Armadillo-300
ソフトウェアマニュアル

WA3000

Version 1.1.2-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-300
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 1.1.2-d308169
2009/08/03

目次

図目次

表目次

1.はじめに

以降、本書では他の Armadillo シリーズにも共通する記述については、製品名を Armadillo と表記します。

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 1.1. 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 1.2. 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 1.3. 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-300 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	D-Sub9 ピン（メス - メス）の「クロス接続用」ケーブルです。
付属 CD-ROM（以降、「付属 CD」と略記）	Armadillo-300 に関する各種マニュアルやソースコードが収録されています。
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。）

2.2. 接続方法

「シリアルクロスケーブル」を使って Armadillo-300 の CON7 と作業用 PC を接続します。

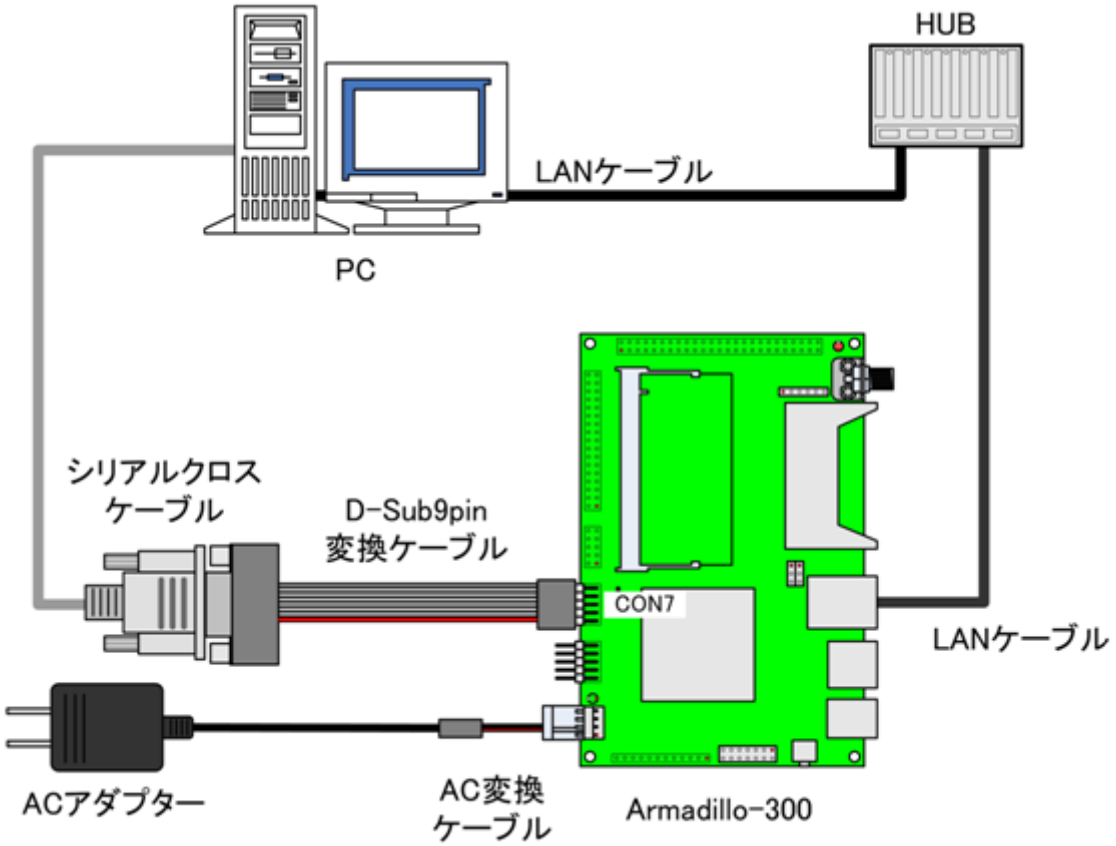

図 2.1. 接続図

2.3. ジャンパピンの設定について

Armadillo-300 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や JTAG 機能の切換えができます。以下の表にジャンパの設定とその機能を記載します。

表 2.1. ジャンパの設定

ジャンパ		機能
JP1	1-2	Linux カーネルを起動
	2-3	ブートローダを起動
JP2	1-2	JTAG 機能無効
	2-3	JTAG 機能有効

ジャンパは上記のいずれかに設定してください。設定されていない場合、ハードウェアの故障につながる恐れがありますのでご注意ください。

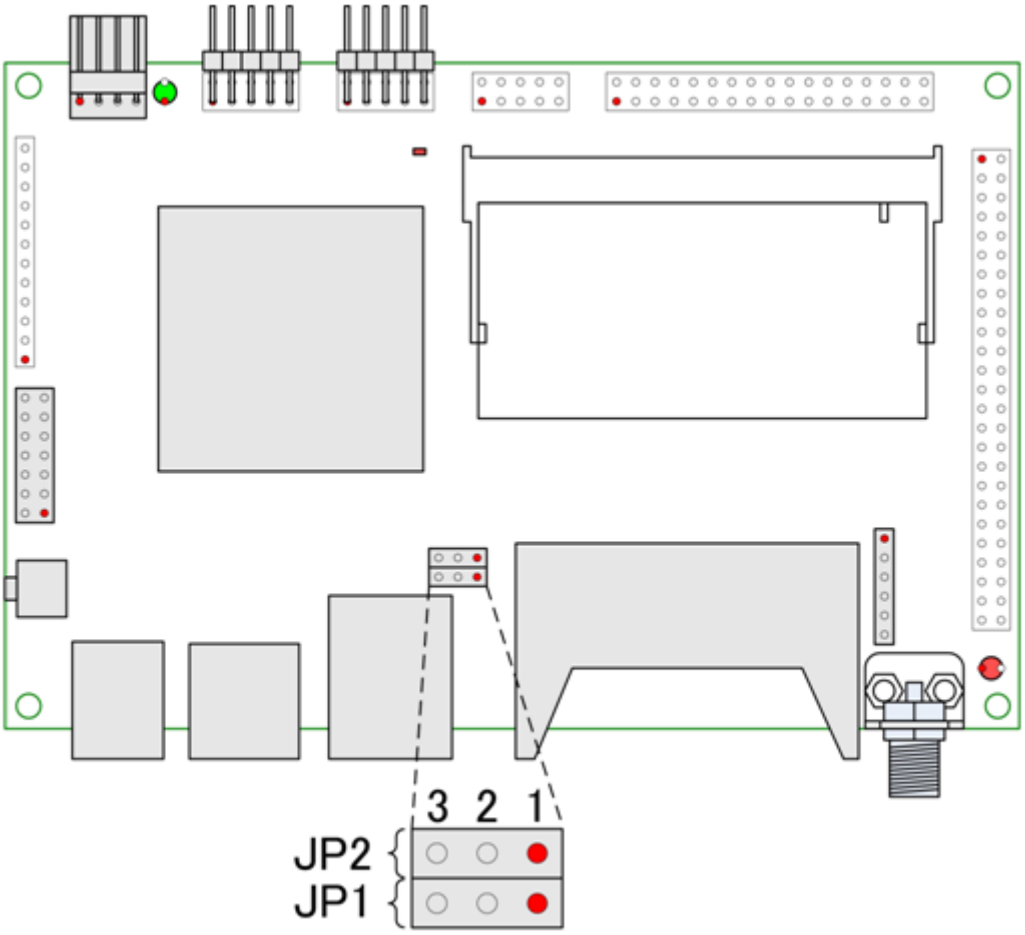

図 2.2. ジャンパピンの位置

3.ソフトウェアについて

この章では、Armadillo-300 で使用されているソフトウェアについて説明します。

3.1. ソフトウェアの種類

Armadillo-300 は 4 つの種類のソフトウェアで構成されています。

3.1.1. 1st ブートローダ「IPL」

1st ブートローダには、オリジナルの IPL(Initial Program Loader)が採用されています。IPL は Shoehorn-At Host と協調動作を行い、2nd ブートローダの復旧を行うことができます。

IPL は通常書き換えを行うことはできないようになっていますが、IPL が破壊された場合、JTAG 経由でフラッシュメモリを復旧しなければなりません。フラッシュメモリの書き換え時には細心の注意を払ってください。

JP1 を「2-3」に設定した場合、Armadillo-300 の起動時に CON7 から数バイトのデータが出力されます。これは、Shoehorn-At とネゴシエーションするために必要な機能です。 **システム設計をされる場合は、本現象を考慮した上で設計してください。**

3.1.2. 2nd ブートローダ「Hermit-At」

2nd ブートローダには、高機能ブートローダ/ダウンローダの Hermit-At が採用されています。Hermit-At は Hermit-At Host と協調動作を行い、2nd ブートローダ、Kernel 又は Userland の復旧を行うことができます。

Armadillo-300 の 2nd ブートローダには、「表 3.1. 2nd ブートローダイメージの種類」に示される種類のフラッシュメモリのイメージファイルが用意されています。

表 3.1. 2nd ブートローダイメージの種類

イメージファイル名	PROFILE 名	シリアルインターフェース	説明
loader-armadillo3x0.bin	(none)	CON7	付加機能のない、小さなイメージです。
loader-armadillo3x0-ttyAM1.bin	ttyAM1	CON6	ログが表示されるシリアルインターフェースが CON6 に変更されます。
loader-armadillo3x0-notty.bin	notty	-	ログを表示しません。
loader-armadillo3x0-eth.bin	eth	CON7	**出荷時のイメージです。** LAN を使用したイメージの書き換えが可能です。
loader-armadillo3x0-boot.bin	boot	CON7	Shoehorn-At で使用します。
loader-armadillo3x0-boot-eth.bin	boot-eth	CON7	Shoehorn-At で使用します。 LAN を使用したイメージの書き換えが可能です。

PROFILE 名は、ソースコードからイメージファイルをビルドするときに指定するオプションです。詳しくは、「6.3. ブートローダーイメージのビルド」を参照してください。

本書で単にブートローダと表記した場合、2nd ブートローダ(Hermit-At)を意味します。

3.1.3. Kernel

Kernel には、Linux-2.6.12.5-at1 が採用されています。これは、Linux-2.6.12.5 をベースにボード固有のオリジナルデバイスドライバを追加したものとなっています。オリジナルデバイスドライバに関しては「3.3. オリジナルデバイスドライバ」を参照してください。

3.1.4. Userland

Userland には、各種ユーティリティ、サーバアプリケーション又は、各種設定ファイル等を EXT2 ファイルシステムイメージにしたものを採用しています。イメージファイルの作成には Atmark-dist を使用しています。

3.2. メモリマップ

表 3.2. メモリマップ(フラッシュメモリ)

物理 アドレス	フラッシュメモリの内容	サイズ	説明
0x50000000 | 0x50001fff	ipl	8KB	1st ブートローダ領域 「ipl-a300.bin」のイメージ
0x50002000 | 0x5000ffff	bootloader	56KB	2nd ブートローダ領域 「loader-a3x0.bin」のイメージ
0x50010000 | 0x5020ffff	kernel	2MB	カーネル領域「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)
0x50210000 | 0x507effff	userland	5.875MB	ユーザランド領域「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)
0x507f0000 | 0x507fffff	config	64KB	コンフィグ領域

表 3.3. メモリマップ(RAM)

論理 アドレス	RAM の内容	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・コピーされます
0xc0800000	userland	ext2fs	Linux の起動前に フラッシュメモリから展開・コピーされます

3.3. オリジナルデバイスドライバ

ここでは Armadillo-300 オリジナルデバイスドライバの仕様を説明します。Armadillo-300 には次に示すオリジナルデバイスドライバがあります。

- GPIO
- LED

3.3.1. GPIO デバイスドライバ

Armadillo-300 の GPIO ポート（CON9）は、GPIO デバイスドライバで設定の変更、及び状態の取得を行うことができます。

GPIO ポートに対応するデバイスノードの情報は、以下の通りです。

表 3.4. GPIO ノード

タイプ	major	minor	node(/dev/xxx)
キャラクタデバイス	10	185	gpio

システムコール（ioctl）を使用してアクセスすることにより、Armadillo-300 の GPIO ポートを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 3.5. 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

第３引数には、(カーネルソース)`/include/asm-arm/arch-ns9750/armadillo3x0_gpio.h` に定義されている構造体「struct gpio_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。

```
struct gpio_param{
        struct gpio_param *next;
        unsigned long no;
        unsigned long mode;
        union{
                struct output_param o;
                struct input_param i;
        }data;
};

/* GPIO ポート 0 を High 出力にする場合 */
struct gpio_param param;
param.no = GPIO0;
param.mode = MODE_OUTPUT;
param.data.o.value = 1;
param.next = NULL;

ioctl(fd, PARAM_SET, &param);
```

図 3.1. ioctl の発行例（GPIO）

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-300/gpioctrl`)のソースコードを参考にしてください。

3.3.2. LED デバイスドライバ

LED デバイスドライバは、Armadillo-300 の LED（D2）を点灯・消灯したり、状態を取得したりすることができます。

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 3.6. LED ノード

タイプ	major	minor	node(/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-300 の LED を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 3.7. LED 操作コマンド

コマンド	説明	第３引数の Type
A3X0_LED_SET	第３引数で指定する構造体で LED を設定します	struct a3x0_led_param
A3X0_LED_GET	第３引数で指定する構造体に LED の状態を格納します	struct a3x0_led_param

第３引数には、(カーネルソース)/include/asm-arm/arch-ns9750/armadillo3x0_led.h に定義されている構造体「struct a3x0_led_param」を使用します。buf メンバは、"1:点灯"、"0:消灯"が設定/取得されます。

```
struct a3x0_led_param {
  unsigned long buf;
};

/* LED を点灯させる場合 */
struct a3x0_led_param param;
param.buf = LED_ON;

ioctl(fd, A3X0_LED_SET, &param);
```

図 3.2. ioctl の発行例（LED）

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-300/ledctrl)のソースコードを参考にしてください。

4.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。

4.1. 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 4.1. 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 4.1. 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 4.1. 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 4.1. 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。

4.2. 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 4.2. 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 4.2. 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 4.2. 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 4.2. インストール情報表示コマンド

4.3. 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 4.3. 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。

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について」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. 2nd ブートローダを出荷状態に戻す」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。

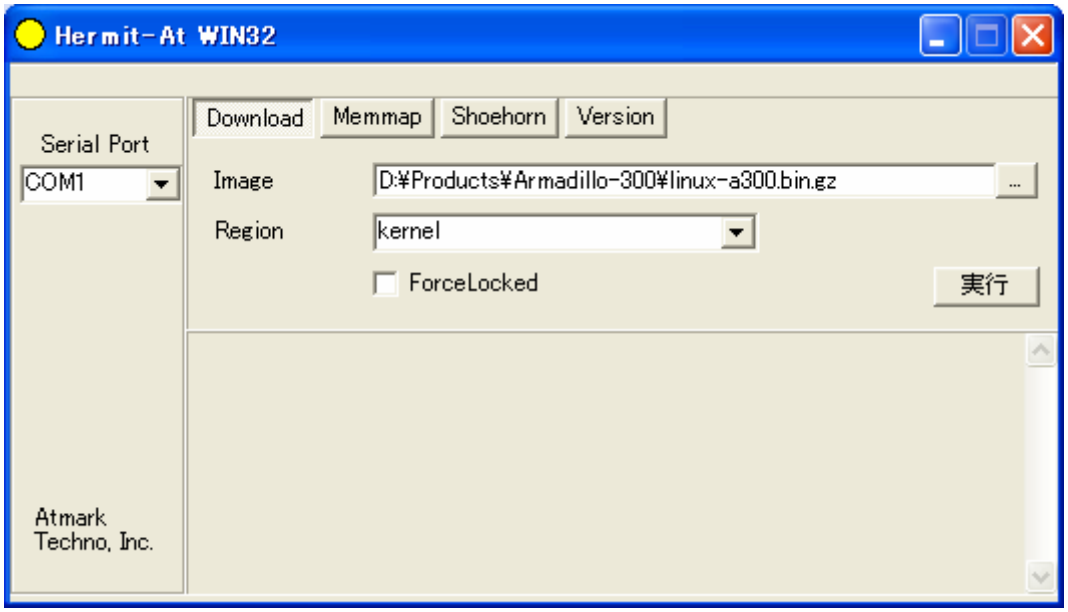

図 5.5. Hermit-At：Download ウィンドウ

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。

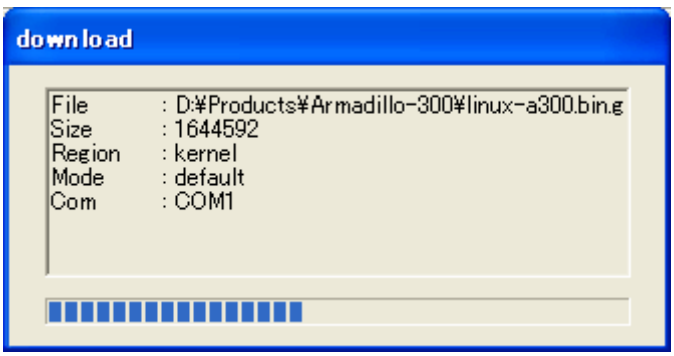

図 5.6. Hermit-At：download ダイアログ

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. 2nd ブートローダを出荷状態に戻す

2nd ブートローダ領域に hermit コマンドプロンプトが表示されないイメージや、不正なイメージを書き込んでしまい、2nd ブートローダを制御できなくなった場合の対処方法について説明します。

Armadillo-300 の 1st ブートローダは、Shoehorn-At Host と協調動作をすることで、Armadillo-300 の RAM 上に直接プログラムを書き込むことができます。この機能を利用して 2nd ブートローダを復旧させることが可能です。

1st ブートローダが破壊されている場合、本手段では復旧することはできません。破壊されている場合は、JTAG を使用し直接フラッシュメモリへ 1st/2nd ブートローダを書き込む必要があります。

5.5.1. 作業用 PC が Linux の場合

1. Armadillo-300 の電源が切断されていることを確認し、作業用 PC と Armadillo-300 をシリアルケーブルで接続します。
2. Armadillo-300 の JP1 を「2-3」に設定します。
3. 作業用 PC で shoehorn コマンドを以下の例[2] のように実行します。

[2] 紙面の都合上、折り返して表現しています。

```
[PC ~]$ shoehorn --boot --terminal --initrd /dev/null
 --kernel /usr/lib/hermit/loader-armadillo3x0-boot.bin
 --loader /usr/lib/shoehorn/shoehorn-armadillo3x0.bin
 --initfile /usr/lib/shoehorn/shoehorn-armadillo3x0.init
 --postfile /usr/lib/shoehorn/shoehorn-armadillo3x0.post

※ シリアルインターフェースが「ttyS0」以外の場合は、オプション「--port "ポート名"」を指定してください
```

図 5.8. shoehorn コマンド例

4. Armadillo-300 の電源を投入します。

電源を投入した場合、起動ログの表示が開始されます。表示が開始されない場合は、Armadillo-300 の電源を切断し、シリアルケーブルの接続やジャンパ設定を確認してください。

5. "hermit>"と表示されたら、「Ctrl + C」キーを入力してください。

以上で作業用 PC から hermit を使用して Armadillo-300 へ 2nd ブートローダをダウンロードする準備が整います。**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、「6.3.2. ビルド」を参照して書き換えを行ってください。

5.5.2. 作業用 PC が Windows の場合

1. Armadillo-300 の電源が切断されていることを確認し、作業用 PC と Armadillo-300 をシリアルケーブルで接続します。

2. Armadillo-300 の JP1 を「2-3」に設定します。

3. 「5.1. ダウンローダのインストール」で展開した `hermit.exe` を実行します。

4. 「Shoehorn」ボタンをクリックすると「図 5.9. Shoehorn モード時の画面」が表示されます。

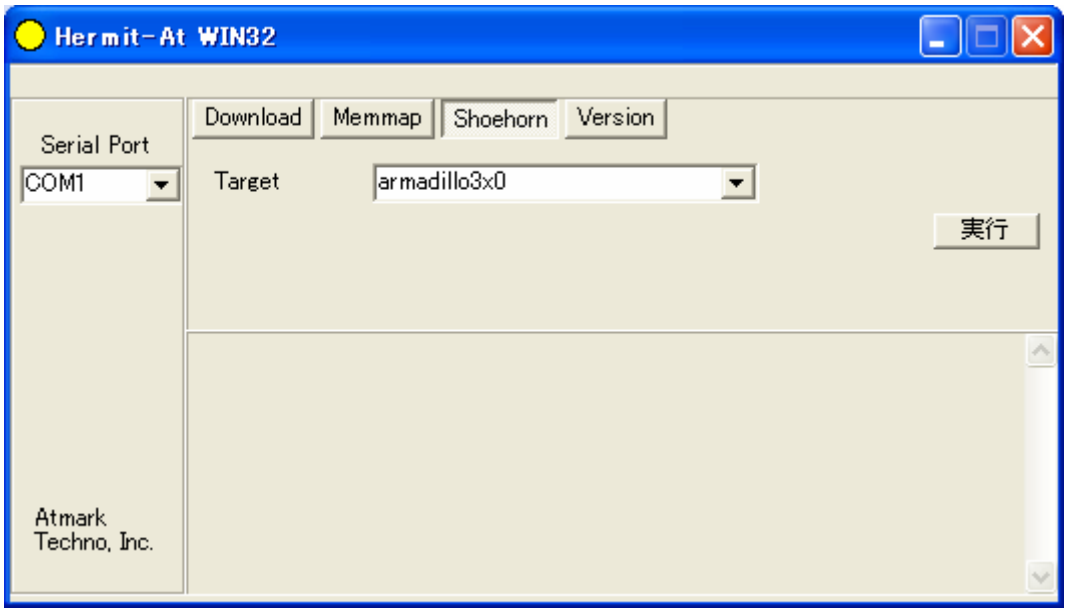

図 5.9. Shoehorn モード時の画面

5. "Target"に armadillo3x0 を指定します。

6. 「実行」ボタンをクリックすると「図 5.10. shoehorn ダイアログ」が表示されます。

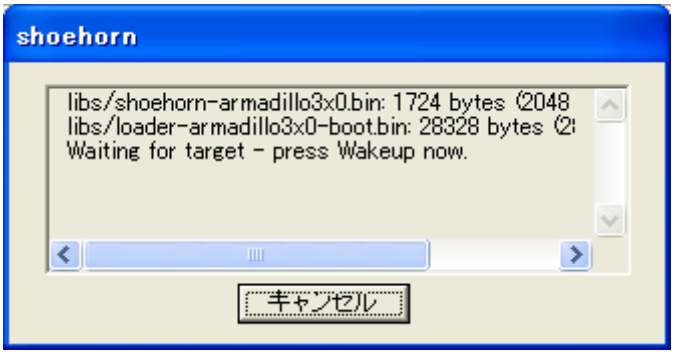

図 5.10. shoehorn ダイアログ

7. Armadillo-300 の電源を投入します。

電源を投入した場合、起動ログの表示が開始されます。表示が開始されない場合は、Armadillo-300 の電源を切断し、シリアルケーブルの接続やジャンパ設定を確認してください。

8. shoehorn ダイアログがクローズするのを待ちます。

以上で作業用 PC から hermit を使用して Armadillo-300 へ 2nd ブートローダをダウンロードする準備が整います。**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」を参照して書き換えを行ってください。

6.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。

6.1. 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、「4. 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。

6.1.1. 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 6.1. 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 6.1. ソースコード準備

6.1.2. 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 6.1. 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 6.1. 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。

6.1.3. ビルド

ビルドするには、atmark-dist ディレクトリで「図 6.2. 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 6.2. ビルド

6.2. 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、「6.1. 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 6.3. 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。

6.3. ブートローダーイメージのビルド

6.3.1. 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 6.4. 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 6.4. ソースコード展開例

6.3.2. 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 6.2. 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 6.2. 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 6.5. 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 6.5. ビルド例 1

同様に、Armadillo-500 FX の場合「図 6.6. 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 6.6. ビルド例 2

7.コンパクトフラッシュシステム構築

7.1. コンパクトフラッシュシステム例

Armadillo では、コンパクトフラッシュに Linux システムを構築することができます。この章では、起動可能なコンパクトフラッシュシステムの構築手順について説明します。

ブートローダがカーネルイメージを読み込むことができるファイルシステムは、EXT2 ファイルシステムとなっています。

この章では、「表 7.1. コンパクトフラッシュシステム例」のようなコンパクトフラッシュシステムを例に、構築手順を説明します。

表 7.1. コンパクトフラッシュシステム例

パーティション[1]	タイプ	容量	説明
/dev/hda1	ext2	32MB	起動パーティション。 カーネルイメージを配置する領域です。
/dev/hda2	ext3	-	ルートファイルシステムを配置する領域です。

[1]Armadillo-9 の場合、パーティション名は/dev/hdc1,/dev/hdc2 となります。以降、適宜読み替えてください。

7.2. コンパクトフラッシュの初期化

ここでは、コンパクトフラッシュをフォーマットし、パーティション 1 に EXT2 ファイルシステムを、パーティション 2 に EXT3 ファイルシステムを作成するところまでの手順を説明します。

作業の前に、ジャンパピンを以下のように設定してください。

表 7.2. コンパクトフラッシュ初期化時のジャンパピン設定

製品	ジャンパピン設定
Armadillo-9	JP1:オープン JP2:オープン
Armadillo-300	JP1:1-2

7.2.1. ディスクフォーマット

「図 7.1. ディスク初期化方法」のように、ディスクをフォーマットします。

```
[armadillo ~]# fdisk /dev/hda
The number of cylinders for this disk is set to 1324.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): d
No partition is defined yet!

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1324, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-1324, default 1324): +32M

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (85-1324, default 85):
Using default value 85
Last cylinder or +size or +sizeM or +sizeK (85-1324, default 1324):
Using default value 1324

Command (m for help): p

Disk /dev/hda: 512 MB, 512483328 bytes
12 heads, 63 sectors/track, 1324 cylinders
Units = cylinders of 756 * 512 = 387072 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/hda1               1          84       31720+  83  Linux
/dev/hda2              85        1324      468720   83  Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
 hda: hda1 hda2
 hda: hda1 hda2
Syncing disks.
```

図 7.1. ディスク初期化方法

7.2.2. ファイルシステムの作成

「図 7.2. ファイルシステムの構築」のように初期化したディスクのパーティションにファイルシステムを作成します。

mke2fs で起動パーティション（カーネルイメージを配置するパーティション）に EXT2 ファイルシステムを作成する場合は、必ず「**-O none**」オプションを指定する必要があります。

```
[armadillo ~]# mke2fs -O none /dev/hda1
mke2fs 1.25 (20-Sep-2001)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
7936 inodes, 31720 blocks
1586 blocks (5%) reserved for the super user
First data block=1
4 block groups
8192 blocks per group, 8192 fragments per group
1984 inodes per group
Superblock backups stored on blocks:
        8193, 16385, 24577

Writing inode tables: done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 36 mounts or
180.00 days, whichever comes first.  Use tune2fs -c or -i to override.
[armadillo ~]# mke2fs -j /dev/hda2
mke2fs 1.25 (20-Sep-2001)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
117392 inodes, 468720 blocks
23436 blocks (5%) reserved for the super user
First data block=1
58 block groups
8192 blocks per group, 8192 fragments per group
2024 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729, 204801, 221185, 401409

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 24 mounts or
180.00 days, whichever comes first.  Use tune2fs -c or -i to override.
```

図 7.2. ファイルシステムの構築

7.3. カーネルイメージを配置する

コンパクトフラッシュシステムから起動する場合は、起動パーティションの/boot ディレクトリにカーネルイメージを配置する必要があります。対応しているカーネルイメージは、非圧縮カーネルイメージ（Image linux.bin）または、圧縮イメージ（Image.gz linux.bin.gz）のどちらかになります。

ここで説明する例では、カーネルイメージの取得に wget コマンドを使用します。wget コマンドで指定する URL は製品によって異なりますので、以下の表を参照し適宜読み替えてください。

表 7.3. カーネルイメージのダウンロード先 URL

製品	URL
Armadillo-9	http://download.atmark-techno.com/armadillo-9/image/linux-*[version]*.bin.gz
Armadillo-300	http://download.atmark-techno.com/armadillo-300/image/linux-a300-*[version]*.bin.gz
Armadillo-500	http://download.atmark-techno.com/armadillo-500/image/linux-a500-*[version]*.bin.gz

以下に Armadillo-500 での配置例を示します。

```
[armadillo ~]# mount /dev/hda1 /mnt
[armadillo ~]# mkdir /mnt/boot
[armadillo ~]# cd /mnt/boot
[armadillo ~]# wget http://download.atmark-techno.com/armadillo-500/image/linux-
a500-[version].bin.gz
Connecting to download.atmark-techno.com [210.191.215.172]:80
linux-a500-[version].bin.gz 100% |**********************|  **** KB    00:00 ETA
[armadillo ~]# mv linux-a500-[version].bin.gz /mnt/boot/Image.gz
[armadillo ~]# sync
[armadillo ~]# umount /mnt
```

図 7.3. カーネルイメージの配置

7.4. ルートファイルシステムの構築

ここでは、コンパクトフラッシュにルートファイルシステムを構築する手順について説明します。

7.4.1. Debian GNU/Linux を構築する

Debian を構築する場合、付属 CD の debian ディレクトリ以下のアーカイブを使用するか、弊社ダウンロードサイトからアーカイブを取得します。これは、純粋な Debian でインストールされるファイルを分割してアーカイブ化したものとなります。これらをファイルシステム上に展開することでルートファイルシステムを構築することができます。

ルートファイルシステムに Debian を構築する場合は、パーティションの空き容量が最低でも 256MB 必要です。

ここで説明する例では、debian アーカイブの取得に wget コマンドを使用します。wget コマンドで指定する URL は製品によって異なりますので、以下の表を参照し適宜読み替えてください。

表 7.4. debian アーカイブのダウンロード先 URL

製品	URL
Armadillo-9	http://download.atmark-techno.com/armadillo-9/debian/debian-etch-a9-#.tgz
Armadillo-300	http://download.atmark-techno.com/armadillo-300/debian/debian-etch-a300-#.tgz
Armadillo-500	http://download.atmark-techno.com/armadillo-500/debian/debian-etch-arm#.tgz

以下に Armadillo-500 での構築例を示します。

```
[armadillo ~]# mount /dev/hda2 /mnt
[armadillo ~]# mount -t ramfs ramfs /tmp
[armadillo ~]# cd /tmp

[LOOP]  debian-etch-arm#.tgzの#の部分を 1~5 まで繰り返します。

[armadillo /tmp]# wget http://download.atmark-techno.com/armadillo-500/debian/
debian-etch-arm#.tgz
Connecting to download.atmark-techno.com [210.191.215.172]:80
debian-etch-#.tgz    100% |************************|  **** KB    00:00 ETA
[armadillo /tmp]# gzip -cd debian-etch-arm#.tgz | (cd /mnt; tar xf -)
[armadillo /tmp]# sync
[armadillo /tmp]# rm -f debian-etch-arm#.tgz

[LOOP] に戻る

[armadillo /tmp]# umount /mnt
```

図 7.4. Debian アーカイブの構築例

7.4.2. atmark-dist イメージから構築する

atmark-dist で作成されるシステムイメージをコンパクトフラッシュのルートファイルシステムとして構築する方法を説明します。Debian を構築する場合に比べ、ディスク容量の少ないコンパクトフラッシュへシステムを構築することができます。

ここで説明する例では、atmark-dist イメージの取得に wget コマンドを使用します。wget コマンドで指定する URL は製品によって異なりますので、以下の表を参照し適宜読み替えてください。

表 7.5. atmark-dist イメージのダウンロード先 URL

製品	URL
Armadillo-9	http://download.atmark-techno.com/armadillo-9/image/romfs-*[version]*.img.gz
Armadillo-300	http://download.atmark-techno.com/armadillo-300/image/romfs-a300-*[version]*.img.gz
Armadillo-500	http://download.atmark-techno.com/armadillo-500/image/romfs-a500-*[version]*.img.gz

以下に Armadillo-500 での構築例を示します。

```
[armadillo ~]# mount -t ramfs ramfs /tmp
[armadillo ~]# cd /tmp
[armadillo /tmp]# wget http://download.atmark-techno.com/armadillo-500/images/
romfs-a500-[version].img.gz
Connecting to download.atmark-techno.com [210.191.215.172]:80
romfs-a500-1.00.img.gz 100% |**********************|  **** KB    00:00 ETA
[armadillo /tmp]# gzip -dc romfs-a500-[version].img.gz > romfs.img
[armadillo /tmp]# mount /dev/hda2 /mnt
[armadillo /tmp]# mkdir romfs
[armadillo /tmp]# mount -o loop romfs.img romfs
[armadillo /tmp]# (cd romfs/; tar cf - *) | (cd /mnt; tar xf -)
[armadillo /tmp]# sync
[armadillo /tmp]# umount romfs
[armadillo /tmp]# umount /mnt
```

図 7.5. romfs.img.gz からの作成例

7.5. コンパクトフラッシュシステムから起動する

前項までで構築したコンパクトフラッシュシステムから、実際に起動させる手順を説明します。

1. Armadillo-300 の電源が切断されていることを確認し、コンパクトフラッシュを取り外します。
2. JP1 を「2-3」に設定し、Armadillo-300 に電源を投入します。
3. clearenv を実行します。
4. setenv でカーネル起動オプションを設定します。

```
hermit> clearenv
hermit> setenv console=ttyAM0,115200 root=/dev/hda2 noinitrd
1: console=ttyAM0,115200
2: root=/dev/hda2
3: noinitrd
hermit>
```

図 7.6. コンパクトフラッシュシステムから起動する

5. Armadillo-300 の電源を切断し、コンパクトフラッシュを挿入します。

6. 再度電源を投入するとコンパクトフラッシュシステムで起動します。

コンパクトフラッシュシステムをシャットダウンする場合、電源を切断する前に halt コマンドで Linux をシャットダウンする必要があります。これを実行しない場合、コンパクトフラッシュのデータが破壊される恐れがあります。

8.Hermit-At について

Mike Touloumtzis 氏がメンテナンスを行っている高機能ダウンローダ/ブートローダ「Hermit」に、Atmark Techno がオリジナルのカスタマイズ、製品の対応を行い派生させたダウンローダ/ブートローダです。

従来の Hermit では、Raw ソケットを使用した Ethernet 対応が実装されていますが、Hermit-At では、UDP/IP を実装し TFTP によるフラッシュメモリの書き換えや、Linux 起動オプションの動的変更等に対応しています。

本章では、Hermit-At に実装されている一部の機能について説明します。

8.1. setenv と clearenv

Linux 起動オプションを動的に変更させるコマンドです。

8.1.1. setenv

setenv は、指定された起動オプションを、フラッシュメモリへ書き込みます。Hermit-At が Linux を起動させる時に自動的にフラッシュメモリから起動オプションを読み込み、設定します。

構文：setenv [起動オプション]...

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

図 8.1. setenv 実行例

8.1.2. clearenv

構文：clearenv

```
hermit> clearenv
```

図 8.2. clearenv 実行例

8.1.3. Linux 起動オプション

表 8.1. よく使用される Linux 起動オプション

オプション	説明
console	シリアルコンソールが使用するデバイスを指示します。
root	ルートファイルシステム関連の設定を指示します。
noinitrd	カーネルが起動した後に initrd データがどうなるのかを指示します。
nfsroot	NFS を使用する場合に、ルートファイルシステムの場所や NFS オプションを指示します。

8.2. frob

指定したアドレスのデータを読み込む、又は変更することができるモードに移行するコマンドです。

表 8.2. frob コマンド

構文	説明
peek [addr]	指定されたアドレスから 32bit のデータを読み出します。
peek8 [addr]	指定されたアドレスから 8bit のデータを読み出します。
peek16 [addr]	指定されたアドレスから 16bit のデータを読み出します。
poke [addr] [value]	指定されたアドレスに 32bit のデータを書き込みます。
poke8 [addr] [value]	指定されたアドレスに 8bit のデータを書き込みます。
poke16 [addr] [value]	指定されたアドレスに 16bit のデータを書き込みます。

8.3. tftpdl

TFTP プロトコルを使用し、フラッシュメモリの書き換えを行うコマンド[1]です。

構文：tftpdl [クライアント IPaddr] [TFTP サーバー IPaddr] [オプション[2]]

オプション	説明
--bootloader=[filepath]	bootloader 領域のイメージファイルを指定します。
--kernel=[filepath]	kernel 領域のイメージファイルを指定します。
--userland=[filepath]	userland 領域のイメージファイルを指定します。
--fake	フラッシュメモリへの書き込みを行いません。

[1] 紙面の都合上、折り返して表現しています。
[2] 一度に複数のオプションを指定することも可能です。

```
hermit> tftpdl 192.168.10.147 192.168.10.140
         --kernel=a300/linux-a300.bin.gz
Client IPaddr   : 192.168.10.147
Server IPaddr   : 192.168.10.140
Kernel file     : a300/linux-a300.bin.gz

initializing net-device...OK
Filename : a300/linux-a300.bin.gz
................................................................
................................................................
................................................................
................................................................
..
Filesize : 1644592

programing: kernel
##########################

completed!!

hermit>
```

図 8.3. tftpdl 実行例

8.4. erase

フラッシュメモリの消去を行うコマンドです。

```
構文：erase [addr]
```

"addr"には、フラッシュメモリの物理アドレスを指定します。入力したアドレスはイレースブロックに自動的にアラインされます。フラッシュメモリの物理アドレスは、「3.2. メモリマップ」を参照してください。

IPL 領域(0x50000000 - 0x50001fff)は、消去できません。

```
hermit> erase 0x507f0000
```

図 8.4. config 領域の消去

改訂履歴

バージョン	年月日	改訂内容
1.0.0	2007/1/5	・初版発行
1.0.1	2007/7/20	・ドキュメントプロパティのタイトルを修正 ・初期不良の保証期間に関する記述修正 ・「4.1. 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 ・「4.1. 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
1.0.2	2007/9/14	・「表 1.2. 表示プロンプトと実行環境の関係」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 ・「表 4.2. atmark-dist のビルドに必要なパッケージ一覧」に libncurses5-dev を追加
1.0.3	2007/10/19	・開発環境のバージョンアップに伴う記述の変更
1.0.4	2007/12/14	・「6.1.2. 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 ・「7.4.1. Debian GNU/Linux を構築する」の Debian イメージファイル名を変更
1.0.5	2008/9/26	・図中の誤記修正 ・CD-ROM ディレクトリ構成変更に伴う修正
1.1.0	2009/03/18	・「1. はじめに」「4. 開発環境の準備」「5. フラッシュメモリの書き換え方法」「6. ビルド」 構成変更 ・誤記、表記ゆれを修正
1.1.1	2009/07/17	・「図 7.6. コンパクトフラッシュシステムから起動する」起動パーティション名の誤記修正 ・本文のレイアウト統一 ・表記ゆれを修正 ・「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 6.1. ソースコード準備」の誤記を修正
1.1.2	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-300 ソフトウェアマニュアル
Version 1.1.2-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