Network Settings

Item	Settings
IP Address	192.168.1.100
Net Mask	255.255.255.0
Gateway	192.168.1.1
DNS Server	192.168.1.1

1) Edit /etc/network/interfaces in coLinux as follows.

Example 13-5 Editing the /etc/network/interfaces file

```
auto lo eth0
iface lo inetloopback
iface eth0 inet static
  address 192.168.1.100
  gateway 192.168.1.1
  netmask 255.255.255.0
```

2) Edit /etc/resolv.conf in coLinux as follows.

Example 13-6 Editing the /etc/resolv.conf file

```
nameserver 192.168.1.1
```

3) Execute the following command to update the network settings with the edited content.

Example 13-7 Network update command

```
colinux:~# /etc/init.d/networking restart
Reconfiguring network interfaces: done.
colinux:~#
```

Revision History

Version	Date	Revisions
1.0.0	2004.12.18	• First edition released
1.0.1	2004.12.28	• Addition of instructions for working with uClinux as a general user. • Unified the icons for “Caution” and “Tips”
1.0.2	2005.2.11	• Addition of “noinitrd” to the option for making an IDE drive a root device • Correction of mistake in the mount point description in section 11.1, “Creating an Armadillo-9 Bootable Compact Flash Card”

Armadillo-9 Software Manual February 11, 2005 version 1.0.2

Atmark Techno, Inc.
AFT Bldg, 6F, East 2 North 5, Chuo-ku, Sapporo, Japan 060-0035
TEL011-207-6550 FAX011-207-6570