

Armadillo-300 ソフトウェアマニュアル

WA3000

Version 1.0.5-4145a23
2008/10/15

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

Armadillo 公式サイト [<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 6FTEL 011-207-6550 FAX 011-207-6570

製作著作 © 2008 Atmark Techno, Inc

Version 1.0.5-4145a23
2008/10/15

目次

1. はじめに	1
1.1. 対象となる読者	1
1.2. 本書の構成	1
1.3. フォントについて	1
1.4. コマンド入力例の表記について	1
1.5. アイコンについて	2
1.6. 謝辞	2
1.7. 注意事項	2
1.8. 保証に関する注意事項	3
1.8.1. 製品保証範囲について	3
1.8.2. 保証対象外になる場合	3
1.8.3. 免責事項	3
2. 作業の前に	4
2.1. 準備するもの	4
2.2. 接続方法	4
2.3. ジャンパピンの設定について	5
3. ソフトウェアについて	6
3.1. ソフトウェアの種類	6
3.1.1. 1st ブートローダ「IPL」	6
3.1.2. 2nd ブートローダ「Hermit-At」	6
3.1.3. Kernel	7
3.1.4. Userland	7
3.2. メモリマップ	8
3.3. オリジナルデバイスドライバ	8
3.3.1. GPIO デバイスドライバ	8
3.3.2. LED デバイスドライバ	9
4. 開発環境の準備	11
4.1. クロス開発環境パッケージのインストール	11
4.2. atmark-dist のビルドに必要なパッケージ	12
4.3. クロス開発用ライブラリパッケージの作成方法	12
5. フラッシュメモリの書き換え方法	13
5.1. ダウンローダのインストール	13
5.1.1. 作業用 PC が Linux の場合	13
5.1.2. 作業用 PC が Windows の場合	13
5.2. フラッシュメモリの書き込み領域について	14
5.3. hermit でフラッシュメモリを書き換える	14
5.3.1. ジャンパピンの設定	14
5.3.2. 作業用 PC が Linux の場合	14
5.3.3. 作業用 PC が Windows の場合	15
5.4. netflash でフラッシュメモリを書き換える	16
5.5. 2nd ブートローダを出荷状態に戻す	17
5.5.1. 作業用 PC が Linux の場合	17
5.5.2. 作業用 PC が Windows の場合	18
6. ソースコードからイメージファイルを作成	20
6.1. デフォルトの kernel/userland イメージを作成する	20
6.1.1. ビルドの準備	20
6.1.2. コンフィギュレーション	20
6.1.3. ビルド	22
6.2. 自作アプリケーションを追加した userland イメージを作成する	23
6.3. デフォルトの 2nd ブートローダイメージを作成する	23

7. CompactFlash システム構築	25
7.1. CompactFlash の初期化	25
7.2. ルートファイルシステムの構築	26
7.2.1. Debian GNU/Linux を構築する場合	26
7.2.2. atmark-dist イメージから構築する場合	27
7.3. Linux カーネルの配置	28
7.4. CompactFlash システムから起動する	29
8. Hermit-At について	30
8.1. setenv と clearenv	30
8.1.1. setenv	30
8.1.2. clearenv	30
8.1.3. Linux 起動オプション	31
8.2. frob	31
8.3. tftpd	31
8.4. erase	32

目次

2.1. 接続図	4
2.2. ジャンパピンの位置	5
3.1. ioctl の発行例 (GPIO)	9
3.2. ioctl の発行例 (LED)	10
4.1. インストールコマンド	11
4.2. インストール情報表示コマンド	12
4.3. クロス開発用ライブラリパッケージの作成	12
5.1. 展開処理コマンド入力例	13
5.2. コマンド入力例	14
5.3. Download モード時の画面	15
5.4. 書き換え進捗ダイアログ	16
5.5. netflash コマンド例	17
5.6. netflash ヘルプコマンド	17
5.7. shoehorn コマンド例	18
5.8. Shoehorn モード時の画面	19
5.9. shoehorn ダイアログ	19
6.1. ビルドの準備	20
6.2. コンフィギュレーション	22
6.3. ビルド	23
6.4. 自作アプリケーションを追加したイメージの作成	23
6.5. 2nd ブートローダイメージのビルド	24
7.1. CompactFlash の初期化	25
7.2. RAM ファイルシステム マウント例	26
7.3. Debian/GNU Linux の構築	27
7.4. atmark-dist イメージから構築	28
7.5. Linux カーネルの配置	28
7.6. CompactFlash システムから起動する	29
8.1. setenv 実行例	30
8.2. clearenv 実行例	30
8.3. tftpd 実行例	32
8.4. config 領域の消去	32

表目次

1.1. 使用しているフォント	1
1.2. 表示プロンプトと実行環境の関係	2
1.3. コマンド入力例での省略表記	2
2.1. ジャンパの設定	5
3.1. 2nd ブートローダイメージの種類	7
3.2. メモリマップ(フラッシュメモリ)	8
3.3. メモリマップ(RAM)	8
3.4. GPIO ノード	8
3.5. GPIO 操作コマンド	9
3.6. LED ノード	10
3.7. LED 操作コマンド	10
4.1. 開発環境一覧	11
4.2. atmark-dist のビルドに必要なパッケージ一覧	12
5.1. 各領域用のイメージファイル名	14
5.2. 各リージョンの対応デバイスファイル	17
8.1. よく使用される Linux 起動オプション	31
8.2. frob コマンド	31

1.はじめに

1.1. 対象となる読者

Armadillo-300 のソフトウェアをカスタマイズして様々な用途でご利用をお考えの方が対象となります。

本書を読む上で、以下についてある程度の知識と経験があることを前提としています。

- Windows / UNIX に関する操作
- C 言語
- 組み込みシステム

1.2. 本書の構成

本書は、Armadillo-300 を使用する上で必要な情報のうち、以下の点について記載されています。

- 開発環境の準備
- フラッシュメモリの書き換え方法
- カーネルとユーザランドのビルド

Armadillo-300 の機能を最大限に引き出すために、ご活用いただければ幸いです。

1.3. フォントについて

本書では以下のようにフォントを使っています。

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

1.4. コマンド入力例の表記について

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 上の特権ユーザで実行
[PC /]\$	作業用 PC 上の一般ユーザで実行
[armadillo300 /]#	Armadillo300 上の特権ユーザで実行
[armadillo300 /]\$	Armadillo300 上の一般ユーザで実行


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
[user]	一般ユーザ名
[group]	グループ名

1.5. アイコンについて

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。

1.6. 謝辞

Armadillo-300 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っています。この場を借りて感謝の意を示したいと思います。

1.7. 注意事項

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。

1.8. 保証に関する注意事項

1.8.1. 製品保証範囲について

付属品（ソフトウェアを含みます）を使用し、取扱説明書、各注意事項に基づく正常なご使用に限り有効です。万一正常なご使用のもと製品が故障した場合は、初期不良保証期間内であれば新品交換をさせていただきます。

1.8.2. 保証対象外になる場合

次のような場合の故障・損傷は、保証期間内であっても保証対象外になります。

1. 取扱説明書に記載されている使用方法、または注意に反したお取り扱いによる場合
2. 改造や部品交換に起因する場合。または正規のものではない機器を接続したことによる場合
3. お客様のお手元に届いた後の輸送、移動時の落下など、お取り扱いの不備による場合
4. 火災、地震、水害、落雷、その他の天災、公害や異常電圧による場合
5. AC アダプタ、専用ケーブルなどの付属品について、同梱のものを使用していない場合
6. 修理依頼の際に購入時の付属品がすべて揃っていない場合

1.8.3. 免責事項

弊社に故意または重大な過失があった場合を除き、製品の使用および、故障、修理によって発生するいかなる損害についても、弊社は一切の責任を負わないものとします。



本製品の初期不良保証期間は商品到着後 2 週間です。本製品をご購入されましたらお手数でも必ず動作確認をおこなってからご使用ください。本製品に対して注意事項を守らずに発生した故障につきましては保証対象外となります。

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ポートを持つ 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。

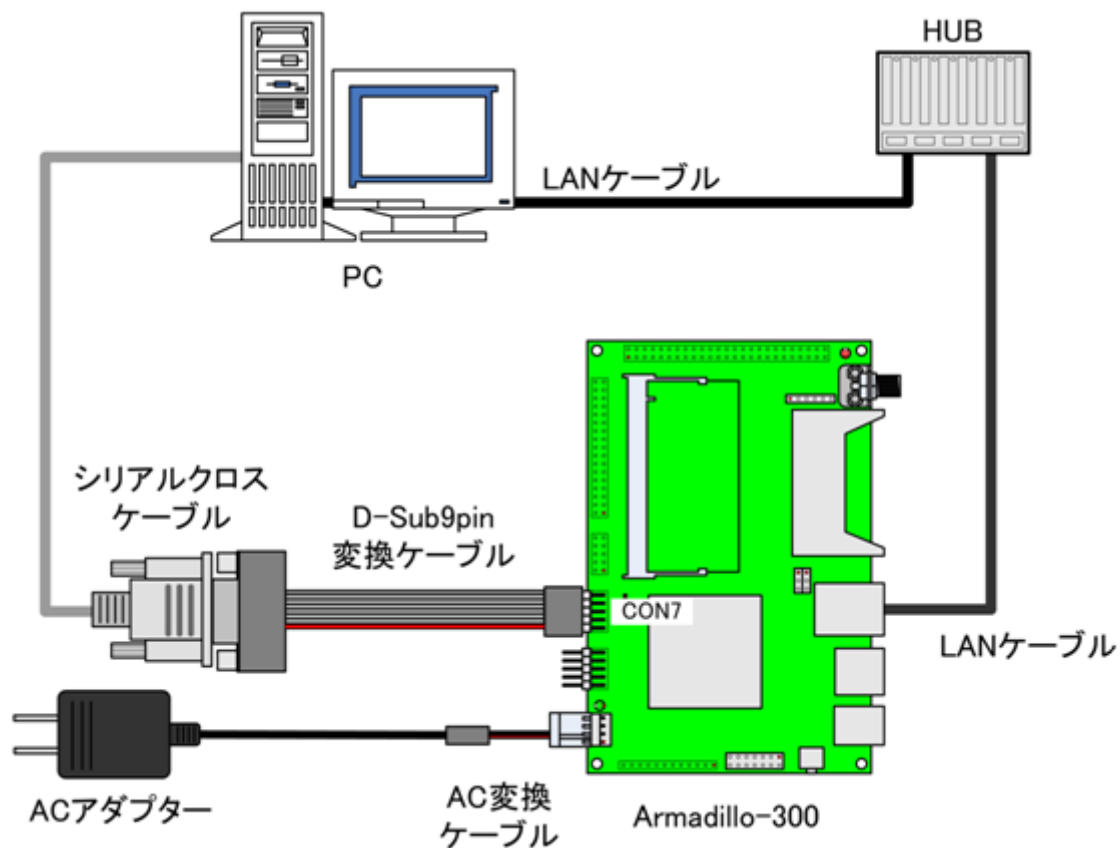



図 2.1. 接続図

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。

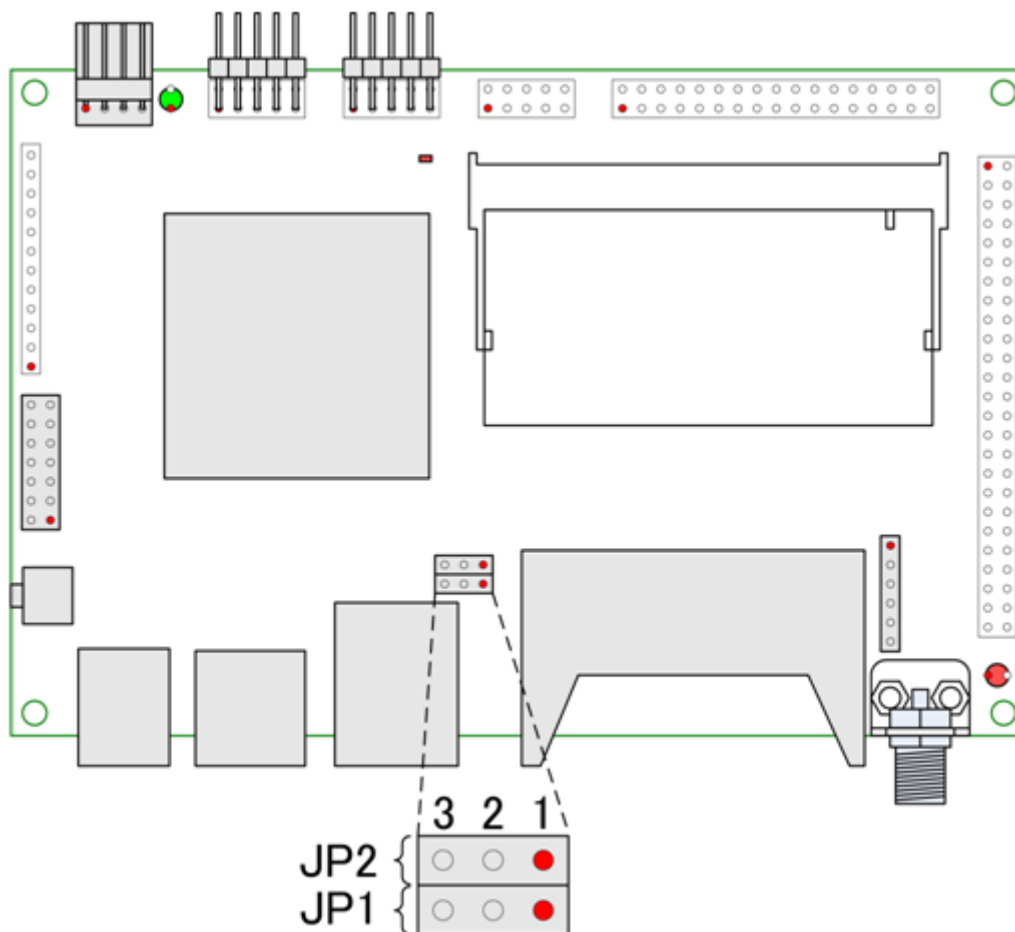


図 2.2. ジャンパピンの位置

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	シリアルポート	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ポートが 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. デフォルトの 2nd ブートローダイメージを作成する」を参照してください。

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

開発ボード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特権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 -i *.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。

¹debian 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。

²Armadillo-300 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。

```
[PC ~]# dpkg -l file
           パッケージ名
```

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なう際に、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_x.x-x_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-300 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-300 が正常に起動しなくなる場合があります。書き換え中は次の点に注意してください。

- Armadillo-300 の電源を切断しない
- Armadillo-300 と開発用 PC を接続しているシリアル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「hermit」と「shoehorn」をインストールします。これらは、Armadillo-300 のフラッシュメモリの書き換えに使用します。



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**特権ユーザ**で行ってください。

```
deb パッケージを使用する場合
[PC ~]# dpkg -i hermit-at_[version]_i386.deb
[PC ~]# dpkg -i shoehorn-at_[version]_i386.deb
```

図 5.1. 展開処理コマンド入力例

5.1.2. 作業用 PC が Windows の場合

付属 CD より「`downloader/win32/hermt-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。書き込み領域は4種類あり、それぞれに書き込むイメージファイルは「表 5.1. 各領域用のイメージファイル名」を参照してください。

表 5.1. 各領域用のイメージファイル名

領域名	ファイル名
ipl	ipl-a300.bin (書き換え不可)
bootloader	loader-armadillo3x0.bin
kernel	linux.bin.gz
userland	romfs.img.gz

5.3. hermit でフラッシュメモリを書き換える

5.3.1. ジャンパピンの設定

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 : 「2-3」に設定
- JP2 : 使用する状況に合わせて設定（通常は「1-2」に設定）

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について」を参照してください。

5.3.2. 作業用 PC が Linux の場合

1. 作業用 PC と Armadillo-300 の CON 7 をシリアルケーブルで接続します。
2. Armadillo-300 の「ジャンパピンの設定」を行い、電源を投入します。
3. 作業用 PC でイメージファイルとリージョンを指定して hermit コマンドを実行します。

```
[PC ~]$ hermit download -i linux.bin.gz -r kernel
```

コマンド指定(固定) ファイル名 リージョン指定

シリアルポートが「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 5.2. コマンド入力例



2nd ブートローダ領域(リージョン : bootloader)を書き換える際は、「--force-locked」オプションを指定する必要があります。これを指定しない場合、警告が表示されフラッシュメモリへの書き込みは実行されません。1st ブートローダ領域(リージョン : ipl)を書き換えることはできません。



2nd ブートローダ領域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2nd ブートローダを復旧してください。

書き換え終了後、JP 1 を「1-2」に設定して Armadillo-300 を再起動すると、書き込んだイメージで起動されます。

5.3.3. 作業用 PC が Windows の場合

1. 作業用 PC と Armadillo-300 の CON 7 をシリアルケーブルで接続します。
2. Armadillo-300 の「5.3.1. ジャンパピンの設定」を行い、電源を投入します。
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. 「Download」ボタンをクリックすると「図 5.3. Download モード時の画面」が表示されます。

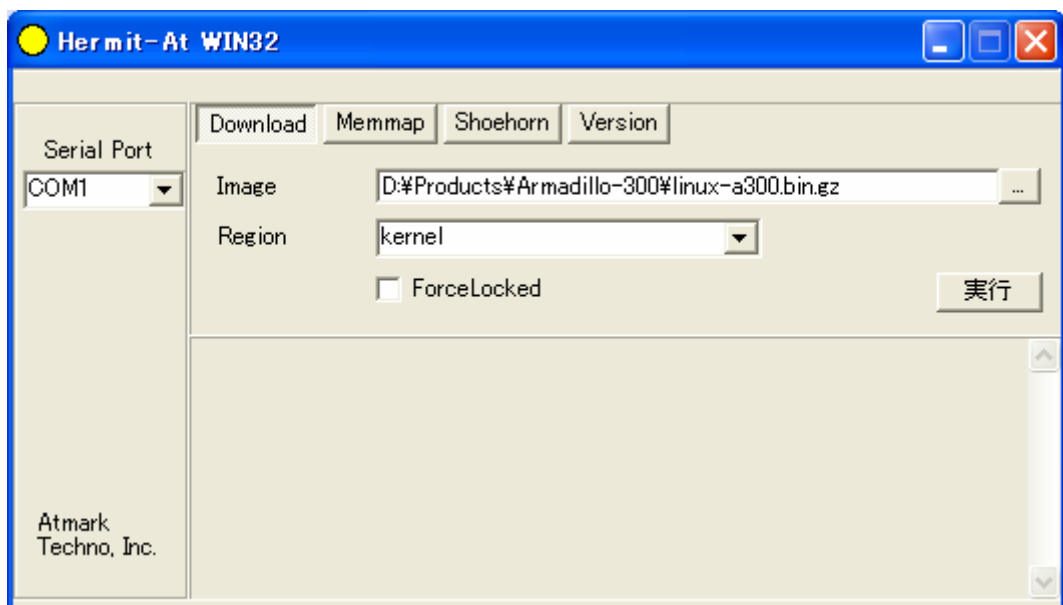


図 5.3. Download モード時の画面

5. "Serial Port"には、Armadillo-300 と接続している作業用 PC のシリアルポートを指定します。
6. "Image"には、書き込みを行いたいイメージファイルを指定します。ファイルダイアログによる指定も可能です。
7. "Region"には、書き込むリージョンを選択します。
8. 「実行」ボタンをクリックすると、フラッシュメモリの書き換えが開始されます。書き換え中は、進捗状況が「図 5.4. 書き換え進捗ダイアログ」のように表示されます。ダイアログは、書き換えが終了すると自動的にクローズされます。

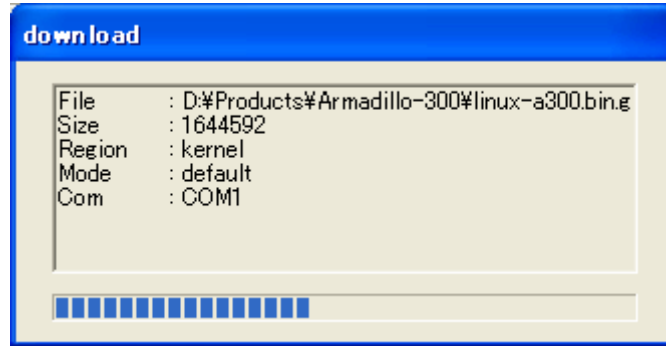


図 5.4. 書き換え進捗ダイアログ



2nd ブートローダ領域(リージョン : bootloader)を書き換える際は、「ForceLocked」をチェックする必要があります。これを指定しない場合、警告が表示されフラッシュメモリへの書き込みは実行されません。1st ブートローダ領域(リージョン : ip1)を書き換えることはできません。



2nd ブートローダ領域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2nd ブートローダを復旧してください。

書き換え終了後、JP 1 を「1-2」に設定して Armadillo-300 を再起動すると、書き込んだイメージで起動されます。

5.4. netflash でフラッシュメモリを書き換える

フラッシュメモリの内容を書き換える方法として、Armadillo-300 上で netflash というアプリケーションを使用することも可能です。ここでは、netflash を使用してフラッシュメモリを書き換える方法を説明します。



何らかの原因により「フラッシュメモリの書き換え」に失敗した場合、Armadillo-300 が正常に起動しなくなる場合があります。書き換え中は Armadillo-300 の電源を切断しないように注意してください。正常に起動しなくなった場合は「5.3. hermit でフラッシュメモリを書き換える」を参照して、復旧してください。

netflash は、HTTP や FTP サーバからファイルを取得し、フラッシュメモリへ書き込みます。netflash を使用する前に、サーバにイメージファイルを置いておく必要があります。

netflash のコマンド実行例¹です。

¹ 紙面の都合上、折り返して表現しています。

```
[armadillo300 ~]# netflash -k -n -r /dev/flash/kernel
                        オプション   リージョン指定
                        http://download.atmark-techno.com/a300/images/linux-a300-[version].bin.gz
                        ファイル名
```

図 5.5. netflash コマンド例

オプションの"-r /dev/flash/kernel"でリージョンを指定しています。リージョンの指定は、下記表を参照してください。

表 5.2. 各リージョンの対応デバイスファイル

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

netflash のヘルプは、以下のコマンドで参照することができます。


```
[armadillo300 ~]# netflash -h
```

図 5.6. netflash ヘルプコマンド

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
```

シリアルポートが「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 5.7. 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、「5.3.2. 作業用 PC が Linux の場合」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 5.8. Shoehorn モード時の画面」が表示されません。

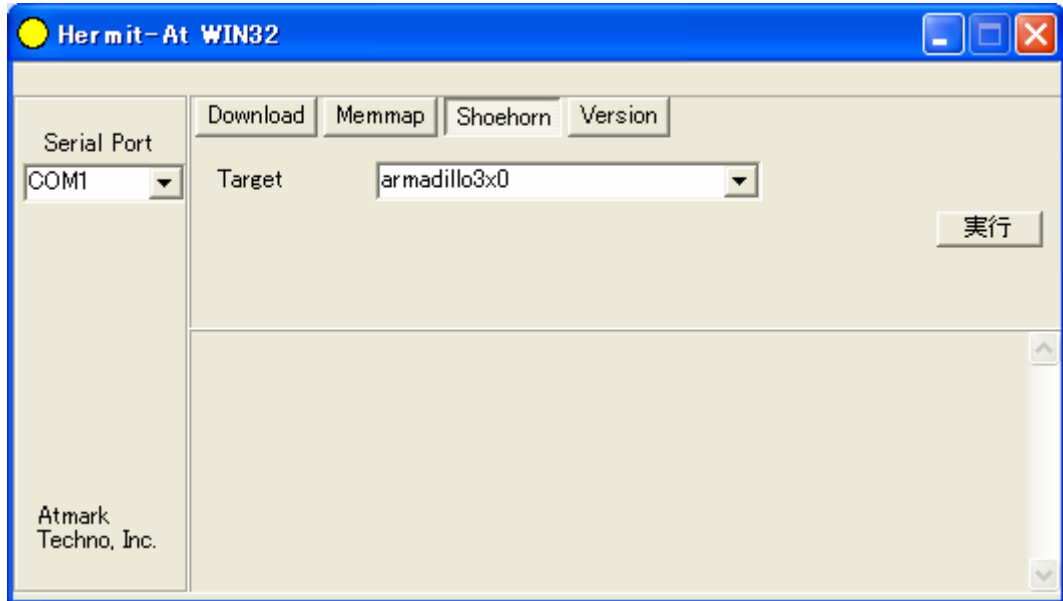


図 5.8. Shoehorn モード時の画面

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 5.9. shoehorn ダイアログ」が表示されます。

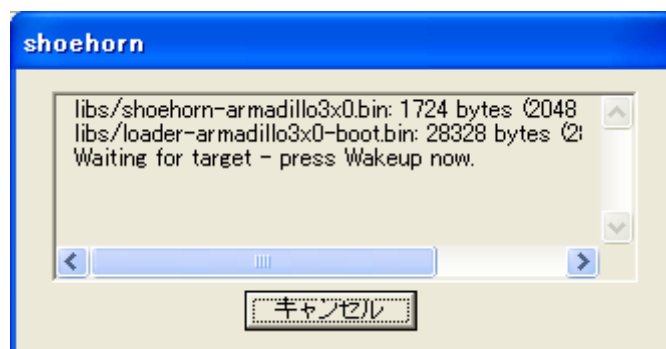


図 5.9. shoehorn ダイアログ

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、「5.3.3. 作業用 PC が Windows の場合」を参照して書き換えを行ってください。

6. ソースコードからイメージファイルを作成

実際にイメージファイルを作成する手順を説明します。

本章では、ホームディレクトリ(~/)を作業ディレクトリとします。



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・配置を行ないます。各ファイルは作業ディレクトリ配下で作成・配置作業を行ないますが、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、すべての作業は特権ユーザではなく一般ユーザで行なってください。

6.1. デフォルトの kernel/userland イメージを作成する

ここでは、付属 CD に収録されているデフォルトのイメージと同じものを作成してみます。

6.1.1. ビルドの準備

- ① 付属 CD の source/dist にある atmark-dist-YYYYMMDD.tar.gz と、source/kernel にある linux-[version].tar.gz というファイル名のアーカイブを作業ディレクトリに展開します。
- ② 展開してできた atmark-dist-YYYYMMDD ディレクトリに移動し、kernel ソースコードへのシンボリックリンクを作成します。

```
[PC ~]$ tar zxvf atamrk-dist-[version].tar.gz           ①
[PC ~]$ tar zxvf linux-[version].tar.gz               ①
[PC ~]$ cd atmark-dist-[version]                       ②
[PC ~/atmark-dist]$ ln -s ../linux-[version] linux-2.6.x ②
```

図 6.1. ビルドの準備



linux-2.6.x の「x」はそのまま記述してください。

6.1.2. コンフィギュレーション

Armadillo-300 用にビルドシステムのコンフィギュレーションを行います。

- ① atmark-dist ディレクトリに移動し、「make config」と入力します。
- ② ベンダー名を聞かれます。「AtmarkTechno」と入力します。

- ③ プロダクト名を聞かれます。「**Armadillo-300**」と入力します。
- ④ 開発環境を聞かれます。「**default**」と入力します。
- ⑤ C ライブラリを聞かれます。「**None**」と入力します。
- ⑥ プロダクトのデフォルト設定にするか聞かれます。「**y**」と入力します。
- ⑦ Kernel の設定を変更するか聞かれます。「**n**」と入力します。
- ⑧ Userland の設定を変更するか聞かれます。「**n**」と入力します。
- ⑨ コンフィギュレーション終了後、プロダクトのデフォルト設定を上書きするか聞かれます。「**n**」と入力します。
- ⑩ 設定が終わると、ビルドシステムの初期化が始まります。初期化が終わるとプロンプトに戻ります。

```

[PC ~/atmark-dist]$ make config ①
*
* 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] AtmarkTechno ②
*
* 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-9, Armadillo-9.PCMCIA, SUZAKU-V.SZ310, SUZAKU-V.SZ310-SIL, SUZAKU-
V.SZ410, SUZAKU-V.SZ410-SIL) [Armadillo-210.Base] (NEW) Armadillo-300 ③
*
* Kernel/Library/Defaults Selection
*
*
* Kernel is linux-2.6.x
*
Cross-dev (default, arm-vfp, arm, armmommu, common, h8300, host, i386, i960,
m68knommu, microblaze, mips, powerpc, sh) [default] (NEW) default ④
Libc Version (None, glibc, uC-libc, uClibc) [uClibc] None ⑤
Default all settings (lose changes) (CONFIG_DEFAULTS_OVERRIDE) [N/y/?] y ⑥
Customize Kernel Settings (CONFIG_DEFAULTS_KERNEL) [N/y/?] n ⑦
Customize Vendor/User Settings (CONFIG_DEFAULTS_VENDOR) [N/y/?] n ⑧
Update Default Vendor Settings (CONFIG_DEFAULTS_VENDOR_UPDATE) [N/y/?] n ⑨
:
:
[PC ~/atmark-dist]$ ⑩

```

図 6.2. コンフィギュレーション

6.1.3. ビルド

- ① (atmark-dist ディレクトリに移動し) 「make all」と入力します。



atmark-dist のバージョンによってはのバージョンによっては、make の途中で一時停止し、未設定項目の問合せが表示されることがあります。

す。通常はデフォルト設定のままで構いませんので、このような場合はそのまま **Enter** キーを入力して進めてください。

- ビルドが終了すると `atmark-dist/images` にカーネルイメージの「`linux.bin.gz`」とユーザーランドイメージの「`romfs.img.gz`」が作成されます。

```
[PC ~/atmark-dist]$ make all ①
[PC ~/atmark-dist]$ ls images
linux.bin  linux.bin.gz  romfs.img  romfs.img.gz ②
```

図 6.3. ビルド

6.2. 自作アプリケーションを追加した userland イメージを作成する

ここでは、「6.1. デフォルトの kernel/userland イメージを作成する」で作成したビルド環境を使用して、userland イメージを作成する手順を説明します。

追加するアプリケーションの作成方法は、「atmark-dist 開発者ガイド」等を参照してください。

- `atmark-dist` ディレクトリに移動し、コンフィギュレーションを行いビルドします。
- 追加したいアプリケーション(ここでは、「hello」を例にとります)を `atmark-dist/romfs/usr/bin` にコピーします。
- 「`make image`」と入力し、イメージを作成します。

```
[PC ~/atmark-dist]$ make config all ①
[PC ~/atmark-dist]$ cp hello romfs/usr/bin ②
[PC ~/atmark-dist]$ make image ③
[PC ~/atmark-dist]$ ls images
linux.bin  linux.bin.gz  romfs.img  romfs.img.gz
```

図 6.4. 自作アプリケーションを追加したイメージの作成

6.3. デフォルトの 2nd ブートローダイメージを作成する

ここでは、Armadillo-300 のデフォルトの 2nd ブートローダイメージを作成する手順を説明します。

- 付属 CD の `sources/bootloader` にある `hermit-at-[version]-source.tar.gz` というファイル名のアーカイブを作業ディレクトリに展開します。
- 展開してできた `hermit-at-[version]` ディレクトリに入ります。
- 「`make TARGET=armadillo3x0 PROFILE=eth`」と入力します。



Armadillo-300 の PROFILE は数種類用意されています。詳しくは「3.1.2. 2nd ブートローダ「Hermit-At」」を参照してください。

- ④ ビルドが終了すると `hermit-at-[version]/src/target/armadillo3x0` にイメージファイル「`loader-armadillo3x0-eth-v[version].bin`」が作成されます。

```
[PC ~]$ tar zxvf hermit-at-[version]-source.tar.gz ①
[PC ~]$ cd hermit-at-[version] ②
[PC ~/hermit-at]$ make TARGET=armadillo3x0 PROFILE=eth ③
[PC ~/hermit-at]$ ls src/target/armadillo3x0/*.bin
loader-armadillo3x0-eth.bin ④
```

図 6.5. 2nd ブートローダイメージのビルド

7.CompactFlash システム構築

Armadillo-300 は、CompactFlash に搭載した Linux システムから起動することができます。ここでは起動可能な CompactFlash を作成する手順を説明します。

7.1. CompactFlash の初期化

Armadillo-300 が起動可能な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。ここでは、CompactFlash を EXT2 ファイルシステムで初期化す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、CompactFlash を挿入します。
- ② JP1 を「1-2」に設定し、電源を投入します。
- ③ ログイン後、fdisk でパーティションを設定します。
- ④ 作成したパーティションを mke2fs で EXT2 ファイルシステムに初期化します。

```
[armadillo300 ~]# fdisk /dev/hda ③
hda: hda1

Command (m for help): d
Selected partition 1

Command (m for help): n
Command action
  e   extended
  p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-993, default 1): (Press Enter)
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-993, default 993): (Press Enter)
Using default value 993

Command (m for help): t
Selected partition 1
Hex code (type L to list codes): 83

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

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

[armadillo300 ~]# mke2fs -O none /dev/hda1 ④
mke2fs 1.25 (20-Sep-2001)
hda: hda1
hda: hda1
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
```

```

Fragment size=1024 (log=0)
125488 inodes, 500440 blocks
25022 blocks (5%) reserved for the super user
First data block=1
62 block groups
8192 blocks per group, 8192 fragments per group
2024 inodes per group
Superblock backups stored on blocks:
    8193, 16385, 24577, 32769, 40961, 49153, 57345, 65537, 73729, 81921,
    90113, 98305, 106497, 114689, 122881, 131073, 139265, 147457, 155649,
    163841, 172033, 180225, 188417, 196609, 204801, 212993, 221185, 229377,
    237569, 245761, 253953, 262145, 270337, 278529, 286721, 294913, 303105,
    311297, 319489, 327681, 335873, 344065, 352257, 360449, 368641, 376833,
    385025, 393217, 401409, 409601, 417793, 425985, 434177, 442369, 450561,
    458753, 466945, 475137, 483329, 491521, 499713

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

This filesystem will be automatically checked every 25 mounts or
180.00 days, whichever comes first.  Use tune2fs -c or -i to override.
[armadillo300 ~]#

```

図 7.1. CompactFlash の初期化



Armadillo-300 の起動パーティションは、mke2fs による初期化の際に必ず「-o none」オプションを指定する必要があります。

7.2. ルートファイルシステムの構築

ここでは、CompactFlash にルートファイルシステムを構築する方法として以下の内容を紹介합니다。

- Debian GNU/Linux を構築する
- atmark-dist イメージから構築する

作業上、Armadillo-300 の /home/ftp/pub には、特定のファイルを保存する場合があります。「図 7.2. RAM ファイルシステム マウント例」のように RAM ファイルシステムをマウントし、書き込み権限を与えておいてください。

```

[armadillo300 ~]# mount -t ramfs none /home/ftp/pub
[armadillo300 ~]# chmod 0777 /home/ftp/pub


```

図 7.2. RAM ファイルシステム マウント例

7.2.1. Debian GNU/Linux を構築する場合

CompactFlash に Debian GNU/Linux のルートファイルシステムを構築します。Debian イメージは、付属 CD の debian ディレクトリに debian-etch-a300-1.tgz ~ debian-etch-a300-5.tgz

として分割されたファイルが用意されています。このファイルを CompactFlash へ展開しインストールします。



Debian GNU/Linux のイメージを使用するには、CompactFlash の空き容量が 300MB 以上必要です。

- ❶ CompactFlash をマウントし、データを読み書きできるようにします。
- ❷ PC から Armadillo-300 へアーカイブファイルを転送します。
- ❸ アーカイブファイルを CompactFlash へ移動します。
- ❹ アーカイブファイルを展開します。
- ❺ CompactFlash に全データが書き込まれるまで待ちます。
- ❻ アーカイブファイルを削除します。
- ❼ ❷ ~ ❻ をアーカイブファイルすべてに対し行います。

```

[armadillo300 ~]# mount -t ext2 /dev/hda1 /mnt ❶
-----
[PC ~]$ ftp xxx.xxx.xxx.xxx ❷
Password:
ftp> cd pub
ftp> bin
ftp> put debian-etch-a300-1.tgz
-----
[armadillo300 ~]# mv /home/ftp/pub/ debian-etch-a300-1.tgz /mnt ❸
[armadillo300 ~]# gzip -cd /mnt/ debian-etch-a300-1.tgz | (cd /mnt; tar xf -) ❹
[armadillo300 ~]# sync ❺
[armadillo300 ~]# rm -f /mnt/debian-etch-a300-1.tgz ❻
    
```

図 7.3. Debian/GNU Linux の構築

7.2.2. atmark-dist イメージから構築する場合

atmark-dist で作成されたルートファイルシステムをそのまま CompactFlash のルートファイルシステムとして構築します。ここでは、ユーザランドイメージ (romfs.img.gz) から構築する手順を説明します。

- ❶ 作業用 PC で romfs.img.gz を展開し、romfs.img を作成します。
- ❷ romfs.img をマウントします。(イメージのマウントには、root 権限が必要となります。)
- ❸ ルートファイルシステムのアーカイブを作成します。
- ❹ 一般ユーザがアーカイブファイルを扱えるように、所有者を変更します。
- ❺ PC から Armadillo-300 へアーカイブファイルを転送します。
- ❻ CompactFlash をマウントし、データを読み書きできるようにします。

- ⑦ アーカイブファイルを展開します。
- ⑧ CompactFlash に全データが書き込まれるまで待ちます。

```

[PC ~]$ gzip -dc romfs.img.gz > romfs.img           ①
[PC ~]$ su -
[PC ~]# mount -t ext2 -o loop romfs.img /mnt         ②
[PC ~]# (cd /mnt; tar czvf - *) > romfs-image.tar.gz ③
[PC ~]# chown [user]:[group] romfs-image.tar.gz     ④
[PC ~]# umount /mnt
[PC ~]# exit
[PC ~]$ ftp xxx.xxx.xxx.xxx                         ⑤
Password:
ftp> cd pub
ftp> bin
ftp> put romfs-image.tar.gz
-----
[armadillo300 ~]# mount -t ext2 /dev/hda1 /mnt       ⑥
[armadillo300 ~]# gzip -cd /home/ftp/pub/romfs-image.tar.gz | \
                    (cd /mnt; tar xf -)           ⑦
[armadillo300 ~]# sync                               ⑧

```

図 7.4. atmark-dist イメージから構築

7.3. Linux カーネルの配置

CompactFlash システムから起動する場合、CompactFlash 内の /boot ディレクトリに非圧縮カーネルイメージ (Image) 圧縮カーネルイメージ (Image.gz) のどちらかが必要となります。

- ① PC から Armadillo-300 へカーネルイメージを転送します。
- ② CompactFlash をマウントし、データを読み書きできるようにします。
- ③ カーネルイメージを CompactFlash の /boot ディレクトリに移動し、名前を変更します。
- ④ CompactFlash に全データが書き込まれるまで待ちます。

```

[PC ~]$ ftp xxx.xxx.xxx.xxx                         ①
Password:
ftp> cd pub
ftp> bin
ftp> put linux.bin.gz
-----
[armadillo300 ~]# mount -t ext2 /dev/hda1 /mnt       ②
[armadillo300 ~]# mv /home/ftp/pub/linux.bin.gz /mnt/boot/Image.gz ③
[armadillo300 ~]# sync                               ④

```

図 7.5. Linux カーネルの配置

7.4. CompactFlash システムから起動する

前項までで構築した CompactFlash システムから、実際に起動させる手順を説明します。

- ❶ Armadillo-300 の電源が切断されていることを確認し、CompactFlash を取り外します。
- ❷ JP1 を「2-3」に設定し、Armadillo-300 に電源を投入します。
- ❸ `clearenv` を実行します。
- ❹ `setenv` でカーネル起動オプションを設定します。
- ❺ Armadillo-300 の電源を切断し、CompactFlash を挿入します。
- ❻ 再度電源を投入すると CompactFlash システムで起動します。

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

図 7.6. CompactFlash システムから起動する



CompactFlash 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、CompactFlash 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. tftpd

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 : tftpd [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> tftpd1 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. tftpd1 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	<ul style="list-style-type: none">• 初版発行
1.0.1	2007/7/20	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• 「表 1.3. コマンド入力例での省略表記」を追加• 「1.8.1. 製品保証範囲について」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	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• 「6.1.2. コンフィギュレーション」について、atmark-dist-20071112 で変更された内容にあわせて修正• 「7.2.1. Debian GNU/Linux を構築する場合」の Debian イメージファイル名を変更
1.0.5	2008/9/26	<ul style="list-style-type: none">• 図中の誤記修正• CD-ROM ディレクトリ構成変更に伴う修正

Armadillo-300 ソフトウェアマニュアル
Version 1.0.5-4145a23
2008/10/15

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
