Armadillo-200 シリーズ
220/230/240
ソフトウェアマニュアル

Version 2.1.2-d308169
2009/08/03

株式会社アットマークテクノ [http://www.atmark-techno.com]

Armadillo 開発者サイト [http://armadillo.atmark-techno.com]

Armadillo-200 シリーズ 220/230/240
ソフトウェアマニュアル

株式会社アットマークテクノ

060-0035 札幌市中央区北 5 条東 2 丁目 AFT ビル 6F
TEL 011-207-6550 FAX 011-207-6570

Version 2.1.2-d308169
2009/08/03

目次

図目次

表目次

1.はじめに

このマニュアルは Armadillo-220 と Armadillo-230、Armadillo-240 の 3 製品のソフトウェア開発について記載されています。特にシリーズ内製品の指定がない場合は Armadillo を代名詞として使用します。シリーズ全体を表わすときは「Armadillo-200 シリーズ」と呼びます。

表 1.1. 製品の呼び名

呼び名	説明
Armadillo	とくに指定がない場合の代名詞として
Armadillo-220	Armadilo-220 固有の場合
Armadillo-230	Armadillo-230 固有の場合
Armadillo-240	Armadilo-240 固有の場合
Armadillo-200 シリーズ	シリーズ製品全体を表わす場合

1.1. 対象となる読者

- Armadillo のソフトウェアをカスタマイズされる方
- 外部ストレージにシステム構築される方

上記以外の方でも、本書を有効に利用していただけたら幸いです。

1.2. 本書の構成

本書は、Armadillo のソフトウェアをカスタマイズする上で必要となる情報について記載しています。

- 開発環境の構築方法
- フラッシュメモリの書き換え方法
- ビルド方法

1.3. 表記について

1.3.1. フォント

本書では以下のような意味でフォントを使いわけています。

表 1.2. 使用しているフォント

フォント例	説明
本文中のフォント	本文
`[PC ~]$ ls`	プロンプトとユーザ入力文字列
text	編集する文字列や出力される文字列。またはコメント

1.3.2. コマンド入力例

本書に記載されているコマンドの入力例は、表示されているプロンプトによって、それぞれに対応した実行環境を想定して書かれています。「/」の部分はカレントディレクトリによって異なります。各ユーザのホームディレクトリは「~」で表わします。

表 1.3. 表示プロンプトと実行環境の関係

プロンプト	コマンドの実行環境
[PC /]#	作業用 PC 上の root ユーザで実行
[PC /]$	作業用 PC 上の一般ユーザで実行
[armadillo /]#	Armadillo 上の root ユーザで実行
[armadillo /]$	Armadillo 上の一般ユーザで実行
hermit>	Armadillo 上の保守モードで実行

コマンド中で、変更の可能性のあるものや、環境により異なるものに関しては以下のように表記します。適時読み替えて入力してください。

表 1.4. コマンド入力例での省略表記

表記	説明
[version]	ファイルのバージョン番号

1.3.3. アイコン

本書では以下のようにアイコンを使用しています。

注意事項を記載します。

役に立つ情報を記載します。

1.4. 謝辞

Armadillo で使用しているソフトウェアは Free Software / Open Source Software で構成されています。Free Software / Open Source Software は世界中の多くの開発者の成果によってなりたっています。この場を借りて感謝の意を表します。

1.5. ソフトウェア使用に関しての注意事項

本製品に含まれるソフトウェアについて	本製品に含まれるソフトウェア(付属のドキュメント等も含みます)は、現状のまま(AS IS)提供されるものであり、特定の目的に適合することや、そ

の信頼性、正確性を保証するものではありません。また、本製品の使用による結果についてもなんら保証するものではありません。

1.6. 商標について

Armadillo は株式会社アットマークテクノの登録商標です。その他の記載の商品名および会社名は、各社・各団体の商標または登録商標です。

2.作業の前に

2.1. 準備するもの

Armadillo を使用する前に、次のものを準備してください。

作業用 PC	Linux もしくは Windows が動作し、1 ポート以上のシリアルインターフェースを持つ PC です。
シリアルクロスケーブル(及び、Armadillo-240 では RS232C レベル変換アダプター)	D-Sub9 ピン(メス-メス)の「クロス接続用」ケーブルです。RS232C レベル変換アダプターをコネクタ基板に接続する際は、黄色のケーブルが 1 ピン側になるよう接続してください。
付属 CD-ROM(以降、付属 CD)	Armadillo-200 シリーズに関する各種マニュアルやソースコードが収納されています。
シリアルコンソールソフト	minicom や Tera Term などのシリアルコンソールソフトです。(Linux 用のソフトは付属 CD の「`tool`」ディレクトリにあります。)作業用 PC にインストールしてください。

2.2. 接続方法

下の図を参照して、シリアルクロスケーブル(と RS232C レベル変換アダプター)、AC アダプター、VGA モニタ、そして LAN ケーブルを Armadillo に接続してください。

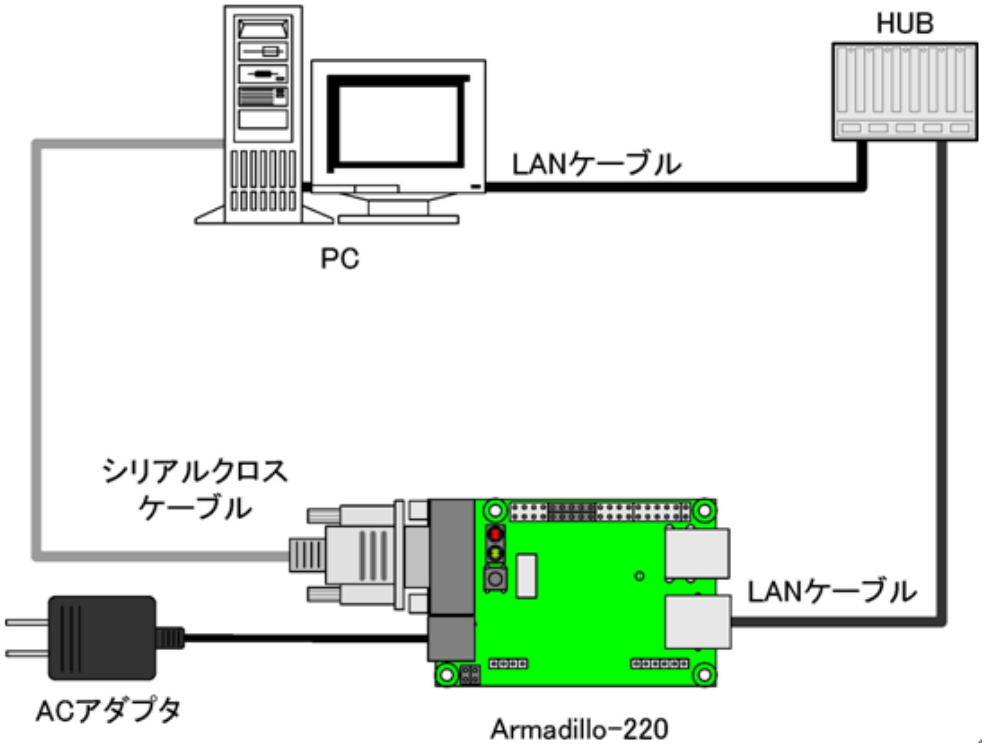

図 2.1. Armadillo-220 接続例

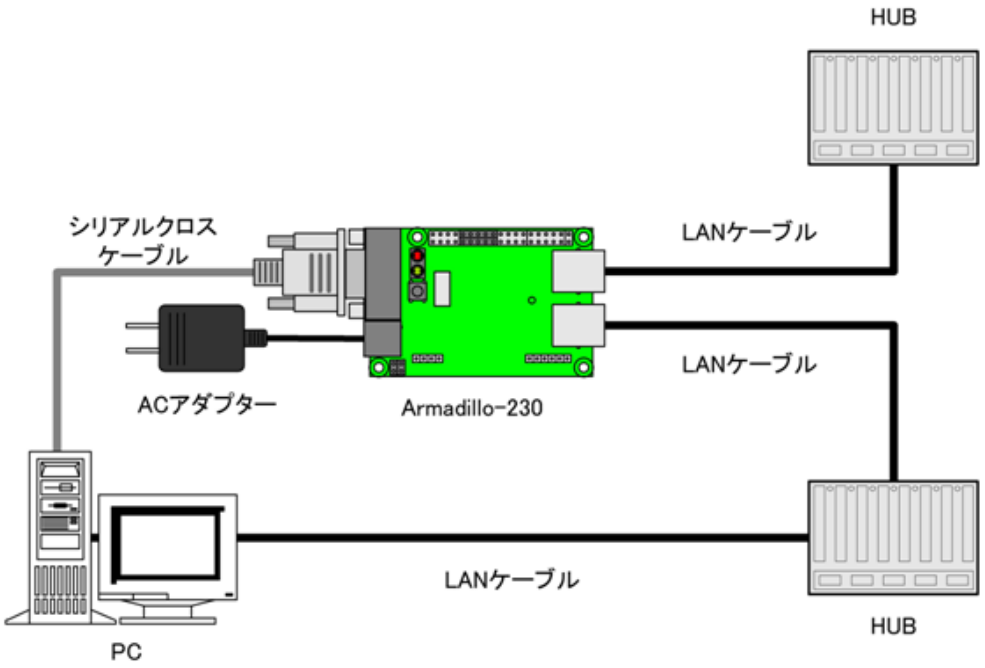

図 2.2. Armadillo-230 接続例

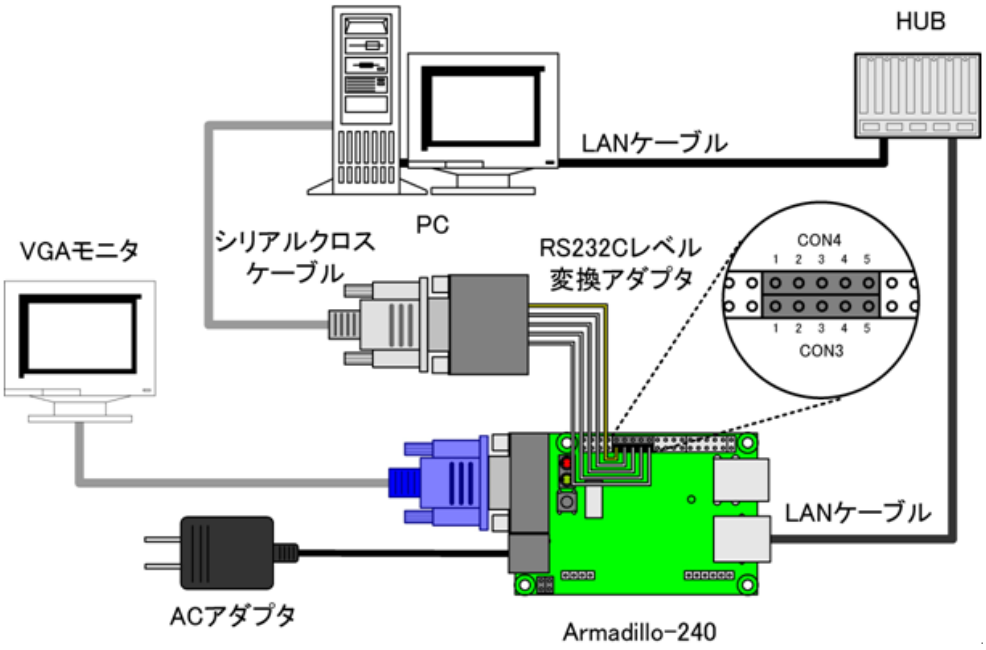

図 2.3. Armadillo-240 接続例

2.3. ジャンパピンの設定

Armadillo-200 シリーズではジャンパの設定を変えることで、ブート時の動作を変更することができます。以下の表に設定と動作の関連を記載します。Hermit や CPU オンチップブート ROM の使用については、 「5. フラッシュメモリの書き換え方法」 や 「5.5. ブートローダーを出荷状態に戻す」 で説明します。

表 2.1. ジャンパの設定とブート時の動作

JP1	JP2	ブート時の動作
オープン	オープン	**Linux カーネルを起動**
オープン	ショート	**Hermit コマンドプロンプトを起動**
ショート	-	**CPU オンチップブート ROM を起動**

ジャンパのオープンまたはショートとは、ジャンパピンにジャンパソケットを

- オープン: 挿さない
- ショート: 挿す

状態を表わします。

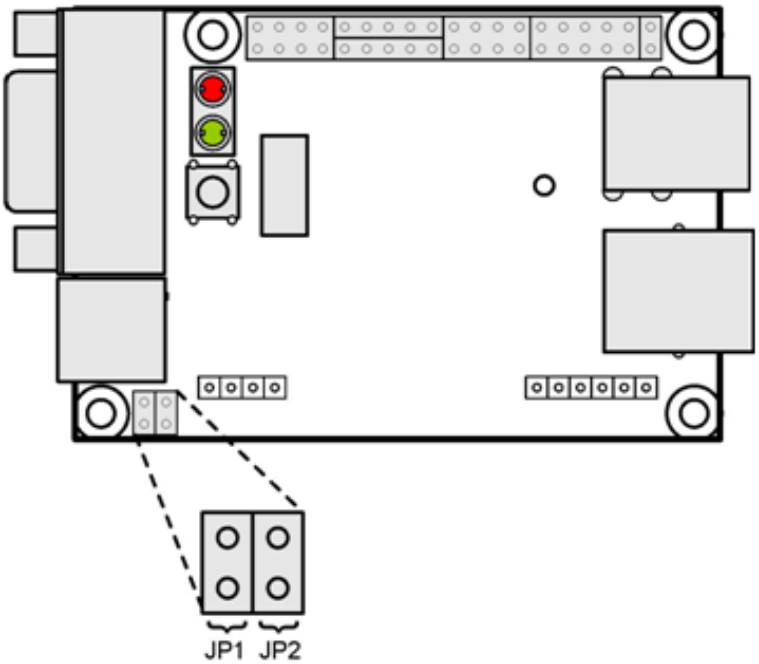

図 2.4. ジャンパの位置

3.開発環境の準備

Armadillo のソフトウェア開発には、Debian/GNU Linux 系の OS 環境[1](Debian etch を標準とします）が必要です。作業用 PC が Windows の場合、仮想的な Linux 環境を構築する必要があります。

Windows 上に Linux 環境を構築する方法として、「VMware」を推奨しています。VMware を使用する場合は、開発に必要なソフトウェアがインストールされた状態の OS イメージ「ATDE（Atmark Techno Development Environment)」[2]を提供しています。

Windows 上に Linux 環境を構築する手順についてのドキュメントは以下のとおりです。詳しくは、こちらを参照してください。

- ATDE Install Guide
- coLinux Guide

ATDE をお使いになる場合は、本章で新たにインストールする必要はありません。

3.1. クロス開発環境パッケージのインストール

付属 CD の `cross-dev/deb` ディレクトリにクロス開発環境パッケージが用意されています。サポートしている開発環境は、「表 3.1. 開発環境一覧」のとおりです。通常は、arm クロス開発環境をインストールしてください。`cross-dev/deb/`クロスターゲットディレクトリ以下のパッケージをすべてインストールしてください。インストールは必ず root ユーザで行ってください。「図 3.1. インストールコマンド」のようにコマンドを実行します。

表 3.1. 開発環境一覧

クロスターゲット	説明
arm	通常の ARM クロス開発環境です。

```
[PC ~]# dpkg --install *.deb
```

図 3.1. インストールコマンド

ご使用の開発環境に既に同一のターゲット用クロス開発環境がインストールされている場合、新しいクロス開発環境をインストールする前に必ずアンインストールするようにしてください。

[1]debian 系以外の Linux でも開発はできますが、本書記載事項すべてが全く同じように動作するわけではありません。各作業はお使いの Linux 環境に合わせた形で自己責任のもと行ってください。
[2]Armadillo の開発環境としては、ATDE v2.0 以降を推奨しています。

3.2. atmark-dist のビルドに必要なパッケージ

atmark-dist をビルドするためには、「表 3.2. atmark-dist のビルドに必要なパッケージ一覧」に示すパッケージを作業用 PC にインストールされている必要があります。作業用 PC の環境に合わせて適切にインストールしてください。

表 3.2. atmark-dist のビルドに必要なパッケージ一覧

パッケージ名	バージョン	備考
genext2fs	1.3-7.1-cvs20050225	付属 CD の cross-dev ディレクトリに収録されています
file	4.12-1 以降	
sed	4.1.2-8 以降	
perl	5.8.4-8 以降	
bison	1.875d 以降	
flex	2.5.31 以降	
libncurses5-dev	5.4-4 以降	

現在インストールされているバージョンを表示するには、「図 3.2. インストール情報表示コマンド」のようにパッケージ名を指定して実行してください。

`--list` はパッケージ情報を表示する dpkg のオプションです。`file` にはバージョンを表示したいパッケージ名を指定します。

```
[PC ~]# dpkg --list file
```

図 3.2. インストール情報表示コマンド

3.3. クロス開発用ライブラリパッケージの作成方法

アプリケーション開発を行う際に、付属 CD には収録されていないライブラリパッケージが必要になることがあります。ここでは、ARM のクロス開発用ライブラリパッケージの作成方法を紹介します。

まず、作成したいクロス開発用パッケージの元となるライブラリパッケージを取得します。元となるパッケージは、ARM 用のパッケージです。例えば、libjpeg6b の場合「`libjpeg6b_[version]_arm.deb`」というパッケージになります。

次のコマンドで、取得したライブラリパッケージをクロス開発用に変換します。

```
[PC ~]$ dpkg-cross --build --arch arm libjpeg6b_[version]_arm.deb
[PC ~]$ ls
libjpeg6b-arm-cross_[version]_all.deb libjpeg6b_[version]_arm.deb
```

図 3.3. クロス開発用ライブラリパッケージの作成

Debian etch 以外の Linux 環境で dpkg-cross を行った場合、インストール可能なパッケージを生成できない場合があります。

4.使用方法

この章では Armadillo の基本的な使用方法の説明を行います。

4.1. シリアル通信ソフトウェアの設定

シリアル通信ソフトウェアを起動し、シリアルの通信設定を、「表 4.1. シリアル通信設定」のように設定してください。

Armadillo-240 では、RS232C レベル変換アダプターを経由させる必要があります。

表 4.1. シリアル通信設定

項目	設定
転送レート	115,200bps
データ長	8bit
ストップビット	1bit
パリティ	なし
フロー制御	なし

4.2. 起動

JP1、JP2 をオープンに設定して電源を接続すると、Linux が起動します。正常に起動した場合、シリアルインターフェース 1 に起動ログが出力されます。以下は、Armadillo-240 における例です。

```
Uncompressing
kernel.....................................................................
.......................done.
Uncompressing
ramdisk....................................................................
.................................................
.....................................................................................don
e.
Doing console=ttyAM0,115200
Doing mtdparts=armadillo2x0-nor:0x10000(bootloader)ro,0x170000(kernel),0x670000
(userland),-(config)
Linux version 2.6.12.3-a9-5 (atmark@pc-nsx) (gcc version 3.4.4 20050314
(prerelease)
(Debian 3.4.3-13)) #1 Fri Jun 16 18:43:41 JST 2006
CPU: ARM920Tid(wb) [41129200] revision 0 (ARMv4T)
CPU0: D VIVT write-back cache
CPU0: I cache: 16384 bytes, associativity 64, 32 byte lines, 8 sets
CPU0: D cache: 16384 bytes, associativity 64, 32 byte lines, 8 sets
Machine: Armadillo-240
ATAG_INITRD is deprecated; please update your bootloader.
```

```
Memory policy: ECC disabled, Data cache writeback
Built 1 zonelists
Kernel command line: console=ttyAM0,115200 mtdparts=armadillo2x0-nor:0x10000
(bootloader)ro,0x170000(kernel),0x670000(userland),-(config)
PID hash table entries: 512 (order: 9, 8192 bytes)
Dentry cache hash table entries: 16384 (order: 4, 65536 bytes)
Inode-cache hash table entries: 8192 (order: 3, 32768 bytes)
Memory: 32MB 32MB = 64MB total
Memory: 54768KB available (2670K code, 609K data, 96K init)
Mount-cache hash table entries: 512
CPU: Testing write buffer coherency: ok
checking if image is initramfs...it isn't (bad gzip magic numbers); looks like an
initrd
Freeing initrd memory: 6591K
NET: Registered protocol family 16
SCSI subsystem initialized
usbcore: registered new driver usbfs
usbcore:  registered new iver hub
Bluetooth: Core ver 2.7
NET: Registered protocol family 31
Bluetooth: HCI device and connection manager initialized
Bluetooth: HCI socket layer initialized
NetWinder Floating Point Emulator V0.97 (double precision)
JFFS2 version 2.2. (NAND) (C) 2001-2003 Red Hat, Inc.
Initializing Cryptographic API
fb0: EP93xx frame buffer at 800x600x24
gpio: Armadillo-2x0 GPIO driver, (C) 2005-2006 Atmark Techno, Inc.
led: Armadillo-2x0 LED driver, (C) 2005-2006 Atmark Techno, Inc.
sw: Armadillo-2x0 Takt Switch driver, (C) 2006 Atmark Techno, Inc.
ttyAM0 at MMIO 0x808c0000 (irq = 52) is a EP93XX
ttyAM1 at MMIO 0x808d0000 (irq = 54) is a EP93XX
ttyAM2 at MMIO 0x808e0000 (irq = 55) is a EP93XX
io scheduler noop registered
io scheduler anticipatory registered
io scheduler deadline registered
io scheduler cfq registered
RAMDISK driver initialized: 16 RAM disks of 16384K size 1024 blocksize
loop: loaded (max 8 devices)
i2c /dev entries driver
i2c-armadillo9: i2c Armadillo-9 driver, (C) 2004-2005 Atmark Techno, Inc.
i2c-at24cxx: i2c at24cxx eeprom driver, (C) 2003-2005 Atmark Techno, Inc.
armadillo2x0-nor: Found 1 x16 devices at 0x0 in 16-bit bank
Amd/Fujitsu Extended Query Table at 0x0040
armadillo2x0-nor: CFI does not contain boot bank location. Assuming top.
number of CFI chips: 1
cfi_cmdset_0002: Disabling erase-suspend-program due to code brokenness.
4 cmdlinepart partitions found on MTD device armadillo2x0-nor
parse_mtd_partitions:4
Creating 4 MTD partitions on "armadillo2x0-nor":
0x00000000-0x00010000 : "bootloader"
0x00010000-0x00180000 : "kernel"
0x00180000-0x007f0000 : "userland"
0x007f0000-0x00800000 : "config"
No NAND device found!!!
ep93xxusb ep93xxusb.0: EP93xx OHCI
ep93xxusb ep93xxusb.0: new USB bus registered, assigned bus number 1
ep93xxusb ep93xxusb.0: irq 56, io base 0xff020000
hub 1-0:1.0: USB hub found
```

```
hub 1-0:1.0: 3 ports detected
Initializing USB Mass Storage driver...
usbcore: registered new driver usb-storage
USB Mass Storage support registered.
usbcore: registered new driver usbhid
drivers/usb/input/hid-core.c: v2.01:USB HID core driver
pegasus: v0.6.12 (2005/01/13), Pegasus/Pegasus II USB Ethernet driver
usbcore: registered new driver pegasus
zd1211 - http://zd1211.ath.cx/
Based on www.zydas.com.tw driver version 2.0.0.0
usbcore: registered new driver zd1211
Bluetooth: HCI USB driver ver 2.8
usbcore: registered new driver hci_usb
NET: Registered protocol family 2
IP: routing cache hash table of 512 buckets, 4Kbytes
TCP established hash table entries: 4096 (order: 3, 32768 bytes)
TCP bind hash table entries: 4096 (order: 2, 16384 bytes)
TCP: Hash tables configured (established 4096 bind 4096)
IPv4 over IPv4 tunneling driver
ip_tables: (C) 2000-2002 Netfilter core team
Initializing IPsec netlink socket
NET: Registered protocol family 1
NET: Registered protocol family 10
Disabled Privacy Extensions on device c02ee504(lo)
IPv6 over IPv4 tunneling driver
NET: Registered protocol family 17
NET: Registered protocol family 15
Bluetooth: L2CAP ver 2.7
Bluetooth: L2CAP socket layer initialized
Bluetooth: RFCOMM ver 1.5
Bluetooth: RFCOMM socket layer initialized
Bluetooth: RFCOMM TTY layer initialized
SCTP: Hash tables configured (established 2048 bind 4096)
RAMDISK: ext2 filesystem found at block 0
RAMDISK: Loading 6591KiB [1 disk] into ram disk... done.
VFS: Mounted root (ext2 filesystem).
Freeing init memory: 96K
init started:  BusyBox v1.00 (2006.06.16-11:12+0000) multi-call binary
Starting fsck for root filesystem.
fsck 1.25 (20-Sep-2001)
ext2fs_check_if_mount: No such file or directory while determining whether /dev/
ram0
is mounted.
/dev/ram0: clean, 614/1024 files, 5354/6591 blocks
Checking root filesystem:        done
Remounting root rw:              done
Mounting proc:                   done
Mounting usbfs:                  done
Mounting sysfs:                  done
Loading /etc/config:             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
Loading /etc/config:             done
```

```
Configuring network interfaces:  done
Starting inetd:                  done
Stating sshd:                    done
Starting thttpd:                 done

atmark-dist v1.5.1 (AtmarkTechno/Armadillo-240.Base)
Linux 2.6.12.3-a9-5 [armv4tl arch]

a240-0 login:
```

図 4.1. 起動ログ(Armadillo-240 の例)

ベースイメージのユーザーランドでは、ログインプロンプトはシリアルインターフェース 1 とシリアルインターフェース 2 に表示されます。

4.3. コンソールログイン時のユーザ名とパスワード

ログインユーザは、次の 2 種類が用意されています。

表 4.2. コンソールログイン時のユーザ名とパスワード

ユーザ名	パスワード	権限
root	root	root ユーザ
guest	(なし)	一般ユーザ

4.4. ディレクトリ構成

Armadillo 内のディレクトリ構成は次のようになっています。

表 4.3. ディレクトリ構成の一覧

ディレクトリ名	説　明
/bin	アプリケーション用
/dev	デバイスノード用
/etc	システム設定用
/etc/config	flatfsd 向け設定用
/lib	共有ライブラリ用
/mnt	マウントポイント用
/proc	プロセス情報用
/root	root ホームディレクトリ
/sbin	システム管理コマンド用
/usr	ユーザ共有情報用
/home	ユーザホームディレクトリ
/home/ftp/pub	ftp データ送受信用
/home/www-data	WEB サーバホームディレクトリ用
/tmp	テンポラリ保存用
/var	変更データ用

4.5. 終了

電源を切断することで Armadillo を終了させます。

ただし IDE ドライブやコンパクトフラッシュがマウントされている場合は、電源切断前にアンマウントするか、halt コマンドを実行してシステムを停止させてから電源を切断してください。これを行わない場合は、IDE ドライブやコンパクトフラッシュのデータが破損する恐れがあります。

4.6. ネットワーク設定

Armadillo の「/etc/config/interfaces」ファイルを編集することで、ネットワークの設定を変更することができます。Armadillo-230 はネットワークインターフェイスを 2 つ搭載しているため、通常の eth0 に加え eth1 も存在します。USB のインターフェイスを持つ Armadillo で USB 対応 LAN アダプターを使用する場合も同じです。eth1 側を設定する場合、以降 eth0 の個所を eth1 に読み替えてください。また詳しい interfaces の書き方については、interfaces のマニュアルを参照してください。

4.6.1. 固定 IP アドレスで使用する場合

固定 IP アドレスを指定する場合の設定例を次に示します。

表 4.4. ネットワーク設定例

項目	設定値
IP アドレス	192.168.10.10
ネットマスク	255.255.255.0
ブロードキャストアドレス	192.168.10.255
デフォルトゲートウェイ	192.168.10.1

```
# /etc/config/interfaces - configuration file for ifup(8), ifdown(8)auto lo eth0
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
```

図 4.2. ネットワーク設定例(固定 IP アドレス時)

ゲートウェイを使用しない場合、gateway に 0.0.0.0 を指定してください。

```
gateway 0.0.0.0
```

図 4.3. ネットワーク設定例(ゲートウェイの無効化)

4.6.2. DNS サーバの設定

DNS サーバを設定する場合、/etc/config/resolv.conf を変更します。

```
nameserver 192.168.10.1
```

図 4.4. DNS サーバの設定

変更は即座に適用されます。

4.6.3. DHCP を使用する場合

DHCP を利用して IP アドレスを取得する場合の設定例を次に示します。

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

auto lo eth0

iface lo inet loopback

iface eth0 inet dhcp
```

図 4.5. ネットワーク設定例(DHCP 使用時)

4.6.4. ネットワーク接続の開始と終了

ネットワーク接続を開始するには ifup を、ネットワーク接続を終了するには ifdown というコマンドを使用します。コマンドには開始または終了させたいインターフェイスを指定してください。

```
[armadillo /]# ifup eth0
```

図 4.6. ネットワーク接続の開始

```
[armadillo /]# ifdown eth0
```

図 4.7. ネットワーク接続の終了

4.6.5. ネットワーク設定をフラッシュメモリに保存する

ネットワーク設定に必要なファイルは、/etc/config/ディレクトリにあります。このディレクトリにあるファイルをフラッシュメモリに保存するには、flatfsd というコマンドを使います。オプション「-s」を指定し、Armadillo 上で flatfsd を実行してください。

```
[armadillo /etc/config]# flatfsd -s
```

これで書き換えたネットワーク設定がフラッシュメモリに書き込まれ、次回以降の起動時に反映されます。

4.7. ネットワークブリッジの設定

複数のネットワークインターフェイスを持つ Armadillo では、ネットワークブリッジ機能を利用することができます。設定にはブリッジユーティリティ「brctl」コマンドを利用します。

ネットワークブリッジインターフェイスには、eth0 などのインターフェイスと同じように IP 番号を割り当てることができます。IP アドレスを割り当てることで、他の LAN インターフェイスと同じように使用することができます。IP 番号の設定については 「4.6. ネットワーク設定」 を参照してください。

この章では 2 つの LAN インターフェイスを持っている Armadillo-230 を例として使用します。

4.7.1. ネットワークブリッジ設定の準備

始めに、Armadillo-230 でネットワーク設定がすでに有効になっている場合、 「4.6.4. ネットワーク接続の開始と終了」 を参考にして無効にしてください。次に、各ネットワークインターフェイスの保持する IP アドレスを完全に解放した上で有効化するために、以下のようにコマンド入力してください。

```
[a230 ~]# ifconfig eth0 0.0.0.0
[a230 ~]# ifconfig eth1 0.0.0.0
```

図 4.8. ブリッジに追加するインターフェイスの有効化

4.7.2. ブリッジ作成

ブリッジを実現するため、brctl を利用して論理的なブリッジインターフェイスを作成します。続けて、このブリッジインターフェイスに 2 つのネットワークインターフェイスを追加します。

```
[a230 ~]# brctl addbr br0
[a230 ~]# brctl addif br0 eth0
device eth0 entered promiscuous mode
[a230 ~]# brctl addif br0 eth1
device eth1 entered promiscuous mod
```

図 4.9. ブリッジの作成

4.7.3. ブリッジの有効化

ブリッジを有効にする方法は、通常のネットワークインターフェイスと同様です。以下のコマンドを入力すると、ブリッジが動作し始めます。

```
[a230 ~]# ifconfig br0 0.0.0.0
br0: port 2(eth1) entering learning state
br0: port 1(eth0) entering learning state
br0: port 2(eth1) entering forwarding state
br0: port 1(eth0) entering forwarding state
```

図 4.10. ブリッジの有効化

4.7.4. ブリッジの廃棄

ブリッジの使用をやめる場合、まずブリッジインターフェイスを無効化するコマンドを入力します。

```
[a230 ~]# ifconfig br0 down
br0: port 2(eth1) entering disabled state
br0: port 1(eth0) entering disabled state
```

図 4.11. ブリッジの無効化

次に、ブリッジを完全に廃棄するために、追加されているネットワークインターフェイスを外し、最後にブリッジインターフェイスを削除します。

```
[a230 ~]# brctl delif br0 eth0
[a230 ~]# brctl delif br0 eth1
[a230 ~]# brctl delbr br0
```

図 4.12. ブリッジの廃棄

4.7.5. ブリッジのスクリプト例

brctl コマンドを利用すると、ブリッジの状態表示や STP というブリッジプロトコルの設定ができます。Armadillo-230 では、こうした機能を利用するためのサンプルスクリプト「/etc/init.d/bridges」が用意されています。このスクリプトを利用することで、簡単にブリッジ設定や STP の設定を行うことができます。以下は、このスクリプトを利用してブリッジを有効化する場合の例です。

```
[a230 ~]# /etc/init.d/bridges create
Creating bridge:
device eth0 entered promiscuous mode
device eth1 entered promiscuous mode
Upping bridge (8sec):
br0: port 2(eth1) entering listening state
br0: port 1(eth0) entering listening state
br0: port 2(eth1) entering learning state
br0: port 1(eth0) entering learning state
br0: topology change detected, propagating
br0: port 2(eth1) entering forwarding state
br0: topology change detected, propagating
br0: port 1(eth0) entering forwarding state
```

図 4.13. ブリッジのスクリプト例

4.8. telnet ログイン

次のユーザ名/パスワードで telnet ログインが可能です。root でのログインは行えません。root 権限が必要な作業を行う場合、guest でログイン後に「su」コマンドで root 権限を取得してください。

表 4.5. telnet ログイン時のユーザ名とパスワード

ユーザ名	パスワード
guest	なし

Armadillo-220/230/240 の Recover イメージ(出荷状態)の起動直後の状態では、telnet ログインをすることができません。telnet ログインをするには、/etc/inetd.conf を編集し、以下のコマンドを実行してください。

```
telnet  stream  tcp   nowait   root    /usr/sbin/telnetd      telnetd -l /bin/login
```

図 4.14. /etc/inetd ファイル編集例

```
[armadillo ~]# iptables --append INPUT --proto tcp --dport telnet --jump ACCEPT
```

図 4.15. ファイアウォールの設定コマンド入力例

```
[armadillo ~]# inetd
```

図 4.16. スーパーサーバ起動コマンド

4.9. ファイル転送

ftp によるファイル転送が可能です。次のユーザ/パスワードでログインしてください。ホームディレクトリは「/home/ftp」です。「/home/ftp/pub」ディレクトリに移動することでデータの書き込みが可能になります。

表 4.6. ftp のユーザ名とパスワード

ユーザ名	パスワード
ftp	なし

Armadillo-220/230/240 の Recover イメージ(出荷状態)の起動直後の状態では、ftp によるファイル転送をすることができません。ftp によるファイル転送をするには、 「図 4.16. スーパーサーバ起動コマンド」 を実行してください。

4.10. Web サーバ

thttpd という小さな HTTP サーバが起動しており、Web ブラウザを使って Armadillo にアクセスすることができます。データディレクトリは「/home/www-data」です。URL は「http://(Armadillo-240 の IP アドレス)/」になります。(例 http://192.168.10.10/)

4.11. ssh ログイン

次のユーザ名/パスワードで ssh ログインが可能です。root でのログインは行えません。root 権限が必要な作業を行う場合、guest でログイン後に「su」コマンドで root 権限を取得してください。

表 4.7. ssh ログイン時のユーザ名とパスワード

ユーザ名	パスワード
guest	なし

5.フラッシュメモリの書き換え方法

フラッシュメモリの内容を書き換えることで、Armadillo の機能を変更することができます。この章ではフラッシュメモリの書き換え方法を説明します。

何らかの原因により「書き換えイメージの転送」に失敗した場合、Armadillo が正常に起動しなくなる場合があります。書き換えの際は次の点に注意してください。

- Armadillo の電源を切らない
- Armadillo と開発用 PC を接続しているシリアルケーブルと LAN ケーブルを外さない

5.1. ダウンローダのインストール

作業用 PC にダウンローダをインストールします。

ダウンローダの種類には、「表 5.1. ダウンローダ一覧」のようなものがあります。

表 5.1. ダウンローダ一覧

ダウンローダ	OS タイプ	説明
hermit-at	Linux	Linux 用の CUI アプリケーションです。
shoehorn-at	Linux	Linux 用の CUI アプリケーションです。
hermit-at-win	Windows	Windows 用の GUI アプリケーションです。

ATDE(Atmark Techno Development Environment)を利用する場合、ダウンローダパッケージはすでにインストールされているので、インストールする必要はありません。

5.1.1. 作業用 PC が Linux の場合

付属 CD の downloader/deb ディレクトリよりパッケージファイルを用意し、インストールします。必ず **root ユーザ**で行ってください。

```
[PC ~]# dpkg --install hermit-at_[version]_i386.deb
[PC ~]# dpkg --install shoehorn-at_[version]_i386.deb
```

図 5.1. ダウンローダのインストール（Linux）

5.1.2. 作業用 PC が Windows の場合

付属 CD の `downloader/win32/hermit-at-win_[version].zip` を任意のフォルダに展開します。

5.2. フラッシュメモリの書き込み領域について

フラッシュメモリの書き込み先頭アドレスは、領域（リージョン）名で指定することができます。書き込み領域毎に指定するイメージファイルは、「表 5.2. リージョン名と対応するイメージファイル 」のようになります。

表 5.2. リージョン名と対応するイメージファイル [1]

製品	領域名	ファイル名
Armadillo-210	bootloader	loader-armadillo2x0-*[version]*.bin
	kernel	linux-a210-*[version]*.bin.gz
	userland	romfs-a210-recover-*[version]*.img.gz romfs-a210-base-*[version]*.img.gz
Armadillo-220/230/240	bootloader	loader-armadillo2x0-eth-*[version]*.bin
	kernel	linux-a2x0-*[version]*.bin.gz
	userland	romfs-a2x0-recover-*[version]*.img.gz romfs-a2x0-base-*[version]*.img.gz
Armadillo-9	bootloader	loader-armadillo9-*[version]*.bin
	kernel	linux-*[version]*.bin.gz
	userland	romfs-*[version]*.img.gz
Armadillo-300	ipl	ipl-a300.bin(※書き換え不可)
	bootloader	loader-armadillo-3x0-*[version]*.bin
	kernel	linux-a300-*[version]*.bin.gz
	userland	romfs-a300-*[version]*.img.gz
Armadillo-500	bootloader	loader-armadillo5x0-*[version]*.bin
	kernel	linux-a500-*[version]*.bin.gz
	userland	romfs-a500-*[version]*.img.gz
Armadillo-500 FX	bootloader	loader-armadillo5x0-fx-*[version]*.bin
	kernel	linux-a500-fx-*[version]*.bin.gz
	userland	romfs-a500-fx-*[version]*.img.gz

[1]「x」にはバージョン番号の任意の数値が入ります。

一部製品のユーザーランドには、Recover と Base という 2 種類のイメージファイルが用意されています。Recover イメージは、出荷状態でオンボードフラッシュメモリに書き込まれていて、各製品の特徴や性能を利用するアプリケーションが含まれています。Base イメージは、開発のベースとなるように、基本的なアプリケーションやツールのみが含まれています。

5.3. Hermit-At ダウンローダを使用してフラッシュメモリを書き換える

ここでは、Hermit-At ダウンローダを使用してフラッシュメモリを書き換える手順について説明します。「5.1. ダウンローダのインストール」でインストールした Hermit-At ダウンローダを使用します。これは、Armadillo のブートローダーと協調動作を行い、作業用 PC から Armadillo のフラッシュメモリを書き換えることができます。

5.3.1. 準備

「2.3. ジャンパピンの設定」を参照し、Hermit-At を起動してください。

Armadillo と接続している作業用 PC のシリアルインターフェースが他のアプリケーションで使用されていないことを確認します。使用されている場合は、該当アプリケーションを終了するなどしてシリアルインターフェースを開放してください。

5.3.2. 作業用 PC が Linux の場合

「図 5.2. ダウンロードコマンド」のようにコマンドを実行します。

`download` は hermit のサブコマンドの一つです。`--input-file` で指定されたファイルをターゲットボードに書き込む時に使用します。`--region` は書き込み対象の領域を指定するオプションです。下記の例では、「kernel 領域に linux.bin.gz を書き込む」という命令になります。

```
[PC ~]$ hermit download --input-file linux.bin.gz --region kernel
```

図 5.2. ダウンロードコマンド

シリアルインターフェースが ttyS0 以外の場合は、「図 5.3. ダウンロードコマンド（ポート指定） 」のように`--port` オプションを使用してポートを指定してください。

```
[PC ~]$ hermit download --input-file linux.bin.gz --region kernel --port ttyS1
```

図 5.3. ダウンロードコマンド（ポート指定） [1]

bootloader リージョンは、誤って書き換えることがないように簡易プロテクトされています。書き換える場合は、「図 5.4. ダウンロードコマンド（アンプロテクト）[1]」のように`--force-locked` オプションを使用して、プロテクトの解除をしてください。

```
[PC ~]$ hermit download --input-file loader-armadillo5x0-fx.bin --region
bootloader --force-locked
```

図 5.4. ダウンロードコマンド（アンプロテクト）[1]

[1] コマンドは 1 行で入力します。

bootloader リージョンに誤ったイメージを書き込んでしまった場合、オンボードフラッシュメモリからの起動ができなくなります。この場合は「5.5. ブートローダーを出荷状態に戻す」を参照してブートローダーを復旧してください。

5.3.3. 作業用 PC が Windows の場合

hermit-at-win.exe を実行します。「図 5.5. Hermit-At：Download ウィンドウ」が表示されます。

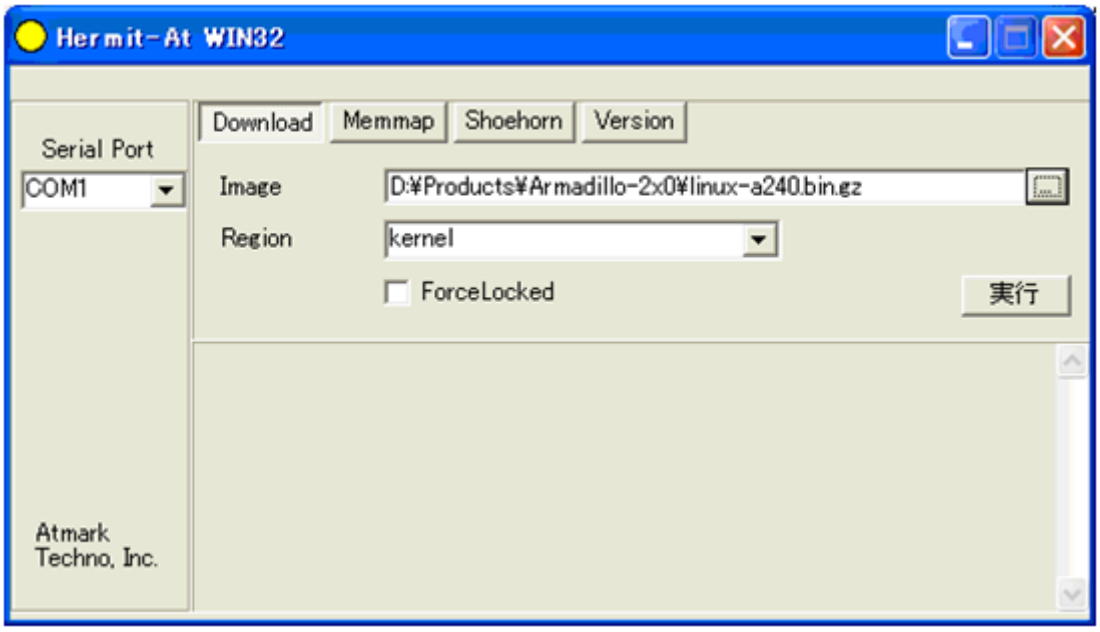

図 5.5. Hermit-At：Download ウィンドウ

Armadillo と接続されているシリアルインターフェースを「Serial Port」に指定してください。ドロップダウンリストに表示されない場合は、直接ポートを入力してください。

Image には書き込むファイルを指定してください。Region には書き込み対象のリージョンを指定してください。all や bootloader リージョンを指定する場合は、Force Locked をチェックしてください。

すべて設定してから実行ボタンをクリックします。「図 5.6. Hermit-At：download ダイアログ」が表示されます。

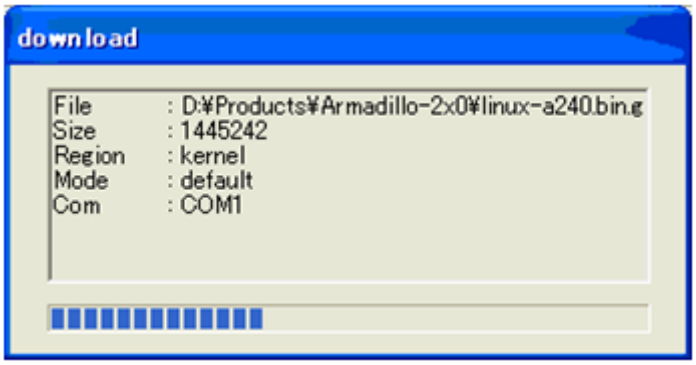

図 5.6. Hermit-At：download ダイアログ

ダウンロードの設定と進捗状況が表示されます。ダウンロードが完了するとダイアログはクローズされます。

5.4. netflash を使用してフラッシュメモリを書き換える

Linux アプリケーションの netflash を使用してフラッシュメモリを書き換えることができます。netflash は、所属するネットワークにある HTTP サーバーや FTP サーバーが公開しているファイルをダウンロードしてフラッシュメモリを書き換えることができます。

Armadillo にログインし、「図 5.7. netflash コマンド例」のようにコマンドを実行します。

```
[armadillo ~]# netflash -k -n -u -r /dev/flash/kernel [URL]
```

図 5.7. netflash コマンド例

オプションの"-r [デバイスファイル名]"で書き込み対象のリージョンを指定しています。「表 5.3. リージョンとデバイスファイルの対応」を参照してください。その他のオプションについては、netflash -h で詳細を確認する事ができます。

表 5.3. リージョンとデバイスファイルの対応

リージョン	デバイスファイル
カーネル	/dev/flash/kernel
ユーザランド	/dev/flash/userland

5.5. ブートローダーを出荷状態に戻す

loader-armadillo-2x0-notty が書き込まれている Armadillo のブートローダーを書き換えるときや、不正なブートローダーを書き込んでしまい Armadillo がブートできなくなってしまった場合の対処方法について説明します。Armadillo-200 シリーズの CPU にはオンチップブート ROM が搭載されており、この ROM に格納されているソフトウェアを使用して、ブートローダーを出荷状態に戻すことができます。以下にその手順を説明します。

5.5.1. ブートローダーの種類

Armadillo には複数のブートローダーが用意されています。ブートローダーの一覧は、「7.3. ブートローダーイメージのビルド」を参照してください。

5.5.2. 準備

Armadillo の電源が**切断**されていることを確認し、Armadillo のジャンパ JP1 をショートに設定してください。

Armadillo と接続している作業用 PC のシリアルインターフェースが他のアプリケーションで使用されていないことを確認します。使用されている場合は、該当アプリケーションを終了するなどしてシリアルインターフェースを開放してください。

5.5.3. 作業用 PC が Linux の場合

「図 5.8. shoehorn コマンド例」のようにコマンド[2]を実行してから、Armadillo の電源を入れてください。

```
[PC ~]$ shoehorn --boot --terminal --initrd /dev/null
            --kernel /usr/lib/hermit/loader-armadillo2x0-boot.bin
            --loader /usr/lib/shoehorn/shoehorn-armadillo2x0.bin
            --initfile /usr/lib/shoehorn/shoehorn-armadillo2x0.init
            --postfile /usr/lib/shoehorn/shoehorn-armadillo2x0.post
```

図 5.8. shoehorn コマンド例

上記は、作業用 PC のシリアルインターフェース"/dev/ttyS0"に Armadillo を接続した場合の例です。他のシリアルインターフェースに接続した場合は、shoehorn コマンドのオプションに

--port [シリアルインターフェース名]

を追加してください。

すぐにメッセージ表示が開始されます。正常に表示されない場合、Armadillo の電源を切断し、シリアルケーブルの接続や Armadillo のジャンパ設定を再度確認してください。

この状態で、**ジャンパの設定変更や電源の切断をしないで** Ctrl-C を押して Shoehorn を終了してから、「5.3. Hermit-At ダウンローダを使用してフラッシュメモリを書き換える」を参照してブートローダの書き込みを行ってください。

5.5.4. 作業用 PC が Windows の場合

`hermit-at-win.exe` を実行し Shoehorn ボタンをクリックすると、「図 5.9. Hermit-At：Shoehorn ウィンドウ」が表示されます。

[2] 書面の都合上折り返して表記しています。通常は 1 行のコマンドとなります。

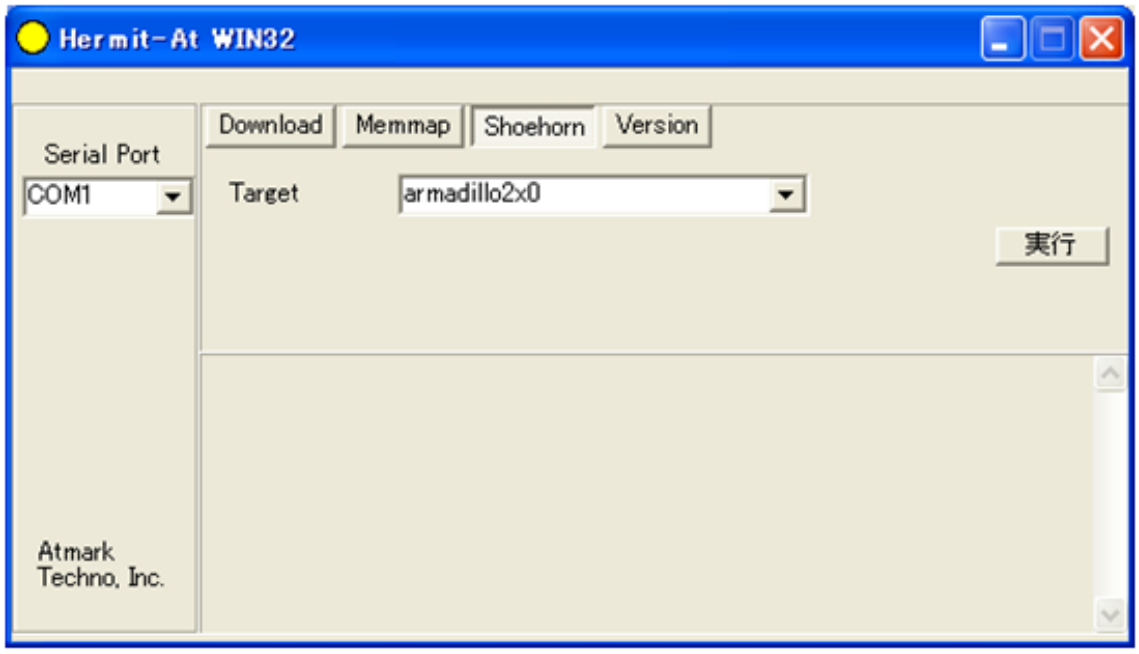

図 5.9. Hermit-At：Shoehorn ウィンドウ

Target に armadillo2x0 を選択して実行ボタンをクリックします。

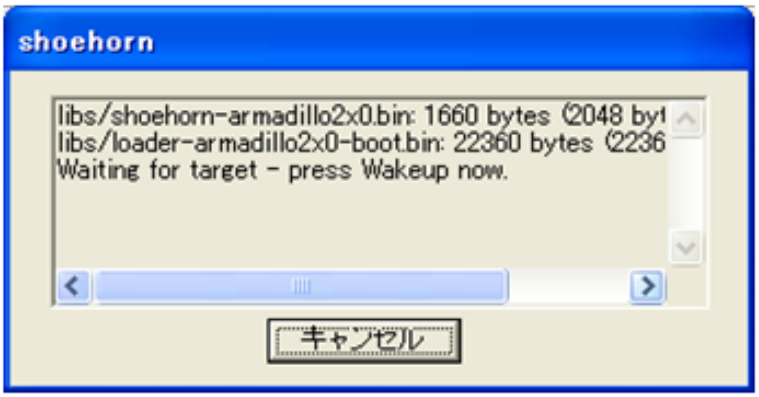

図 5.10. Hermit-At：shoehorn ダイアログ

ダイアログの表示後、Armadillo の電源を入れてください。

すぐにダイアログにメッセージが表示されます。正常に表示されない場合、Armadillo の電源を切断し、シリアルケーブルの接続や Armadillo のジャンパ設定を再度確認してください。

ダウンロードするための準備が完了するとダイアログは自動的にクローズされます。

この状態で、**ジャンパの設定変更や電源の切断をしないで**「5.3. Hermit-At ダウンローダを使用してフラッシュメモリを書き換える」を参照してブートローダの書き込みを行ってください。

6.Linux ブートオプション

Armadillo-200 シリーズでは、自動起動する Linux のブートオプションを設定することができます。設定はフラッシュメモリ上に保存され、次回の Linux 起動時から使用されます。Linux ブートオプションの設定は、Hermit コマンドプロンプトから行います。

設定する Linux ブートオプションを決定するためには、使用する Linux カーネルについての知識が必要です。オプションの内容と効果については、Linux カーネルについての文献や、ソースファイル付属ドキュメントを参照してください。

6.1. Hermit コマンドプロンプトの起動

1. シリアルコンソールソフトの起動

   Armadillo のシリアルインターフェース 1(と作業用 PC をシリアルケーブルで接続し、シリアルコンソールソフトを起動します。次のように通信設定を行ってください。

表 6.1. シリアル通信設定

項目	設定
転送レート	115,200bps
データ長	8bit
ストップビット	1bit
パリティ	なし
フロー制御	なし

2. ジャンパピンの設定

   Armadillo に電源を投入する前に、ジャンパピンを次のように設定します。
   • JP1: オープン
   • JP2: ショート
   詳しいジャンパピンの設定については、 「2.3. ジャンパピンの設定」 を参照してください。

3. Armadillo の起動

   Armadillo に電源を投入すると、Hermit コマンドプロンプトが表示されます。

```
Hermit-At v1.0.4 (armadillo2x0) compiled at 00:00:00, Jun 1 2006
hermit>
```

6.2. Linux ブートオプションの設定

Linux ブートオプションを設定するには、Hermit コマンドプロンプトから setenv コマンドを使用します。setenv に続けて、設定したい Linux ブートオプションを入力します。

```
hermit>setenv console=ttyAM0,115200
```

Linux ブートオプションが未設定(デフォルト)の場合、ブートローダーは Linux の起動時に自動的にオプション「console=ttyAM0,115200」を使用してシリアルインターフェース 1(ttyAM0)をコンソールにしますが、setenv により任意のブートオプションを設定した場合は、このオプションは自動使用されません。setenv した場合でもシリアルコンソールを使用する場合、オプションに「console=ttyAM0,115200」を含めてください。

設定したブートオプションを使用して Linux を起動するには、一旦 Armadillo の電源を切断し、適切なジャンパ設定を行ってから再度電源を入れ直してください。

6.3. 設定されている Linux ブートオプションの確認

現在設定されている Linux ブートオプションを表示して確認するには、setenv コマンドをパラメータなしで入力します。

```
hermit>setenv
1: console=ttyAM0,115200
```

6.4. Linux ブートオプションを初期化する

現在設定されている Linux ブートオプションをクリアし、デフォルトの状態に初期化するには、clearenv コマンドを入力します。

```
hermit>clearenv
```

ブートローダーを書き換えた場合、Linux ブートオプションの領域が壊れてしまい正常に起動しない場合があります。この場合、一度 clearenv を実行し、Linux ブートオプション領域を初期化する必要があります。

6.5. Linux ブートオプションの例

Linux ブートオプションの設定例を紹介します。

ex.1) シリアルコンソールを使用し、Linux 起動ログをシリアルインターフェース 1(ttyAM0)に表示させる場合

```
hermit>setenv console=ttyAM0,115200
```

ex.2) Linux 起動ログを表示させない場合

```
hermit>setenv console=null
```

7.ビルド

この章では、ソースコードからデフォルトイメージを作成する手順を説明します。以下の例では、作業ディレクトリとしてホームディレクトリ（~/）を使用していきます。

開発作業では、基本ライブラリ・アプリケーションやシステム設定ファイルの作成・配置を行います。各ファイルは作業ディレクトリ配下で作成・配置作業を行いますが、作業ミスにより誤って作業用 PC 自体の OS を破壊しないために、すべての作業は root ユーザではなく**一般ユーザ**で行ってください。

7.1. カーネルイメージとユーザーランドイメージのビルド

ここでは、付属 CD に収録されているデフォルトイメージを作成してみます。開発環境を構築してない場合は、「3. 開発環境の準備」を参照して作業用 PC に開発環境を構築してください。

7.1.1. ソースコードの準備

付属 CD の source/dist にある atmark-dist.tar.gz と source/kernel にある linux.tar.gz を作業ディレクトリに展開します。展開後、atmark-dist にカーネルソースを登録します。「図 7.1. ソースコード準備」のように作業してください。

```
[PC ~]$ tar zxvf atmark-dist-[version].tar.gz
[PC ~]$ tar zxvf linux-[version].tar.gz
[PC ~]$ ls
atmark-dist-[version].tar.gz  atmark-dist-[version]
linux-[version].tar.gz  linux-[version]
[PC ~]$ ln -s ../linux-[version] atmark-dist-[version]/linux-2.6.x
```

図 7.1. ソースコード準備

7.1.2. コンフィグレーション

ターゲットボード用の dist をコンフィグレーションします。以下の例のようにコマンドを入力し、コンフィグレーションを開始します。

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

続いて、使用するボードのベンダー名を聞かれます。「AtmarkTechno」と入力してください。

```
[PC ~/atmark-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
```

次にプロダクト名を聞かれます。「表 7.1. プロダクト名一覧」から、使用する製品に対応するプロダクト名を入力してください。

表 7.1. プロダクト名一覧

製品	プロダクト名	備考
Armadillo-210	Armadillo-210.Base	
	Armadillo-210.Recover	出荷時イメージ
Armadillo-220	Armadillo-220.Base	
	Armadillo-220.Recover	出荷時イメージ
Armadillo-230	Armadillo-230.Base	
	Armadillo-230.Recover	出荷時イメージ
Armadillo-240	Armadillo-240.Base	
	Armadillo-240.Recover	出荷時イメージ
Armadillo-9	Armadillo-9	出荷時イメージ
	Armadillo-9.PCMCIA	
Armadillo-300	Armadillo-300	出荷時イメージ
Armadillo-500	Armadillo-500	出荷時イメージ
Armadillo-500 FX	Armadillo-500-FX.dev	出荷時イメージ

以下は、Armadillo-210.Base の例です。

```
*
* Select the Product you wish to target
*
AtmarkTechno Products (Armadillo-210.Base, Armadillo-210.Recover,
Armadillo-220.Base, Armadillo-220.Recover, Armadillo-230.Base,
Armadillo-230.Recover, Armadillo-240.Base, Armadillo-240.Recover, Armadillo-300,
Armadillo-500, Armadillo-500-FX.dev, Armadillo-9, Armadillo-9.PCMCIA, SUZAKU-
V.SZ310, SUZAKU-V.SZ310-SIL, SUZAKU-V.SZ410, SUZAKU-V.SZ410-SIL)
[Armadillo-210.Base] (NEW) Armadillo-210.Base
```

ビルドする開発環境を聞かれます。「default」と入力してください。

```
*
* Kernel/Library/Defaults Selection
*
```

```
*
* Kernel is linux-2.6.x
*
Cross-dev (default, arm-vfp, arm, armnommu, common, h8300, host, i386, i960,
m68knommu, microblaze, mips, powerpc, sh) [default] (NEW) default
```

使用する C ライブラリを指定します。「None」を選択してください。

```
Libc Version (None, glibc, uC-libc, uClibc) [uClibc] (NEW) None
```

デフォルトの設定にするかどうか質問されます。「y」(Yes)を選択してください。

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

最後の 3 つの質問は「n」(No)と答えてください。

```
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
```

質問事項が終わるとビルドシステムの設定を行います。すべての設定が終わるとプロンプトに戻ります。

7.1.3. ビルド

ビルドするには、atmark-dist ディレクトリで「図 7.2. ビルド」のようにコマンドを実行します。ビルドが完了すると、`atmark-dist/images` ディレクトリに `linux.bin.gz` と `romfs.img.gz` が作成されます。

```
[PC ~/atmark-dist]$ make


[PC ~/atmark-dist]$ ls images
linux.bin  linux.bin.gz  romfs.img  romfs.img.gz
```

図 7.2. ビルド

7.2. ユーザーランドイメージをカスタマイズする

自作のアプリケーションを`/bin` に追加したユーザーランドイメージの作成方法について説明します。ここでは、「7.1. カーネルイメージとユーザーランドイメージのビルド」が完了している前提で説明します。

自作アプリケーションは、`~/sample/hello` にある仮定とします。

```
[PC ~/atmark-dist]$ cp ~/sample/hello romfs/bin/
[PC ~/atmark-dist]$ make image


[PC ~/atmark-dist]$ ls images
linux.bin  linux.bin.gz  romfs.img  romfs.img.gz
```

図 7.3. ユーザーランドイメージのカスタマイズ

できた romfs.img 及び romfs.img.gz の/bin には、hello がインストールされています。

7.3. ブートローダーイメージのビルド

7.3.1. ソースコードの準備

付属 CD の source/bootloader にある hermit-at-*[version]*-source.tar.gz を作業ディレクトリに展開します。「図 7.4. ソースコード展開例」のように作業してください。

```
[PC ~]$ tar zxvf hermit-at-[version]-source.tar.gz
```

図 7.4. ソースコード展開例

7.3.2. ビルド

ビルドオプションに TARGET と PROFILE を指定します。製品毎にパラメータが異なりますので、「表 7.2. ビルドオプション一覧」を参照してください。

また、生成されるイメージファイル名は loader-*[TARGET]*-*[PROFILE]*.bin(PROFILE が未指定の場合は loader-*[TARGET]*.bin)になります。

表 7.2. ビルドオプション一覧

製品	TARGET	PROFILE	説明
Armadillo-210 Armadillo-220 Armadillo-230 Armadillo-240	armadillo2x0	指定なし	hermit コンソールにシリアルインターフェース 1 を使用。
		eth	出荷時イメージ。 hermit コンソールにシリアルインターフェース 1 を使用。 tftp によるフラッシュメモリ書き換えが可能。
		ttyAM1	hermit コンソールにシリアルインターフェース 2 を使用。
		notty	hermit コンソールにシリアルインターフェースを使用しない。
		boot	Shoehorn-At で使用。
		boot-eth	Shoehorn-At で使用。 LAN 経由でのフラッシュメモリ書き換えが可能。

製品	TARGET	PROFILE	説明
Armadillo-9	armadillo9	指定なし	出荷時イメージ。 hermit コンソールにシリアルインターフェース 1 を使用。
		eth	hermit コンソールにシリアルインターフェース 1 を使用。 tftp によるフラッシュメモリ書き換えが可能。
		ttyAM1	hermit コンソールにシリアルインターフェース 2 を使用。
		notty	hermit コンソールにシリアルインターフェースを使用しない。
		boot	Shoehorn-At で使用。
		boot-eth	Shoehorn-At で使用。 LAN 経由でのフラッシュメモリ書き換えが可能。
Armadillo-300	armadillo3x0	指定なし	hermit コンソールにシリアルインターフェース 2 を使用。
		eth	出荷時イメージ。 hermit コンソールにシリアルインターフェース 2 を使用。 tftp によるフラッシュメモリ書き換えが可能。
		ttyAM1	hermit コンソールにシリアルインターフェース 1 を使用。
		notty	hermit コンソールにシリアルインターフェースを使用しない。
		boot	Shoehorn-At で使用。
		boot-eth	Shoehorn-At で使用。 LAN 経由でのフラッシュメモリ書き換えが可能。
Armadillo-500 Armadillo-500 FX	armadillo5x0	指定なし	Armadillo-500 開発ボード用のイメージ。
		fx	Armadillo-500 FX 液晶モデル用のイメージ。
		boot	Shoehorn-At で使用。
		zero	Armadillo-500 CPU モジュール単体用のイメージ。

例えば、Armadillo-210(PROFILE=指定なし)の場合「図 7.5. ビルド例 1」のように実行します。

```
[PC ~]$ cd hermit-at-[version]
[PC ~/hermit-at]$ make TARGET=armadillo2x0


[PC ~/hermit-at]$ ls src/target/armadillo2x0/*.bin
loader-armadillo2x0.bin
```

図 7.5. ビルド例 1

同様に、Armadillo-500 FX の場合「図 7.6. ビルド例 2」のように実行します。

```
[PC ~]$ cd hermit-at-[version]
[PC ~/hermit-at]$ make TARGET=armadillo5x0 PROFILE=fx


[PC ~/hermit-at]$ ls src/target/armadillo5x0/*.bin
loader-armadillo5x0-fx.bin
```

図 7.6. ビルド例 2

8.メモリマップについて

表 8.1. メモリマップ(フラッシュメモリ)

アドレス	リージョン	サイズ	説明
0x60000000 0x6000ffff	bootloader	64KB	Hermit ブートローダー loader-armadillo-2x0.bin」のイメージ
0x60010000 0x6017ffff	kernel [1]	約 1.44MB	Linux カーネル 「linux.bin(.gz)」のイメージ (非圧縮イメージ、gz 圧縮イメージに対応)
0x60180000 0x607effff	userland [1]	約 6.44MB	ユーザーランド 「romfs.img(.gz)」のイメージ (非圧縮イメージ、gz 圧縮イメージに対応)
0x607f0000 0x607fffff	config	64KB	コンフィグ領域

[1]kernel とユーザーランドのみ、linux の起動前に RAM へ展開・コピーされる

表 8.2. メモリマップ(RAM)

アドレス	内容	ファイルシステム	説明
0xc0018000	kernel	-	linux 起動前に フラッシュメモリから展開・コピー
0xc4800000	userland	EXT2	linux 起動前に フラッシュメモリから展開・コピー

9.デバイスドライバ仕様

9.1. GPIO ポート

GPIO ポートに対応するデバイスノードのパラメータは、以下の通りです。

表 9.1. GPIO ノード

タイプ	メジャー番号	マイナー番号	ノード名(/dev/xxx)
キャラクタデバイス	10	185	gpio

ioctl を使用してアクセスすることにより、Armadillo の GPIO を直接操作することができます。

第 1 引数には、デバイスファイルのファイルディスクリプタを指定します。第 2 引数には、GPIO を操作するためのコマンドを指定します。

表 9.2. GPIO 操作コマンド

コマンド	説明	第 3 引数の Type
PARAM_SET	第 3 引数で指定する内容で GPIO の状態を設定します	struct gpio_param
PARAM_GET	第 3 引数で指定する内容で GPIO の状態を取得します	struct gpio_param
INTERRUPT_WAIT	第 3 引数で指定する内容で GPIO の割込みが発生するまで WAIT します	struct wait_param

第 3 引数には、(カーネルソース)/include/asm-arm/arch-ep93xx/armadillo2x0_gpio.h に定義されている構造体「struct gpio_param」と「struct wait_param」を使用します。「struct gpio_param」は単方向リストになっているので、複数の GPIO を一度に制御する場合は next メンバを使用してください。また、リストの最後の next メンバには"0(NULL)"を指定してください。GPIO デバイスドライバの詳細な使用方法については、サンプルの GPIO 制御アプリケーション(atmark-dist/vendors/AtmarkTechno/Armadillo-2x0.Common/gpiod)のソースコードを参考にしてください。

9.2. LED

LED に対応するデバイスノードのパラメータは、以下の通りです。

表 9.3. LED ノード

タイプ	メジャー番号	マイナー番号	ノード名(/dev/xxx)
キャラクタデバイス	10	215	led

ioctl を使用してアクセスすることにより、Armadillo-200 シリーズの LED を直接操作することができます。

第 1 引数には、デバイスファイルのファイルディスクリプタを指定します。第 2 引数には、LED を操作するためのコマンドを指定します。

表 9.4. LED 操作コマンド

コマンド	説明	第 3 引数の Type
LED_RED_ON	LED(赤)を点灯します	なし
LED_RED_OFF	LED(赤)を消灯します	なし
LED_RED_STATUS	LED(赤)の点灯状態を取得します	状態を保存するバッファ(最小 1Byte)
LED_RED_BLINKON	LED(赤)を点滅を開始します	なし
LED_RED_BLINKOFF	LED(赤)を点滅を停止します	なし
LED_RED_BLINKSTATUS	LED(赤)の点滅状態を取得します	状態を保存するバッファ(最小 1Byte)
LED_GREEN_ON	LED(緑)を点灯します	なし
LED_GREEN_OFF	LED(緑)を消灯します	なし
LED_GREEN_STATUS	LED(緑)の点灯状態を取得します	状態を保存するバッファ(最小 1Byte)
LED_GREEN_BLINKON	LED(緑)を点滅を開始します	なし
LED_GREEN_BLINKOFF	LED(緑)を点滅を停止します	なし
LED_GREEN_BLINKSTATUS	LED(緑)の点滅状態を取得します	状態を保存するバッファ(最小 1Byte)

LED デバイスドライバの詳細な使用方法については、サンプルの LED 制御アプリケーション(atmark-dist/vendors/AtmarkTechno/Armadillo-2x0.Common/ledctrl)のソースコードを参考にしてください。

9.3. オンボードフラッシュメモリ/NAND フラッシュメモリ(オプション)

オンボードフラッシュメモリは、Memory Technology Device(MTD)としてリージョン単位で扱われます。オンボードフラッシュメモリのリージョンについては、「8. メモリマップについて」を参照してください。また、オプション品の NAND フラッシュメモリ(受注生産品)についても、オンボードフラッシュメモリに続く形でリージョンで扱われます。各リージョンに対応するデバイスノードのパラメータは、以下の通りです。

表 9.5. MTD ノード

タイプ	メジャー番号	マイナー番号	ノード名 (/dev/xxx)	デバイス名
キャラクタデバイス	90	0	mtd0	bootloader
		1	mtdr0	bootloader(read only)
		2	mtd1	kernel
		3	mtdr1	kernel(read only)
		4	mtd2	userland
		5	mtdr2	userland(read only)
		6	mtd3	config
		7	mtdr3	config(read only)
		8	mtd4	NAND Flash(接続時のみ)
		9	mtdr4	NAND Flash(接続時のみ/read only)
ブロックデバイス	31	0	mtdblock0	bootloader
		1	mtdblock1	kernel
		2	mtdblock2	userland
		3	mtdblock3	config
		4	mtdblock4	NAND Flash(接続時のみ)

9.4. USB ホスト

EP9307 は、OHCI 互換の USB ホスト機能を持っています。いくつかのデバイスについては初期状態のカーネルでドライバを有効化しており、接続するだけで使用できるようになっています。

9.4.1. USB Storage

USB メモリやディスクドライブ、メモリカードリーダなどをサポートします。Linux からは一般的な SCSI 機器と同様に認識され、/dev/sda(ブロックデバイス、メジャー番号:8、マイナー番号:0)や/dev/sda1(ブロックデバイス、メジャー番号:8、マイナー番号:1)などから扱うことができます。

9.4.2. USB Human Interface Device(HID)

USB キーボードやマウスなど、各種入力機器をサポートします。

9.5. VGA(Armadillo-240 のみ)

VGA 出力はフレームバッファドライバが用意されており、コンソール画面として使用することができます。初期状態では SVGA サイズ(解像度:800x600)の 24 ビットカラー設定となっていますが、VGA サイズ(640x480)及び XGA サイズ(1024x768)や 8/16 ビットカラーにも対応しています。ここでは、この設定の変更方法について説明します。

現在のソフトウェアでは、デバイスが提供する設定の全てに対応していません。また、Armadillo-240 の VGA 出力は、VESA などの規格化されているタイミングを完全に満しているわけではありません。そのため、許容範囲の狭いモニタでは同期ずれが起こる場合があります。

9.5.1. デフォルト設定の変更

デフォルト設定の変更には、カーネルのリコンパイルが必要となります。

まず、コンフィギュレーションします。

```
[PC ~/atmark-dist]$ make menuconfig
```

メニューが表示されるので、

```
Kernel/Library/Defaults Selection --->
 --- Kernel is linux-2.4.x
 (None)Libc Version
 [ ] Default all settings
 [*] Customize Kernel Settings                    ここを選択する
 [ ] Customize Vendor/User Settings
 [ ] Update Default Vendor Settings
```

とします。続いて Kernel Configuration のメニューが表示されるので、

```
Device drivers --->
  Graphics support --->
[*]  EP93xx frame buffer support
     EP93xx frame buffer display(CRT display) --->
     EP93xx frame buffer resolution(SVGA(60Hz)) ---> デフォルトの解像度
     EP93xx frame buffer depth(24bpp true color) ---> カラー設定
```

上記の項目を変更した後、コンフィギュレーションを終了させます。

続いて、ビルドします。

```
[PC ~/atmark-dist]$ make all
```

ビルドしてできたカーネルイメージ(linux.bin.gz)を Armadillo-240 へ書き込み、VGA のデフォルトの設定は完了です。

9.5.2. 解像度・色深度の変更

デフォルトの解像度・色深度以外で VGA を動作させるときは、Linux ブートオプションに設定を追加するだけで変更ができます。

「6. Linux ブートオプション」を参考に hermit を起動させます。ブートオプションに"**video=ep93xxfb:???**"を追加します。"**???**"には、「表 9.6. 解像度一覧」「表 9.7. 色深度一覧」からモード名を挿入してください。

表 9.6. 解像度一覧

モード名	解像度
CRT-640x480	640x480 60Hz
CRT-640x480@75	640x480 75Hz
CRT-800x600	800x600 60Hz
CRT-800x600@75	800x600 75Hz
CRT-1024x768	1024x768 60Hz
CRT-1024x768@75	1024x768 75Hz

表 9.7. 色深度一覧

モード名	解像度
8bpp	8 ビットカラー
16bpp	16 ビットカラー
24bpp	24 ビットカラー
32bpp	32 ビットカラー

以下は 800x600 60Hz, 8 ビットカラーの設定例です。

```
hermit> setenv video=ep93xxfb:CRT-800x600,8bpp
```

10.RTC/NAND フラッシュメモリモジュール

10.1. フラッシュメモリ用デバイスドライバの組み込み

NAND フラッシュメモリ用のデバイスドライバは、誤認識による起動不具合[1]を防ぐため、デフォルトの状態では組込まれていません。デバイスドライバを組み込むには、カーネルのコンフグレーションで以下のオプションを選択してください。

```
Device Drivers  --->
    Memory Technology Devices (MTD)  --->
        NAND Flash Device Drivers  --->
            <*> NAND Device Support
            <*> Support for Armadillo-220/230/240
```

10.2. フラッシュメモリの認識

認識された場合は、Linux ブート中に以下のメッセージが表示されます。

```
NAND device: Manufacturer ID: 0x??, Chip ID: 0x?? (??) [2]
```

NAND フラッシュメモリのフォーマット方法は以下のとおりです。

```
[root@armadillo (ttyAM0) ~]# flash_eraseall -j /dev/mtd4
```

NAND フラッシュメモリのマウント方法は以下のとおりです。以下の例では、/mnt へ NAND フラッシュメモリをマウントしています。

```
[root@armadillo (ttyAM0) ~]# mount -t jffs2 /dev/mtdblock4 /mnt
```

10.3. リアルタイムクロック用デバイスドライバの組み込み

RTC/NAND フラッシュメモリモジュール用のデバイスドライバは、誤認識による起動不具合 [1] を防ぐため、デフォルトの状態では組込まれていません。デバイスドライバを組み込むには、カーネルのコンフィグレーションで以下のオプションを選択してください。

[1]http://armadillo.atmark-techno.com/faq/nand-kernel-panic
[2]Manufacturer ID」および「Chip ID」は、製造ロットやフラッシュメモリの容量により変わります。

```
Device Drivers  --->
    I2C support  --->
        I2C Hardware Bus support  --->
            [*] External I2C interface for Armadillo-220/230/240
        Other I2C Chip support  --->
            <*> Armadillo-9 Real Time Clock
```

hwclock コマンドを使用するために、ユーザーランドのコンフィグレーションで以下のオプションを選択してください。

```
BusyBox  --->
    [*] hwclock
```

10.4. リアルタイムクロックの認識

リアルタイムクロックへの書き込みは以下のコマンドで行います。例では、2009 年 1 月 23 日 12 時 34 分に時間を設定してリアルタムクロックへ書き込み、時間を表示します。

```
[root@armadillo (ttyAM0) ~]# date -s 012312342009……時間の設定
[root@armadillo (ttyAM0) ~]# hwclock --systohc --utc……リアルタイムクロックへ書き込み
[root@armadillo (ttyAM0) ~]# hwclock --utc……時間の表示
```

改訂履歴

バージョン	年月日	改訂内容
2.0.0	2006/8/17	・Armadillo-220 ソフトウェアマニュアル v1.01 と Armadillo-240 ソフトウェアマニュアル v1.01 をベースに統一し、Armadillo-230 の記述を加え新規作成
2.0.1	2006/9/5	・「4.8. telnet ログイン」 を追加 ・「9.2. LED」 仕様に点滅状態制御についての記述を追加
2.0.2	2006/10/20	・「2.2. 保証に関する注意事項」を追加 ・「ユーザランド」を「ユーザーランド」に統一
2.0.3	2007/7/20	・初期不良の保証期間に関する記述修正 ・「Flash メモリ」を「フラッシュメモリ」に統一 ・「表 1.1. 製品の呼び名」 の修正 ・「表 3.1. クロス開発環境パッケージ一覧」へパッケージ追加 ・「3.1. クロス開発環境パッケージのインストール」 へ rpm パッケージを使用した場合の注意点追記 ・「3.1. クロス開発環境パッケージのインストール」 にパッケージの一括インストール方法を追加 ・「表 3.2. atmark-dist のビルドに必要なパッケージ一覧」 へパッケージ追加 ・「4.8. telnet ログイン」 に Recover イメージでの telnet ログイン方法に関する記述追加 ・「4.9. ファイル転送」 に Recover イメージでのファイル転送方法に関する記述追加 ・「7.1.1. ソースコードの準備」 のカーネルディレクトリへのシンボリックリンク作成に注意書きを追加
2.0.4	2007/8/2	・「表 3.1. クロス開発環境パッケージ一覧」から libjpeg62、libjpeg62-dev を削除
2.0.5	2007/9/14	・「表 1.4. コマンド入力例での省略表記」 を追加 ・「1.5. 保証に関する注意事項」の製品の保証方法を修正 ・コマンド入力例で、バージョン番号などの省略の表記方法を修正 ・「5.2. フラッシュメモリの書き込み領域について」 に Recover イメージと Base イメージの違いを追加。 ・「表 3.2. atmark-dist のビルドに必要なパッケージ一覧」 に libncurses5-dev を追加
2.0.6	2007/10/19	・開発環境のバージョンアップに伴う記述の変更 ・「10.1. Windows 上に開発環境を構築する方法」を削除
2.0.7	2007/12/14	・「7.1.2. コンフィグレーション」 について、atmark-dist-20071112 で変更された内容にあわせて修正
2.0.8	2008/3/14	・「9.5. VGA(Armadillo-240 のみ)」 に注記を追加
2.0.9	2008/9/26	・CD-ROM ディレクトリ構成変更に伴う修正 ・タイトルを英語表記からカタカナ表記に
2.0.10	2008/12/29	・「表 1.1. 製品の呼び名」「3.2. atmark-dist のビルドに必要なパッケージ」「図 4.15. ファイアウォールの設定コマンド入力例」「7.1.3. ビルド」「9.5.2. 解像度・色深度の変更」誤記修正 ・「10. RTC/NAND フラッシュメモリモジュール」追加
2.1.0	2009/03/19	・「1. はじめに」、「3. 開発環境の準備」、「4. 使用方法」、「5. フラッシュメモリの書き換え方法」、「6. Linux ブートオプション」、「7. ビルド」 構成変更 ・誤記、表記ゆれ修正

2.1.1	2009/07/17	・本文のレイアウト統一 ・「9. デバイスドライバ仕様」の表記を変更 ・表記ゆれ修正 ・「5. フラッシュメモリの書き換え方法」にコマンド例の説明を追記 ・「図 7.1. ソースコード準備」の誤記を修正
2.1.2	2009/07/29	・製品保証に関する記載を http://www.atmark-techno.com/support/warranty-policy に移動(2009/08/03 適用)

Armadillo-200 シリーズ 220/230/240 ソフトウェアマニュアル
Version 2.1.2-d308169
2009/08/03

株式会社アットマークテクノ
060-0035 札幌市中央区北 5 条東 2 丁目 AFT ビル 6F TEL 011-207-6550 FAX 011-207-6570