

Armadillo-800 EVA 製品マニュアル

A8000-D00Z

Version 1.4.0
2015/02/06

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

Armadillo サイト [<http://armadillo.atmark-techno.com>]

Armadillo-800 EVA 製品マニュアル

株式会社アットマークテクノ

札幌本社

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

横浜営業所

〒221-0835 横浜市神奈川区鶴屋町3丁目 30-4 明治安田生命横浜西口ビル 7F
TEL 045-548-5651 FAX 050-3737-4597

製作著作 © 2012-2015 Atmark Techno, Inc.

Version 1.4.0
2015/02/06

目次

1. はじめに	12
1.1. 本書および関連ファイルについて	12
1.2. 本書の構成	12
1.3. 表記について	12
1.3.1. フォント	12
1.3.2. コマンド入力例	13
1.3.3. アイコン	13
1.4. 謝辞	13
2. 注意事項	14
2.1. 製品本体開封についてのご注意	14
2.2. 評価ボードについてのご注意	14
2.3. 安全に関する注意事項	14
2.4. 無線 LAN 機能搭載製品の使用上のご注意	15
2.5. 保証について	16
2.6. 輸出について	16
2.7. 商標について	16
3. 概要	17
3.1. ボード概要	17
3.2. ブロック図	19
4. インターフェースレイアウト	21
4.1. Armadillo-800 EVA のインターフェース配置	21
4.2. ディップスイッチ	23
5. 電源を入れる前に	25
5.1. 準備するもの	25
5.2. 接続方法	25
5.3. ディップスイッチの設定	27
5.4. シリアル通信ソフトウェアの設定	27
5.5. インストール済みのソフトウェア	28
6. Android で起動する	29
6.1. 準備	29
6.2. 起動	30
6.3. 終了	30
6.4. 基本操作	31
6.4.1. ロック画面	32
6.4.2. ホーム画面	32
6.4.3. アプリケーション一覧	33
7. Debian GNU/Linux で起動する	35
7.1. 準備	35
7.2. 起動	36
7.3. ログイン	37
7.4. 終了	37
8. Debian GNU/Linux で各種機能を使用する	38
8.1. 有線 LAN	38
8.2. 無線 LAN	39
8.2.1. 準備	40
8.2.2. 無線設定	41
8.3. 時刻設定	47
8.4. パッケージ管理	48
8.5. GStreamer	50
8.5.1. GStreamer のインストール	50

8.5.2. GStreamer の使用	50
8.6. X サーバー	51
8.6.1. X サーバーのインストール	51
8.6.2. X サーバーの起動	52
8.7. ビデオ	53
8.7.1. ビデオに画像を表示する	53
8.7.2. ディスプレイのビデオモードを設定する	54
8.8. カメラ	55
8.9. USB ガジェット	55
8.10. オーディオ	57
8.11. ストレージ	59
8.11.1. ストレージの使用方法	60
8.11.2. ストレージのパーティション変更とフォーマット	60
8.12. LED バックライト	62
8.13. LED	63
8.13.1. LED を点灯/消灯する	63
8.13.2. LED トリガを使用する	63
8.14. ユーザースイッチ	64
8.15. タッチパネル	65
9. SD ブートの活用	67
9.1. ブートディスクの作成	67
9.1.1. ブートディスクの作成に必要なファイルの取得	68
9.1.2. パーティションの作成	68
9.1.3. ファイルシステムの構築	70
9.1.4. システムイメージの展開	71
9.1.5. 設定ファイルの編集	71
9.2. ブートディスクから起動する	71
10. リカバリ手順	73
10.1. パーティション構成	73
10.2. eMMC 全体をリカバリする	74
10.2.1. リカバリディスクの作成	74
10.2.2. リカバリを実行する	78
10.3. eMMC の特定ルートファイルシステムをリカバリする	78
10.3.1. Debian GNU/Linux をリカバリする	79
10.3.2. Android をリカバリする	80
10.4. eMMC のブートローダーをリカバリする	81
10.4.1. ブートローダーのリカバリに必要なファイルの取得	81
10.4.2. ブートローダーのリカバリを実行する	81
11. 開発環境の準備	83
11.1. ATDE の使用方法	83
11.1.1. ATDE の取得	83
11.1.2. ATDE の起動	84
11.2. クロス開発ツールのインストール方法	84
11.2.1. クロス開発ツールが依存する Debian パッケージのインストール	84
11.2.2. クロス開発ツール Debian パッケージの取得	84
11.2.3. クロス開発ツール Debian パッケージのインストール	85
12. カーネルのビルド	86
12.1. ソースアーカイブの取得	86
12.2. ソースアーカイブの展開	86
12.3. ビルド	86
12.4. インストール	87
13. SGX 用カーネルモジュールのビルド	88
13.1. ソースアーカイブ取得	88

13.2. カーネルの準備	88
13.3. ビルド	88
13.4. インストール	89
14. 無線 LAN(AWL13) 用 Linux デバイスドライバーのビルド	91
14.1. ソースアーカイブ取得	91
14.2. カーネルの準備	91
14.3. ビルド	91
14.4. インストール	92
15. ブートローダーのビルド	94
15.1. ソースアーカイブの取得	94
15.2. ソースアーカイブの展開	94
15.3. ビルド	94
15.4. インストール	95
16. JTAG ICE を利用する	96
16.1. 準備	96
16.1.1. JTAG ケーブルの接続	96
16.1.2. ディップスイッチの設定	96
16.2. 接続確認	96
16.3. 各種デバッガへの対応について	96
17. Linux カーネル仕様	97
17.1. デフォルトコンフィギュレーション	97
17.2. Android 機能	97
17.3. Linux ドライバー 一覧	98
18. インターフェース仕様	101
18.1. CON1(カメラモジュールインターフェース)	101
18.2. CON2(拡張バスインターフェース)	101
18.3. CON3(デジタル HD 出カインターフェース)	103
18.4. CON4(コンポジットビデオ出カインターフェース)	103
18.5. CON5(H-UDI JTAG インターフェース)	103
18.6. CON6(ARM JTAG インターフェース)	104
18.7. CON7(SD インターフェース 1)	105
18.8. CON8(SD インターフェース 2)	105
18.9. CON9(RTC 外部バックアップインターフェース)	106
18.10. CON10~CON13(オーディオインターフェース)	106
18.10.1. CON10(モノラルマイク入カインターフェース)	106
18.10.2. CON11(ステレオヘッドホン出カインターフェース)	106
18.10.3. CON12(ステレオライン出力(L)インターフェース)	107
18.10.4. CON13(ステレオライン出力(R)インターフェース)	107
18.11. CON14(AWL13 モジュールインターフェース)	107
18.12. CON15(拡張インターフェース)	108
18.13. CON16(LCD 拡張インターフェース 1)	110
18.14. CON17(LCD 拡張インターフェース 2)	111
18.15. CON19(電源入カインターフェース)	112
18.16. CON20(USB インターフェース 1)	112
18.17. CON21(USB インターフェース 2)	113
18.18. CON22(シリアルインターフェース)	113
18.19. CON23(LAN インターフェース)	113
18.20. CON24(USB インターフェース 3)	114
18.21. LED1(カメラ LED)	114
18.22. LED2(電源 LED)	114
18.23. LED3~LED6(ユーザー LED)	115
18.24. LED7、LED8(LAN LED)	115
18.25. SW1(機能選択スイッチ)	115

18.26. SW2(リセットスイッチ)	115
18.27. SW3~SW6(ユーザースイッチ)	116
19. 基板形状図	117
A. Hermit-At ブートローダー	119
A.1. version	119
A.1.1. version 使用例	120
A.2. info	120
A.2.1. info 使用例	120
A.3. mac	120
A.3.1. mac 使用例	121
A.4. setenv と clearenv	121
A.4.1. setenv/clearenv 使用例	121
A.4.2. Linux カーネルパラメーター	121
A.5. setbootdevice	122
A.5.1. setbootdevice の使用例	122
A.6. frob	122
A.7. boot	122
A.7.1. boot 使用例	123
B. LCD パネルのドット欠けについて	124
B.1. 点欠陥の定義	124
B.2. 検査基準	124
C. LCD パネル固定部品の変更	125
C.1. 変更による影響	125
C.2. 外観・寸法図	125

目次

3.1. Armadillo-800 EVA のブロック図	20
4.1. Armadillo-800 EVA のインターフェース配置図	21
4.2. ディップスイッチ(SW1)の設定(出荷時)	23
5.1. Armadillo-800 EVA 接続例	26
5.2. ディップスイッチの設定	27
6.1. ディップスイッチの起動モード設定	29
6.2. 起動 OS 設定の変更手順(Android)	29
6.3. 起動ログ(Android)	30
6.4. 携帯電話オプション画面: Power off	31
6.5. 終了ダイアログ画面: OK ボタン	31
6.6. 終了プログレス画面: 終了スピナー	31
6.7. ロック画面	32
6.8. ホーム画面	33
6.9. ホーム画面: アプリケーション一覧ボタン	33
6.10. アプリケーション一覧画面	33
7.1. ディップスイッチの起動モード設定	35
7.2. 起動設定の変更手順(Debian)	35
7.3. 起動ログ(Linux)	36
7.4. 終了方法	37
8.1. ifup コマンドによる有線 LAN の有効化	38
8.2. ifdown コマンドによる有線 LAN の無効化	39
8.3. ディップスイッチの SDH11 設定(AWL13 モジュールインターフェース有効)	39
8.4. インフラストラクチャモード: WPA2-PSK(AES)設定手順	44
8.5. インフラストラクチャモード: WPA2-PSK(AES)設定確認手順	45
8.6. インフラストラクチャモード: WEP 設定手順	45
8.7. インフラストラクチャモード: WEP 設定確認手順	46
8.8. アドホックモード: WEP 設定手順	47
8.9. アドホックモード: WEP 設定確認手順	47
8.10. システムクロックの設定	48
8.11. ハードウェアクロックの設定	48
8.12. GStreamer のインストール	50
8.13. テスト音声の再生	51
8.14. Armadillo-800 EVA でのテスト音声の再生	51
8.15. X サーバーのインストール	51
8.16. X サーバーの起動(LCD)	52
8.17. X サーバーの起動(ディスプレイ)	52
8.18. X サーバー起動画面	52
8.19. JPEG 画像の表示(LCD)	53
8.20. JPEG 画像の表示(ディスプレイ)	53
8.21. gnome-backgrounds パッケージの画像ファイルを利用	54
8.22. 設定可能なビデオモードの表示(ディスプレイ)	54
8.23. ビデオモードの設定(ディスプレイ)	55
8.24. カメラの映像を LCD に表示	55
8.25. ディップスイッチの USB0 設定(USB インターフェース 3 有効)	55
8.26. USB Ethernet ガジェット ネットワークインターフェースの有効化(Armadillo-800 EVA) ...	56
8.27. USB Ethernet ガジェット ネットワークインターフェースの有効化(作業用 PC)	56
8.28. 音声の録音	58
8.29. 音声ファイルの再生(hw:0)	58
8.30. 音声ファイルの再生(hw:1)	58
8.31. マイク入力を hw:0 に出力	58

8.32. gnome-audio パッケージの音声ファイルを利用	58
8.33. ディップスイッチの SDHI1/USB1 設定	59
8.34. ストレージのマウント	60
8.35. ストレージのアンマウント	60
8.36. ストレージのパーティション変更	61
8.37. ストレージのフォーマット	62
8.38. LED バックライトの輝度を変更	62
8.39. LED3 を点灯させる	63
8.40. LED3 を消灯させる	63
8.41. LED3 の状態を表示する	63
8.42. LED3 のトリガに timer を設定する	64
8.43. LED3 のトリガ表示する	64
8.44. evtest のインストール	65
8.45. ユーザースイッチのイベントを取得	65
8.46. タッチパネルのイベントを取得	66
9.1. パーティション作成手順	68
9.2. パーティション確認手順	69
9.3. ファイルシステム作成手順	70
9.4. システムイメージ展開手順	71
9.5. fstab の編集	71
9.6. ブートディスクからの起動	72
10.1. パーティション作成手順	74
10.2. パーティション確認手順	76
10.3. ファイルシステム作成手順	76
10.4. ファイル展開手順	77
10.5. Debian GNU/Linux のリカバリ手順	79
10.6. Android のリカバリ手順	80
10.7. ブートローダーのリカバリ手順	81
11.1. クロス開発ツールが依存する Debian パッケージインストールコマンド	84
11.2. 64-bit PC 用クロス開発ツール Debian パッケージインストールコマンド	85
11.3. 32-bit PC 用クロス開発ツール Debian パッケージインストールコマンド	85
12.1. ソースアーカイブの展開	86
12.2. カーネルのビルド	86
12.3. Android システムへのカーネルイメージのインストール	87
12.4. Debian GNU/Linux システムへのカーネルイメージのインストール	87
13.1. SGX 用カーネルモジュールのビルド	89
13.2. SGX 用カーネルモジュールのインストール	90
14.1. AWL13 ドライバーのビルド	92
14.2. AWL13 用カーネルモジュールのインストール	93
15.1. ソースアーカイブの展開	94
15.2. ブートローダーのビルド	95
15.3. ブートローダーイメージのインストール	95
16.1. ディップスイッチの JTAG 設定(ARM)	96
18.1. AC アダプターの極性マーク	112
18.2. リセットブロック図	116
19.1. 基板形状および固定穴寸法	117
19.2. コネクタ中心寸法	118
A.1. version 構文	120
A.2. version の使用例	120
A.3. info 構文	120
A.4. info の使用例	120
A.5. mac 構文	120
A.6. mac の使用例	121

A.7. setenv/clearenv 構文	121
A.8. setenv と clearenv の使用例	121
A.9. setbootdevice 構文	122
A.10. ブートデバイスに内蔵ストレージのパーティション 4 を指定する	122
A.11. ブートデバイスに SD カードを指定する	122
A.12. boot 構文	123
A.13. boot の使用例	123
C.1. 外観: 変更前	125
C.2. 外観: 変更後	126
C.3. 寸法図	126

表目次

1.1. 使用しているフォント	13
1.2. 表示プロンプトと実行環境の関係	13
1.3. コマンド入力例での省略表記	13
3.1. Armadillo-800 EVA の仕様	17
4.1. Armadillo-800 EVA のインターフェース内容	22
4.2. ディップスイッチの設定(出荷時)	23
4.3. ディップスイッチ(SW1)のスイッチの機能	23
5.1. ディップスイッチの設定	27
5.2. シリアル通信設定	27
5.3. インストールされている OS	28
6.1. ユーザースイッチの名称と機能	32
7.1. シリアルコンソールログイン時のユーザー名とパスワード	37
8.1. インフラストラクチャモード: WPA-PSK/WPA2-PSK パラメーター例	44
8.2. インフラストラクチャモード: WEP パラメーター例	45
8.3. アドホックモード: WEP パラメーター例	46
8.4. エレメントの種類	51
8.5. ALSA デバイスとインターフェースの対応	57
8.6. オーディオインターフェースに接続する機材	57
8.7. ストレージデバイス	59
8.8. 輝度設定に使用するファイル	62
8.9. LED と LED クラスディレクトリの対応	63
8.10. trigger に設定可能なトリガ	64
8.11. スイッチとインプットデバイスファイルの対応	64
9.1. ブートディスクの構成	67
9.2. ブートディスクの作成に必要なファイル	68
10.1. 内蔵ストレージのパーティション構成	73
10.2. eMMC のブートパーティション	73
10.3. 内蔵ストレージ各領域の用途	73
10.4. リカバリディスクの作成に必要なファイル	74
10.5. リカバリ進捗と LED の対応	78
10.6. リカバリ対象のルートファイルシステム	78
10.7. Debian GNU/Linux のリカバリに必要なファイル	79
10.8. Android のリカバリに必要なファイル	80
10.9. ブートローダーのリカバリに必要なファイル	81
11.1. ATDE4 の種類	83
11.2. ユーザー名とパスワード	84
17.1. OS とデフォルトコンフィギュレーション	97
17.2. Android 機能の主要コンフィギュレーション	97
18.1. CON1 信号配列	101
18.2. CON2 信号配列	101
18.3. CON3 信号配列	103
18.4. CON4 信号配列	103
18.5. CON5 信号配列	104
18.6. CON6 信号配列	104
18.7. CON7 信号配列	105
18.8. CON8 信号配列	105
18.9. CON9 信号配列	106
18.10. CON10 信号配列	106
18.11. CON11 信号配列	107
18.12. CON12 信号配列	107

18.13. CON13 信号配列	107
18.14. CON14 信号配列	107
18.15. CON15 信号配列	108
18.16. CON16 信号配列	110
18.17. CON17 信号配列	111
18.18. CON19 信号配列	112
18.19. CON20 信号配列	113
18.20. CON21 信号配列	113
18.21. CON22 信号配列	113
18.22. CON23 信号配列	114
18.23. CON24 信号配列	114
18.24. LED1 の挙動	114
18.25. LED の挙動	115
18.26. LED3～LED6 の挙動	115
18.27. LED7、LED8 の挙動	115
18.28. SW1 信号配列	115
18.29. SW2 の機能	116
18.30. SW3～SW6 の機能	116
A.1. よく使用される Linux カーネルパラメーター	121
A.2. frob コマンド	122
B.1. 点欠陥の定義	124
B.2. 点欠陥許容範囲	124
C.1. 変更部品	125
C.2. 変更部品	125

1. はじめに

このたびは Armadillo-800 EVA をお求めいただき、誠にありがとうございます。

1.1. 本書および関連ファイルについて

本書を含めた関連マニュアル、ソースファイルやイメージファイルなどの関連ファイルは最新版を使用することをおすすめいたします。本書を読み進める前に、Armadillo サイト [<http://armadillo.atmark-techno.com>]から最新版の情報をご確認ください。

また、Armadillo-800 EVA をご購入の上でアットマークテクノ ユーザーズサイト [<https://users.atmark-techno.com>]から購入製品登録を行った方に限定して、下記のユーザー限定コンテンツを公開しています。

- ・ ご購入ユーザー限定のソフトウェア
- ・ Armadillo-800 EVA 回路図

購入製品登録を行うには、ユーザーズサイトの「ユーザー限定コンテンツ」メニューにアクセスしてください。

購入製品登録では、購入製品から取り出した正規認証ファイルをアップロードする必要があります。正規認証ファイルの取り出し手順は、「Armadillo-800 EVA 購入製品登録」ページからリンクされています。

1.2. 本書の構成

本書は、1 章から 19 章および 付録 から構成されています。

1 章から 4 章で、Armadillo-800 EVA を紹介します。

5 章から 10 章では、Armadillo-800 EVA を使用方法について説明します。インストール済みソフトウェアの説明や、Armadillo-800 EVA にインストールされているソフトウェアを工場出荷状態に戻すリカバリ手順などの説明が含まれています。

11 章から 17 章では、ソフトウェアの開発に必要な情報について説明します。開発環境の構築方法や、各種ソフトウェアのビルド方法などの説明が含まれています。

18 章と 19 章では、ハードウェア仕様について説明します。

最後に 付録では、ブートローダーの機能について説明します。

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の多くは 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表します。

2. 注意事項

2.1. 製品本体開封についてのご注意

製品本体を開封する前に、以下の事項をご確認ください。



- ・ 本製品をご利用いただくには、あらかじめ「ソフトウェア使用許諾契約書」(本製品に同梱されている資料「はじめにお読みください」に記載)に同意いただくことが必要です。はじめに「ソフトウェア使用許諾契約書」をご確認いただき、同意の上で開封してください。

2.2. 評価ボードについてのご注意

「開発セット」「評価セット」として販売している製品(以下「評価ボード」といいます)は、評価目的、技術開発またはデモンストレーション用途向けです。以下の事項をご理解・ご了承いただいた上で、ご使用いただきますようお願いいたします。



- ・ 評価ボードは、電子工学に関する技術知識と実務経験を有する技術者によって、良識ある技術的・実務的基準に従って取り扱われることを想定しています。
- ・ 評価ボードは、一般消費者が利用する最終製品において通常要求されるような設計上、販売上、または製造上の保護的措置については未完成品です。
- ・ 弊社は評価ボードについて、弊社の製品保証規定に従いご購入後 1 年間の交換保証のみを行うものとします。
- ・ 弊社は評価ボードのご購入者に対し、上記の交換保証を除き、評価ボードが特定目的に合致することの保証を含む明示的・黙示的な保証、その他ありとあらゆる保証に関する一切の責任を負わないものとします。
- ・ 評価ボードまたはその構成部品に不具合が発生した場合であっても、弊社はその原因の解析を行いません。

2.3. 安全に関する注意事項

本製品を安全にご使用いただくために、特に以下の点にご注意ください。



- ・ ご使用前に必ず製品マニュアルおよび関連資料をお読みになり、使用上の注意を守って正しく安全にお使いください。

- ・ マニュアルに記載されていない操作・拡張などを行う場合は、弊社 Web サイトに掲載されている資料やその他技術情報を十分に理解した上で、お客様自身の責任で安全にお使いください。
- ・ 水・湿気・ほこり・油煙等の多い場所に設置しないでください。火災、故障、感電などの原因になる場合があります。
- ・ 本製品に搭載されている部品の一部は、発熱により高温になる場合があります。周囲温度や取扱いによってはやけどの原因となる恐れがあります。本体の電源が入っている間、または電源切断後本体の温度が下がるまでの間は、基板上の電子部品、及びその周辺部分には触れないでください。
- ・ 本製品を使用して、お客様の仕様による機器・システムを開発される場合は、製品マニュアルおよび関連資料、弊社 Web サイトで提供している技術情報のほか、関連するデバイスのデータシート等を熟読し、十分に理解した上で設計・開発を行ってください。また、信頼性および安全性を確保・維持するため、事前に十分な試験を実施してください。
- ・ 本製品は、機能・精度において極めて高い信頼性・安全性が必要とされる用途(医療機器、交通関連機器、燃焼制御、安全装置等)での使用を意図しておりません。これらの設備や機器またはシステム等に使用された場合において、人身事故、火災、損害等が発生した場合、当社はいかなる責任も負いかねます。
- ・ 本製品には、一般電子機器用(OA 機器・通信機器・計測機器・工作機械等)に製造された半導体部品を使用しています。外来ノイズやサージ等により誤作動や故障が発生する可能性があります。万一誤作動または故障などが発生した場合に備え、生命・身体・財産等が侵害されることのないよう、装置としての安全設計(リミットスイッチやヒューズ・ブレーカー等の保護回路の設置、装置の多重化等)に万全を期し、信頼性および安全性維持のための十分な措置を講じた上でお使いください。
- ・ 無線 LAN 機能を搭載した製品は、心臓ペースメーカーや補聴器などの医療機器、火災報知器や自動ドアなどの自動制御器、電子レンジ、高度な電子機器やテレビ・ラジオに近接する場所、移動体識別用の構内無線局および特定小電力無線局の近くで使用しないでください。製品が発生する電波によりこれらの機器の誤作動を招く恐れがあります。

2.4. 無線 LAN 機能搭載製品の使用上のご注意

本製品は、2.4GHz 帯域の電波を使用します。稼動時に電波を使用しますので、電磁妨害や電波干渉が発生する恐れがあります。



- ・ 心臓ペースメーカーや補聴器などの医療機器、火災報知器や自動ドアなどの自動制御器、電子レンジ、高度な電子機器やテレビ・ラジオに近接する場所で使用しないでください。

- ・ 移動体識別用の構内無線局および特定小電力無線局の近くで使用しないでください。
- ・ 万一、本製品と同種無線局や他の機器との電波干渉が発生した場合は、すみやかに使用場所を変えるか、製品の稼働を停止（電波の使用を停止）してください。

2.5. 保証について

本製品の本体基板は、製品に添付もしくは弊社 Web サイトに記載している「製品保証規定」に従い、ご購入から 1 年間の交換保証を行っています。添付品およびソフトウェアは保証対象外となりますのでご注意ください。

製品保証規定 <http://www.atmark-techno.com/support/warranty-policy>

2.6. 輸出について

- ・ 当社製品は、原則として日本国内での使用を想定して開発・製造されています。
- ・ 海外の法令および規則への適合については当社はなんらの保証を行うものではありません。
- ・ 当社製品を輸出するときは、輸出者の責任において、日本国および関係する諸外国の輸出関連法令に従い、必要な手続きを行っていただきますようお願いいたします。
- ・ 日本国およびその他関係諸国による制裁または通商停止を受けている国家、組織、法人または個人に対し、当社製品を輸出、販売等することはできません。
- ・ 当社製品および関連技術は、大量破壊兵器の開発等の軍事目的、その他国内外の法令により製造・使用・販売・調達が禁止されている機器には使用することができません。

2.7. 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。™、®マークは省略しています。
- ・ SD、SDHC、SDXC、microSD、microSDHC、microSDXC、SDIO ロゴは SD-3C, LLC の商標です。



3. 概要

3.1. ボード概要

Armadillo-800 EVA はルネサス エレクトロニクス製のプロセッサ「R-Mobile A1」(ARM Cortex-A9、CPU クロック 最大800MHz)と、512MB DRAM(DDR3-800)、8GB フラッシュメモリ(eMMC)を搭載した評価ボードです。5 インチ LCD(WVGA)静電容量方式タッチパネル、デジタル HD 出力対応、3.1M ピクセル CMOS カメラモジュールなど、マルチメディア機能を多数搭載しています。また、IEEE802.11b/g/n 準拠の産業用組み込み機器向け無線 LAN モジュール「Armadillo-WLAN(AWL13)」を標準搭載しています。入手しづらいモジュールを一式取り揃え、多数の機能を評価することができます。

Armadillo-800 EVA の主な仕様は次の通りです。

表 3.1 Armadillo-800 EVA の仕様

プロセッサ	ルネサス エレクトロニクス R-Mobile A1 (R8A77404DBA)	
CPU コア	ARM Cortex-A9 シングルコア 命令/データキャッシュ 32kByte/32kByte L2 キャッシュ 256kByte メディアプロセッシングエンジン(NEON)搭載 浮動小数点コプロセッサ(VFPv3)搭載	
システムクロック	CPU コアクロック: 792MHz (最大 800MHz ^[a]) DDR クロック: 396MHz (最大 400MHz ^[a]) 内部 BUS クロック: 198MHz (最大 200MHz ^[a]) 拡張 BUS クロック: 99MHz (最大 100MHz ^[a]) 源発振クロック: 24MHz	
RAM	DDR3 SDRAM : 512MByte (32bit 幅) (DDR3-800)	
フラッシュメモリ	eMMC NAND フラッシュメモリ : 8GByte	
LAN(Ethernet)	10BASE-T/100BASE-TX AUTO-MDIX 対応	
無線 LAN	SDHI1 ^[b]	Armadillo-WLAN モジュール(AWL13) IEEE 802.11b/g/n 対応 (最大 72.2Mbps ^[c])
シリアル(UART)	最大 7 チャンネル	
	SCIFA0 ^[d]	+3.3V CMOS レベル フロー制御ピン有り(CTS、RTS) 拡張コネクタ
	SCIFA1	RS-232C レベル フロー制御ピンなし D-Sub9 ピンコネクタ
	SCIFA2 ^[e]	+3.3V CMOS レベル フロー制御ピン有り(CTS、RTS) 同期クロックピン有り(SCK) 拡張コネクタ
	SCIFA4 ^[f]	+3.3V CMOS レベル フロー制御ピンなし 拡張コネクタ
	SCIFA6 ^[d]	+3.3V CMOS レベル フロー制御ピンなし 同期クロックピン有り(SCK) 拡張コネクタ
	SCIFA7 ^[g]	+3.3V CMOS レベル フロー制御ピンなし LCD 拡張コネクタ
	SCIFB ^[h]	+3.3V CMOS レベル フロー制御ピンなし 同期クロックピン有り(SCK) 拡張コネクタまたは LCD 拡張コネクタ

USB	2 チャンネル	
	USB0 ^[i]	USB HOST(High Speed 対応) TYPE A コネクタ USB DEVICE(High Speed 対応) TYPE B コネクタ
	USB1	USB HOST(High Speed 対応) TYPE A コネクタ
SD/MMC	最大 2 チャンネル ^[j]	
	SDHI0	SD スロット
	SDHI1 ^[k]	SD スロット
ビデオ	LCDC0 ^[l]	RGB/YCrCb インターフェース 最大 1440×900 ドット/24bit カラー LCD モジュール(LCD 拡張コネクタ 2)または LCD 拡張コネクタ 1 ^[m] - AMPIRE 社製 AM-800480L1TMQW-T00H または AM-800480L1TMQW-TN0H - 5 インチ WVGA(800×480 ドット/24bit カラー) - LED バックライト - 静電容量式マルチタッチパネル付
	LCDC1	デジタル HD 出力 ^[n] HDMI Type-A コネクタ ^[o] コンポジットビデオ出力 ^[n] RCA ピンジャック
	VOU ^[l]	デジタルビデオ出力 LCD 拡張コネクタ
地デジチューナー I/F	SSP ^[d]	8bit パラレルバスまたは 1bit シリアルバス 拡張コネクタ
スマートカード I/F	SIM	拡張コネクタ
カメラ	最大 2 チャンネル ^[p]	
	CEU0	最大 16bit データバス ^[q] カメラモジュールまたは拡張コネクタ - CRESYN 社製 DCB-NSB55QFMRB - 3.1M ピクセル QXGA:最大 15fps, VGA:最大 30fps 8bit データバス)
	CEU1	8bit データバス 拡張コネクタ
オーディオ	FSIA	ステレオヘッドホン出力ミニジャック ステレオライン出力 RCA ピンジャック モノラルマイク入力ミニジャック
	FSIB	デジタル出力 HDMI Type-A コネクタ ^[o]
I2C	3 チャンネル	
	I2C0	ボード内部デバイス用 LCD 拡張コネクタ
	I2C1	外部拡張利用専用 拡張コネクタ
	I2C2(GPIO)	RTC 専用
SPI I2S Microwire	最大 2 チャンネル	
	MSIOF1 ^[r]	拡張バスコネクタ
	MSIOF2 ^[s]	LCD 拡張コネクタ
GPIO	最大 136bit ^[t]	
拡張バス	アドレスバス: 25bit データバス: 最大 16bit ^[u] チップセレクト: 最大 4bit ^[u]	
カレンダー時計	リアルタイムクロック	
RTC バックアップ電源	コイン電池対応 ^[v]	
スイッチ	タクトスイッチ x 4	
LED	LED(黄色、面実装) x 4	
JTAG	ARM 標準 20 ピンコネクタまたは H-UDI 14 ピンコネクタ ^[w]	

電源電圧	5V±5%
使用周囲温度	10~40°C(ただし結露なきこと)
基板サイズ	183×135mm(突起部含まず)

- [a]R-Mobile A1 の最大クロック周波数
- [b]SD スロットと排他で利用可能
- [c]規格上の理論値
- [d]CEU1 の一部の信号と排他で利用可能
- [e]SSP および SIM の一部の信号と排他で利用可能
- [f]SSP の一部の信号と排他で利用可能
- [g]LCDC0 の一部の信号と排他で利用可能
- [h]SSP または LCDC0 の一部の信号と排他で利用可能
- [i]USB HOST と USB DEVICE は排他で利用可能
- [j]SDHI1 を SD スロットに選択した場合のチャンネル数
- [k]無線 LAN と排他で利用可能
- [l]LCDC0 と VOU は排他で利用可能
- [m]LCD 拡張コネクタ 2 と LCD 拡張コネクタ 1 は排他で利用可能
- [n]デジタル HD 出力とコンポジットビデオは排他で利用可能
- [o]ビデオとオーディオの HDMI Type-A コネクタは同一
- [p]CEU0 を 8bit データバスで使った場合のチャンネル数
- [q]CEU1 を使わない場合に可能な最大 bit 数
- [r]拡張バスの一部の信号と排他で利用可能
- [s]LCDC0 信号の一部と排他で利用可能
- [t]拡張バスインターフェース、拡張インターフェース、LCD 拡張インターフェース 1 において、R-Mobile A1 のピンマルチプレクス機能で、GPIO として利用可能なピンの合計
- [u]eMMC を使わない場合に可能な最大 bit 数
- [v]電池は付属しません
- [w]ARM コネクタと H-UDI コネクタは信号線を共有しているため排他で利用可能

3.2. ブロック図

Armadillo-800 EVA のブロック図は次の通りです。

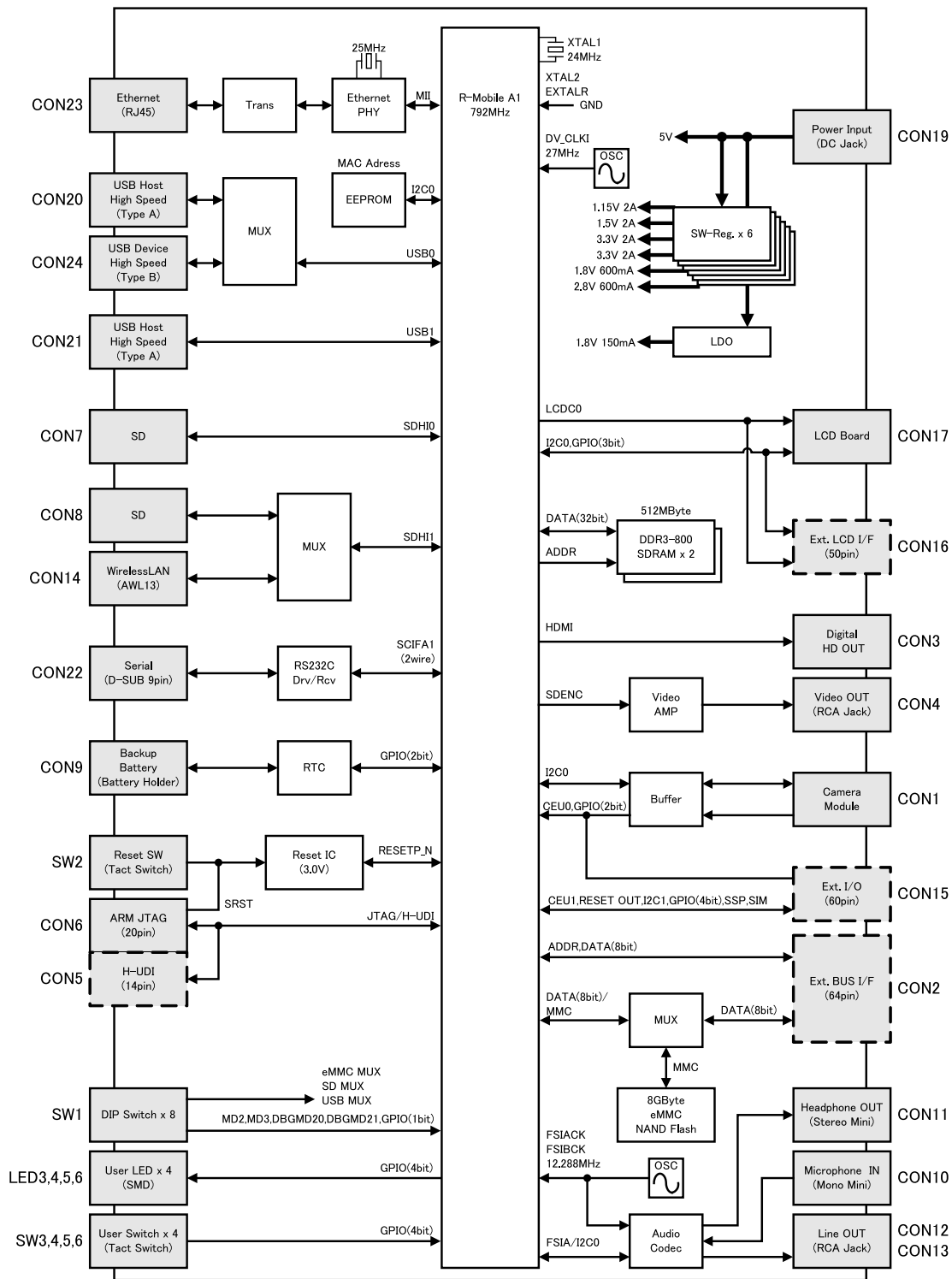


図 3.1 Armadillo-800 EVA のブロック図

4. インターフェースレイアウト

Armadillo-800 EVA のインターフェースレイアウトです。各インターフェースの配置場所等を確認してください。

4.1. Armadillo-800 EVA のインターフェース配置

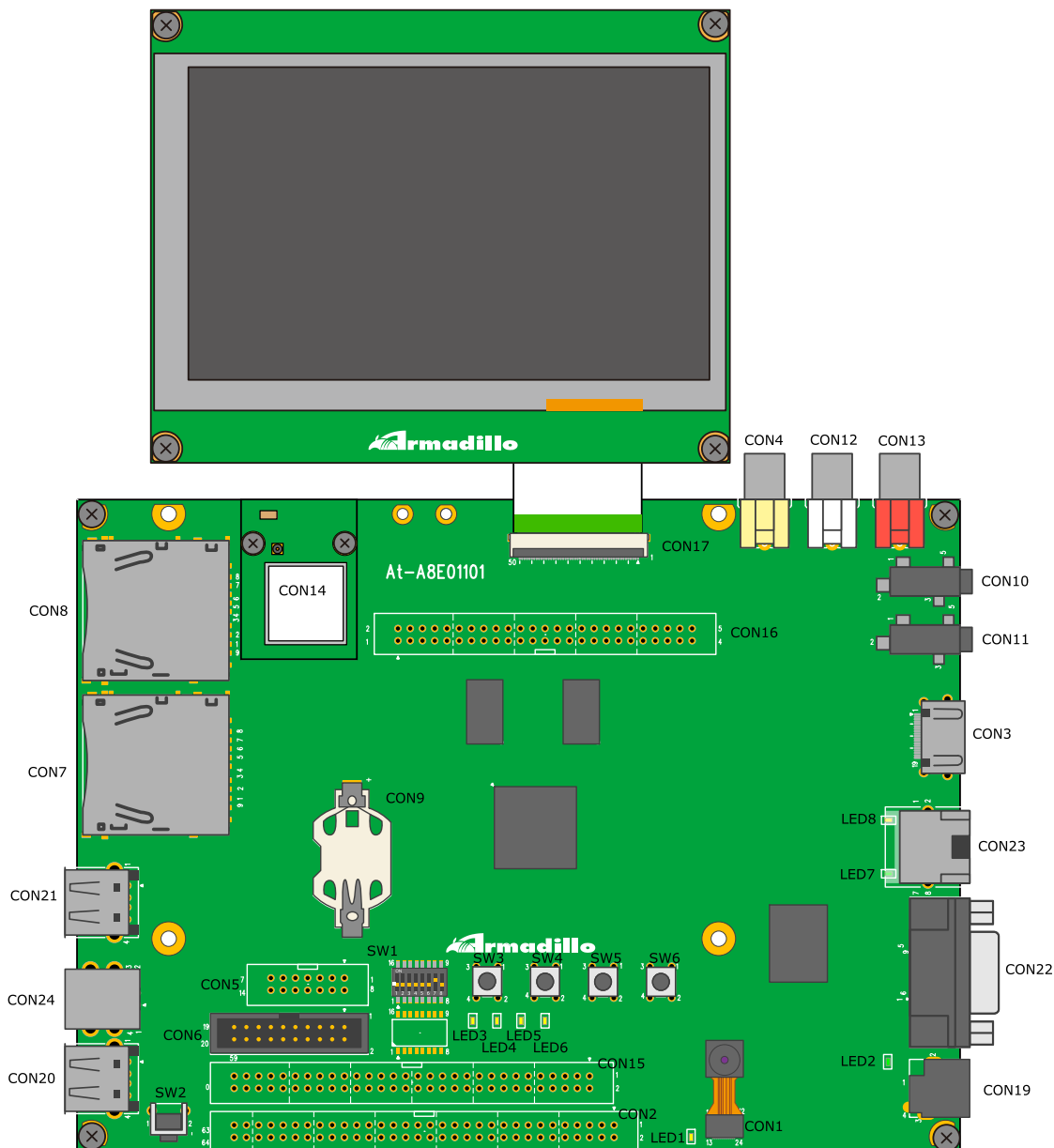


図 4.1 Armadillo-800 EVA のインターフェース配置図



2015年2月出荷品から、LCDパネルの固定方法が変更されています。
詳しくは、付録C LCDパネル固定部品の変更をご確認ください。

表 4.1 Armadillo-800 EVA のインターフェース内容

部品番号	インターフェース	形状	備考
CON1	カメラモジュール	FPC コネクタ(24P) (0.4mm ピッチ)	挿抜寿命:50 回
CON2	拡張バス	ピンヘッダ(64P) (2.54mm ピッチ)	コネクタ非搭載
CON3	デジタル HD 出力	HDMI Type A コネクタ	
CON4	コンポジットビデオ出力	RCA ジャック黄色	
CON5	H-UDI JTAG	ピンヘッダ(14P) (2.54mm ピッチ)	コネクタ非搭載
CON6	ARM JTAG	ピンヘッダ(20P) (2.54mm ピッチ)	
CON7	SD1	SD スロット	
CON8	SD2	SD スロット	
CON9	RTC 外部バックアップ	電池ボックス	対応電池: CR2032
CON10	モノラルマイク入力	ミニジャック (φ3.5mm)	
CON11	ステレオヘッドホン出力	ミニジャック (φ3.5mm)	
CON12	ステレオライン出力(L)	RCA ジャック白色	
CON13	ステレオライン出力(R)	RCA ジャック赤色	
CON14	AWL13 モジュール	狭ピッチコネクタ(34P) (0.5mm ピッチ)	挿抜寿命: 50 回
CON15	拡張	ピンヘッダ(60P) (2.54mm ピッチ)	コネクタ非搭載
CON16	LCD 拡張 1	ピンヘッダ(50P) (2.54 ピッチ)	コネクタ非搭載
CON17	LCD 拡張 2	FFC コネクタ(50P) (0.5mm ピッチ)	挿抜寿命: 20 回
CON19	電源入力	DC ジャック	対応プラグ: EIAJ #2
CON20	USB1	USB Type A コネクタ	
CON21	USB2	USB Type A コネクタ	
CON22	シリアル	D-Sub9 ピン(オス)	
CON23	LAN	RJ-45 コネクタ	
CON24	USB3	USB Type B コネクタ	
LED1	カメラ LED(黄色)	LED(面実装)	
LED2	電源 LED(緑色)	LED(面実装)	
LED3~LED6	ユーザー LED(黄色)	LED(面実装)	
LED7	LAN リンク LED(緑色)	LED(面実装)	
LED8	LAN アクティビティ LED(黄色)	LED(面実装)	
SW1	機能選択スイッチ	ディップスイッチ(8 接点)	
SW2	リセットスイッチ	タクトスイッチ(l=3.85mm) ライトアングル	
SW3~SW6	ユーザースイッチ	タクトスイッチ(h=5mm) ストレート	

4.2. ディップスイッチ

Armadillo-800 EVA に電源を入れる前にディップスイッチ(SW1)を設定する必要があります。出荷時は「表 4.2. ディップスイッチの設定(出荷時)」のように設定されています。

表 4.2 ディップスイッチの設定(出荷時)

SW1.1	SW1.2	SW1.3	SW1.4	SW1.5	SW1.6	SW1.7	SW1.8
OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF

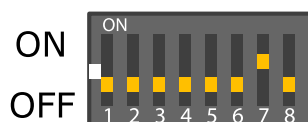


図 4.2 ディップスイッチ(SW1)の設定(出荷時)

各スイッチの機能は次の通りです。

表 4.3 ディップスイッチ(SW1)のスイッチの機能

機能	スイッチ		動作
起動モード設定	SW1.1		
	OFF		OS 自動起動モード
	ON		保守モード
起動デバイス設定	SW1.2	SW1.3	
	OFF	OFF	eMMC
	ON	OFF	SDHI0(CON7)
	OFF	ON	(未設定)
	ON	ON	拡張バス(CS0)
拡張バス設定	SW1.4		
	OFF		データバス上位 8bit(D8~D15)無効/eMMC 有効
	ON		データバス上位 8bit(D8~D15)有効/eMMC 無効
SDHI1 設定	SW1.5		
	OFF		SD インターフェース 2(CON8)無効/AWL13 モジュールインターフェース (CON14)有効
	ON		SD インターフェース 2(CON8)有効/AWL13 モジュールインターフェース (CON14)無効
USB0 設定	SW1.6		
	OFF		USB インターフェース 1(CON20)有効/USB インターフェース 3(CON24)無効
	ON		USB インターフェース 1(CON20)無効/USB インターフェース 3(CON24)有効
JTAG 設定	SW1.7	SW1.8	
	OFF	OFF	SH-X2
	ON	OFF	ARM
	OFF	ON	(未設定)
	ON	ON	パウンダリスキャン



起動デバイスを拡張バス(CS0)に設定して起動させる場合、拡張バスインターフェース(CON2)の 29 ピンの信号ではなく、R-Mobile A1 の P23 の信号をウェイト信号として使用します。

Armadillo-800 EVA は R-Mobile A1 の P23 の信号を LED6 の制御信号として使用しており、ウェイト信号として使用する場合、リセット解除後

アサート状態となりアクセスが止まってしまうため、トランジスタ(Q10)を取り外す必要があります^[1]。

詳細につきましては、Armadillo-800 EVA 回路図でご確認ください。

^[1]トランジスタ(Q10)は 2 素子入りのトランジスタで、取り外した場合には LED5、LED6 が使用できなくなります。

5. 電源を入れる前に

5.1. 準備するもの

Armadillo-800 EVA を評価するには、以下の機材を準備する必要があります。

作業用 PC	Debian GNU/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	Armadillo-800 EVA と作業用 PC を接続するための、D-Sub9 ピン（メス - メス）のクロス接続用ケーブルです。
シリアル通信ソフトウェア	Linux では「minicom」、Windows では「Tera Term Pro」などです。Armadillo を制御するために使用します。作業用 PC にインストールしてください。

また、以下の機材を利用する場合があります。必要に応じて準備してください。

SD カード	容量が 1GByte 以上のものを準備してください。Armadillo-800 EVA を工場出荷状態に戻すためなどに利用します。
USB メモリ	Armadillo-800 EVA と作業用 PC 間のデータ転送等に利用します。
USB ケーブル(A オス - B オス)	Armadillo-800 EVA と USB ガジェット を経由した通信を行う場合に必要となります。
LAN ケーブル	Armadillo-800 EVA と LAN を経由した通信を行う場合に必要となります。 ^[1]
HDMI ケーブル	CON3 から映像を出力させる場合に必要となります。
RCA ケーブル	CON12, CON13 および CON4 から映像または音声出力させる場合に必要となります。
スピーカー又はヘッドホン	CON11 から音声出力させる場合に必要となります。スピーカーを利用する場合は、別途ステレオミニプラグケーブルが必要となる場合があります。
マイク	CON10 から音声を入力させる場合に必要となります。
ディスプレイ	HDMI 端子または RCA 入力端子をもつものです。映像または音声出力させる場合に必要となります。

5.2. 接続方法

Armadillo-800 EVA の接続例です。

^[1]Armadillo-800 EVA は Auto MDIX に対応しているため、作業用 PC と LAN ケーブルで直接接続することができます。

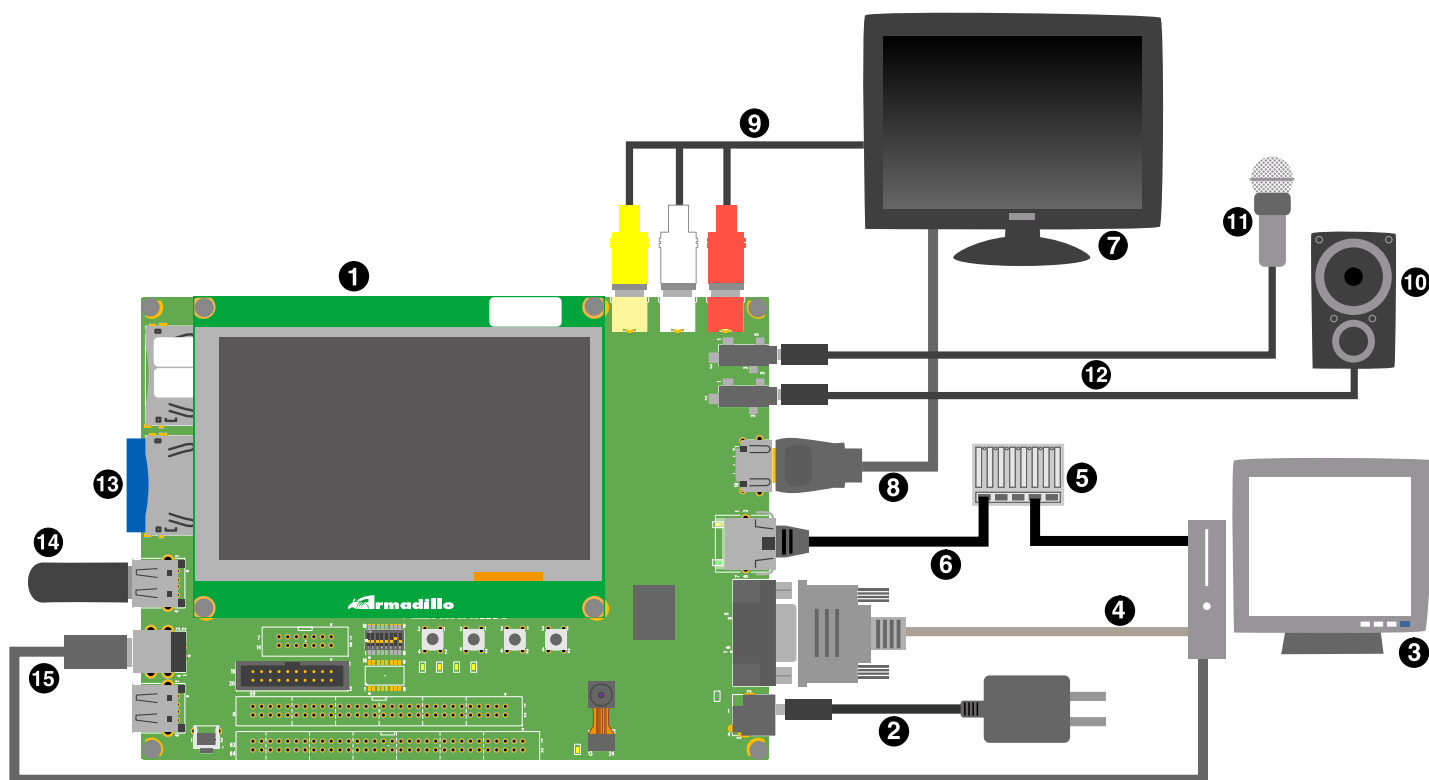



図 5.1 Armadillo-800 EVA 接続例

- ① Armadillo-800 EVA
- ② AC アダプター(付属品)
- ③ 作業用 PC
- ④ シリアルクロスケーブル(付属品)
- ⑤ LAN HUB
- ⑥ LAN ケーブル
- ⑦ ディスプレイ
- ⑧ HDMI ケーブル
- ⑨ RCA ケーブル
- ⑩ スピーカ
- ⑪ マイク
- ⑫ ステレオミニプラグケーブル
- ⑬ SD カード
- ⑭ USB メモリ
- ⑮ USB ケーブル(A オス - B オス)



AC アダプターを接続する際は、HDMI ケーブルで接続したディスプレイの電源を OFF にしてください。ディスプレイの電源が ON の状態で AC アダプターを接続すると、Armadillo-800 EVA 付属の液晶が正常に表示されない場合があります。

本現象の詳細につきましては、「Armadillo-800 EVA リビジョン情報 v1.1.0」に記載されている「A800-EVA-ERRATUM #1」を参照してください。



製品に付属の LCD モジュールおよびカメラモジュールには出荷時、保護フィルムが貼られています。保護フィルムをはがした状態でご使用ください。

5.3. ディップスイッチの設定

SW1 を「表 5.1. ディップスイッチの設定」の設定にしてください。SW1 の機能詳細については「表 4.3. ディップスイッチ(SW1)のスイッチの機能」で確認してください。

表 5.1 ディップスイッチの設定

1	2	3	4	5	6	7	8
OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF

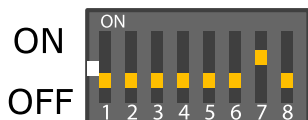


図 5.2 ディップスイッチの設定

5.4. 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 5.2. シリアル通信設定」のように設定してください。

表 5.2 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



シリアル通信ソフトウェアの横幅を 80 文字以上にしてください。横幅が 80 文字より小さい場合、コマンド入力中に表示が乱れることがあります。

5.5. インストール済みのソフトウェア

工場出荷状態で内蔵ストレージにインストールされている OS については、次のとおりとなります。

表 5.3 インストールされている OS

OS	バージョン
Debian GNU/Linux	6.0
Android	2.3.7



工場出荷状態の Armadillo-800 EVA は、電源を投入すると Android が起動するよう設定されています。電源を投入後に起動する OS は、ブートローダーの設定(起動 OS 設定)により決定されます。起動する OS を変更するには、次に示す項を参照してください。

- ・「6. Android で起動する」
- ・「7. Debian GNU/Linux で起動する」

6. Android で起動する

6.1. 準備

Android を起動させる場合の起動 OS 設定の変更について説明します。

ブートローダーを保守モードで起動し、起動 OS 設定の変更を行うことで起動する OS を変更することができます。Armadillo-800 EVA に電源を投入する前にディップスイッチの起動モード設定(SW1.1)を ON に設定し、Armadillo-800 EVA に電源を投入するとブートローダーが保守モードで起動します。

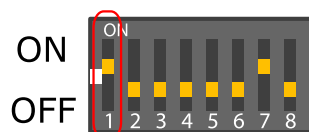


図 6.1 ディップスイッチの起動モード設定

ブートローダーを保守モードで起動すると、ブートローダーのコマンドプロンプトが表示されます。

```
Hermit-At v3.0.0 (Armadillo-800 EVA) compiled at 22:22:10, Dec 21 2011
hermit>
```

Android を起動するには「図 6.2. 起動 OS 設定の変更手順(Android)」のように起動 OS 設定を変更します。

```
hermit> setbootdevice mmcblk0p4 ❶
hermit> setenv console=ttySC1,115200 noinitrd rootwait root=/dev/mmcblk0p4 init=/init ❷
```

- ❶ Android 用の Linux カーネルを格納しているブートデバイスを設定
- ❷ Linux カーネルパラメーターを設定

図 6.2 起動 OS 設定の変更手順(Android)

以上で起動 OS 設定の変更は完了です。boot コマンドで Android を起動することができます。

```
hermit> boot
```



Armadillo-800 EVA の電源を切断し、ディップスイッチの起動モード設定(SW1.1)を OFF に設定すると、電源投入後自動的に Android が起動するようになります。

6.2. 起動

Android を起動します。次のように起動ログがシリアル通信ソフトウェアに表示されます。

```
mmcscd: SD card at address 0x00000001
mmcscd: M8G2FA 1048576KiB
gendisk: /dev/mmcblk0p4: start=0x000f4280, size=0x001dc0c0
gendisk: Image.bin is found. (4390496 Bytes)
Copying      kernel...done.
Doing console=ttySC1,115200
Doing noinitrd
Doing rootwait
Doing root=/dev/mmcblk0p4
Doing init=/init
Linux version 2.6.35.7 (atmark@atde4) (gcc version 4.4.5 (Debian 4.4.5-8) ) #1 P
REEMPT Wed Dec 21 22:37:47 JST 2011
CPU: ARMv7 Processor [412fc093] revision 3 (ARMv7), cr=10c53c7f
CPU: VIPT nonaliasing data cache, VIPT nonaliasing instruction cache
Machine: Armadillo-800EVA
:
:
:
VFS: Mounted root (ext3 filesystem) on device 179:4.
Freeing init memory: 124K
init: cannot open '/initlogo.rle'
init: cannot find '/system/etc/install-recovery.sh', disabling 'flash_recovery'
sh: can't access tty; job control turned off
$ net eth0: attached phy 0 to driver Generic PHY
WM8978 0-001a: Imprecise sampling rate: 48000Hz, consider using PLL
PHY: 0:00 - Link is Up - 100/Full
warning: `zygote' uses 32-bit capabilities (legacy support in use)
request_suspend_state: wakeup (3->0) at 11694431586 (2000-01-01 00:12:25.6519288
34 UTC)
```

図 6.3 起動ログ(Android)

6.3. 終了

安全に終了させるには、「手順 6.1. Android の終了手順」に示す手順を実行します。



linux-2.6.35-a800eva-at1 は本手順に対応していません。終了させるには電源を切断してください。電源を切断する際、リムーバブルディスクにデータを書き込み中の場合は、ファイルシステムおよびデータが破損する恐れがあります。

手順 6.1 Android の終了手順

1. SW3(電源ボタン)を長押しします。
2. 「携帯電話オプション画面」で、「Power off」(図中、赤枠の項目)を押下します。

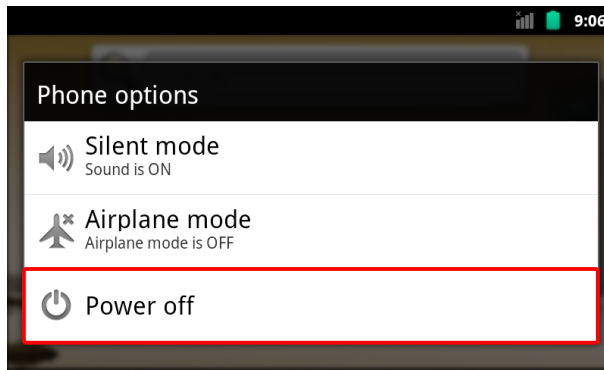


図 6.4 携帯電話オプション画面: Power off

3. 「終了ダイアログ画面」で、「OK ボタン」(図中、赤枠のボタン)を押下します。

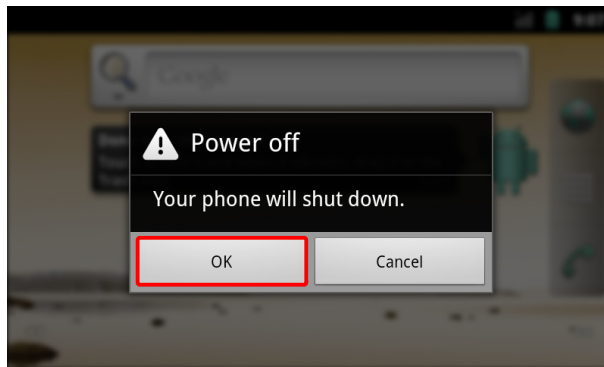


図 6.5 終了ダイアログ画面: OK ボタン

4. 「終了プロセス画面」で、「終了スピナー」(図中、赤枠のスピナー)が停止するのを待ちます。

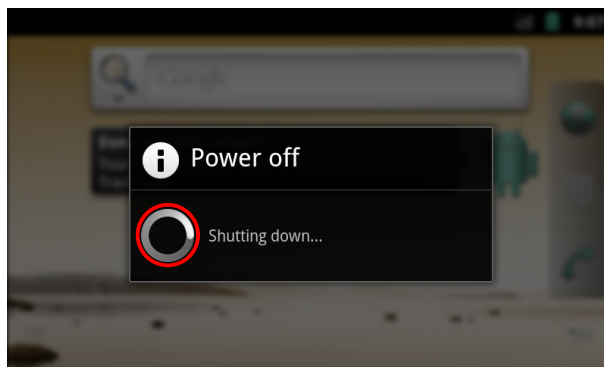


図 6.6 終了プロセス画面: 終了スピナー

5. 電源を切断します。

6.4. 基本操作

Android の基本的な操作方法について説明します。

Android の操作の多くは、タッチパネルとユーザースイッチから行います。ユーザースイッチの名称と主な機能については、「表 6.1. ユーザースイッチの名称と機能」を参照してください。

表 6.1 ユーザースイッチの名称と機能

ユーザースイッチ	名称	主な機能
SW3	電源ボタン	ロック画面に遷移します/安全に終了します
SW4	戻るボタン	1 つ前の画面に戻ります
SW5	メニューボタン	メニューを開きます
SW6	ホームボタン	ホーム画面に遷移します



SW3 は、linux-2.6.35-a800eva-at1 では 決定ボタン に割り当てられていました。linux-2.6.35-a800eva-at2 から、電源ボタンに変更されています。決定ボタンは、フォーカスのあるキーを実行する機能を提供します。

6.4.1. ロック画面

Android の起動直後または、ロック画面以外の任意の画面で「電源ボタン」を押下すると、「図 6.7. ロック画面」が表示されます。

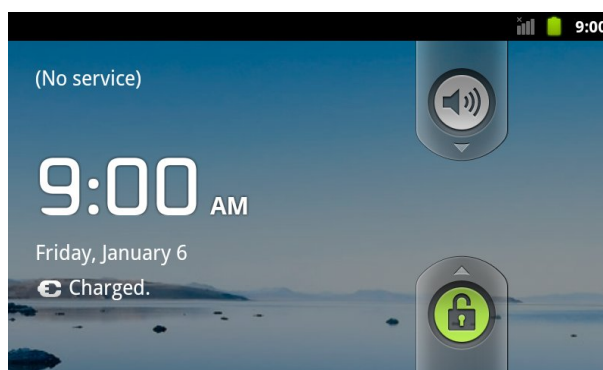


図 6.7 ロック画面

ロックは「メニューボタン」を 2 回押下すると解除できます。Android が起動直後の場合は、「6.4.2. ホーム画面」に示すホーム画面が表示されます。

ロック画面以外の画面が表示されている状態で、何も操作をせずに一定時間が経過すると、ロック画面が表示されます。

6.4.2. ホーム画面

ロック画面以外の任意の画面で「ホームボタン」を押下すると、「図 6.8. ホーム画面」が表示されます。

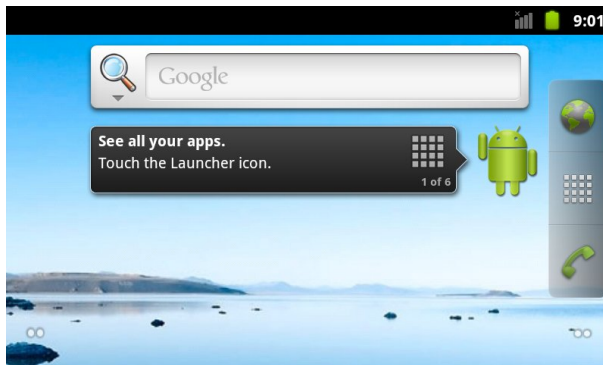


図 6.8 ホーム画面

「図 6.9. ホーム画面: アプリケーション一覧ボタン」で「アプリケーション一覧ボタン」(図中、赤枠のアイコン)を押下すると、「図 6.10. アプリケーション一覧画面」に示す画面が表示されます。

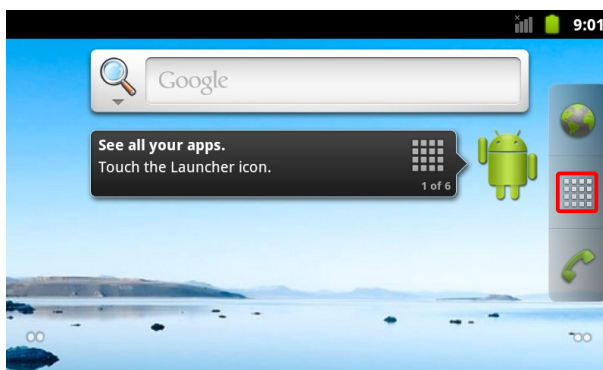


図 6.9 ホーム画面: アプリケーション一覧ボタン

6.4.3. アプリケーション一覧

ホーム画面で「アプリケーション一覧ボタン」を押下すると、「図 6.10. アプリケーション一覧画面」が表示されます。この画面には、インストールされているアプリケーションの一覧が表示されます。

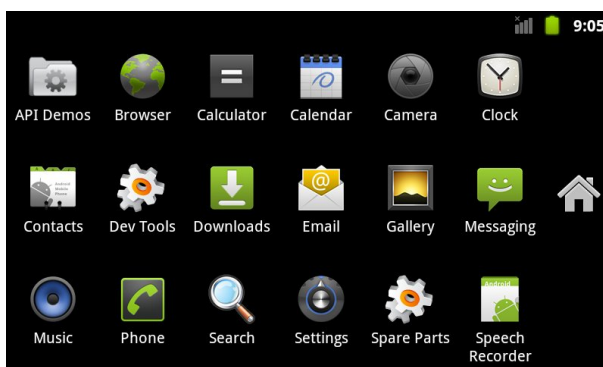


図 6.10 アプリケーション一覧画面

各アプリケーションのアイコンを押下すると、アプリケーションを起動することができます。「API Demos」のアイコンを押下すると、画面描画や音楽再生のデモを行うことができます。



工場出荷状態の Android では、以下のデバイスには対応していません。

- ・ カメラ
- ・ 無線 LAN
- ・ SD カード



Android の ソースコード^[1] は、Armadillo サイト (<http://armadillo.atmark-techno.com>)から取得可能です。

^[1]利用制限のあるソフトウェアは含まれていません。

7. Debian GNU/Linux で起動する

7.1. 準備

Debian GNU/Linux(以降、単に Debian と表記します)を起動させる場合の起動 OS 設定の変更について説明します。

ブートローダーを保守モードで起動し、起動 OS 設定の変更を行うことで起動する OS を変更することができます。Armadillo-800 EVA に電源を投入する前にディップスイッチの起動モード設定(SW1.1)を ON に設定し、Armadillo-800 EVA に電源を投入するとブートローダーが保守モードで起動します。

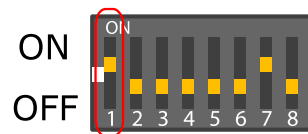


図 7.1 ディップスイッチの起動モード設定

ブートローダーを保守モードで起動すると、ブートローダーのコマンドプロンプトが表示されます。

```
Hermit-At v3.0.0 (Armadillo-800 EVA) compiled at 22:22:10, Dec 21 2011
hermit>
```

Debian を起動するには「図 7.2. 起動設定の変更手順(Debian)」のように起動設定を変更します。

```
hermit> setbootdevice mmcblk0p2 ❶
hermit> setenv console=ttySC1,115200 noinitrd rootwait root=/dev/mmcblk0p2 ❷
```

- ❶ Debian 用の Linux カーネルを格納しているブートデバイスを設定
- ❷ Linux カーネルパラメーターを設定

図 7.2 起動設定の変更手順(Debian)

以上で起動設定の変更は完了です。boot コマンドで Debian を起動することができます。

```
hermit> boot
```



Armadillo-800 EVA の電源を切断し、ディップスイッチの起動モード設定(SW1.1)を OFF に設定すると、電源投入後自動的に Debian が起動するようになります。

7.2. 起動

Debian を起動します。次のように起動ログがシリアル通信ソフトウェアに表示されます。

```
mmcscd: SD card at address 0x00000001
mmcscd: M8G2FA 1048576KiB
gendisk: /dev/mmcblk0p2: start=0x004ade00, size=0x009ba200
gendisk: Image.bin is found. (4390496 Bytes)
Copying      kernel...done.
Doing console=ttySC1,115200
Doing noinitrd
Doing rootwait
Doing root=/dev/mmcblk0p2
Linux version 2.6.35.7 (atmark@atde4) (gcc version 4.4.5 (Debian 4.4.5-8) ) #1 P
REEMPT Wed Dec 21 22:37:47 JST 2011
CPU: ARMv7 Processor [412fc093] revision 3 (ARMv7), cr=10c53c7f
CPU: VIPT nonaliasing data cache, VIPT nonaliasing instruction cache
Machine: Armadillo-800EVA
:
:
:
VFS: Mounted root (ext3 filesystem) on device 179:2.
Freeing init memory: 124K
INIT: version 2.88 booting
Using makefile-style concurrent boot in runlevel S.
Starting the hotplug events dispatcher: udevd.
Synthesizing the initial hotplug events...done.
Waiting for /dev to be fully populated...done.
Activating swap...done.
Checking root file system...fsck from util-linux-ng 2.17.2
/dev/mmcblk0p2: Backing up journal inode block information.

/dev/mmcblk0p2: clean, 11589/637728 files, 88453/1274944 blocks
done.
EXT3-fs (mmcblk0p2): using internal journal
Cleaning up ifupdown...
Setting up networking...
Loading kernel modules...done.
Activating lvm and md swap...done.
Checking file systems...fsck from util-linux-ng 2.17.2
done.
Mounting local filesystems...done.
Activating swapfile swap...done.
Cleaning up temporary files...
Configuring network interfaces...done.
Cleaning up temporary files...
Setting kernel variables ...done.
INIT: Entering runlevel: 2
Using makefile-style concurrent boot in runlevel 2.
Starting enhanced syslogd: rsyslogd.
Starting periodic command scheduler: cron.

Debian GNU/Linux 6.0 debian ttySC1
```

```
debian login:
```

図 7.3 起動ログ(Linux)

7.3. ログイン

起動が完了するとログインプロンプトが表示されます。「表 7.1. シリアルコンソールログイン時のユーザー名とパスワード」に示すユーザーでログインすることができます。

表 7.1 シリアルコンソールログイン時のユーザー名とパスワード

ユーザー名	パスワード	権限
root	root	root ユーザー

7.4. 終了

安全に終了させる場合は、次のように halt コマンドを実行し、「System halted.」と表示されたのを確認してから電源を切断します。

```
[armadillo ~]# halt
Broadcast message from root@debian (ttySC1) (Sat Jan 1 04:25:31 2000):

The system is going down for system halt NOW!
INIT: Switching to runlevel: 0
INIT: Sending processes the TERM signal
Using makefile-style concurrent boot in runlevel 0.
Asking all remaining processes to terminate...done.
All processes ended within 1 seconds....done.
Stopping enhanced syslogd: rsyslogd.
Deconfiguring network interfaces...done.
Cleaning up ifupdown....
Saving the system clock.
Deactivating swap...done.
Will now halt.
System halted.
```

図 7.4 終了方法

8. Debian GNU/Linux で各種機能を使用する

本章では、Debian で Armadillo-800 EVA の機能を使用する方法について記載します。本章に記載のある手順を実行するためには、事前に「7. Debian GNU/Linux で起動する」を参照し、起動およびログインが完了している必要があります。

8.1. 有線 LAN

有線 LAN を使用する方法について記載します。

工場出荷状態の Debian は有線 LAN が無効になっています。 `ifup` コマンドを使用して有線 LAN を有効化する手順を、「図 8.1. ifup コマンドによる有線 LAN の有効化」に示します。

```
[armadillo ~]# ifup eth0 ❶
Internet Systems Consortium DHCP Client 4.1.1-P1
Copyright 2004-2010 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

net eth0: attached phy 0 to driver Generic PHY
Listening on LPF/eth0/00:11:0c:xx:xx:xx
Sending on   LPF/eth0/00:11:0c:xx:xx:xx
Sending on   Socket/fallback
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 3
PHY: 0:00 - Link is Up - 100/Full
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 7
DHCPOFFER from 172.16.0.1
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 172.16.0.1
bound to 172.16.2.237 -- renewal in 33138 seconds.
[armadillo ~]# ifconfig eth0 ❷
eth0      Link encap:Ethernet  HWaddr 00:11:0c:xx:xx:xx
          inet addr:172.16.2.237  Bcast:172.16.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:876 errors:0 dropped:0 overruns:0 frame:0
          TX packets:109 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:316114 (308.7 KiB)  TX bytes:13236 (12.9 KiB)
          Interrupt:142 DMA chan:ff
```

- ❶ 有線 LAN インターフェースを有効化します。工場出荷状態の Debian では、DHCP を利用してネットワーク設定が行われるようになっています。
- ❷ 有線 LAN インターフェースの状態を表示します。

図 8.1 ifup コマンドによる有線 LAN の有効化


`ifup` コマンドを使用して有効化された有線 LAN を再度無効化するには、`ifdown` コマンドを使用します。手順を「図 8.2. ifdown コマンドによる有線 LAN の無効化」に示します。

```
[armadillo ~]# ifdown eth0 ❶
Internet Systems Consortium DHCP Client 4.1.1-P1
Copyright 2004-2010 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/eth0/00:11:0c:xx:xx:xx
Sending on   LPF/eth0/00:11:0c:xx:xx:xx
Sending on   Socket/fallback
DHCPRELEASE on eth0 to 172.16.0.1 port 67
[armadillo ~]# ifconfig eth0 ❷
eth0      Link encap:Ethernet  HWaddr 00:11:0c:xx:xx:xx
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:963 errors:0 dropped:0 overruns:0 frame:0
          TX packets:111 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:332666 (324.8 KiB)  TX bytes:13638 (13.3 KiB)
          Interrupt:142 DMA chan:ff
```

- ❶ 有線 LAN インターフェースを無効化します。
- ❷ 有線 LAN の状態を表示します。

図 8.2 ifdown コマンドによる有線 LAN の無効化



ifup および **ifdown** コマンドは、`/etc/network/interfaces` に記述された動作を行います。工場出荷状態では、DHCP サーバーから IP アドレスを取得するよう設定されています。`interfaces` を変更することで、静的 IP アドレスの設定や、起動時に自動的に有効化することができます。詳しくは `interfaces` のマニュアルページを参照してください。

```
[armadillo ~]# man 5 interfaces
```

8.2. 無線 LAN

Armadillo-800 EVA に搭載されている無線 LAN モジュール Armadillo-WLAN(AWL13)を使用する方法について記載します。AWL13 を使用するためには、Armadillo-800 EVA に電源を投入する前にディップスイッチの SDHI1 設定(SW1.5)を OFF に設定する必要があります。

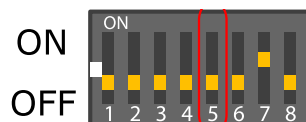



図 8.3 ディップスイッチの SDHI1 設定(AWL13 モジュールインターフェース有効)



CON14(AWL13 モジュールインターフェース)と CON8(SD インターフェース 2)は排他になっており、どちらか一方しか使用することができます。

せん。どちらのインターフェースを使用するかは、ディップスイッチの SDHI1 設定(SW1.5)で選択できるようになっています。

無線通信を行うためには以下の作業が必要となりますが、Armadillo-800 EVA に標準でインストールされている Debian では、udev によって 1.および 2.を自動的に行うように設定することが可能です。

1. カーネルモジュールをロードする
2. ファームウェアをロードする
3. 無線設定を行う

8.2.1. 準備

8.2.1.1. カーネルモジュール/ファームウェアロードの自動化

udev が AWL13 を検出した場合に、自動的にカーネルモジュールをロードできるようにするために、**depmod** コマンドを実行します。

```
[armadillo ~]# ls /lib/modules/$(uname -r)
awl13
[armadillo ~]# depmod
[armadillo ~]# ls /lib/modules/$(uname -r)
awl13                modules.dep          modules.softdep
modules.alias        modules.dep.bin     modules.symbols
modules.alias.bin    modules.devname     modules.symbols.bin
```

再起動を行うと、自動的にカーネルモジュールがロードされるようになります。再起動を行うには、**reboot** コマンドを実行します。

```
[armadillo ~]# reboot
```

カーネルモジュールがロードされると、udev が自動的にファームウェアのロードを行います。起動ログに以下のように表示されることで、カーネルモジュールおよびファームウェアのロードに成功したことが確認できます。

```
awl13: Version 3.0.0 Load.
awl13: MAC is 00:1d:12:xx:xx:xx
```



depmod コマンド実行後、Armadillo-800 EVA を再起動させずに、udev に AWL13 を検出させたい場合は、以下のコマンドを実行してください。

```
[armadillo ~]# udevadm trigger
```




カーネルモジュールがロードされない場合は、SW1(ディップスイッチ)の設定を確認してください。AWL13を使用する場合は、ディップスイッチのSDHI1設定(SW1.5)をOFFに設定する必要があります。

8.2.2. 無線設定

インフラストラクチャモードおよびアドホックモードの無線設定について記載します。より詳細な設定方法については、「Armadillo-WLAN(AWL13) ソフトウェアマニュアル」を参照してください。



有線 LAN インターフェースを使用してネットワークに接続している場合、ネットワーク通信時に AWL13 が使用されない場合があります。確実に無線通信を行いたい場合は、「図 8.2. ifdown コマンドによる有線 LAN の無効化」を参照して有線 LAN インターフェースを無効にしてください。



使用環境によっては、無線 LAN 通信が不安定になる場合があります。そのような環境下では、外付けアンテナを取り付けることで問題が改善する場合があります。外付けアンテナに関する情報は、「Armadillo-WLAN(AWL13) ハードウェアマニュアル」を参照してください。

8.2.2.1. 無線設定パラメーター

無線通信を行うための基本的な設定パラメーターについて記載します。設定には、Wireless Tools^[1]に含まれる `iwconfig/iwpriv` コマンドを使用します。

SSID

アクセスポイントの識別子です。

ID 長 1～32 文字 (デフォルト: "wifi")

設定例 `iwconfig awlan0 essid wifi`

取得例 `iwconfig awlan0`

^[1]標準でインストールされています。

暗号化方式

WPA/WPA2 などの暗号化方式を設定、参照します。

値

設定値	暗号化方式
"none"	暗号化無効
"WEP64"	WEP(キー長: 64bits)
"WEP128"	WEP(キー長: 128bits)
"WPA-TKIP"	WPA-PSK(TKIP)
"WPA-CCMP"	WPA-PSK(AES)
"WPA-AES"	WPA-PSK(AES)
"WPA2-TKIP"	WPA2-PSK(TKIP)
"WPA2-CCMP"	WPA2-PSK(AES)
"WPA2-AES"	WPA2-PSK(AES)
"WPA-MIX"	WPA-PSK(TKIP, AES)
"WPA2-MIX"	WPA-PSK(AES)
"WPA/2-TKIP"	WPA-PSK(TKIP) WPA2-PSK(TKIP)
"WPA/2-CCMP"	WPA-PSK(AES) WPA2-PSK(AES)
"WPA/2-AES"	WPA-PSK(AES) WPA2-PSK(AES)
"WPA/2-MIX"	WPA-PSK(TKIP, AES) WPA2-PSK(TKIP, AES)

設定例 `iwpriv awlan0 set_cryptmode WPA2-AES`

取得例 `iwpriv awlan0 get_cryptmode`

備考 アドホックモードでは、WPA/WPA2 には対応していません。

TKIP や、AES を指定すると、キー生成のために計算が行われるため、コマンド終了までに時間がかかります。

SSID や、事前共有キーを設定した場合にも、キーが再作成されコマンド終了までに時間がかかります。キー生成を 1 回で済ませるために、SSID とパスワードを設定した後に、暗号化方式の設定をすることをお勧めします。

事前共有キー(PSK)

WPA/WPA2 の PSK または、ネットワークパスワードです。

キー長 PSK の場合 64 文字の 16 進数列

パスワードの場合 8~63 文字の文字列

設定例 `iwpriv awlan0 set_psk PreSharedKey` (パスワードの場合)

通信モード

通信形態の種別です。

モード	"Managed"	インフラストラクチャ
	"Ad-Hoc"	アドホック
	"Auto"	無線 LAN 機能オフ (デフォルト)
設定例	iwconfig awlan0 mode <u>Managed</u>	
取得例	iwconfig awlan0	
備考	ファームウェアをロードした直後は、通信モードが Auto に設定されているため、無線機能がオフ状態となっています。無線通信を開始させる前には必ず"Managed"または"Ad-Hoc"に設定してください。	

WEP キー

パケットを暗号化するとき使用する秘密鍵です。

キー長	WEP-64 ビットの場合は 10 文字、WEP-128 ビットの場合は 26 文字の 16 進数文字
設定例	iwconfig awlan0 enc <u>1234567890</u>
取得例	iwconfig awlan0
注意	WEP キーのインデックス番号 1~3 には対応していません。 WEP キーの設定をすると、キー長に応じて 暗号化方式 が適切に設定されます。

チャンネル

使用する周波数帯域です。

チャンネル	1~13 (デフォルト: 11)
設定例	iwconfig awlan0 channel <u>11</u>

8.2.2.2. インフラストラクチャモード設定例

AWL13 をインフラストラクチャモードに設定し、アクセスポイントに接続する設定例を示します。アクセスポイントの暗号化方式が WPA-PSK、または WPA2-PSK の場合は「8.2.2.2.1. インフラストラクチャモード: WPA-PSK/WPA2-PSK」を、WEP の場合は「8.2.2.2.2. インフラストラクチャモード: WEP」を参照してください。

8.2.2.2.1. インフラストラクチャモード: WPA-PSK/WPA2-PSK

WPA-PSK/WPA2-PSK に設定されたアクセスポイントに接続する例として、設定パラメーターを「表 8.1. インフラストラクチャモード: WPA-PSK/WPA2-PSK パラメーター例」に、設定手順を「図 8.4. インフラストラクチャモード: WPA2-PSK(AES)設定手順」に示します。

表 8.1 インフラストラクチャモード: WPA-PSK/WPA2-PSK パラメーター例

項目	設定値
ESSID	myssid
パスワード	my passphrase
暗号化方式	WPA2-PSK(AES)
IP アドレス	192.168.0.1

```
[armadillo ~]# iwconfig wlan0 essid myssid ❶
[armadillo ~]# iwconfig wlan0 mode Managed ❷
[armadillo ~]# iwpriv wlan0 set_psk mypassphrase ❸
[armadillo ~]# iwpriv wlan0 set_cryptmode WPA2-AES ❹
[armadillo ~]# ifconfig wlan0 192.168.0.1 up ❺
```

- ❶ SSID に myssid を設定します。
- ❷ 通信モードをインフラストラクチャモードに設定します。
- ❸ パスワードに mypassphrase を設定します。
- ❹ 暗号化方式に WPA2-PSK(AES) を設定します。
- ❺ 無線 LAN インターフェースを有効化します。IP アドレスは 192.168.0.1 を設定します。

図 8.4 インフラストラクチャモード: WPA2-PSK(AES)設定手順

設定確認手順を「図 8.5. インフラストラクチャモード: WPA2-PSK(AES)設定確認手順」に示します。
iwconfig コマンドの結果で、**Access Point:**に BSSID(アクセスポイントの MAC アドレス)が表示されていれば、アクセスポイントへの接続は完了です。

```
[armadillo ~]# iwconfig awlan0 ❶
awlan0 IEEE 802.11bgn ESSID:"myessid"
Mode:Managed Frequency:2.412 GHz Access Point: XX:XX:XX:XX:XX:XX
Bit Rate=65 Mb/s
Encryption key:off
Power Management:off
Link Signal level=-37 dBm
Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
Tx excessive retries:0 Invalid misc:0 Missed beacon:0

[armadillo ~]# ifconfig awlan0 ❷
awlan0 Link encap:Ethernet HWaddr 00:1d:12:xx:xx:xx
inet addr:192.168.0.1 Bcast:192.168.0.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:47 errors:0 dropped:0 overruns:0 frame:0
TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:6760 (6.6 KiB) TX bytes:1196 (1.1 KiB)
```

- ❶ 無線 LAN 固有の状態を表示します。
- ❷ 無線 LAN インターフェースの状態を表示します。

図 8.5 インフラストラクチャモード: WPA2-PSK(AES)設定確認手順

8.2.2.2.2. インフラストラクチャモード: WEP

WEP に設定されたアクセスポイントに接続する例として、設定パラメーターを「表 8.2. インフラストラクチャモード: WEP パラメーター例」に、設定手順を「図 8.6. インフラストラクチャモード: WEP 設定手順」に示します。

表 8.2 インフラストラクチャモード: WEP パラメーター例

項目	設定値
ESSID	myessid
WEP キー(WEP-64 ビット)	1234567890
IP アドレス	192.168.0.1

```
[armadillo ~]# iwconfig awlan0 essid myessid ❶
[armadillo ~]# iwconfig awlan0 mode Managed ❷
[armadillo ~]# iwconfig awlan0 enc 1234567890 ❸
[armadillo ~]# ifconfig awlan0 192.168.0.1 up ❹
```

- ❶ SSID に myessid を設定します。
- ❷ 通信モードをインフラストラクチャモードに設定します。
- ❸ WEP キーに 1234567890 を設定します。暗号化方式は、キー長に応じて適切に設定されます。
- ❹ 無線 LAN インターフェースを有効化します。IP アドレスは 192.168.0.1 を設定します。

図 8.6 インフラストラクチャモード: WEP 設定手順

設定確認手順を「[図 8.7. インフラストラクチャモード: WEP 設定確認手順](#)」に示します。 `iwconfig` コマンドの結果で、**Access Point:**に BSSID(アクセスポイントの MAC アドレス)が表示されていれば、アクセスポイントへの接続は完了です。

```
[armadillo ~]# iwconfig awlan0 ❶
awlan0 IEEE 802.11bgn ESSID:"myessid"
Mode:Managed Frequency:2.412 GHz Access Point: XX:XX:XX:XX:XX:XX
Bit Rate=65 Mb/s
Encryption key:1234-5678-90
Power Management:off
Link Signal Level=-37 dBm
Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
Tx excessive retries:0 Invalid misc:0 Missed beacon:0


[armadillo ~]# ifconfig awlan0 ❷
awlan0 Link encap:Ethernet HWaddr 00:1d:12:xx:xx:xx
inet addr:192.168.0.1 Bcast:192.168.0.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:47 errors:0 dropped:0 overruns:0 frame:0
TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:6760 (6.6 KiB) TX bytes:1196 (1.1 KiB)
```

- ❶ 無線 LAN 固有の状態を表示します。
- ❷ 無線 LAN インターフェースの状態を表示します。

図 8.7 インフラストラクチャモード: WEP 設定確認手順

8.2.2.3. アドホックモード設定例

AWL13 をアドホックモードに設定し、アドホック端末に接続する設定例を示します。



アドホックモードでは、WPA/WPA2 には対応していません。

8.2.2.3.1. アドホックモード: WEP-128 ビット

WEP に設定されたアドホック端末に接続する例として、設定パラメーターを「[表 8.3. アドホックモード: WEP パラメーター例](#)」に、設定手順を「[図 8.8. アドホックモード: WEP 設定手順](#)」に示します。

表 8.3 アドホックモード: WEP パラメーター例

項目	設定値
ESSID	myessid
WEP キー(WEP-128 ビット)	12345678901234567890123456
チャンネル	1
IP アドレス	192.168.0.1

```
[armadillo ~]# iwconfig wlan0 essid myessid ❶
[armadillo ~]# iwconfig wlan0 mode Ad-Hoc ❷
[armadillo ~]# iwconfig wlan0 enc 12345678901234567890123456 ❸
[armadillo ~]# iwconfig wlan0 channel 1 ❹
[armadillo ~]# ifconfig wlan0 192.168.0.1 up ❺
```

- ❶ SSID に myessid を設定します。
- ❷ 通信モードをアドホックモードに設定します。
- ❸ WEP キーに 12345678901234567890123456 を設定します。暗号化方式は、キー長に応じて適切に設定されます。
- ❹ チャンネルに 1 を設定します。
- ❺ 無線 LAN インターフェースを有効化します。IP アドレスは 192.168.0.1 を設定します。

図 8.8 アドホックモード: WEP 設定手順

設定確認手順を「図 8.9. アドホックモード: WEP 設定確認手順」に示します。

```
[armadillo ~]# iwconfig wlan0 ❶
wlan0 IEEE 802.11bgn ESSID:"myessid"
      Mode:Ad-Hoc Frequency:2.412 GHz Cell: 8E:1F:40:6D:7A:3F
      Bit Rate=0 kb/s
      Encryption key:1234-5678-9012-3456-7890-1234-56
      Power Management:off
      Link Signal level=-37 dBm
      Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
      Tx excessive retries:0 Invalid misc:0 Missed beacon:0

[armadillo ~]# ifconfig wlan0 ❷
wlan0 Link encap:Ethernet HWaddr 00:1d:12:xx:xx:xx
      inet addr:192.168.0.1 Bcast:192.168.0.255 Mask:255.255.255.0
      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
      RX packets:47 errors:0 dropped:0 overruns:0 frame:0
      TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:6760 (6.6 KiB) TX bytes:1196 (1.1 KiB)
```

- ❶ 無線 LAN 固有の状態を表示します。
- ❷ 無線 LAN インターフェースの状態を表示します。

図 8.9 アドホックモード: WEP 設定確認手順

8.3. 時刻設定

時刻を設定する方法について記載します。時刻には、Linux カーネルが管理するシステムクロックと、リアルタイムクロックが管理するハードウェアクロックの 2 種類があります。ハードウェアクロックに時刻を設定すると、電源を切断しても一定時間時刻を保持することができます。

ハードウェアクロックに時刻を設定するためには、まずシステムクロックを設定します。その後に、ハードウェアクロックをシステムクロックと一致させる手順となります。

```
[armadillo ~]# date 010609002012 ❶  
[armadillo ~]# date ❷  
Fri Jan 6 09:00:05 UTC 2012
```

- ❶ システムクロックを 2012 年 1 月 6 日 9 時 00 分^[2]に設定します。
- ❷ システムクロックを確認します。

図 8.10 システムクロックの設定

```
[armadillo ~]# hwclock -wu ❶  
[armadillo ~]# hwclock ❷  
Fri Jan 6 09:00:14 2012 -0.319894 seconds
```

- ❶ ハードウェアクロックにシステムクロックを設定します。
- ❷ ハードウェアクロックを確認します。

図 8.11 ハードウェアクロックの設定

リアルタイムクロックを設定しておく、電源を切断/接続した場合でも Debian の起動時にリアルタイムクロックの値がシステムクロックに設定されるため、システムクロックを再設定する必要がなくなります。電源を長時間切断する場合は、RTC 外部バックアップインターフェース(CON9)に電池を接続する必要があります。

8.4. パッケージ管理

パッケージ管理システム APT(Advanced Packaging Tool)を使用して、パッケージを管理する方法について記載します。工場出荷状態の Debian には動作に必要な最低限のパッケージしかインストールされていませんが、APT を使用することで、簡単にパッケージを追加することができます。

工場出荷状態では、APT はインターネット上の Debian サイト(HTTP サーバー)から利用可能なパッケージのインデックスを取得します^[3]。そのため、APT を使用するためにはネットワークを有効化し、インターネットに接続できる状態にしておく必要があります。

ネットワークを有効化する方法については、「8.1. 有線 LAN」または、「8.2. 無線 LAN」を参照してください。



システムクロックが大幅にずれた状態で、APT を利用すると警告メッセージが出力される場合があります。事前に「8.3. 時刻設定」を参照してシステムクロックを合わせてください。

^[2]フォーマットは、MMDDhhmm[[CC]YY][.ss]です。詳しくは、`date` のマニュアルページを参照してください。

^[3]/etc/apt/sources.list で設定しています。記述ルールなどについては、sources.list のマニュアルページを参照してください。

apt-get update

パッケージインデックスファイルを最新の状態にアップデートします。

引数 無し

使用例

```
[armadillo ~]# apt-get update
```

apt-get upgrade

現在インストールされている全てのパッケージを最新バージョンにアップグレードします。

引数 無し

使用例

```
[armadillo ~]# apt-get upgrade
```

apt-get install [パッケージ名]

引数に指定したパッケージをインストールします。すでにインストール済みの場合はアップグレードします。

引数 パッケージ名(複数指定可能)

使用例

```
[armadillo ~]# apt-get install gcc
```

apt-get remove [パッケージ名]

引数に指定したパッケージをアンインストールします。インストールされていない場合は何もありません。

引数 パッケージ名(複数指定可能)

使用例

```
[armadillo ~]# apt-get remove apache2
```

apt-cache search [キーワード]

引数に指定したキーワードをパッケージ名または説明文に含むパッケージを検索します。

引数 キーワード(正規表現が使用可能)

使用例

```
[armadillo ~]# apt-cache search "Bourne Again SHell"
bash-doc - Documentation and examples for the The GNU Bourne Again SHell
bash-static - The GNU Bourne Again SHell (static version)
bash - The GNU Bourne Again SHell
```

8.5. GStreamer

GStreamer のインストールと使用方法について記載します。

GStreamer はオープンソースのマルチメディアのフレームワークです。動画や音声などを使用するマルチメディアアプリケーションの基礎となる機能を提供します。

GStreamer には、動作確認用のテストツールとして **gst-launch** コマンドが用意されています。**gst-launch** を使用すると、プログラムを作成することなく GStreamer の機能を使用することができます。

以下に示す機能の使用には、**gst-launch** を利用します。

- ・「8.7. ビデオ」
- ・「8.8. カメラ」
- ・「8.10. オーディオ」

8.5.1. GStreamer のインストール

GStreamer のインストールには APT を使用します。APT の使用方法については、「8.4. パッケージ管理」を参照してください。

APT を使用して GStreamer をインストールするには、次のようにコマンドを実行します。

```
[armadillo ~]# apt-get install gstreamer0.10 gstreamer-tools
```

図 8.12 GStreamer のインストール

8.5.2. GStreamer の使用

本書では、GStreamer を **gst-launch** コマンドを使用して利用します。

GStreamer では、特定の機能を持ったエレメントをリンクすることによって、目的とする機能を実現します。**gst-launch** では、エクスクラメーションマーク「!」によってエレメントをリンクします。

エレメントは、マルチメディアデータを入力または出力することのできるパッドを持ちます。エレメントは、エレメントが持つパッドによって 3 種類に分類されます。

表 8.4 エレメントの種類

エレメント	説明
ソースエレメント	出力用のパッドのみを持ちます。
フィルタエレメント	入力用と出力用のパッドをそれぞれ1つ以上持ちます。
シンクエレメント	入力用のパッドのみを持ちます。

ソースエレメントとシンクエレメントのみを持った **gst-launch** の使用例を以下に示します。

```
[armadillo ~]# gst-launch audiotestsrc ! alsasink
```

図 8.13 テスト音声の再生

audiotestsrc は、テスト音声(440Hz の正弦波)を生成するソースエレメントです。alsasink は、ALSA デバイスに音声を入力するシンクエレメントです。この2つのエレメントをリンクすることで、テスト音声を ALSA デバイスに出力します。

Armadillo-800 EVA は、audiotestsrc がデフォルトで生成するテスト音声のフォーマットに対応していません。audiotestsrc のプロパティを指定して、Armadillo-800 EVA のステレオヘッドホン出力 (CON11) に接続したヘッドホンからテスト音声を出力する例を以下に示します。

```
[armadillo ~]# gst-launch -v audiotestsrc ! 'audio/x-raw-int,channels=2,width=16' ! alsasink
```

図 8.14 Armadillo-800 EVA でのテスト音声の再生

GStreamer のより詳細な情報については、GStreamer 公式サイト (<http://gstreamer.freedesktop.org/>) を参照してください。

8.6. X サーバー

X サーバーのインストールと起動方法について記載します。

以下に示す機能の使用には、X サーバーを利用します。

- ・「8.7. ビデオ」
- ・「8.8. カメラ」

8.6.1. X サーバーのインストール

X サーバーのインストールには APT を使用します。APT の使用方法については、「8.4. パッケージ管理」を参照してください。

APT を使用して X サーバーをインストールするには、次のようにコマンドを実行します。

```
[armadillo ~]# apt-get install xserver-xorg-core
```

図 8.15 X サーバーのインストール

8.6.2. X サーバーの起動

X サーバーが使用するビデオ出力を指定して、X サーバーを起動する方法について記載します。Armadillo-800 EVA では、以下のビデオ出力をサポートしています。

ビデオ出力	フレームバッファデバイス	備考
LCD	/dev/fb0	CON17 に接続された Armadillo-800 EVA 付属の液晶パネルです。
ディスプレイ	/dev/fb1	CON3 または CON4 に接続されたディスプレイです。



CON3(デジタル HD 出力インターフェース)と CON4(コンポジットビデオ出力インターフェース)は排他になっており、どちらか一方しか使用することができません。工場出荷状態では、CON3 が使用されます。CON4 を使用する場合は、「8.7.2. ディスプレイのビデオモードを設定する」を参照してビデオモードを **U:720x480i-60** に設定してください。

X サーバーはデフォルトで /dev/fb0 を使用しますが、環境変数 \$FRAMEBUFFER によって使用するフレームバッファデバイスを指定することができます。

X サーバーが使用するビデオ出力として、LCD を使用するには「図 8.16. X サーバーの起動(LCD)」を、ディスプレイを使用するには「図 8.17. X サーバーの起動(ディスプレイ)」のようにコマンドを実行します。

```
[armadillo ~]# X -retro &
```

図 8.16 X サーバーの起動(LCD)

```
[armadillo ~]# FRAMEBUFFER=/dev/fb1 X -retro :1 &
```

図 8.17 X サーバーの起動(ディスプレイ)

X サーバーが起動するとビデオに「図 8.18. X サーバー起動画面」のように起動画面が表示されます。

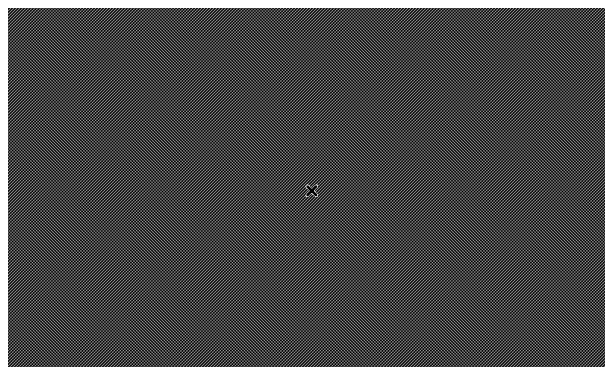




図 8.18 X サーバー起動画面



X サーバーの起動オプション "-retro" を指定する目的は、起動画面によって X サーバーの正常起動を確認しやすくするためです。不要であれば指定する必要はありません。



ディスプレイに表示された X サーバー起動画面のちらつきが気になる場合は、次のようにコマンドを実行してください。

```
[armadillo ~]# FRAMEBUFFER=/dev/fb1 X -wr :1 &
```

X サーバーの起動オプション "-retro" を指定した場合は網目模様の背景が表示されますが、"-wr"を指定した場合は単色白色の背景が表示されます。

8.7. ビデオ

ビデオを使用する方法について記載します。工場出荷状態では、以下のビデオ出力をサポートしています。

ビデオ出力	フレームバッファデバイス	解像度(デフォルト)	備考
LCD	/dev/fb0	800 × 480	CON17 に接続された Armadillo-800 EVA 付属の液晶パネルです。
ディスプレイ	/dev/fb1	1920 × 1080	CON3 または CON4 に接続されたディスプレイです。

8.7.1. ビデオに画像を表示する

ビデオに画像を表示させる方法について記載します。画像の表示には、GStreamer を利用します。インストールされていない場合は、「[図 8.12. GStreamer のインストール](#)」を参照してインストールしてください。

また、GStreamer からビデオへの直接描画はできないため、X サーバーを利用します。事前に「[8.6. X サーバー](#)」を参照して X サーバーを起動してください。

任意の JPEG 画像を LCD に表示する手順を「[図 8.19. JPEG 画像の表示\(LCD\)](#)」に、ディスプレイに表示する手順を「[図 8.20. JPEG 画像の表示\(ディスプレイ\)](#)」に示します。

```
[armadillo ~]# DISPLAY=:0 gst-launch filesrc location=sample.jpg ! jpegdec ! freeze !
ffmpegcolorspace ! autovideosink
```

図 8.19 JPEG 画像の表示(LCD)

```
[armadillo ~]# DISPLAY=:1 gst-launch filesrc location=sample.jpg ! jpegdec ! freeze !
ffmpegcolorspace ! autovideosink
```

図 8.20 JPEG 画像の表示(ディスプレイ)



JPEG ファイルのサンプルには、gnome-backgrounds パッケージに含まれるファイルを利用することができます。JPEG ファイルを LCD の解像度に合わせて表示するコマンド例を以下に示します。

```
[armadillo ~]# apt-get install gnome-backgrounds imagemagick
[armadillo ~]# cp /usr/share/pixmaps/backgrounds/gnome/nature/
YellowFlower.jpg .
[armadillo ~]# convert -geometry 800x480! YellowFlower.jpg sample.jpg
[armadillo ~]# X -retro &
[armadillo ~]# DISPLAY=:0 gst-launch filesrc location=sample.jpg !
jpegdec ! freeze ! ffmpegcolorspace ! autovideosink
```

図 8.21 gnome-backgrounds パッケージの画像ファイルを利用

8.7.2. ディスプレイのビデオモードを設定する

ディスプレイのビデオモードを設定する方法について記載します。



linux-2.6.35-a800eva-at1 では、ビデオモードの設定に対応していません。以下に示す手順を実行しても、設定が正常に反映されません。

設定可能なビデオモードは、以下のように表示することができます。

```
[armadillo ~]# cat /sys/class/graphics/fb1/modes
U:720x480i-60
U:720x480p-60
U:1280x720p-60
U:1920x1080p-24
U:1920x1080p-60
U:1920x1080i-60
```

図 8.22 設定可能なビデオモードの表示(ディスプレイ)



U:1920x1080p-60 はサポートしていません。U:1920x1080p-60 へのビデオモードの設定は、自己責任のもと行ってください。

Armadillo-800 EVA がサポートしているビデオモードを以下に示します。

ビデオモード	解像度	走査方式	フレームレート	出力先インターフェース
U:720x480i-60	720 × 480	インターレース	60	コンポジットビデオ出力(CON4)
U:720x480p-60	720 × 480	プログレッシブ	60	デジタル HD 出力(CON3)
U:1280x720p-60	1280 × 720	プログレッシブ	60	デジタル HD 出力(CON3)
U:1920x1080p-24	1920 × 1080	プログレッシブ	24	デジタル HD 出力(CON3)
U:1920x1080i-60	1920 × 1080	インターレース	60	デジタル HD 出力(CON3)

ディスプレイのビデオモードを **U:1280x720p-60** に設定する例を「[図 8.23. ビデオモードの設定\(ディスプレイ\)](#)」に示します。

```
[armadillo ~]# echo U:1280x720p-60 > /sys/class/graphics/fb1/mode
```

図 8.23 ビデオモードの設定(ディスプレイ)

8.8. カメラ

カメラモジュールインターフェース(CON1)に搭載されたカメラから取得した映像を LCD に表示させる方法について記載します。映像の取得および表示には、GStreamer を利用します。インストールされていない場合は「[図 8.12. GStreamer のインストール](#)」を参照してインストールしてください。

また、GStreamer からビデオへの直接描画はできないため、X サーバーを利用します。事前に「[8.6. X サーバー](#)」を参照して X サーバーを起動してください。

カメラから取得した映像を LCD に出力するには、次のようにコマンドを実行します。アプリケーションを停止するには、Ctrl+c を押下してください。

```
[armadillo ~]# DISPLAY=:0 gst-launch v4l2src ! "video/x-raw-yuv,width=800,height=480,format=(fourcc)NV12" ! ffmpegcolorspace ! autovideosink
```

図 8.24 カメラの映像を LCD に表示



LCD にカメラから取得した映像が表示されない場合は、Armadillo-800 EVA を再起動し、再度「[図 8.24. カメラの映像を LCD に表示](#)」を実行してください。

8.9. USB ガジェット

USB ガジェットの Ethernet 機能を使用する方法について記載します。USB ガジェットを使用するためには、Armadillo-800 EVA に電源を投入する前にディップスイッチの USB0 設定(SW1.6)を ON に設定する必要があります。

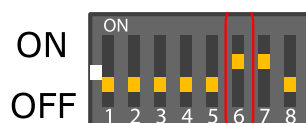


図 8.25 ディップスイッチの USB0 設定(USB インターフェース 3 有効)



CON24(USB デバイスインターフェース)と CON20(USB ホストインターフェース)は排他になっており、どちらか一方しか使用することができません。どちらのインターフェースを使用するかは、ディップスイッチの USB0 設定(SW1.6)で選択できるようになっています。

USB Ethernet ガジェットは、Armadillo-800 EVA と 作業用 PC 間で 1 対 1 の排他的なネットワーク通信機能を提供します。



linux-2.6.35-a800eva-at2 までは USB ガジェットに対応していません。
linux-2.6.35-a800eva-at3 で対応しました。

この機能を利用するためには、Armadillo-800 EVA と作業用 PC の両方でネットワークインターフェースを有効化する必要があります。ネットワークインターフェースを有効化するには ifconfig コマンドを使用します。Armadillo-800 EVA での手順を「[図 8.26. USB Ethernet ガジェット ネットワークインターフェースの有効化\(Armadillo-800 EVA\)](#)」に、作業用 PC での手順を「[図 8.27. USB Ethernet ガジェット ネットワークインターフェースの有効化\(作業用 PC\)](#)」に示します。

```
[armadillo ~]# ifconfig usb0 192.168.10.1 up ①
[armadillo ~]# ifconfig usb0 ②
usb0      Link encap:Ethernet  HWaddr xx:xx:xx:xx:xx:xx
          inet addr:192.168.10.1  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:21760 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2013 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:32593185 (31.0 MiB)  TX bytes:133441 (130.3 KiB)
```


- ① USB Ethernet ガジェットのネットワークインターフェースを有効化します。IP アドレスは 192.168.10.1 を設定します。
- ② USB Ethernet ガジェットのネットワークインターフェースの状態を表示します。

図 8.26 USB Ethernet ガジェット ネットワークインターフェースの有効化(Armadillo-800 EVA)


```
[PC ~]$ sudo ifconfig usb0 192.168.10.2 up ①
[PC ~]$ ifconfig usb0 ②
usb0      Link encap:イーサネット  ハードウェアアドレス xx:xx:xx:xx:xx:xx
          inet アドレス:192.168.10.2  ブロードキャスト:192.168.10.255  マスク:255.255.255.0
          inet6 アドレス: fe80::1c2f:29ff:fe31:565/64  範囲:リンク
          UP BROADCAST RUNNING MULTICAST  MTU:1500  メトリック:1
          RX パケット:2013 エラー:0 損失:0 オーバーラン:0 フレーム:0
          TX パケット:21771 エラー:0 損失:0 オーバーラン:0 キャリア:0
          衝突(Collision):0 TX キュー長:1000
          RX バイト:105259 (102.7 KiB)  TX バイト:32900664 (31.3 MiB)
```


- ❶ USB Ethernet ガジェット ネットワークインターフェースを有効化します。IP アドレスは 192.168.10.2 を設定します。
- ❷ USB Ethernet ガジェット ネットワークインターフェースの状態を表示します。

図 8.27 USB Ethernet ガジェット ネットワークインターフェースの有効化(作業用 PC)



Armadillo-800 EVA および 作業用 PC で他のネットワークインターフェースが有効になっている場合は、それらとは別のネットワークを利用する必要があります。



USB Ethernet ガジェット ネットワークインターフェースの MAC アドレスには、Linux カーネルが生成したランダム値が設定されます。この MAC アドレスのうち、特定の意味を持つ I/G(Individual/Group)および U/L(Universal/Local)ビットについてはランダム値ではなく固定の値が設定されます。I/G ビットは 0(ユニキャストアドレス)に、U/L ビットは 1(ローカルアドレス)に設定されます。

8.10. オーディオ

オーディオの再生・録音を行う方法について記載します。Armadillo-800 EVA のオーディオ機能は、ALSA デバイスとして実装されています。ALSA デバイスとインターフェースの対応を、「表 8.5. ALSA デバイスとインターフェースの対応」に示します。

表 8.5 ALSA デバイスとインターフェースの対応

ALSA デバイス	インターフェース	機能	サンプリング周波数	フォーマット
hw:0	モノラルマイク入力(CON10)	録音	48k, 32k, 16k, 8k Hz	Signed 16/24 bit, Little-endian
	ステレオヘッドホン出力(CON11)	再生		
	ステレオライン出力(CON12,13)			
hw:1	デジタル出力(CON3)	再生	48k Hz	Signed 16 bit, Little-endian

オーディオの再生および録音には GStreamer を使用します。インストールされていない場合は「図 8.12. GStreamer のインストール」を参照してインストールしてください。

オーディオを動作させるためには、各インターフェースに「表 8.6. オーディオインターフェースに接続する機材」に示す機材を接続しておく必要があります。

表 8.6 オーディオインターフェースに接続する機材

インターフェース	機材
モノラルマイク入力(CON10)	マイク
ステレオヘッドホン出力(CON11)	スピーカーまたはヘッドホン
ステレオライン出力(CON12,13)	ディスプレイ(RCA ケーブルで接続)
デジタル出力(CON3)	ディスプレイ(HDMI ケーブルで接続)

音声を録音する手順を「図 8.28. 音声の録音」に示します。サンプリングレートが 48kHz、ステレオの WAV ファイル(sample.wav)を作成します。

```
[armadillo ~]# gst-launch alsasrc ! 'audio/x-raw-int,channels=2,rate=48000,width=16' !
audioreample ! wavenc ! filesink location=sample.wav
[armadillo ~]# file sample.wav
sample.wav: RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, stereo 48000 Hz
```

図 8.28 音声の録音

任意の音声ファイルを再生する手順を「図 8.29. 音声ファイルの再生(hw:0)」および「図 8.30. 音声ファイルの再生(hw:1)」に示します。音声ファイルはカレントディレクトリ以下にあることを想定しています。ここで使用する音声ファイルは、サンプリングレートが 48kHz、ステレオの WAV ファイル (sample.wav) を想定しています。

```
[armadillo ~]# file sample.wav
sample.wav: RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, stereo 48000 Hz
[armadillo ~]# gst-launch filesrc location=sample.wav ! decodebin ! audioconvert ! audioreample !
alsasink
```

図 8.29 音声ファイルの再生(hw:0)

```
[armadillo ~]# file sample.wav
sample.wav: RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit, stereo 48000 Hz
[armadillo ~]# gst-launch filesrc location=sample.wav ! decodebin ! audioconvert ! audioreample !
alsasink device=hw:1
```

図 8.30 音声ファイルの再生(hw:1)

マイクの音声をそのまま hw:0 に出力する手順を「図 8.31. マイク入力を hw:0 に出力」に示します。

```
[armadillo ~]# gst-launch alsasrc ! 'audio/x-raw-int,channels=2,rate=48000,width=16' !
audioreample ! alsasink
```

図 8.31 マイク入力を hw:0 に出力



音声ファイルのサンプルには、gnome-audio パッケージに含まれるファイルを利用することができます。パッケージに含まれる音声ファイル(WAV)は、44.1kHz でサンプリングされているため、48kHz にリサンプリングする必要があります。

```
[armadillo ~]# apt-get install gnome-audio resample
[armadillo ~]# cp /usr/share/sounds/login.wav .
[armadillo ~]# resample -to 48000 login.wav sample.wav
[armadillo ~]# gst-launch filesrc location=sample.wav ! decodebin !
audioconvert ! audioreample ! alsasink
```

図 8.32 gnome-audio パッケージの音声ファイルを利用

8.11. ストレージ

ストレージを使用する方法について記載します。Armadillo-800 EVA では、内蔵ストレージ(eMMC)の他に SD カードや USB メモリなどの外部ストレージを使用することができます。外部ストレージを使用することで、容易に作業用 PC などとデータ転送を行うことができます。


ストレージとして使用可能なデバイスを「表 8.7. ストレージデバイス」に示します。

表 8.7 ストレージデバイス

デバイス	ディスクデバイス	先頭パーティション	備考
eMMC	/dev/mmcblk0	/dev/mmcblk0p1	内蔵ストレージ。容量 8GB。
SD/MMC カード	/dev/mmcblk*[a]	/dev/mmcblk*p1	外部ストレージ。SD1(CON7)または SD2(CON8)に接続。
USB メモリ	/dev/sd*[b]	/dev/sd*1	外部ストレージ。USB1(CON20)または USB2(CON21)に接続。

[a]ストレージが認識された順に、mmcblk1 mmcblk2 となります。

[b]ストレージが認識された順に、sda sdb sdc ... となります。



工場出荷状態の eMMC はいくつかのパーティションに分割されており、それぞれのパーティションに OS やブートローダーなどのソフトウェアが格納されています。eMMC をストレージとして使用する場合は、使用しているソフトウェアが格納されたパーティションへの書き込みを行わないようご注意ください。

SD2(CON8)または USB1(CON20)に接続したストレージを使用するためには、Armadillo-800 EVA に電源を投入する前にディップスイッチを設定する必要があります。SD2(CON8)に接続した SD/MMC カードを使用する場合は SDH1 設定(SW1.5)を ON に、USB1(CON20)に接続した USB メモリを使用する場合は USB0 設定(SW1.6)を OFF に設定してください。

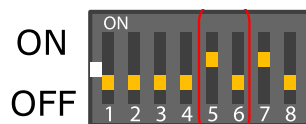




図 8.33 ディップスイッチの SDH1/USB1 設定



CON8(SD インターフェース 2)と CON14(AWL13 モジュールインターフェース)は排他になっており、どちらか一方しか使用することができません。どちらのインターフェースを使用するかは、ディップスイッチの SDH1 設定(SW1.5)で選択できるようになっています。



CON20(USB ホストインターフェース)と CON24(USB デバイスインターフェース)は排他になっており、どちらか一方しか使用することができません。どちらのインターフェースを使用するかは、ディップスイッチの USB0 設定(SW1.6)で選択できるようになっています。


8.11.1. ストレージの使用方法

ここでは、USB メモリを例にストレージの使用方法を説明します。

ストレージを使用可能にするには、ルートファイルシステムにストレージのファイルシステムをマウントする必要があります。マウントを行うコマンドは **mount** です。USB メモリの先頭パーティションに構築された FAT32 ファイルシステムを、ルートファイルシステムの `/mnt/` ディレクトリにマウントするには、次のようにコマンドを実行します。


```
[armadillo ~]# mount -t vfat /dev/sda1 /mnt
```

図 8.34 ストレージのマウント



他のファイルシステムタイプが構築されたストレージのマウント方法など、詳しくは **mount** のマニュアルページを参照してください。

```
[armadillo ~]# man 8 mount
```



Linux カーネルがサポートしているファイルシステムは、以下のように調べることができます。

```
[armadillo ~]# cat /proc/filesystems
nodev  sysfs
nodev  rootfs
:
: (割愛)
:
```

マウントされたストレージを安全に取り外すには、アンマウントする必要があります。アンマウントを行うコマンドは **umount** です。 `/mnt/` ディレクトリにマウントされたストレージをアンマウントするには、次のようにコマンドを実行します。

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

図 8.35 ストレージのアンマウント

8.11.2. ストレージのパーティション変更とフォーマット

ここでは、SD カードを例にストレージのパーティション変更とフォーマット方法を説明します。

パーティション構成を変更するには、**fdisk** コマンドを使用します。**fdisk** コマンドの使用例として、一つのプライマリパーティションで構成されている SD カードのパーティションを、2 つに分割する例を「図 8.36. ストレージのパーティション変更」に示します。一度、既存のパーティションを削除して

から、新たにプライマリパーティションを2つ作成しています。先頭のパーティションには100MByte、二つめのパーティションに残りの容量を割り当てています。

```
[armadillo ~]# fdisk /dev/mmcblk1 ❶
WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
switch off the mode (command 'c') and change display units to
sectors (command 'u').

Command (m for help): d ❷
Selected partition 1

Command (m for help): n ❸
Command action
  e  extended
  p  primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-7182, default 1): ❹
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-7182, default 7182): +100M ❺

Command (m for help): n ❻
Command action
  e  extended
  p  primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (373-7182, default 373): ❼
Using default value 373
Last cylinder, +cylinders or +size{K,M,G} (373-7182, default 7182): ❽
Using default value 7182

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

Calling ioctl() to re-read partition tab mmcblk1:le.
p1 p2
Syncing disks.
```

- ❶ SD カードのパーティションテーブル操作を開始します。複数の SD カードを接続している場合は、SD カードのデバイスファイルが mmcblk2 など本実行例と異なる場合があります。
- ❷ 既存のパーティションを削除します。
- ❸ 新しくプライマリパーティション 1 を作成します。
- ❹ 開始シリンダにはデフォルト値(先頭シリンダ)を使用するので、そのまま改行を入力してください。
- ❺ 最終シリンダは、100MByte 分を指定します。
- ❻ 新しくプライマリパーティション 2 を作成します。
- ❼ 開始シリンダにはデフォルト値(プライマリパーティション 1 直後のシリンダ)を使用するので、そのまま改行を入力してください。
- ❽ 最終シリンダにはデフォルト値(末尾シリンダ)を使用するので、そのまま改行を入力してください。

- ⑨ 変更を SD カードに書き込みます。

図 8.36 ストレージのパーティション変更

SD カードの先頭のパーティションを EXT3 ファイルシステムでフォーマットするには、次のようにコマンドを実行します。

```
[armadillo ~]# mkfs.ext3 /dev/mmcblk1p1
```

図 8.37 ストレージのフォーマット



EXT2 ファイルシステムでフォーマットするには `mkfs.ext2` コマンドを、FAT32 ファイルシステムでフォーマットするには、`dosfstools` パッケージに含まれる `mkfs.vfat` コマンドを使用します。使用方法については、それぞれのマニュアルページを参照してください。

8.12. LED バックライト

LED バックライトの輝度を変更する方法について記載します。Armadillo-800 EVA の LED バックライト機能は、バックライトクラスとして実装されています。LED バックライトの輝度変更には、`/sys/class/backlight/pwm-backlight.0/` ディレクトリ以下の「表 8.8. 輝度設定に使用するファイル」に示すファイルを使用します。

表 8.8 輝度設定に使用するファイル

ファイル	説明
<code>brightness</code>	0(消灯) ~ <code>max_brightness</code> (最高輝度)までの数値を書き込むことで輝度を変更します。
<code>max_brightness</code>	<code>brightness</code> に書きこむ数値の最大値(最高輝度)が読み出せます。

LED バックライトの輝度を変更する手順を「図 8.38. LED バックライトの輝度を変更」に示します。

```
[armadillo ~]# cd /sys/class/backlight/pwm-backlight.0/
[armadillo ~]# cat max_brightness ①
255 ②
[armadillo ~]# echo 0 > brightness ③
[armadillo ~]# cat max_brightness > brightness ④
```

- ① 輝度の最大値を表示します。
- ② `brightness` に設定可能な数値が 0 ~ 255 であることが確認できます。
- ③ 消灯させます。
- ④ 最大輝度で点灯させます。

図 8.38 LED バックライトの輝度を変更



/sys/class/backlight/pwm-backlight.0/以下の他のファイルの使用方法については、Linux カーネルソースファイルに含まれる Documentation/ABI/stable/sysfs-class-backlight を参照してください。

8.13. LED

LED を制御する方法について記載します。Armadillo-800 EVA の LED は、LED クラスとして実装されています。LED の制御は、LED クラスディレクトリ以下のファイルを使用します。制御可能な LED と LED クラスディレクトリの対応を「表 8.9. LED と LED クラスディレクトリの対応」に示します。

表 8.9 LED と LED クラスディレクトリの対応

LED	LED クラスディレクトリ
LED3	/sys/class/leds/LED3/
LED4	/sys/class/leds/LED4/
LED5	/sys/class/leds/LED5/
LED6	/sys/class/leds/LED6/

8.13.1. LED を点灯/消灯する

LED を点灯/消灯する方法について記載します。LED を点灯/消灯させるには、LED クラスディレクトリ以下の brightness に数値を書き込みます。

brightness に 0 を書き込むと消灯し、1 を書き込むと点灯^[4]します。

LED3 を例に、点灯/消灯 および状態取得を行う手順を以下に示します。

```
[armadillo ~]# echo 1 > /sys/class/leds/LED3/brightness
```

図 8.39 LED3 を点灯させる

```
[armadillo ~]# echo 0 > /sys/class/leds/LED3/brightness
```

図 8.40 LED3 を消灯させる

```
[armadillo ~]# cat /sys/class/leds/LED3/brightness
0
```

図 8.41 LED3 の状態を表示する

8.13.2. LED トリガを使用する

LED トリガを使用する方法について記載します。LED クラスディレクトリ以下の trigger に設定したトリガで、LED を点灯/消灯させることができます。trigger に設定可能なトリガを以下に示します。

^[4]Armadillo-800 EVA の LED には輝度制御の機能がないため、1 より大きい数値も点灯動作になります。

表 8.10 trigger に設定可能なトリガ

トリガ	説明
none	トリガを設定しません(デフォルト)。
battery-charging-or-full	ダミーのトリガです。常に消灯します。
battery-charging	ダミーのトリガです。常に消灯します。
battery-full	ダミーのトリガです。常に消灯します。
usb-online	ダミーのトリガです。常に消灯します。
ac-online	ダミーのトリガです。常に消灯します。
mmc0	eMMC のアクセスランプにします。
mmc1	SD1 のアクセスランプにします。
mmc2	SD2 または AWL13 モジュールのアクセスランプにします。
timer	任意のタイミングで点灯/消灯を行います。この設定にすることにより、LED クラスディレクトリ以下に delay_on, delay_off ファイルが出現し、それぞれ点灯時間, 消灯時間をミリ秒単位で指定します。

LED3 を例に、トリガの設定および状態取得を行う手順を以下に示します。

```
[armadillo ~]# echo timer > /sys/class/leds/LED3/trigger
[armadillo ~]# echo 1000 > /sys/class/leds/LED3/delay_on
[armadillo ~]# echo 500 > /sys/class/leds/LED3/delay_off
```

図 8.42 LED3 のトリガに timer を設定する

```
[armadillo ~]# cat /sys/class/leds/LED3/trigger
none battery-charging-or-full battery-charging battery-full usb-online ac-online mmc0 mmc1 mmc2
[timer]
```


図 8.43 LED3 のトリガ表示する

8.14. ユーザースイッチ

ユーザースイッチのイベントを取得する方法について記載します。Armadillo-800 EVA のユーザースイッチはインプットデバイスとして実装されています。ユーザースイッチとして使用可能なスイッチと、インプットデバイスファイルの対応を「表 8.11. スイッチとインプットデバイスファイルの対応」に示します。

表 8.11 スイッチとインプットデバイスファイルの対応

ユーザースイッチ	インプットデバイスファイル	イベントコード
SW3	/dev/input/event0	116 (Power)
SW4		158 (Back)
SW5		139 (Menu)
SW6		102 (Home)



SW3 は、linux-2.6.35-a800eva-at1 では 28 (Enter) に割り当てられていました。linux-2.6.35-a800eva-at2 から、116 (Power)に変更されています。



SW2 はリセットスイッチです。ユーザースイッチとしての使用はできません。誤って押下した場合は、Armadillo-800 EVA がリセットされてしまいますのでご注意ください。



イベントデバイスファイルの番号は、検出された順番に割り振られます。そのため、USB キーボードなど他の入力デバイスが起動時に検出されると、デバイス番号は変わる可能性があります。

イベントを取得するには、**evtest** を使用します。**evtest** のインストールには APT を使用します。APT の使用方法については、「8.4. パッケージ管理」を参照してください。

APT を使用して **evtest** をインストールするには、次のようにコマンドを実行します。

```
[armadillo ~]# apt-get install evtest
```

図 8.44 evtest のインストール

ユーザースイッチのイベントを取得するには、次のようにコマンドを実行します。アプリケーションを停止するには、Ctrl+c を押下してください。

```
[armadillo ~]# evtest /dev/input/event0
Input driver version is 1.0.0
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
Input device name: "gpio-keys"
Supported events:
  Event type 0 (Sync)
  Event type 1 (Key)
    Event code 28 (Enter)
    Event code 102 (Home)
    Event code 139 (Menu)
    Event code 158 (Back)
Testing ... (interrupt to exit)
:
: (イベントを表示)
:
^C
[armadillo ~]#
```

図 8.45 ユーザースイッチのイベントを取得

8.15. タッチパネル

タッチパネルのイベントを取得する方法について記載します。Armadillo-800 EVA のタッチパネルは入力デバイスとして実装されています。タッチパネルに対応する入力デバイスファイルは、`/dev/input/event1` です。



イベントデバイスファイルの番号は、検出された順番に割り振られます。そのため、USB キーボードなど他の入力デバイスが起動時に検出されると、デバイス番号は変わる可能性があります。

イベントの取得には **evtest** を使用します。インストールされていない場合は、「[図 8.44. evtest のインストール](#)」を参照してインストールしてください。

タッチパネルのイベントを取得するには、次のようにコマンドを実行します。アプリケーションを停止するには、Ctrl+c を押下してください。

```
[armadillo ~]# evtest /dev/input/event1
Input driver version is 1.0.0
Input device ID: bus 0x18 vendor 0x0 product 0x0 version 0x0
Input device name: "st1232-touchscreen"
Supported events:
  Event type 0 (Sync)
  Event type 1 (Key)
  Event type 3 (Absolute)
    Event code 48 (?)
      Value      0
      Min        0
      Max       255
    Event code 53 (?)
      Value      0
      Min        0
      Max       799
    Event code 54 (?)
      Value      0
      Min        0
      Max       479
Testing ... (interrupt to exit)
:
: (イベントを表示)
:
^C
[armadillo ~]#
```

図 8.46 タッチパネルのイベントを取得

9. SD ブートの活用

本章では、SD カードから直接起動(以降「SD ブート」と表記します)する手順を示します。SD ブートを活用すると、SD カードを取り替えることでシステムイメージを変更することができます。本章に示す手順を実行するためには、容量が 1GByte 以上の SD カードを必要とします。以下では、例として Debian GNU/Linux 6.0(コードネーム squeeze)を SD ブートする手順を示しますが、他の OS を SD ブートすることも可能です。

9.1. ブートディスクの作成

作業用 PC で、SD ブートを行うための SD カードを作成します。この SD カードをブートディスクと呼びます。本章で作成するブートディスクの構成については「表 9.1. ブートディスクの構成」を参照してください。

表 9.1 ブートディスクの構成

パーティション番号	ファイルシステム	配置するファイル
1	VFAT	ブートローダーイメージファイル
2	ext3	ルートファイルシステムおよびカーネルイメージファイル



ブートローダーイメージファイルをブートディスクに配置する場合、次の制約があります。

- ・ ブートディスクのパーティション 1 に配置すること
- ・ ルートディレクトリ直下に配置すること
- ・ ファイル名が"sdboot.bin"であること
- ・ パーティション ID が、0xb(Win95 FAT32)であること
- ・ ファイルシステムが FAT32 または FAT16(32MByte 以上)であること

本章に示す手順を実行した場合、問題になることはありませんが、独自のブートディスクを作成する場合は注意してください。



SD ブートは、CON7(SD インターフェース 1)にのみ対応しています。CON8 で SD ブートすることはできません。



ブートディスク作成時のコマンドは、ATDE で実行した場合の例です。Armadillo-800 EVA でも同様の手順でブートディスクを作成することができますがデバイスファイル名など一部異なるため、適宜読み替えてください。

9.1.1. ブートディスクの作成に必要なファイルの取得

「表 9.2. ブートディスクの作成に必要なファイル」に示す、ブートディスクの作成に必要なファイルを取得します。これらのファイルは Armadillo サイト (<http://armadillo.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。

表 9.2 ブートディスクの作成に必要なファイル

ファイル	説明
loader-armadillo800eva-mmcsd-v[version].bin	ブートローダーイメージファイル
debian-squeeze_a800eva_[version].tar.gz	Debian GNU/Linux 6.0 ルートファイルシステムアーカイブ
linux-a800eva-[version].bin	カーネルイメージファイル

9.1.2. パーティションの作成

SD カードに 2 つのプライマリパーティションを作成します。

作業用 PC に SD カードを接続して「図 9.1. パーティション作成手順」のようにパーティションを作成します。

```
[PC ~]# fdisk /dev/sdb ❶
WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
switch off the mode (command 'c') and change display units to
sectors (command 'u').

Command (m for help): o ❷
Building a new DOS disklabel with disk identifier 0x65314ac5.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
switch off the mode (command 'c') and change display units to
sectors (command 'u').

Command (m for help): n ❸
Command action
e extended
p primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1790, default 1): ❹
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-1790, default 1790): +128M ❺

Command (m for help): n ❻
Command action
e extended
p primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (62-1790, default 62): ❼
Using default value 62
Last cylinder, +cylinders or +size{K,M,G} (62-1790, default 1790): ❽
```

```
Using default value 1790

Command (m for help): t ⑨
Partition number (1-4): 1
Hex code (type L to list codes): b
Changed system type of partition 1 to b (W95 FAT32)

Command (m for help): w ⑩
The partition table has been altered!

Calling ioctl() to re-read partition table.
sdb: p1 p2
Syncing disks.
[PC ~]#
```

- ① SD カードのパーティションテーブル操作を開始します。USB メモリなどを接続している場合は、SD カードのデバイスファイルが sdc や sdd など本実行例と異なる場合があります。
- ② 新しく DOS パーティションテーブルを作成します。
- ③ 新しくプライマリパーティション 1 を作成します。
- ④ 開始シリンダにはデフォルト値(先頭シリンダ)を使用するので、そのまま改行を入力してください。
- ⑤ 最終シリンダは、128MByte 分を指定します。
- ⑥ 新しくプライマリパーティション 2 を作成します。
- ⑦ 開始シリンダにはデフォルト値(プライマリパーティション 1 直後のシリンダ)を使用するので、そのまま改行を入力してください。
- ⑧ 最終シリンダにはデフォルト値(末尾シリンダ)を使用するので、そのまま改行を入力してください。
- ⑨ プライマリパーティション 1 のパーティションタイプを Win95 FAT32(0x0b)に変更します。
- ⑩ 変更を SD カードに書き込みます。

図 9.1 パーティション作成手順

パーティションが作成されていることを確認します。「図 9.2. パーティション確認手順」のように 2 つのパーティションが作成されていることを確認してください。

```
[PC ~]# fdisk -l /dev/sdb
Disk /dev/mmcblk1: 3983 MB, 3983540224 bytes
106 heads, 41 sectors/track, 1790 cylinders
Units = cylinders of 4346 * 512 = 2225152 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x61e03ae6

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            1           61     132532+    b   W95 FAT32
/dev/sdb2            62          1790     3757117   83   Linux
```

図 9.2 パーティション確認手順

9.1.3. ファイルシステムの構築

「9.1.2. パーティションの作成」で作成したそれぞれのパーティションにファイルシステムを構築します。

作業用 PC に SD カードを接続したまま、「図 9.3. ファイルシステム作成手順」のようにファイルシステムを作成します。



ファイルシステムの構築に `mkfs.vfat` コマンド (`dosfstools` パッケージ) を使用します。作業用 PC にインストールされていない場合は、次のようにインストールを行ってください。

```
[PC ~]# apt-get update
[PC ~]# apt-get install dosfstools
```

```
[PC ~]# mkfs.vfat /dev/sdb1 ❶
mkfs.vfat 3.0.9 (31 Jan 2010)
[PC ~]# mkfs.ext3 /dev/sdb2 ❷
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
235248 inodes, 939279 blocks
46963 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=964689920
29 block groups
32768 blocks per group, 32768 fragments per group
8112 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736

Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 21 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
[PC ~]#
```

- ❶ プライマリパーティション 1 に VFAT ファイルシステムを構築します。
- ❷ プライマリパーティション 2 に ext3 ファイルシステムを構築します。

図 9.3 ファイルシステム作成手順

9.1.4. システムイメージの展開

「9.1.1. ブートディスクの作成に必要なファイルの取得」で取得したファイルを、「9.1.3. ファイルシステムの構築」でファイルシステムを構築した SD カードに展開します。

作業用 PC に SD カードを接続したまま、「図 9.4. システムイメージ展開手順」のようにファイルを展開します。以下のコマンド例では、「9.1.1. ブートディスクの作成に必要なファイルの取得」で取得したファイルはカレントディレクトリ以下にあることを想定しています。

```
[PC ~]# ls
debian-squeeze_a800eva_[version].tar.gz loader-armadillo800eva-mmcsd-v[version].bin
linux-a800eva-[version].bin
[PC ~]# mount -t vfat /dev/sdb1 /mnt
[PC ~]# cp ~/loader-armadillo800eva-mmcsd-v[version].bin /mnt/sdboot.bin ❶
[PC ~]# umount /mnt
[PC ~]# mount -t ext3 /dev/sdb2 /mnt
[PC ~]# tar zxf ~/debian-squeeze_a800eva_[version].tar.gz -C /mnt ❷
[PC ~]# cp ~/linux-a800eva-[version].bin /mnt/boot/Image.bin ❸
[PC ~]# umount /mnt
[PC ~]#
```

- ❶ ブートローダーイメージをブートディスクのパーティション 1 にコピーします。ファイル名は "sdboot.bin" にリネームする必要があります。
- ❷ Debian GNU/Linux 6.0 ルートファイルシステムアーカイブをブートディスクのパーティション 2 に展開します。
- ❸ カーネルイメージファイルをブートディスクのパーティション 2 の "/boot/" 以下にコピーします。ファイル名は、"Image.bin" または "linux.bin" にリネームする必要があります。

図 9.4 システムイメージ展開手順

9.1.5. 設定ファイルの編集

「9.1.4. システムイメージの展開」で展開した Debian GNU/Linux 6.0 ルートファイルシステムは、eMMC パーティション 2 をルートファイルシステムとして使用するよう設定されています。「図 9.5. fstab の編集」に示すように、"/dev/mmcblk0p2" を "/dev/mmcblk1p2" に修正します。

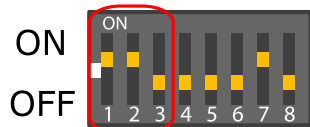
```
[PC ~]# mount -t ext3 /dev/sdb2 /mnt
[PC ~]# vi /mnt/etc/fstab
# <file system> <mount> <type> <options> <dump> <pass>
proc /proc proc defaults 0 0
/dev/mmcblk1p2 / ext3 defaults,errors=remount-ro 0 1
[PC ~]# umount /mnt
[PC ~]#
```

図 9.5 fstab の編集

9.2. ブートディスクから起動する

「9.1. ブートディスクの作成」で作成したブートディスクから起動します。Armadillo-800 EVA に電源を投入する前に以下の準備を行います。

- ・ SD スロット 1(CON7)にブートディスクを接続します。
- ・ ブートディスクのブートローダーから起動するためにディップスイッチの起動デバイス設定をSDHIO(SW1.2 を ON、SW1.3 を OFF)に設定します。
- ・ ブートローダーを保守モードで起動するためにディップスイッチの起動モード設定を(SW1.1)を ON に設定します。



準備の完了後、電源を投入すると保守モードで起動します。「図 9.6. ブートディスクからの起動」を参照して、ブートディスクのカーネルおよびルートファイルシステムを使用するよう設定後、boot コマンドで起動してください。

```
hermit> setbootdevice mmcblk1p2
hermit> setenv console=ttySC1,115200 noinitrd rootwait root=/dev/mmcblk1p2
hermit> boot
```

図 9.6 ブートディスクからの起動

10. リカバリ手順

本章では、Armadillo-800 EVA に搭載された内蔵ストレージ(eMMC)の内容を工場出荷状態に戻す(リカバリする)方法を示します。本章に示す手順を実行するためには、容量が 1GByte 以上の SD カードを必要とします。

10.1. パーティション構成

Armadillo-800 EVA の内蔵ストレージのパーティション構成を示します。Armadillo-800 EVA の内蔵ストレージは、PC 同様にいくつかのパーティションに分割されています。そのため複数の OS をインストールできるようになっています。

工場出荷状態の内蔵ストレージのパーティション構成を「表 10.1. 内蔵ストレージのパーティション構成」に示します。

表 10.1 内蔵ストレージのパーティション構成

デバイスファイル名	サイズ	パーティションタイプ
/dev/mmcblkOp1	512MByte	0b (Win95 FAT32)
/dev/mmcblkOp2	約 5GByte	83 (Linux)
/dev/mmcblkOp3	1GByte	83 (Linux)
/dev/mmcblkOp4	1GByte	83 (Linux)

現行規格である MMC 4.3 では、eMMC デバイスに最大 2 つのブートパーティションを持つことが出来ます。Armadillo-800 EVA に搭載している eMMC には、2 つのブートパーティションが用意されています。

表 10.2 eMMC のブートパーティション

デバイスファイル名	サイズ
/dev/mmcblk0boot0	約 512kByte
/dev/mmcblk0boot1	約 512kByte

「表 10.1. 内蔵ストレージのパーティション構成」および「表 10.2. eMMC のブートパーティション」に示す通り、内蔵ストレージを 6 つの領域に分けて使用しています。それぞれの工場出荷状態での用途を「表 10.3. 内蔵ストレージ各領域の用途」に示します。

表 10.3 内蔵ストレージ各領域の用途

パーティション	用途
パーティション 1 (/dev/mmcblkOp1)	主に Android の SD カード領域に使用します。VFAT にフォーマットされています。
パーティション 2 (/dev/mmcblkOp2)	Debian GNU/Linux がインストールされています。/boot/直下にはカーネルイメージが格納されています。ext3 にフォーマットされています。
パーティション 3 (/dev/mmcblkOp3)	予約領域 ^[a] です。ext3 にフォーマットされています。
パーティション 4 (/dev/mmcblkOp4)	Android がインストールされています。/boot/直下にはカーネルイメージが格納されています。ext3 にフォーマットされています。
ブートパーティション 1 (/dev/mmcblk0boot0)	ブートローダーイメージが格納されています。

パーティション	用途
ブートパーティション 2 (/dev/mmcbk0boot1)	ブートローダーが使用するパラメーターを保存しています。

[a]今後のアップデートで該当パーティション用のシステムイメージを公開する予定です。

10.2. eMMC 全体をリカバリする

eMMC 全体をリカバリする方法を示します。

10.2.1. リカバリディスクの作成

作業用 PC で、リカバリを行うための SD カードを作成します。この SD カードをリカバリディスクと呼びます。



リカバリディスク作成時のコマンドは、ATDE で実行した場合の例です。Armadillo-800 EVA でも同様の手順でリカバリディスクを作成することができますがデバイスファイル名など一部異なるため、適宜読み替えてください。

10.2.1.1. リカバリディスクの作成に必要なファイルの取得

「表 10.4. リカバリディスクの作成に必要なファイル」に示す、リカバリディスクの作成に必要なファイルを取得します。これらのファイルはアットマークテクノ ユーザーズサイト (<https://users.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。



アットマークテクノ ユーザーズサイトからファイルを取得するためには、製品本体をご購入の上で「アットマークテクノ ユーザーズサイト」から「購入製品登録」を行う必要があります。

表 10.4 リカバリディスクの作成に必要なファイル

ファイル	説明
recovery-image_a800eva_[version].tar.gz	eMMC に書き込むためのイメージファイルなどが格納されています
recovery-system_a800eva_[version].tar.gz	リカバリを実行するプログラムなどが格納されています

10.2.1.2. パーティションの作成

SD カードに 2 つのプライマリパーティションを作成します。

作業用 PC に SD カードを接続して「図 10.1. パーティション作成手順」のようにパーティションを作成します。

```
[PC ~]# fdisk /dev/sdb ①
```

```
WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
switch off the mode (command 'c') and change display units to
sectors (command 'u').
```

```

Command (m for help): o ②
Building a new DOS disklabel with disk identifier 0x65314ac5.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
switch off the mode (command 'c') and change display units to
sectors (command 'u').

Command (m for help): n ③
Command action
e extended
p primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1790, default 1): ④
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-1790, default 1790): +512M ⑤

Command (m for help): n ⑥
Command action
e extended
p primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (243-1790, default 243): ⑦
Using default value 243
Last cylinder, +cylinders or +size{K,M,G} (243-1790, default 1790): ⑧
Using default value 1790

Command (m for help): t ⑨
Partition number (1-4): 1
Hex code (type L to list codes): b
Changed system type of partition 1 to b (W95 FAT32)

Command (m for help): w ⑩
The partition table has been altered!

Calling ioctl() to re-read partition table.
sdb: p1 p2
Syncing disks.
[PC ~]#

```

- ① SD カードのパーティショニングを開始します。USB メモリなどを接続している場合は、SD カードのデバイスファイルが sdc や sdd など本実行例と異なる場合があります。
- ② 新しく DOS パーティションテーブルを作成します。
- ③ 新しくプライマリパーティション 1 を作成します。
- ④ 開始シリンダにはデフォルト値(先頭シリンダ)を使用するので、そのまま改行を入力してください。
- ⑤ 最終シリンダは、512MByte 分を指定します。

- ⑥ 新しくプライマリパーティション 2 を作成します。
- ⑦ 開始シリンダにはデフォルト値(プライマリパーティション 1 直後のシリンダ)を使用するので、そのまま改行を入力してください。
- ⑧ 最終シリンダにはデフォルト値(末尾シリンダ)を使用するので、そのまま改行を入力してください。
- ⑨ プライマリパーティション 1 のパーティションタイプを Win95 FAT32(0x0b)に変更します。
- ⑩ 変更を SD カードに書き込みます。

図 10.1 パーティション作成手順

パーティションが作成されていることを確認します。「図 10.2. パーティション確認手順」のように 2 つのパーティションが作成されていることを確認してください。

```
[PC ~]# fdisk -l /dev/sdb
Disk /dev/sdb: 3983 MB, 3983540224 bytes
106 heads, 41 sectors/track, 1790 cylinders
Units = cylinders of 4346 * 512 = 2225152 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x65314ac5
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sdb1		1	242	525845+	b	W95 FAT32
/dev/sdb2		243	1790	3363804	83	Linux

図 10.2 パーティション確認手順

10.2.1.3. ファイルシステムの構築

「10.2.1.2. パーティションの作成」で作成したそれぞれのパーティションにファイルシステムを構築します。

作業用 PC に SD カードを接続したまま、「図 10.3. ファイルシステム作成手順」のようにファイルシステムを作成します。



ファイルシステムの構築に dosfstools を使用します。作業用 PC にインストールされていない場合は、次のようにインストールを行ってください。

```
[PC ~]# apt-get update
[PC ~]# apt-get install dosfstools
```

```
[PC ~]# mkfs.vfat /dev/sdb1 ①
mkfs.vfat 3.0.9 (31 Jan 2010)
[PC ~]# mkfs.ext3 /dev/sdb2 ②
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
```

```
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
210496 inodes, 840951 blocks
42047 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=864026624
26 block groups
32768 blocks per group, 32768 fragments per group
8096 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200

Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 27 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
[PC ~]#
```

- ① プライマリパーティション 1 に VFAT ファイルシステムを構築します。
- ② プライマリパーティション 2 に ext3 ファイルシステムを構築します。

図 10.3 ファイルシステム作成手順

10.2.1.4. リカバリイメージの展開

「10.2.1.1. リカバリディスクの作成に必要なファイルの取得」で取得したファイルを、「10.2.1.3. ファイルシステムの構築」でファイルシステムを構築した SD カードに展開します。

作業用 PC に SD カードを接続したまま、「図 10.4. ファイル展開手順」のようにファイルを展開します。以下のコマンド例では、「10.2.1.1. リカバリディスクの作成に必要なファイルの取得」で取得したファイルはカレントディレクトリ以下にあることを想定しています。

```
[PC ~]# ls
recovery-image_a800eva_[version].tar.gz  recovery-system_a800eva_[version].tar.gz
[PC ~]# mount -t vfat /dev/sdb1 /mnt
[PC ~]# cd /mnt
[PC /mnt]# tar zxf ~/recovery-image_a800eva_[version].tar.gz
[PC /mnt]# cd
[PC ~]# umount /mnt

[PC ~]# mount -t ext3 /dev/sdb2 /mnt
[PC ~]# cd /mnt
[PC /mnt]# tar zxf ~/recovery-system_a800eva_[version].tar.gz
[PC /mnt]# cd
[PC ~]# umount /mnt
[PC ~]#
```

図 10.4 ファイル展開手順

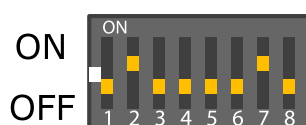
10.2.2. リカバリを実行する



以下に示す手順を実行すると、Armadillo-800 EVA に搭載された eMMC の全ての内容が工場出荷状態になります。一部のパーティションのみを工場出荷状態に戻したい場合は、「10.3. eMMC の特定ルートファイルシステムをリカバリする」または、「10.4. eMMC のブートローダーをリカバリする」を参照してください。

「10.2.1. リカバリディスクの作成」で作成したリカバリディスクを使用して、リカバリを実行します。Armadillo-800 EVA に電源を投入する前に以下の準備を行います。

- ・ ディップスイッチの SW1.2 と SW1.7 を ON に、その他を全て OFF に設定します。



- ・ SD スロット 1(CON7)にリカバリディスクを接続します。

準備の完了後、SW3 を押しながら電源を投入するとリカバリが開始されます。リカバリの進捗具合は、LED の点灯パターンで確認できます。

表 10.5 リカバリ進捗と LED の対応

LED3	LED4	LED5	LED6	リカバリ進捗
点滅	消灯	消灯	消灯	eMMC のユーザーパーティションをフォーマット中
消灯	点滅	消灯	消灯	eMMC のブートパーティションをフォーマット中
点滅	点灯	点灯	点灯	eMMC のパーティション 1 にシステムを構築中
点灯	点滅	点灯	点灯	eMMC のパーティション 2 にシステムを構築中
点灯	点灯	点滅	点灯	eMMC のパーティション 3 にシステムを構築中
点灯	点灯	点灯	点滅	eMMC のパーティション 4 にシステムを構築中
点灯	点灯	点灯	点灯	リカバリが正常に完了
点滅	点滅	点滅	点滅	リカバリが異常終了

全ての LED が点灯するとリカバリは完了です。全ての LED が点滅した場合、エラーが発生したことを示しています。手順を見直して再度リカバリを実行してください。リカバリ後、eMMC から起動するには、ディップスイッチの起動デバイス設定を eMMC(SW1.2 を OFF、SW1.3 を OFF)に設定し、SD カードを抜いた後電源を投入してください。

10.3. eMMC の特定ルートファイルシステムをリカバリする

eMMC の特定ルートファイルシステムのみをリカバリする方法を紹介します。「表 10.6. リカバリ対象のルートファイルシステム」をリカバリの対象とします。

表 10.6 リカバリ対象のルートファイルシステム

ルートファイルシステム	パーティション番号	ファイルシステム
Debian GNU/Linux	2	ext3
Android	4	ext3

事前に「9. SD ブートの活用」を参照して、Armadillo-800 EVA が SD ブートしている必要があります。

10.3.1. Debian GNU/Linux をリカバリする

eMMC のパーティション 2 に、Debian GNU/Linux をリカバリする方法を示します。

10.3.1.1. Debian GNU/Linux のリカバリに必要なファイルの取得

「表 10.7. Debian GNU/Linux のリカバリに必要なファイル」に示す、Debian GNU/Linux のリカバリに必要なファイルを取得します。これらのファイルは Armadillo サイト (<http://armadillo.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。

表 10.7 Debian GNU/Linux のリカバリに必要なファイル

ファイル	説明
debian-squeeze_a800eva_[version].tar.gz	Debian GNU/Linux 6.0 のルートファイルシステムアーカイブ
linux-a800eva-[version].bin	カーネルイメージファイル

10.3.1.2. Debian GNU/Linux のリカバリを実行する

「図 10.5. Debian GNU/Linux のリカバリ手順」のように Debian GNU/Linux のリカバリを実行します。「10.3.1.1. Debian GNU/Linux のリカバリに必要なファイルの取得」で取得したファイルはカレントディレクトリ以下にあることを想定しています。

```
[armadillo ~]# ls
debian-squeeze_a800eva_[version].tar.gz  linux-a800eva-[version].bin
[armadillo ~]# mkfs.ext3 /dev/mmcblk0p2 ❶
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
318864 inodes, 1274944 blocks
63747 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1308622848
39 block groups
32768 blocks per group, 32768 fragments per group
8176 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 24 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
[armadillo ~]# mount -t ext3 /dev/mmcblk0p2 /mnt ❷
[armadillo ~]# tar xzf ~/debian-squeeze_a800eva_[version].tar.gz -C /mnt ❸
[armadillo ~]# cp ~/linux-a800eva-[version].bin /mnt/boot/Image.bin ❹
[armadillo ~]# umount /mnt
[armadillo ~]#
```

- ❶ ext3 ファイルシステムを構築します。

- ② マウントします。
- ③ ルートファイルシステムアーカイブを展開します。
- ④ カーネルイメージファイルを/mnt/boot/以下にコピーします。ファイル名は"Image.bin"または"linux.bin"にリネームする必要があります。

図 10.5 Debian GNU/Linux のリカバリ手順

10.3.2. Android をリカバリする

eMMC のパーティション 4 に、Android をリカバリする方法を示します。

10.3.2.1. Android のリカバリに必要なファイルの取得

「表 10.8. Android のリカバリに必要なファイル」に示す、Android のリカバリに必要なファイルを取得します。Android のルートファイルシステムアーカイブはアットマークテクノ ユーザーズサイト (<https://users.atmark-techno.com>) または付属 DVD-ROM から、カーネルイメージファイルは Armadillo サイト (<http://armadillo.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。



アットマークテクノ ユーザーズサイトからファイルを取得するためには、製品本体をご購入の上で「アットマークテクノ ユーザーズサイト」から「購入製品登録」を行う必要があります。

表 10.8 Android のリカバリに必要なファイル

ファイル	説明
android-2.3.7_a800eva_[version].tar.gz	Android 2.3.7 のルートファイルシステムアーカイブ
linux-a800eva-[version].bin	カーネルイメージファイル

10.3.2.2. Android のリカバリを実行する

「図 10.6. Android のリカバリ手順」のように Android のリカバリを実行します。「10.3.2.1. Android のリカバリに必要なファイルの取得」で取得したファイルはカレントディレクトリ以下にあることを想定しています。

```
[armadillo ~]# ls
android-2.3.7_a800eva_[version].tar.gz  linux-a800eva-[version].bin
[armadillo ~]# mkfs.ext3 /dev/mmcblk0p4 ①
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
61056 inodes, 244152 blocks
12207 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=251658240
8 block groups
32768 blocks per group, 32768 fragments per group
7632 inodes per group
```



```

Superblock backups stored on blocks:
    32768, 98304, 163840, 229376

Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 23 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.
[armadillo ~]# mount -t ext3 /dev/mmcblk0p4 /mnt ②
[armadillo ~]# tar xzf ~/android-2.3.7_a800eva_[version].tar.gz -C /mnt ③
[armadillo ~]# mkdir /mnt/boot
[armadillo ~]# cp ~/linux-a800eva-[version].bin /mnt/boot/Image.bin ④
[armadillo ~]# umount /mnt
[armadillo ~]#

```

- ① ext3 ファイルシステムを構築します。
- ② マウントします。
- ③ ルートファイルシステムアーカイブを展開します。
- ④ カーネルイメージファイルを/mnt/boot/以下にコピーします。ファイル名は"Image.bin"または"linux.bin"にリネームする必要があります。

図 10.6 Android のリカバリ手順

10.4. eMMC のブートローダーをリカバリする

eMMC のブートパーティション 0 に、ブートローダーをリカバリする方法を示します。事前に「9. SD ブートの活用」を参照して、Armadillo-800 EVA が SD ブートしている必要があります。

10.4.1. ブートローダーのリカバリに必要なファイルの取得

「表 10.9. ブートローダーのリカバリに必要なファイル」に示す、ブートローダーのリカバリに必要なファイルを取得します。これらのファイルは Armadillo サイト (<http://armadillo.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。

表 10.9 ブートローダーのリカバリに必要なファイル

ファイル	説明
loader-armadillo800eva-mmcsd-v[version].bin	ブートローダーイメージファイル

10.4.2. ブートローダーのリカバリを実行する

「図 10.7. ブートローダーのリカバリ手順」のようにブートローダーのリカバリを実行します。「10.4.1. ブートローダーのリカバリに必要なファイルの取得」で取得したファイルはカレントディレクトリ以下にあることを想定しています。

```

[armadillo ~]# ls
loader-armadillo800eva-mmcsd-v[version].bin
[armadillo ~]# echo 0 > /sys/block/mmcblk0boot0/force_ro ①
[armadillo ~]# cat ~/loader-armadillo800eva-mmcsd-v[version].bin > /dev/mmcblk0boot0 ②

```

```
[armadillo ~]# sync  
[armadillo ~]#
```

- ① ブートパーティションのソフトライトプロテクトを解除します。
- ② ブートローダーイメージを書き込みます。

図 10.7 ブートローダーのリカバリ手順

11. 開発環境の準備

本章では、Armadillo-800 EVA のソフトウェア開発を行うための開発環境を作業用 PC に構築する方法について説明します。

作業用 PC が Debian GNU/Linux 6.0(コードネーム squeeze)^[1]の場合、クロス開発ツールをインストールすることで Armadillo-800 EVA のソフトウェア開発環境を構築することができます。クロス開発ツールをインストールする方法は、「11.2. クロス開発ツールのインストール方法」を参照してください。

作業用 PC が Windows または、Debian GNU/Linux 6.0(コードネーム squeeze)以外の Linux の場合、「ATDE(Atmark Techno Development Environment)」を使用することで Armadillo-800 EVA のソフトウェア開発環境を構築することができます。ATDE を使用する方法は、「11.1. ATDE の使用方法」を参照してください。

11.1. ATDE の使用方法

作業用 PC で、ATDE を使用する方法について記載します。

ATDE は、Armadillo の開発環境を提供する VMware 仮想マシンのデータイメージです。Armadillo-800 EVA のソフトウェア開発が可能な ATDE4 は、Debian GNU/Linux 6.0(コードネーム squeeze)をベースに、クロス開発ツールやその他の必要なツールが事前にインストールされています。

VMware の取得や、インストール方法については、VMware 社 Web ページ (<http://www.vmware.com/>)を参照してください。



VMware は、非商用利用限定で無償のものから、商用利用可能な有償のものまで複数の製品があります。製品ごとに異なるライセンス、エンドユーザー使用許諾契約書(EULA)が存在するため、十分に確認した上で利用目的に合う製品をご利用ください。

11.1.1. ATDE の取得

「表 11.1. ATDE4 の種類」に示す ATDE4 のアーカイブは Armadillo サイト (<http://armadillo.atmark-techno.com>)または、付属 DVD-ROM から取得可能です。

表 11.1 ATDE4 の種類

ATDE4 アーカイブ	ベースの Debian GNU/Linux
atde4-[version]-amd64.zip	64-bit PC(「amd64」)アーキテクチャ用 Debian GNU/Linux 6.0
atde4-[version]-i386.zip	32-bit PC(「i386」)アーキテクチャ用 Debian GNU/Linux 6.0

^[1]他の 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。

11.1.2. ATDE の起動

ATDE4 のアーカイブを展開したディレクトリに存在する `atde4.vmx` が、ATDE4 の仮想マシン構成ファイルです。このファイルを VMware 上で開くと、ATDE4 を起動することができます。ATDE4 にログイン可能なユーザーを、「表 11.2. ユーザー名とパスワード」に示します^[2]。

表 11.2 ユーザー名とパスワード

ユーザー名	パスワード	権限
<code>atmark</code>	<code>atmark</code>	一般ユーザー
<code>root</code>	<code>root</code>	特権ユーザー



以下の VMware で、ATDE4 が起動できることを確認しています。

- ・ VMware Player 3.1.5
- ・ VMware Workstation 7.1.5
- ・ VMware Workstation 8.0.1



作業用 PC の動作環境(ハードウェア、VMware、ATDE4 の種類など)により、ATDE4 が正常に動作しない可能性があります。VMware 社 Web ページ(<http://www.vmware.com/>)から、使用している VMware のドキュメントなどを参照して動作環境を確認してください。

11.2. クロス開発ツールのインストール方法

Debian GNU/Linux 6.0(コードネーム squeeze)が動作している作業用 PC に、armel アーキテクチャ用のクロス開発ツールを、Debian パッケージでインストールする方法について記載します。

11.2.1. クロス開発ツールが依存する Debian パッケージのインストール

クロス開発ツールが依存する Debian パッケージをインストールします。「図 11.1. クロス開発ツールが依存する Debian パッケージインストールコマンド」のようにインストールを行ってください。

```
[PC ~]$ sudo apt-get update
[PC ~]$ sudo apt-get install binutils gdbserver libgmp3c2 libmpfr4 libpython2.6
```

図 11.1 クロス開発ツールが依存する Debian パッケージインストールコマンド

11.2.2. クロス開発ツール Debian パッケージの取得

クロス開発ツールの Debian パッケージは Armadillo サイト(<http://armadillo.atmark-techno.com>)または、付属 DVD-ROM から取得可能です。

^[2]特権ユーザーで GUI ログインを行うことはできません。

11.2.3. クロス開発ツール Debian パッケージのインストール

作業用 PC に、Debian パッケージをインストールします。以下では、「11.2.2. クロス開発ツール Debian パッケージの取得」で取得したクロス開発ツールの Debian パッケージが全て~/に配置されている場合を前提としています。

64-bit PC(「amd64」)アーキテクチャ用 Debian GNU/Linux 6.0 を使用している場合は「図 11.2. 64-bit PC 用クロス開発ツール Debian パッケージインストールコマンド」を、32-bit PC(「i386」)アーキテクチャ用 Debian GNU/Linux 6.0 を使用している場合は「図 11.3. 32-bit PC 用クロス開発ツール Debian パッケージインストールコマンド」を参照してください。

```
[PC ~]$ sudo dpkg --install *_amd64.deb *_all.deb
```

図 11.2 64-bit PC 用クロス開発ツール Debian パッケージインストールコマンド

```
[PC ~]$ sudo dpkg --install *_i386.deb *_all.deb
```

図 11.3 32-bit PC 用クロス開発ツール Debian パッケージインストールコマンド



作業用 PC に既に同一ターゲットのアーキテクチャ用クロス開発環境がインストールされている場合、クロス開発環境をインストールする前に必ず既存のクロス開発環境をアンインストールするようにしてください。

12. カーネルのビルド

本章では、Linux カーネルのソースコードを作業用 PC でクロスビルドし、カーネルイメージファイルを作成する方法について記載します。

事前に「11. 開発環境の準備」を参照して、作業用 PC に開発環境が構築されている必要があります。

12.1. ソースアーカイブの取得

Linux カーネルのソースアーカイブ `linux-2.6.35-a800eva-[version].tar.gz` を取得します。ソースアーカイブは Armadillo サイト (<http://armadillo.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。

12.2. ソースアーカイブの展開

取得したソースアーカイブを展開します。「図 12.1. ソースアーカイブの展開」のように作業してください。

```
[PC ~]# ls
linux-2.6.35-a800eva-[version].tar.gz
[PC ~]# tar zxvf linux-2.6.35-a800eva-[version].tar.gz
linux-2.6.35-a800eva-[version] linux-2.6.35-a800eva-[version].tar.gz
```

図 12.1 ソースアーカイブの展開

12.3. ビルド

展開したソースアーカイブをビルドして、カーネルイメージファイルを作成します。「図 12.2. カーネルのビルド」のように作業してください。

```
❶ [PC ~]$ cd linux-2.6.35-a800eva-[version]
❷ [PC ~/linux-2.6.35-a800eva-[version]]$ make ARCH=arm armadillo800eva_android_defconfig
❸ [PC ~/linux-2.6.35-a800eva-[version]]$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi-
❹ [PC ~/linux-2.6.35-a800eva-[version]]$ ls arch/arm/boot/Image
Image
```

- ❶ ソースコードディレクトリに移動します
- ❷ デフォルトコンフィギュレーションを適用します
- ❸ ソースコードをビルドします
- ❹ カーネルイメージファイルが作成できたことを確認します

図 12.2 カーネルのビルド

12.4. インストール

カーネルイメージを、内蔵ストレージにインストールします。事前に「9. SD ブートの活用」を参照して、Debian GNU/Linux を SD ブートしている必要があります。

「12.3. ビルド」で作成したカーネルイメージファイルはカレントディレクトリ以下にあることを想定しています。

Android システムにインストールする場合は「[図 12.3. Android システムへのカーネルイメージのインストール](#)」を、Debian GNU/Linux システムにインストールする場合は「[図 12.4. Debian GNU/Linux システムへのカーネルイメージのインストール](#)」を参照してください。

```
[armadillo ~]# ls
Image
❶ [armadillo ~]# mount /dev/mmcblk0p4 /mnt
❷ [armadillo ~]# rm -f /mnt/boot/*.bin
❸ [armadillo ~]# cp Image /mnt/boot/Image.bin
[armadillo ~]# umount /mnt
```

- ❶ eMMC パーティション 4 をマウントします。
- ❷ eMMC パーティション 4 の/boot/以下のカーネルイメージファイルを削除します。
- ❸ eMMC パーティション 4 の/boot/以下にカーネルイメージファイルをコピーします。ファイル名は"Image.bin"または"linux.bin"にリネームする必要があります。

図 12.3 Android システムへのカーネルイメージのインストール

```
[armadillo ~]# ls
Image
❶ [armadillo ~]# mount /dev/mmcblk0p2 /mnt
❷ [armadillo ~]# rm -f /mnt/boot/*.bin
❸ [armadillo ~]# cp Image /mnt/boot/Image.bin
[armadillo ~]# umount /mnt
```

- ❶ eMMC パーティション 2 をマウントします。
- ❷ eMMC パーティション 2 の/boot/以下のカーネルイメージファイルを削除します。
- ❸ eMMC パーティション 2 の/boot/以下にカーネルイメージファイルをコピーします。ファイル名は"Image.bin"または"linux.bin"にリネームする必要があります。

図 12.4 Debian GNU/Linux システムへのカーネルイメージのインストール

13. SGX 用カーネルモジュールのビルド

本章では、PowerVR SGX DDK(Driver Development Kit)のソースコードを作業用 PC でクロスビルドし、カーネルモジュールを作成する方法について記載します。



Kernel features など、カーネルの構成に大きく影響のあるコンフィギュレーションを変更したカーネルイメージを Android で利用した場合、稀に SGX 用カーネルモジュールが動作しなくなる場合があります。カスタマイズを行ったカーネルを使用する場合は、そのカーネルにあわせた SGX 用カーネルモジュールを作成する必要があります。

事前に「11. 開発環境の準備」を参照して、作業用 PC に開発環境が構築されている必要があります。

13.1. ソースアーカイブ取得

SGX DDK のソースコードアーカイブ `eurasia_km-[version].tar.gz` を取得します。ソースアーカイブは付属 DVD-ROM から取得可能です。

13.2. カーネルの準備

SGX 用カーネルモジュールをビルドするためには、ビルド済みのカーネルソースコードが必要です。「12. カーネルのビルド」を参考にしてカーネルソースコードをビルドします。

13.3. ビルド

「13.1. ソースアーカイブ取得」で取得したソースアーカイブを展開し、ビルド環境にあわせたパラメータを設定してビルドを行います。「13.2. カーネルの準備」でビルドしたカーネルソースコードが `~/linux-2.6.35-a800eva` に配置されている場合、以下のコマンドを実行します。



SGX 用カーネルモジュールのビルドには `dos2unix` を使用します。作業用 PC にインストールされていない場合は、次のようにインストールを行ってください。

```
[PC ~]# apt-get update
[PC ~]# apt-get install dos2unix
```



```

❶[PC ~]$ tar zxvf eurasia_km-[version].tar.gz
❷[PC ~]$ cd eurasia_km-[version]/eurasiacon/build/linux2/r8a7740_android
❸[PC ~/eurasia_km-[version]/eurasiacon/build/linux2/r8a7740_android]$ make ARCH=arm
KERNEL_CROSS_COMPILE=arm-linux-gnueabi- CROSS_COMPILE=arm-linux-gnueabi-
TARGET_PRODUCT=armadillo-800eva HAL_VARIANT=armadillo-800eva KERNELDIR=~/.linux-2.6.35-a800eva
(省略)
MODPOST 3 modules
CC      /home/atmark/eurasia_km-[version]/eurasiacon/binary2_r8a7740_android_release/target/
kbuild/bc_example.mod.o
LD [M]  /home/atmark/eurasia_km-[version]/eurasiacon/binary2_r8a7740_android_release/target/
kbuild/bc_example.ko
CC      /home/atmark/eurasia_km-[version]/eurasiacon/binary2_r8a7740_android_release/target/
kbuild/pvrsvkm.mod.o
LD [M]  /home/atmark/eurasia_km-[version]/eurasiacon/binary2_r8a7740_android_release/target/
kbuild/pvrsvkm.ko
CC      /home/atmark/eurasia_km-[version]/eurasiacon/binary2_r8a7740_android_release/target/
kbuild/shmobilelfb.mod.o
LD [M]  /home/atmark/eurasia_km-[version]/eurasiacon/binary2_r8a7740_android_release/target/
kbuild/shmobilelfb.ko
❹[PC ~/eurasia_km-[version]/eurasiacon/build/linux2/r8a7740_android]$ ls ../../../../
binary2_r8a7740_android_release/target/kbuild/*ko
../../../../binary2_r8a7740_android_release/target/kbuild/bc_example.ko
../../../../binary2_r8a7740_android_release/target/kbuild/pvrsvkm.ko
../../../../binary2_r8a7740_android_release/target/kbuild/shmobilelfb.ko

```

- ❶ SGX DDK のソースコードアーカイブを展開します
- ❷ ソースコードディレクトリに移動します
- ❸ ビルドを実行します
- ❹ 3つのカーネルモジュールが作成されます

図 13.1 SGX 用カーネルモジュールのビルド

13.4. インストール

SGX 用カーネルモジュールを、内蔵ストレージの Android システムにインストールします。事前に「9. SD ブートの活用」を参照して、Debian GNU/Linux を SD ブートしている必要があります。

「13.3. ビルド」で作成した SGX 用カーネルモジュールはカレントディレクトリ以下にあることを想定しています。

```
[armadillo ~]# ls
bc_example.ko  pvrsrvkm.ko  shmobilelfb.ko
❶ [armadillo ~]# mount /dev/mmcblk0p4 /mnt
❷ [armadillo ~]# cp *.ko /mnt/lib/modules/
[armadillo ~]# umount /mnt
```

- ❶ eMMC パーティション 4 をマウントします。
- ❷ eMMC パーティション 4 の /lib/modules/ 以下にカーネルモジュールをコピーします。

図 13.2 SGX 用カーネルモジュールのインストール

14. 無線 LAN(AWL13) 用 Linux デバイスドライバーのビルド

本章では、無線 LAN モジュール Armadillo-WLAN(AWL13)用のデバイスドライバーソースコードを作業用 PC でクロスビルドし、カーネルモジュールを作成する方法について記載します。

事前に 「11. 開発環境の準備」 を参照して、作業用 PC に開発環境が構築されている必要があります。

14.1. ソースアーカイブ取得

AWL13 用のデバイスドライバーソースアーカイブ `awl13-[version].tar.gz` を取得します。ソースアーカイブは Armadillo サイト (<http://armadillo.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。

14.2. カーネルの準備

AWL13 用のデバイスドライバーソースコードをビルドするためには、ビルド済みのカーネルソースコードが必要です。「12. カーネルのビルド」 を参照してカーネルソースコードをビルドします。以下のコンフィギュレーションが有効になっている必要があります。^[1]

- ・ CONFIG_MMC
- ・ CONFIG_SYSFS
- ・ CONFIG_WIRELESS_EXT

14.3. ビルド

「14.1. ソースアーカイブ取得」 で取得したソースアーカイブを展開し、ビルド環境にあわせたパラメータを設定しビルドを行います。「14.2. カーネルの準備」 でビルドしたカーネルソースコードが `~/linux-2.6.35-a800eva-[version]` に配置されている場合、以下のコマンドを実行します。

^[1]Armadillo-800 EVA のデフォルトのコンフィギュレーションでは有効になっています。

```

❶ [PC ~]$ tar zxvf awl13-[version].tar.gz
❷ [PC ~]$ cd awl13-[version]
❸ [PC ~/awl13-[version]]$ AWL13_DEVICE=SDIO make ARCH=arm CROSS_COMPILE=arm-linux-gnueabi-
KERNELDIR=~/linux-2.6.35-a800eva-[version]
make -C ../linux-2.6.35-a800eva-[version]/ M=/home/atmark/aerial-awl13 modules
make[1]: ディレクトリ `~/home/atmark/linux-2.6.35-a800eva-[version]` に入ります
CC [M] /home/atmark/aerial-awl13/src/awl13_log.o
CC [M] /home/atmark/aerial-awl13/src/awl13_sdiodrv.o
CC [M] /home/atmark/aerial-awl13/src/awl13_ioctl.o
CC [M] /home/atmark/aerial-awl13/src/awl13_fw.o
CC [M] /home/atmark/aerial-awl13/src/awl13_sysfs.o
CC [M] /home/atmark/aerial-awl13/src/awl13_wid.o
LD [M] /home/atmark/aerial-awl13/src/awl13_sdio.o
Building modules, stage 2.
MODPOST 1 modules
CC /home/atmark/aerial-awl13/src/awl13_sdio.mod.o
LD [M] /home/atmark/aerial-awl13/src/awl13_sdio.ko
make[1]: ディレクトリ `~/home/atmark/linux-2.6.35-a800eva-[version]` から出ます
❹ [PC ~/awl13-[version]]$ ls src/awl13_sdio.ko
awl13_sdio.ko

```

- ❶ AWL13 のソースコードアーカイブを展開します
- ❷ ソースコードディレクトリに移動します
- ❸ ビルドを実行します
- ❹ カーネルモジュール `awl13_sdio.ko` が作成されます

図 14.1 AWL13 ドライバーのビルド



AWL13_DEVICE には、ホストインターフェースを指定します。Armadillo-800 EVA では、AWL13 が SDIO 起動モードで動作するように設定されています。

14.4. インストール

AWL13 用カーネルモジュールを、内蔵ストレージの Debian GNU/Linux システムにインストールします。事前に「9. SD ブートの活用」を参照して、Debian GNU/Linux を SD ブートしている必要があります。

「14.3. ビルド」で作成した AWL13 用カーネルモジュールはカレントディレクトリ以下にあることを想定しています。

```
[armadillo ~]# ls
awl13_sdio.ko
❶ [armadillo ~]# mount /dev/mmcblk0p2 /mnt
❷ [armadillo ~]# cp awl13_sdio.ko /lib/modules/$(uname -r)/awl13/
[armadillo ~]# umount /mnt
```

- ❶ eMMC パーティション 2 をマウントします。
- ❷ eMMC パーティション 2 の /lib/modules/\$(uname -r)/awl13/ 以下にカーネルモジュールをコピーします。

図 14.2 AWL13 用カーネルモジュールのインストール

15. ブートローダーのビルド

本章では、ブートローダーのソースコードを作業用 PC でクロスビルドし、ブートローダーイメージファイルを作成する方法について記載します。

事前に「11. 開発環境の準備」を参照して、作業用 PC に開発環境が構築されている必要があります。

15.1. ソースアーカイブの取得

ブートローダーのソースアーカイブ `hermit-at-[version]-source.tar.gz` を取得します。ソースアーカイブは Armadillo サイト (<http://armadillo.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。



Armadillo-800 EVA のブートローダーには、Hermit-At v3 系を採用しています。

15.2. ソースアーカイブの展開

取得したソースアーカイブを展開します。「図 15.1. ソースアーカイブの展開」のように作業してください。

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

図 15.1 ソースアーカイブの展開

15.3. ビルド

展開したソースアーカイブをビルドして、ブートローダーイメージファイルを作成します。「図 15.2. ブートローダーのビルド」のように作業してください。

```

❶ [PC ~]$ cd hermit-at-[version]
❷ [PC ~/hermit-at-[version]]$ make armadillo800eva_mmcsd_defconfig
❸ [PC ~/hermit-at-[version]]$ make
❹ [PC ~/hermit-at-[version]]$ ls src/target/armadillo8x0/loader-armadillo800eva-mmcsd-
v[version].bin
src/target/armadillo8x0/loader-armadillo800eva-mmcsd-v[version].bin

```

↩

- ❶ ソースコードディレクトリに移動します
- ❷ デフォルトコンフィギュレーションを適用します
- ❸ ソースコードをビルドします
- ❹ ブートローダーイメージファイルが作成できたことを確認します

図 15.2 ブートローダーのビルド

15.4. インストール

ブートローダーイメージを、内蔵ストレージのブートパーティションにインストールします。事前に「9. SD ブートの活用」を参照して、Debian GNU/Linux を SD ブートしている必要があります。

「15.3. ビルド」で作成したブートローダーイメージファイルはカレントディレクトリ以下にあることを想定しています。

```

[armadillo ~]# ls
loader-armadillo800eva-mmcsd-v[version].bin
❶ [armadillo ~]# echo 0 > /sys/block/mmcblk0boot0/force_ro
❷ [armadillo ~]# cat ~/loader-armadillo800eva-mmcsd-v[version].bin > /dev/mmcblk0boot0
[armadillo ~]# sync

```

- ❶ ブートパーティションのソフトライトプロテクトを解除します。
- ❷ ブートローダーイメージを書き込みます。

図 15.3 ブートローダーイメージのインストール

16. JTAG ICE を利用する

本章では ARM のデバッグを行うために、JTAG ICE を接続する方法について説明します。

16.1. 準備

16.1.1. JTAG ケーブルの接続

JTAG ICE のケーブルを、ARM JTAG インターフェース(ARM 標準 20 ピンコネクタ)に接続します。信号配列などの詳細な情報については、「18.6. CON6(ARM JTAG インターフェース)」を参照してください。

16.1.2. ディップスイッチの設定

ディップスイッチの JTAG 設定を ARM に設定します。「図 16.1. ディップスイッチの JTAG 設定 (ARM)」を参照して、JTAG 設定 1(SW1.7)を ON に、JTAG 設定 2(SW1.8)を OFF に設定してください。

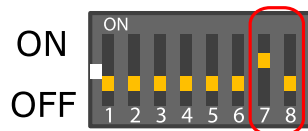


図 16.1 ディップスイッチの JTAG 設定(ARM)

16.2. 接続確認

「16.1. 準備」に従って設定されている場合に、CPU は以下のように見えます。

項目	値
デバイス ID	0x4BA00477
コマンド長	4

16.3. 各種デバッガへの対応について

お使いのデバッガが Armadillo-800 EVA に対応しているか等の情報につきましては、各メーカーにお問い合わせください。

17. Linux カーネル仕様

本章では、Linux カーネルの仕様について説明します。Linux カーネルのソースアーカイブ linux-2.6.35-a800eva-[version].tar.gz は Armadillo サイト (<http://armadillo.atmark-techno.com>) または、付属 DVD-ROM から取得可能です。

17.1. デフォルトコンフィギュレーション

工場出荷状態の内蔵ストレージにインストールされている Linux カーネルにはデフォルトコンフィギュレーションが適用されています。Armadillo-800 EVA 用のデフォルトコンフィギュレーションが記載されているファイルは、Linux カーネルソースファイルに含まれる arch/arm/configs/armadillo800eva_android_defconfig です。

Armadillo-800 EVA では、Linux カーネルが格納されているブートデバイスは OS ごとに異なりますが、Linux カーネルは同じものを使用しています。そのため、デフォルトコンフィギュレーションは、工場出荷状態の内蔵ストレージにインストールされている全ての OS で動作するように共通化されています。

表 17.1 OS とデフォルトコンフィギュレーション

OS	ブートデバイス	コンフィギュレーション
Debian GNU/Linux	mmcblk0p2	armadillo800eva_android_defconfig
Android	mmcblk0p4	

17.2. Android 機能

Armadillo-800 EVA では、Android 向け機能を一部のみ追加したカーネルを Debian GNU/Linux と Android 両方で使用しているため、PC 上で動作する Debian GNU/Linux や、スマートフォンやタブレット機器上で動作する Android とは、一部機能の動作が異なる場合があります。

Android 機能は、カーネルのコンフィギュレーションで、その有効/無効を選択できます。主要な Android 機能の一覧と、デフォルトコンフィギュレーションでの状態を「表 17.2. Android 機能の主要コンフィギュレーション」に示します。

表 17.2 Android 機能の主要コンフィギュレーション

コンフィギュレーション	デフォルト	機能
ANDROID_BINDER_IPC	有効	Android 用プロセス間通信
ASHMEM	有効	無名共有メモリ(Anonymous Shared Memory)
ANDROID_PMEM	有効	Android 用プロセスメモリアロケーター
UID_STAT	有効	/proc/uid_stat/サポート
ANDROID_LOGGER	有効	Android 用ロガー
WAKELOCK	有効	拡張パワーマネジメント(Wake lock)
WAKELOCK_STAT	有効	/proc/wake_locks サポート
USER_WAKELOCK	有効	ユーザー空間でのサスペンド防止
EARLYSUSPEND	有効	拡張パワーマネジメント(Early suspend)
ANDROID_LOW_MEMORY_KILLER	有効	Android 用 OOM(Out Of Memory)ハンドリング
RTC_INTF_ALARM	有効	Android 用アラーム
ANDROID_TIMED_OUTPUT	有効	Android 用タイマー出力
ANDROID_TIMED_GPIO	無効	Android 用 GPIO タイマー出力
ANDROID_PARANOID_NETWORK	無効	Android 用ネットワーク保護

コンフィギュレーション	デフォルト	機能
ANDROID_RAM_CONSOLE	無効	Android 用コンソール RAM バッファ

17.3. Linux ドライバー 一覧

Armadillo-800 EVA を制御する Linux ドライバーのソースコードのパスや制御可能なデバイスを示します。

ボード固有設定

ソースコード arch/arm/mach-shmobile/board-armadillo800eva.c

タイマードライバー

ソースコード arch/arm/mach-shmobile/sh_cmt.c

割り込みコントローラードライバー

ソースコード arch/arm/mach-shmobile/intc-sh7740.c

PWM ドライバー

ソースコード arch/arm/mach-shmobile/tpu-pwm.c

UART ドライバー

ソースコード drivers/serial/sh-sci.c

デバイスファイル /dev/ttySC1 (CON22)

Ethernet ドライバー

ソースコード drivers/net/sh_eth.c

デバイス eth0 (CON23)

MMC ホストドライバ

ソースコード drivers/mmc/host/sh_mmcif.c

デバイス /dev/mmcblk0

SD ホストドライバ

ソースコード drivers/mmc/host/sh_mobile_sdhi.c

デバイス /dev/mmcblk1

/dev/mmcblk2

USB ホストドライバ

ソースコード drivers/usb/host/ehci-rmobile.c

drivers/usb/host/ohci-rmobile.c

USB ファンクションドライバ

ソースコード drivers/usb/gadget/r8a66597-udc.c

フレームバッファドライバ

ソースコード drivers/video/rmobile_lcdcfb.c

デバイス /dev/fb0 (CON17)
 /dev/fb1 (CON3 or CON14)

キャプチャーインターフェースドライバ

ソースコード drivers/media/video/sh_mobile_ceu_camera.c

カメラドライバ

ソースコード drivers/media/video/mt9t112.c

デバイス /dev/video0

リアルタイムクロックドライバ

ソースコード drivers/rtc/rtc-s35390a.c

デバイス /dev/rtc0

タッチスクリーンドライバ

ソースコード drivers/input/touchscreen/st1232.c

デバイス /dev/input/event1

ボタンスイッチキーボードドライバ

ソースコード drivers/input/keyboard/gpio_keys.c

デバイス /dev/input/event0

LCD バックライトドライバ

ソースコード drivers/video/backlight/pwm_bl.c

デバイス /sys/class/backlight/pwm-backlight.0

LED ドライバ

ソースコード drivers/leds/leds-gpio.c

デバイス /sys/class/leds/LED3 (LED3)
 /sys/class/leds/LED4 (LED4)
 /sys/class/leds/LED5 (LED5)
 /sys/class/leds/LED6 (LED6)

オーディオドライバー

ソースコード	sound/soc/sh/fsi-wm8978.c sound/soc/sh/fsi-hdmi.c sound/soc/sh/fsi.c sound/soc/codecs/wm8978.c
デバイス	hw:0 (CON10/11/12/13) hw:1 (CON3)

18. インターフェース仕様

18.1. CON1(カメラモジュールインターフェース)

CON1 はカメラモジュール接続用インターフェースです。

- ・ 対応カメラモジュール： DCB-NSB55QFMRB-S-05-02(CRESYN)

表 18.1 CON1 信号配列

ピン番号	信号名	I/O	機能
1	CAM0_2V8_D6	In	データバス(bit6)、R-Mobile A1 の VIO_D6_0 ピンに接続
2	CAM0_2V8_D7	In	データバス(bit7)、R-Mobile A1 の VIO_D7_0 ピンに接続
3	NC	-	未接続
4	NC	-	未接続
5	CAM0_2V8_STANDBY	Out	スタンバイ信号、R-Mobile A1 の D22 ピンに接続
6	VCC_CAM2.8V	Power	電源(VCC_CAM2.8V)
7	VCC_CAM2.8V	Power	電源(VCC_CAM2.8V)
8	DGND	Power	電源(DGND)
9	CAM0_2V8_PCLK	In	クロック信号入力、R-Mobile A1 の VIO_CLK_0 ピンに接続
10	DGND	Power	電源(DGND)
11	CAM0_2V8_MCLK	Out	クロック信号出力、R-Mobile A1 の VIO_CKO_0 ピンに接続
12	CAM0_2V8_SCL	Out	I2C クロック、R-Mobile A1 の I2C_SCL_0 ピンに接続
13	CAM0_2V8_D3	In	データバス(bit3)、R-Mobile A1 の VIO_D3_0 ピンに接続
14	CAM0_2V8_D2	In	データバス(bit2)、R-Mobile A1 の VIO_D2_0 ピンに接続
15	CAM0_2V8_D1	In	データバス(bit1)、R-Mobile A1 の VIO_D1_0 ピンに接続
16	CAM0_2V8_D0	In	データバス(bit0)、R-Mobile A1 の VIO_D0_0 ピンに接続
17	CAM0_2V8_D4	In	データバス(bit4)、R-Mobile A1 の VIO_D4_0 ピンに接続
18	CAM0_2V8_D5	In	データバス(bit5)、R-Mobile A1 の VIO_D5_0 ピンに接続
19	CAM0_2V8_SDA	In/Out	I2C データ、R-Mobile A1 の I2C_SDA_0 ピンに接続
20	GPIO4_FLASH	In/Out	LED1 に接続(High: 点灯、Low: 消灯)
21	VCC_CAM1.8V	Power	電源(VCC_CAM1.8V)
22	CAM0_2V8_RESET#	Out	リセット信号、R-Mobile A1 の D23 ピンに接続
23	CAM0_2V8_VSYNC	In	VSYNC 信号、R-Mobile A1 の VIO_VD_0 ピンに接続
24	CAM0_2V8_HSYNC	In	HSYNC 信号、R-Mobile A1 の VIO_HD_0 ピンに接続

18.2. CON2(拡張バスインターフェース)

CON2 はバス拡張用のインターフェースです。コネクタは実装していません。

- ・ 基板側コネクタ例： XG4C-6431(オムロン)
- ・ 対向コネクタ例： XG4M-6431(オムロン)

表 18.2 CON2 信号配列

ピン番号	信号名	I/O	機能
1	VCC_3.3V	Power	電源(VCC_3.3V)
2	DGND	Power	電源(DGND)
3	A1_D0	In/Out	データバス(bit0)、R-Mobile A1 の D0 ピンに接続
4	A1_D1	In/Out	データバス(bit1)、R-Mobile A1 の D1 ピンに接続
5	A1_D2	In/Out	データバス(bit2)、R-Mobile A1 の D2 ピンに接続

ピン番号	信号名	I/O	機能
6	A1_D3	In/Out	データバス(bit3)、R-Mobile A1 の D3 ピンに接続
7	A1_D4	In/Out	データバス(bit4)、R-Mobile A1 の D4 ピンに接続
8	A1_D5	In/Out	データバス(bit5)、R-Mobile A1 の D5 ピンに接続
9	A1_D6	In/Out	データバス(bit6)、R-Mobile A1 の D6 ピンに接続
10	A1_D7	In/Out	データバス(bit7)、R-Mobile A1 の D7 ピンに接続
11	A1_D8	In/Out	データバス(bit8)、R-Mobile A1 の D8 ピンに接続
12	A1_D9	In/Out	データバス(bit9)、R-Mobile A1 の D9 ピンに接続
13	A1_D10	In/Out	データバス(bit10)、R-Mobile A1 の D10 ピンに接続
14	A1_D11	In/Out	データバス(bit11)、R-Mobile A1 の D11 ピンに接続
15	A1_D12	In/Out	データバス(bit12)、R-Mobile A1 の D12 ピンに接続
16	A1_D13	In/Out	データバス(bit13)、R-Mobile A1 の D13 ピンに接続
17	A1_D14	In/Out	データバス(bit14)、R-Mobile A1 の D14 ピンに接続
18	A1_D15	In/Out	データバス(bit15)、R-Mobile A1 の D15 ピンに接続
19	EXT_MMC_RST_B	In/Out	外部リセット信号、R-Mobile A1 の MEMC_NWE ピンに接続
20	A1_CS6A_B	In/Out	チップセレクト 6A、R-Mobile A1 の CS6A#ピンに接続
21	A1_CS5B_B	In/Out	チップセレクト 5B、R-Mobile A1 の CS5B#ピンに接続
22	A1_CS5A_B	In/Out	チップセレクト 5A、R-Mobile A1 の CS5A#ピンに接続
23	A1_CS0_B	In/Out	チップセレクト 0、R-Mobile A1 の CS0#ピンに接続
24	A1_RESETOUTS_B	In/Out	リセット信号、R-Mobile A1 の RESETOUTS#ピンに接続
25	A1_RD_B	In/Out	リード信号、R-Mobile A1 の RD#ピンに接続
26	A1_RDWR	In/Out	リードライト信号、R-Mobile A1 の RDWR ピンに接続
27	A1_WE1_B	In/Out	ライトイネーブル信号 1、R-Mobile A1 の WE1#ピンに接続
28	A1_WE0_B	In/Out	ライトイネーブル信号 0、R-Mobile A1 の WE0#ピンに接続
29	A1_DACK0_B	In/Out	DACK 信号 0、R-Mobile A1 の DACK_0 ピンに接続
30	A1_DREQ0_B	In/Out	DREQ 信号 0、R-Mobile A1 の DREQ_0 ピンに接続
31	A1_ICIORD	In/Out	ICIORD 信号、R-Mobile A1 の WE2#ピンに接続
32	A1_ICIOWR	In/Out	ICIOWR 信号、R-Mobile A1 の WE3#ピンに接続
33	A1_IOIS16_B	In/Out	IOIS16 信号、R-Mobile A1 の IOIS16#ピンに接続
34	A1_BSCMD	In/Out	BSCMD 信号、R-Mobile A1 の BSCMD ピンに接続
35	A1_CKO	In/Out	CKO 信号、R-Mobile A1 の CKO ピンに接続
36	DGND	Power	電源(DGND)
37	A1_A0	In/Out	アドレスバス(bit0)、R-Mobile A1 の A0 ピンに接続
38	A1_A1	In/Out	アドレスバス(bit1)、R-Mobile A1 の A1 ピンに接続
39	A1_A2	In/Out	アドレスバス(bit2)、R-Mobile A1 の A2 ピンに接続
40	A1_A3	In/Out	アドレスバス(bit3)、R-Mobile A1 の A3 ピンに接続
41	A1_A4	In/Out	アドレスバス(bit4)、R-Mobile A1 の A4 ピンに接続
42	A1_A5	In/Out	アドレスバス(bit5)、R-Mobile A1 の A5 ピンに接続
43	A1_A6	In/Out	アドレスバス(bit6)、R-Mobile A1 の A6 ピンに接続
44	A1_A7	In/Out	アドレスバス(bit7)、R-Mobile A1 の A7 ピンに接続
45	A1_A8	In/Out	アドレスバス(bit8)、R-Mobile A1 の A8 ピンに接続
46	A1_A9	In/Out	アドレスバス(bit9)、R-Mobile A1 の A9 ピンに接続
47	A1_A10	In/Out	アドレスバス(bit10)、R-Mobile A1 の A10 ピンに接続
48	A1_A11	In/Out	アドレスバス(bit11)、R-Mobile A1 の A11 ピンに接続
49	A1_A12	In/Out	アドレスバス(bit12)、R-Mobile A1 の A12 ピンに接続
50	A1_A13	In/Out	アドレスバス(bit13)、R-Mobile A1 の A13 ピンに接続
51	A1_A14	In/Out	アドレスバス(bit14)、R-Mobile A1 の A14 ピンに接続
52	A1_A15	In/Out	アドレスバス(bit15)、R-Mobile A1 の A15 ピンに接続
53	A1_A16	In/Out	アドレスバス(bit16)、R-Mobile A1 の A16 ピンに接続
54	A1_A17	In/Out	アドレスバス(bit17)、R-Mobile A1 の A17 ピンに接続
55	A1_A18	In/Out	アドレスバス(bit18)、R-Mobile A1 の A18 ピンに接続
56	A1_A19	In/Out	アドレスバス(bit19)、R-Mobile A1 の A19 ピンに接続
57	A1_A20	In/Out	アドレスバス(bit20)、R-Mobile A1 の A20 ピンに接続
58	A1_A21	In/Out	アドレスバス(bit21)、R-Mobile A1 の A21 ピンに接続
59	A1_A22	In/Out	アドレスバス(bit22)、R-Mobile A1 の A22 ピンに接続

ピン番号	信号名	I/O	機能
60	A1_A23	In/Out	アドレスバス(bit23)、R-Mobile A1 の A23 ピンに接続
61	A1_A24	In/Out	アドレスバス(bit24)、R-Mobile A1 の A24 ピンに接続
62	A1_A25	In/Out	アドレスバス(bit25)、R-Mobile A1 の A25 ピンに接続
63	VCC_3.3V	Power	電源(VCC_3.3V)
64	DGND	Power	電源(DGND)

18.3. CON3(デジタル HD 出カインターフェース)

CON3 はデジタル HD 出カインターフェースです。HDMI Type-A コネクタを実装しています。

表 18.3 CON3 信号配列

ピン番号	信号名	I/O	機能
1	TX2_P	Out	TMDS データ 2+, R-Mobile A1 の TODP2 ピンに接続
2	-	-	TMDS データ 2 シールド、DGND に接続
3	TX2_N	Out	TMDS データ 2-, R-Mobile A1 の TODN2 ピンに接続
4	TX1_P	Out	TMDS データ 1+, R-Mobile A1 の TODP1 ピンに接続
5	-	-	TMDS データ 1 シールド、DGND に接続
6	TX1_N	Out	TMDS データ 1-, R-Mobile A1 の TODN1 ピンに接続
7	TX0_P	Out	TMDS データ 0+, R-Mobile A1 の TODP0 ピンに接続
8	-	-	TMDS データ 0 シールド、DGND に接続
9	TX0_N	Out	TMDS データ 0-, R-Mobile A1 の TODN0 ピンに接続
10	TMC_P	Out	TMDS クロック+, R-Mobile A1 の TOCP ピンに接続
11	-	-	TMDS クロックシールド DGND に接続
12	TXC_N	Out	TMDS クロック-, R-Mobile A1 の TOCN ピンに接続
13	DDC_CEC	In/Out	未接続 ^[a]
14	NC	-	未接続
15	DDC_SCL	In/Out	SCL、R-Mobile A1 の HDMI_SCL ピンに接続
16	DDC_SDA	In/Out	SDA、R-Mobile A1 の HDMI_SDA ピンに接続
17	DGND	Power	電源(DGND)
18	DDC_5.0V	Power	電源(DDC_5.0V)
19	DDC_HPD	In	ホットプラグ検出、R-Mobile A1 の HDMI_HPD ピンに接続

^[a] 「A800-EVA-ERRATUM #2 : HDMI の信号線 CEC に HDMI 規格を超えた電圧が印加される」への対応のため CEC は未接続となっています。詳しくは「Armadillo-800 EVA リビジョン情報」を参照してください。

18.4. CON4(コンポジットビデオ出カインターフェース)

CON4 はコンポジットビデオ出カインターフェースです。RCA ジャック黄色を実装しています。


表 18.4 CON4 信号配列

ピン番号	信号名	I/O	機能
1	YCVBSOUT	Out	コンポジットビデオ出力、R-Mobile A1 の YCVBSOUT ピンに接続
2	DGNC	Power	電源(DGND)

18.5. CON5(H-UDI JTAG インターフェース)

CON5 は H-UDI JTAG インターフェースです。コネクタは実装していません。

- ・ 基板側コネクタ例： XG4C-1431(オムロン)
- ・ 対向コネクタ例： XG4M-1431(オムロン)



CON5 と CON6 の信号は共通となっており、CON6 と同時に利用することはできません。


表 18.5 CON5 信号配列

ピン番号	信号名	I/O	機能
1	A1_TCK	In	TCK 信号、R-Mobile A1 の TCK ピンに接続
2	A1_TRST_B	In	TRST 信号、R-Mobile A1 の TRST#ピンに接続
3	A1_TDO	Out	TDO 信号、R-Mobile A1 の TDO ピンに接続
4	A1_EDBGREQ/ ASEBRK_B	In	ASEBRK 信号、R-Mobile A1 の EDBGREQ ピンに接続
5	A1_TMS	In	TMS 信号、R-Mobile A1 の TMS ピンに接続
6	A1_TDI	In	TDI 信号、R-Mobile A1 の TDI ピンに接続
7	RESETP_B	In	リセット信号(Low: リセット状態、High: リセット解除)、R-Mobile A1 の RESETP#ピンに接続
8	NC	-	未接続
9	DGND	Power	電源(DGND)
10	DGND	Power	電源(DGND)
11	VCC_3.3V	Power	電源(VCC_3.3V)
12	DGND	Power	電源(DGND)
13	DGND	Power	電源(DGND)
14	DGND	Power	電源(DGND)

18.6. CON6(ARM JTAG インターフェース)

CON6 は ARM JTAG インターフェースです。XG4C-2031(オムロン)を実装しています。

- ・ 対向コネクタ例： XG4M-2031(オムロン)



CON5 と CON6 の信号は共通となっており、CON5 と同時に利用することはできません。

表 18.6 CON6 信号配列

ピン番号	信号名	I/O	機能
1	VCC_3.3V	Power	電源(VCC_3.3V)
2	VCC_3.3V	Power	電源(VCC_3.3V)
3	A1_TRST_B	In	TRST 信号、R-Mobile A1 の TRST#ピンに接続
4	DGND	Power	電源(DGND)
5	A1_TDI	In	TDI 信号、R-Mobile A1 の TDI ピンに接続
6	DGND	Power	電源(DGND)
7	A1_TMS	In	TMS 信号、R-Mobile A1 の TMS ピンに接続
8	DGND	Power	電源(DGND)
9	A1_TCK	In	TCK 信号、R-Mobile A1 の TCK ピンに接続
10	DGND	Power	電源(DGND)
11	A1_RTCK	Out	RTCK 信号、R-Mobile A1 の RTCK ピンに接続
12	DGND	Power	電源(DGND)
13	A1_TDO	Out	TDO 信号、R-Mobile A1 の TDO ピンに接続

ピン番号	信号名	I/O	機能
14	DGND	Power	電源(DGND)
15	SRST_IN_B_OD	In	リセット信号(Low: リセット状態、High: リセット解除)、R-Mobile A1 の RESETP#ピンに接続
16	DGND	Power	電源(DGND)
17	A1_EDBGREQ/ ASEBRK_B	In	DBGREQ 信号、R-Mobile A1 の EDBGREQ ピンに接続
18	DGND	Power	電源(DGND)
19	NC	-	R45 に抵抗を実装すると CON6 の 17 ピンと接続
20	DGND	Power	電源(DGND)

18.7. CON7(SD インターフェース 1)

CON7 は SD スロットです。

表 18.7 CON7 信号配列

ピン番号	信号名	I/O	機能
1	A1_SDHIO_D3	In/Out	SD データ(bit3)、R-Mobile A1 の SDHID3_0 ピンに接続
2	A1_SDHIO_CMD	In/Out	SD コマンド、R-Mobile A1 の SDHICMD_0 ピンに接続
3	DGND	Power	電源(DGND)
4	VCC_SD0	Power	電源(VCC_SD0)
5	A1_SDHIO_CLK	Out	SD クロック、R-Mobile A1 の SDHICLK_0 ピンに接続
6	DGND	Power	電源(DGND)
7	A1_SDHIO_D0	In/Out	SD データ(bit0)、R-Mobile A1 の SDHID0_0 ピンに接続
8	A1_SDHIO_D1	In/Out	SD データ(bit1)、R-Mobile AC2 の SDHID1_0 ピンに接続
9	A1_SDHIO_D2	In/Out	SD データ(bit2)、R-Mobile A1 の SDHID2_0 ピンに接続
10	A1_SDHIO_CD	In	SD カード検出(Low: カード挿入、High: カード抜去)、R-Mobile A1 の D20 ピンに接続
11	FG	Power	電源(DGND)
12	A1_SDHIO_WP	In	ライトプロテクト検出(Low: 書き込み可能、High: 書き込み不可能)、R-Mobile A1 の SDHIWP_0 ピンに接続
13	DGND	Power	電源(DGND)
14	DGND	Power	電源(DGND)
15	DGND	Power	電源(DGND)
18	DGND	Power	電源(DGND)

18.8. CON8(SD インターフェース 2)

CON8 は SD スロットです。



CON8 と CON14 の信号は共通となっており、同時に使用することはできません。

CON8 と CON14 で使用する各信号をどちらのインターフェースに接続させるかは、ディップスイッチ(SW1)の SDHI1 設定で選択できるようになっています。詳しくは、「表 4.3. ディップスイッチ(SW1)のスイッチの機能」または、「18.25. SW1(機能選択スイッチ)」を参照してください。

表 18.8 CON8 信号配列

ピン番号	信号名	I/O	機能
1	SDSLOT2_SDHI1_D3	In/Out	SD データ(bit3)、R-Mobile A1 の MEMC_AD11 ピンに接続

ピン番号	信号名	I/O	機能
2	SDSLOT2_SDHI1_CMD	In/Out	SD コマンド、R-Mobile A1 の MEMC_CS0 ピンに接続
3	DGND	Power	電源(DGND)
4	VCC_SD0	Power	電源(VCC_SD0)
5	SDSLOT2_SDHI1_CLK	In/Out	SD クロック、R-Mobile A1 の MEMC_INT ピンに接続
6	DGND	Power	電源(DGND)
7	SDSLOT2_SDHI1_D0	In/Out	SD データ(bit0)、R-Mobile A1 の MEMC_AD8 ピンに接続
8	SDSLOT2_SDHI1_D1	In/Out	SD データ(bit1)、R-Mobile A1 の MEMC_AD9 に接続
9	SDSLOT2_SDHI1_D2	In/Out	SD データ(bit2)、R-Mobile A1 の MEMC_AD10 ピンに接続
10	SDSLOT2_SDHI1_CD	In	SD カード検出(Low: カード挿入、High: カード抜去)、R-Mobile A1 の MEMC_AD12 ピンに接続
11	FG	Power	電源(DGND)
12	SDSLOT2_SDHI1_WP	In	ライトプロテクト検出(Low: 書き込み可能、High: 書き込み不可能)、R-Mobile A1 の MEMC_AD13 ピンに接続
13	DGND	Power	電源(DGND)
14	DGND	Power	電源(DGND)
15	DGND	Power	電源(DGND)
18	DGND	Power	電源(DGND)

18.9. CON9(RTC 外部バックアップインターフェース)

CON9 は RTC の外部バックアップインターフェースです。

- ・ 対応電池：CR2032

表 18.9 CON9 信号配列

ピン番号	信号名	I/O	機能
1	BAT	Power	RTC のバックアップ用電源入力
2	DGND	Power	電源(DGND)

18.10. CON10~CON13(オーディオインターフェース)

CON10~CON13 はオーディオインターフェースです。オーディオ CODEC を経由して R-Mobile A1 に接続されています。

18.10.1. CON10(モノラルマイク入力インターフェース)

モノラルマイク入力インターフェースです。φ3.5mm ミニジャックを実装しています。

表 18.10 CON10 信号配列

ピン番号	信号名	I/O	機能
1	MIC_IN	In	マイク入力信号
2	AGND_DA	Power	電源(AGND_DA)
3	AGND_DA	Power	電源(AGND_DA)
5	AGND_DA	Power	電源(AGND_DA)

18.10.2. CON11(ステレオヘッドホン出力インターフェース)

CON11 はステレオヘッドホン出力インターフェースです。φ3.5mm ミニジャックを実装しています。

表 18.11 CON11 信号配列

ピン番号	信号名	I/O	機能
1	HP_OUT_L	Out	ヘッドホン出力(左チャンネル)
2	GPIO	In	ヘッドホン検知
3	HP_OUT_R	Out	ヘッドホン出力(右チャンネル)
5	AGND_DA	Power	電源(AGND_DA)

18.10.3. CON12(ステレオライン出力(L)インターフェース)

CON12 はステレオライン出力(L)インターフェースです。RCA ジャック白色を実装しています。

表 18.12 CON12 信号配列

ピン番号	信号名	I/O	機能
1	LINE_OUT_L	Out	ライン出力(左チャンネル)
2	AGND_DA	Power	電源(AGND_DA)

18.10.4. CON13(ステレオライン出力(R)インターフェース)

CON13 はステレオライン出力(R)インターフェースです。RCA ジャック赤色を実装しています。

表 18.13 CON13 信号配列

ピン番号	信号名	I/O	機能
1	LINE_OUT_R	Out	ライン出力(右チャンネル)
2	AGND_DA	Power	電源(AGND_DA)

18.11. CON14(AWL13 モジュールインターフェース)

CON14 は AWL13 モジュールインターフェースです。AWL13 の制御信号が接続されており、SDIO 起動モードで動作するよう設定されています。^[1]



CON8 と CON14 の信号は共通となっており、同時に使用することはできません。

CON8 と CON14 で使用する各信号をどちらのインターフェースに接続させるかは、ディップスイッチ(SW1)の SDHI1 設定で選択できるようになっています。詳しくは、「表 4.3. ディップスイッチ(SW1)のスイッチの機能」または、「18.25. SW1(機能選択スイッチ)」を参照してください。

表 18.14 CON14 信号配列

ピン番号	信号名	I/O	機能
1	SDDATA1	In/Out	SDIO データ(bit1)、R-Mobile A1 の MEMC_AD9 ピンに接続
2	SDDATA0	In/Out	SDIO データ(bit0)、R-Mobile A1 の MEMC_AD8 ピンに接続
3	DGND	Power	電源(DGND)
4	DGND	Power	電源(DGND)
5	NC	-	未接続
6	NC	-	未接続
7	SDCLK	Out	SDIO クロック、R-Mobile A1 の MEMC_INT ピンに接続

^[1]AWL13 モジュールの詳細につきましては、「Armadillo-WLAN(AWL13)ハードウェアマニュアル」を参照してください。

ピン番号	信号名	I/O	機能
8	VCC_SD1	Power	電源(VCC_SD1)
9	NC	-	未接続
10	SDCMD	Out	SDIO コマンド、R-Mobile A1 の MEMC_CS0 ピンに接続
11	SDDATA3	In/Out	SDIO データ(bit3)、R-Mobile A1 の MEMC_AD11 ピンに接続
12	SDDATA2	In/Out	SDIO データ(bit2)、R-Mobile A1 の MEMC_AD10 ピンに接続
13	NC	-	未接続
14	NC	-	未接続
15	BOOT_SEL1	Out	SDIO 起動モード選択 (BOOT_SEL1: Low、BOOT_SEL0: High、HOST_SEL: High)
16	BOOT_SEL0	Out	
17	HOST_SEL	Out	
18	NC	-	未接続
19	NC	-	未接続
20	NC	-	未接続
21	NC	-	未接続
22	NC	-	未接続
23	NC	-	未接続
24	NC	-	未接続
25	NC	-	未接続
26	NC	-	未接続
27	NC	-	未接続
28	HRST	In	VCC_SD1 に接続
29	NC	-	未接続
30	NC	-	未接続
31	NC	-	未接続
32	NC	-	未接続
33	NC	-	未接続
34	NC	-	未接続

18.12. CON15(拡張インターフェース)

CON15 は拡張インターフェースです。コネクタは実装していません。

- ・ 基板側コネクタ例： XG4C-6031(オムロン)
- ・ 対向コネクタ例： XG4M-6031(オムロン)

表 18.15 CON15 信号配列

ピン番号	信号名	I/O	機能
1	VCC_5V	Power	電源(VCC_5V)
2	VCC_5V	Power	電源(VCC_5V)
3	VCC_3.3V	Power	電源(VCC_3.3V)
4	VCC_3.3V	Power	電源(VCC_3.3V)
5	DGND	Power	電源(DGND)
6	DGND	Power	電源(DGND)
7	A1_VIO0_CKO	In/Out	R-Mobile A1 の VIO_CKO_0 ピンに接続
8	A1_VIO0_CLK	In/Out	R-Mobile A1 の VIO_CLK_0 ピンに接続
9	A1_VIO0_HD	In/Out	R-Mobile A1 の VIO_HD_0 ピンに接続
10	A1_VIO0_VD	In/Out	R-Mobile A1 の VIO_VD_0 ピンに接続
11	A1_VIO0_FIELD	In/Out	R-Mobile A1 の VIO_FIELD_0 ピンに接続
12	A1_VIO0_D0	In/Out	R-Mobile A1 の VIO_D0_0 ピンに接続
13	A1_VIO0_D1	In/Out	R-Mobile A1 の VIO_D1_0 ピンに接続
14	A1_VIO0_D2	In/Out	R-Mobile A1 の VIO_D2_0 ピンに接続
15	A1_VIO0_D3	In/Out	R-Mobile A1 の VIO_D3_0 ピンに接続
16	A1_VIO0_D4	In/Out	R-Mobile A1 の VIO_D4_0 ピンに接続

ピン番号	信号名	I/O	機能
17	A1_VIO0_D5	In/Out	R-Mobile A1 の VIO_D5_0 ピンに接続
18	A1_VIO0_D6	In/Out	R-Mobile A1 の VIO_D6_0 ピンに接続
19	A1_VIO0_D7	In/Out	R-Mobile A1 の VIO_D7_0 ピンに接続
20	A1_VIO0_STANDBY	In/Out	R-Mobile A1 の D22 ピンに接続
21	A1_VIO0_RST_B	In/Out	R-Mobile A1 の D23 ピンに接続
22	DGND	Power	電源(DGND)
23	A1_VIO1_CLK	In/Out	R-Mobile A1 の SCIFA_RXD_0 ピンに接続
24	A1_VIO1_HD	In/Out	R-Mobile A1 の D29 ピンに接続
25	A1_VIO1_VD	In/Out	R-Mobile A1 の SCIFA_TXD_0 ピンに接続
26	A1_VIO1_D0	In/Out	R-Mobile A1 の VIO_D8_0 ピンに接続
27	A1_VIO1_D1	In/Out	R-Mobile A1 の VIO_D9_0 ピンに接続
28	A1_VIO1_D2	In/Out	R-Mobile A1 の VIO_D10_0 ピンに接続
29	A1_VIO1_D3	In/Out	R-Mobile A1 の VIO_D11_0 ピンに接続
30	A1_VIO1_D4	In/Out	R-Mobile A1 の VIO_D12_0 ピンに接続
31	A1_VIO1_D5	In/Out	R-Mobile A1 の VIO_D13_0 ピンに接続
32	A1_VIO1_D6	In/Out	R-Mobile A1 の VIO_D14_0 ピンに接続
33	A1_VIO1_D7	In/Out	R-Mobile A1 の VIO_D15_0 ピンに接続
34	A1_VIO1_STANDBY	In/Out	R-Mobile A1 の SCIFA_RTS_0#ピンに接続
35	A1_VIO1_RST_B	In/Out	R-Mobile A1 の SCIFA_CTS_0#ピンに接続
36	CAM_DISABLE	In	CON1 に搭載されているカメラモジュールの信号線のイネーブル/ディスエーブル切り替え信号 (Low: イネーブル、High: ディスエーブル)
37	VCC_3.3V	Power	電源(VCC_3.3V)
38	DGND	Power	電源(DGND)
39	A1_STP0_IPCLK	In/Out	R-Mobile A1 の STP_IPCLK_0 ピンに接続
40	A1_STP0_IPSYNC	In/Out	R-Mobile A1 の SCIFB_TXD ピンに接続
41	A1_STP0_IPEN	In/Out	R-Mobile A1 の SCIFB_RXD ピンに接続
42	A1_STP0_IPD0	In/Out	R-Mobile A1 の SCIFB_SCK ピンに接続
43	A1_STP0_IPD1	In/Out	R-Mobile A1 の MEMC_AD1 ピンに接続
44	A1_STP0_IPD2	In/Out	R-Mobile A1 の SCIFA_RTS_1#ピンに接続
45	A1_STP0_IPD3	In/Out	R-Mobile A1 の MEMC_AD2 ピンに接続
46	A1_STP0_IPD4	In/Out	R-Mobile A1 の MEMC_AD3 ピンに接続
47	A1_STP0_IPD5	In/Out	R-Mobile A1 の SCIFA_CTS_1#ピンに接続
48	A1_STP0_IPD6	In/Out	R-Mobile A1 の MEMC_WAIT ピンに接続
49	A1_STP0_IPD7	In/Out	R-Mobile A1 の MEMC_NOE ピンに接続
50	A1_SIM_D	In/Out	R-Mobile A1 の SCIFA_SCK_2 ピンに接続
51	A1_SIM_CLK	In/Out	R-Mobile A1 の MEMC_ADV ピンに接続
52	A1_SIM_RST	In/Out	R-Mobile A1 の MEMC_CS1 ピンに接続
53	A1_PORT15	In/Out	R-Mobile A1 の FMISOIBT ピンに接続
54	A1_PORT14	In/Out	R-Mobile A1 の FMISOILR ピンに接続
55	RESETP_B	In/Out	R-Mobile A1 の RESETP#ピンに接続
56	NC	-	未接続
57	VCC_3.3V	Power	電源(VCC_3.3V)
58	DGND	Power	電源(DGND)
59	A1_SCL1	In/Out	R-Mobile A1 の I2C_SCL_1 ピンに接続
60	A1_SDA1	In/Out	R-Mobile A1 の I2C_SDA_1 ピンに接続



CON15 の 7 ピンから 21 ピンの信号線を使用する場合、CON15 の 36 ピン(CAM_DISABLE)を High に設定し、CON1 に搭載されているカメラモジュールの信号線をディスエーブルにしてください。イネーブルで使用した場合、信号線が衝突して CON1 に搭載されているカメラモジュールや CON15 に接続した機器を破損する可能性があります。

18.13. CON16(LCD 拡張インターフェース 1)

CON16 は LCD 拡張インターフェースです。コネクタは実装していません。



CON16 と CON17 の信号は共通となっており、同時に使用することはできません。

- ・ 基板側コネクタ例： XG4C-5031(オムロン)
- ・ 対向コネクタ例： XG4M-5031(オムロン)

表 18.16 CON16 信号配列

ピン番号	信号名	I/O	機能
1	VCC_5V	Power	電源(VCC_5V)
2	VCC_5V	Power	電源(VCC_5V)
3	VCC_3.3V	Power	電源(VCC_3.3V)
4	VCC_3.3V	Power	電源(VCC_3.3V)
5	DGND	Power	電源(DGND)
6	DGND	Power	電源(DGND)
7	A1_LCDO_D16	Out	LCD データ(bit16)、R-Mobile A1 の LCDD16_0 ピンに接続
8	A1_LCDO_D17	Out	LCD データ(bit17)、R-Mobile A1 の LCDD17_0 ピンに接続
9	A1_LCDO_D18	Out	LCD データ(bit18)、R-Mobile A1 の LCDD18_0 ピンに接続
10	LCD0_D19	Out	LCD データ(bit19)、R-Mobile A1 の DBGMD20 ピンに接続
11	LCD0_D20	Out	LCD データ(bit20)、R-Mobile A1 の DBGMD21 ピンに接続
12	LCD0_D21	Out	LCD データ(bit21)、R-Mobile A1 の DBGMDT0 ピンに接続
13	LCD0_D22	Out	LCD データ(bit22)、R-Mobile A1 の DBGMDT2 ピンに接続
14	LCD0_D23	Out	LCD データ(bit23)、R-Mobile A1 の DBGMDT1 ピンに接続
15	A1_LCDO_D8	Out	LCD データ(bit8)、R-Mobile A1 の LCDD8_0 ピンに接続
16	A1_LCDO_D9	Out	LCD データ(bit9)、R-Mobile A1 の LCDD9_0 ピンに接続
17	A1_LCDO_D10	Out	LCD データ(bit10)、R-Mobile A1 の LCDD10_0 ピンに接続
18	A1_LCDO_D11	Out	LCD データ(bit11)、R-Mobile A1 の LCDD11_0 ピンに接続
19	A1_LCDO_D12	Out	LCD データ(bit12)、R-Mobile A1 の LCDD12_0 ピンに接続
20	A1_LCDO_D13	Out	LCD データ(bit13)、R-Mobile A1 の LCDD13_0 ピンに接続
21	A1_LCDO_D14	Out	LCD データ(bit14)、R-Mobile A1 の LCDD14_0 ピンに接続
22	A1_LCDO_D15	Out	LCD データ(bit15)、R-Mobile A1 の LCDD15_0 ピンに接続
23	A1_LCDO_D0	Out	LCD データ(bit0)、R-Mobile A1 の LCDD0_0 ピンに接続
24	A1_LCDO_D1	Out	LCD データ(bit1)、R-Mobile A1 の LCDD1_0 ピンに接続
25	A1_LCDO_D2	Out	LCD データ(bit2)、R-Mobile A1 の LCDD2_0 ピンに接続
26	A1_LCDO_D3	Out	LCD データ(bit3)、R-Mobile A1 の LCDD3_0 ピンに接続
27	A1_LCDO_D4	Out	LCD データ(bit4)、R-Mobile A1 の LCDD4_0 ピンに接続
28	A1_LCDO_D5	Out	LCD データ(bit5)、R-Mobile A1 の LCDD5_0 に接続
29	A1_LCDO_D6	Out	LCD データ(bit6)、R-Mobile A1 の LCDD6_0 ピンに接続
30	A1_LCDO_D7	Out	LCD データ(bit7)、R-Mobile A1 の LCDD7_0 ピンに接続
31	DGND	Power	電源(DGND)
32	A1_LCDO_DCK	Out	DCK 信号、R-Mobile A1 の LCDDCK_0 ピンに接続
33	A1_LCDO_HSYN	Out	HSYNC 信号、R-Mobile A1 の LCDHSYN_0 ピンに接続
34	A1_LCDO_VSYN	Out	VSYNC 信号、R-Mobile A1 の LCDVSYN_0 ピンに接続
35	A1_LCDO_DISP	Out	DISP 信号、R-Mobile A1 の LCDDISP_0 ピンに接続
36	A1_LCDO_DON	Out	DON 信号、R-Mobile A1 の LCDDON_0 ピンに接続
37	A1_LCDO_VEPWC	Out	VEPWC 信号、R-Mobile A1 の LCDVEPWC_0 ピンに接続
38	A1_LCDO_VCPWC	Out	VCPWC 信号、R-Mobile A1 の LCDVCPWC_0 ピンに接続

ピン番号	信号名	I/O	機能
39	A1_LCD0_LCLK	Out	LCLK 信号、R-Mobile A1 の D24 ピンに接続
40	A1_LCD0_LED_CONT	Out	LED_CONT 信号、R-Mobile A1 の MEMC_BUSCLK ピンに接続
41	NC	-	未接続
42	NC	-	未接続
43	NC	-	未接続
44	NC	-	未接続
45	A1_SDA0	In/Out	SDA 信号、R-Mobile A1 の I2C_SDA_0 ピンに接続
46	A1_SCL0	In/Out	SCL 信号、R-Mobile A1 の I2C_SCL_0 ピンに接続
47	TP_INT	In	割り込み信号、R-Mobile A1 の FMSICK ピンに接続
48	TP_RST_B	Out	リセット信号、R-Mobile A1 の D21 ピンに接続
49	VCC_3.3V	Power	電源(VCC_3.3V)
50	DGND	Power	電源(DGND)

18.14. CON17(LCD 拡張インターフェース 2)

CON17 は LCD モジュール接続用インターフェースです。

- ・ 対応 LCD モジュール：AM-800480L1TMQW-T00H または AM-800480L1TMQW-TNOH(AMPIRE)



CON16 と CON17 の信号は共通となっており、同時に使用することはできません。

表 18.17 CON17 信号配列

ピン番号	信号名	I/O	機能
1	VCC_5V	Power	電源(VCC_5V)
2	VCC_5V	Power	電源(VCC_5V)
3	VCC_5V	Power	電源(VCC_5V)
4	NC	-	電源(VCC_3.3V)
5	VCC_3.3V	Power	電源(VCC_3.3V)
6	VCC_3.3V	Power	電源(VCC_3.3V)
7	NC	-	未接続
8	DGND	Power	電源(DGND)
9	DGND	Power	電源(DGND)
10	TP_RST_B	Out	リセット信号、R-Mobile A1 の D21 ピンに接続
11	TP_INT	In	割り込み信号、R-Mobile A1 の FMSICK ピンに接続
12	DGND	Power	電源(DGND)
13	A1_SCL0	In/Out	SCL 信号、R-Mobile A1 の I2C_SCL_0 ピンに接続
14	A1_SDA0	In/Out	SDA 信号、R-Mobile A1 の I2C_SDA_0 ピンに接続
15	DGND	Power	電源(DGND)
16	A1_LCD0_LED_CONT	Out	LED_CONT 信号、R-Mobile A1 の MEMC_BUSCLK ピンに接続
17	DGND	Power	電源(DGND)
18	A1_LCD0_DON	Out	DON 信号、R-Mobile A1 の LCDDON_0 ピンに接続
19	A1_LCD0_DISP	Out	DISP 信号、R-Mobile A1 の LCDDISP_0 ピンに接続
20	A1_LCD0_VSYN	Out	VSYNC 信号、R-Mobile A1 の LCDVSYN_0 ピンに接続
21	A1_LCD0_HSYN	Out	HSYNC 信号、R-Mobile A1 の LCDHSYN_0 ピンに接続
22	A1_LCD0_DCK	Out	DCK 信号、R-Mobile A1 の LCDDCK_0 ピンに接続
23	DGND	Power	電源(DGND)
24	A1_LCD0_D7	Out	LCD データ(bit7)、R-Mobile A1 の LCDD7_0 ピンに接続

ピン番号	信号名	I/O	機能
25	A1_LCD0_D6	Out	LCD データ(bit6)、R-Mobile A1 の LCDD6_0 ピンに接続
26	A1_LCD0_D5	Out	LCD データ(bit5)、R-Mobile A1 の LCDD5_0 ピンに接続
27	A1_LCD0_D4	Out	LCD データ(bit4)、R-Mobile A1 の LCDD4_0 ピンに接続
28	A1_LCD0_D3	Out	LCD データ(bit3)、R-Mobile A1 の LCDD3_0 ピンに接続
29	A1_LCD0_D2	Out	LCD データ(bit2)、R-Mobile A1 の LCDD2_0 ピンに接続
30	A1_LCD0_D1	Out	LCD データ(bit1)、R-Mobile A1 の LCDD1_0 ピンに接続
31	A1_LCD0_D0	Out	LCD データ(bit0)、R-Mobile A1 の LCDD0_0 ピンに接続
32	DGND	Power	電源(DGND)
33	A1_LCD0_D15	Out	LCD データ(bit15)、R-Mobile A1 の LCDD15_0 ピンに接続
34	A1_LCD0_D14	Out	LCD データ(bit14)、R-Mobile A1 の LCDD14_0 ピンに接続
35	A1_LCD0_D13	Out	LCD データ(bit13)、R-Mobile A1 の LCDD13_0 ピンに接続
36	A1_LCD0_D12	Out	LCD データ(bit12)、R-Mobile A1 の LCDD12_0 ピンに接続
37	A1_LCD0_D11	Out	LCD データ(bit11)、R-Mobile A1 の LCDD11_0 ピンに接続
38	A1_LCD0_D10	Out	LCD データ(bit10)、R-Mobile A1 の LCDD10_0 ピンに接続
39	A1_LCD0_D9	Out	LCD データ(bit9)、R-Mobile A1 の LCDD9_0 ピンに接続
40	A1_LCD0_D8	Out	R-Mobile A1 の LCDD8_0 ピンに接続
41	DGND	Power	電源(DGND)
42	LCD0_D23	Out	LCD データ(bit23)、R-Mobile A1 の DBGMDT1 ピンに接続
43	LCD0_D22	Out	LCD データ(bit22)、R-Mobile A1 の DBGMDT2 ピンに接続
44	LCD0_D21	Out	LCD データ(bit21)、R-Mobile A1 の DBGMDT0 ピンに接続
45	LCD0_D20	Out	LCD データ(bit20)、R-Mobile A1 の DBGMD21 ピンに接続
46	LCD0_D19	Out	LCD データ(bit19)、R-Mobile A1 の DBGMD20 ピンに接続
47	A1_LCD0_D18	Out	LCD データ(bit18)、R-Mobile A1 の LCDD18_0 ピンに接続
48	A1_LCD0_D17	Out	LCD データ(bit17)、R-Mobile A1 の LCDD17_0 ピンに接続
49	A1_LCD0_D16	Out	LCD データ(bit16)、R-Mobile A1 の LCDD16_0 ピンに接続
50	DGND	Power	電源(DGND)

18.15. CON19(電源入力インターフェース)

CON19 は電源を供給する DC ジャックです。AC アダプターのジャック形状は EIAJ RC-5320A 準拠 (電圧区分 2) です。




図 18.1 AC アダプターの極性マーク

表 18.18 CON19 信号配列

ピン番号	信号名	I/O	機能
1	VCC_5V	Power	電源(VCC_5V)
2	DGND	Power	電源(DGND)
3	DGND	Power	電源(DGND)
4	DGND	Power	電源(DGND)

18.16. CON20(USB インターフェース 1)

CON20 は USB ホストインターフェースです。Type A コネクタを実装しています。



CON20 と CON24 の信号は共通となっており、同時に使用することはできません。

CON20 と CON24 で使用する各信号をどちらのインターフェースに接続させるかは、ディップスイッチ(SW1)の USB0 設定で選択できるように

なっています。詳しくは、「表 4.3. ディップスイッチ(SW1)のスイッチの機能」または、「18.25. SW1(機能選択スイッチ)」を参照してください。

表 18.19 CON20 信号配列

ピン番号	信号名	I/O	機能
1	VCC_5V	Power	電源(VCC_5V)
2	A1_DM0	In/Out	USB マイナス側信号、R-Mobile A1 の DM_0 ピンに接続
3	A1_DP0	In/Out	USB プラス側信号、R-Mobile A1 の DP_0 ピンに接続
4	DGND	Power	電源(DGND)
5	DGND	Power	電源(DGND)
6	DGND	Power	電源(DGND)
7	DGND	Power	電源(DGND)

18.17. CON21(USB インターフェース 2)

USB ホストインターフェースです。Type A コネクタを実装しています。

表 18.20 CON21 信号配列

ピン番号	信号名	I/O	機能
1	VCC_5V	Power	電源(VCC_5V)
2	A1_DM1	In/Out	USB マイナス側信号、R-Mobile A1 の DM_1 ピンに接続
3	A1_DP1	In/Out	USB プラス側信号、R-Mobile A1 の DP_1 ピンに接続
4	DGND	Power	電源(DGND)
5	DGND	Power	電源(DGND)
6	DGND	Power	電源(DGND)
7	DGND	Power	電源(DGND)

18.18. CON22(シリアルインターフェース)

非同期(調歩同期)シリアルインターフェースです。D-Sub 9 ピンを実装しています。RS232C トランシーバを経由して R-Mobile A1 と接続されています。

表 18.21 CON22 信号配列

ピン番号	信号名	I/O	機能
1	NC	-	未接続
2	UART_RXD	In	UART 受信データ
3	UART_TXD	Out	UART 送信データ
4	NC	-	未接続
5	DGND	Power	電源(DGND)
6	RTS	Out	常時 High
7	CTS	-	未接続
8	NC	-	未接続
9	NC	-	未接続

18.19. CON23(LAN インターフェース)

10BASE-T/100BASE-TX の LAN インターフェースです。RJ45 コネクタを実装しています。イーサネット PHY を経由して R-Mobile A1 と接続されています。

表 18.22 CON23 信号配列

ピン番号	信号名	I/O	機能
1	TX+	Out	差動のツイストペア送信出力(+)
2	TX-	Out	差動のツイストペア送信出力(-)
3	RX+	In	差動のツイストペア受信入力(+)
4	-	-	75Ω 終端、CON23(5 ピン)とコネクタ内部で接続
5	-	-	75Ω 終端、CON23(4 ピン)とコネクタ内部で接続
6	RX-	In	差動のツイストペア受信入力(-)
7	-	-	75Ω 終端、CON23(8 ピン)とコネクタ内部で接続
8	-	-	75Ω 終端、CON23(7 ピン)とコネクタ内部で接続

18.20. CON24(USB インターフェース 3)

CON24 は USB デバイスインターフェースです。Type B コネクタを実装しています。



CON20 と CON24 の信号は共通となっており、同時に使用することはできません。

CON20 と CON24 で使用する各信号をどちらのインターフェースに接続させるかは、ディップスイッチ(SW1)の USB0 設定で選択できるようになっています。詳しくは、「表 4.3. ディップスイッチ(SW1)のスイッチの機能」または、「18.25. SW1(機能選択スイッチ)」を参照してください。

表 18.23 CON24 信号配列

ピン番号	信号名	I/O	機能
1	A1_VBUS	In	VBUS 検出 ^[a] 、R-Mobile A1 の VBUS ピンに接続
2	A1_DM0	In/Out	USB マイナス側信号、R-Mobile A1 の DM_0 ピンに接続
3	A1_DP0	In/Out	USB プラス側信号、R-Mobile A1 の DP_0 ピンに接続
4	DGND	Power	電源(DGND)
5	FG	Power	電源(DGND)
6	FG	Power	電源(DGND)
7	FG	Power	電源(DGND)

^[a]Armadillo-800 EVA 本体への電源供給としては使用できません。

18.21. LED1(カメラ LED)

LED1 はカメラモジュールの FLASH LED です。

表 18.24 LED1 の挙動

LED	名称(色)	説明
LED1	LED(黄色)	カメラモジュールの GPIO4_FLASH に接続 (Low: 消灯、High: 点灯)

18.22. LED2(電源 LED)

LED2 は電源 LED です。VCC_3.3V が供給されると点灯します。

表 18.25 LED の挙動

LED	名称(色)	点灯	消灯
LED2	電源 LED(緑色)	VCC_3.3V が供給されていることを示します。	VCC_3.3V が供給されていないことを示します。

18.23. LED3～LED6(ユーザー LED)

LED3～LED6 はユーザー LED です。

表 18.26 LED3～LED6 の挙動

LED	名称(色)	説明
LED3	ユーザー LED(黄色)	R-Mobile A1 の FRB ピンに接続(Low: 消灯、High: 点灯)
LED4	ユーザー LED(黄色)	R-Mobile A1 の CS4#ピンに接続(Low: 消灯、High: 点灯)
LED5	ユーザー LED(黄色)	R-Mobile A1 の CS2#ピンに接続(Low: 消灯、High: 点灯)
LED6	ユーザー LED(黄色)	R-Mobile A1 の WAIT#ピンに接続(Low: 消灯、High: 点灯)

18.24. LED7、LED8(LAN LED)

LED7、LED8 は LAN インターフェース(CON23)のステータス LED です。

表 18.27 LED7、LED8 の挙動

LED	名称(色)	点灯	消灯	点滅
LED7	LINK/ACT LED(緑色)	LAN ケーブルが接続されており、リンクが確立されている。	LAN ケーブルが接続されており、リンクが確立されていない。	データを送受信している。
LED8	SPEED LED(黄色)	100BASE-TX のリンクが確立されている。	LAN ケーブルが接続されている場合、10BASE-T のリンクが確立されている。または、リンクが確立されていない。	-

18.25. SW1(機能選択スイッチ)

SW1 はディップスイッチ 8 接点です。スイッチを OFF に設定すると High、ON に設定すると Low が各信号に入力されます。

表 18.28 SW1 信号配列

ピン番号	信号名	機能
1	A1_PORT101_JP2	R-Mobile A1 の FCE0#ピンに接続
2	A1_MD2	R-Mobile A1 の MD2 ピンに接続
3	A1_MD3	R-Mobile A1 の MD3 ピンに接続
4	MMC_DISABLE	eMMC と拡張バスインターフェース(CON2)の切り替え (ON: eMMC、OFF: 拡張バスインターフェース)
5	SDSLOT2_ENABLE	SD インターフェース 2(CON8)と AWL13 モジュール(CON14)の切り替え (ON: SD インターフェース 2、OFF: AWL13 モジュール)
6	USB_DEVICE_MODE	USB デバイスインターフェース(CON24)と USB ホストインターフェース(CON20)の切り替え (ON: USB デバイスインターフェース、OFF: USB ホストインターフェース)
7	DBGMD20	R-Mobile A1 の DBGMD20 ピンに接続
8	DBGMD21	R-Mobile A1 の DBGMD21 ピンに接続

18.26. SW2(リセットスイッチ)

SW2 はリセットスイッチです。タクトスイッチを実装しています。基板上的リセット IC に接続されています。

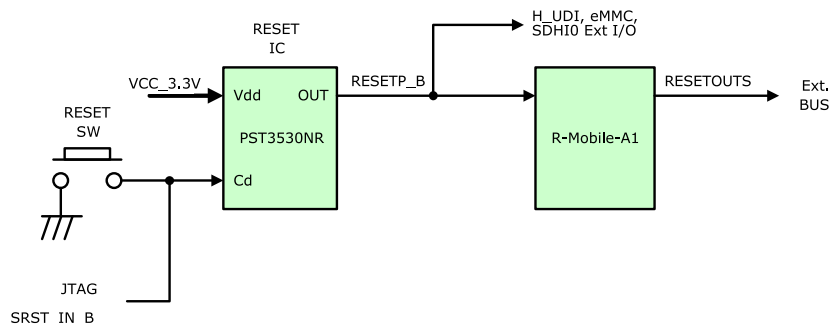


図 18.2 リセットブロック図

表 18.29 SW2 の機能

SW	機能
SW2	リセット(押された状態: リセット状態、押されていない状態: リセット解除)

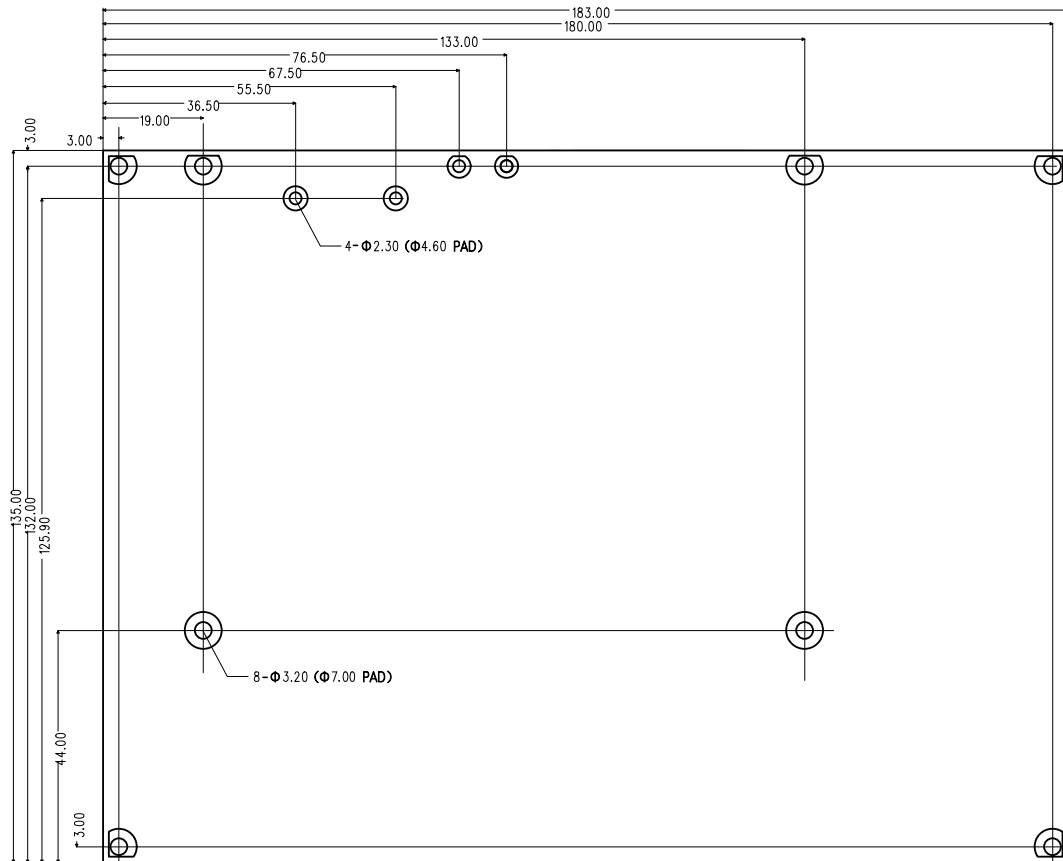
18.27. SW3～SW6(ユーザースイッチ)

SW3～SW6 はユーザースイッチです。タクトスイッチを実装しています。スイッチが押されている時は Low、押されていない時は High が各ピンに入力されます。

表 18.30 SW3～SW6 の機能

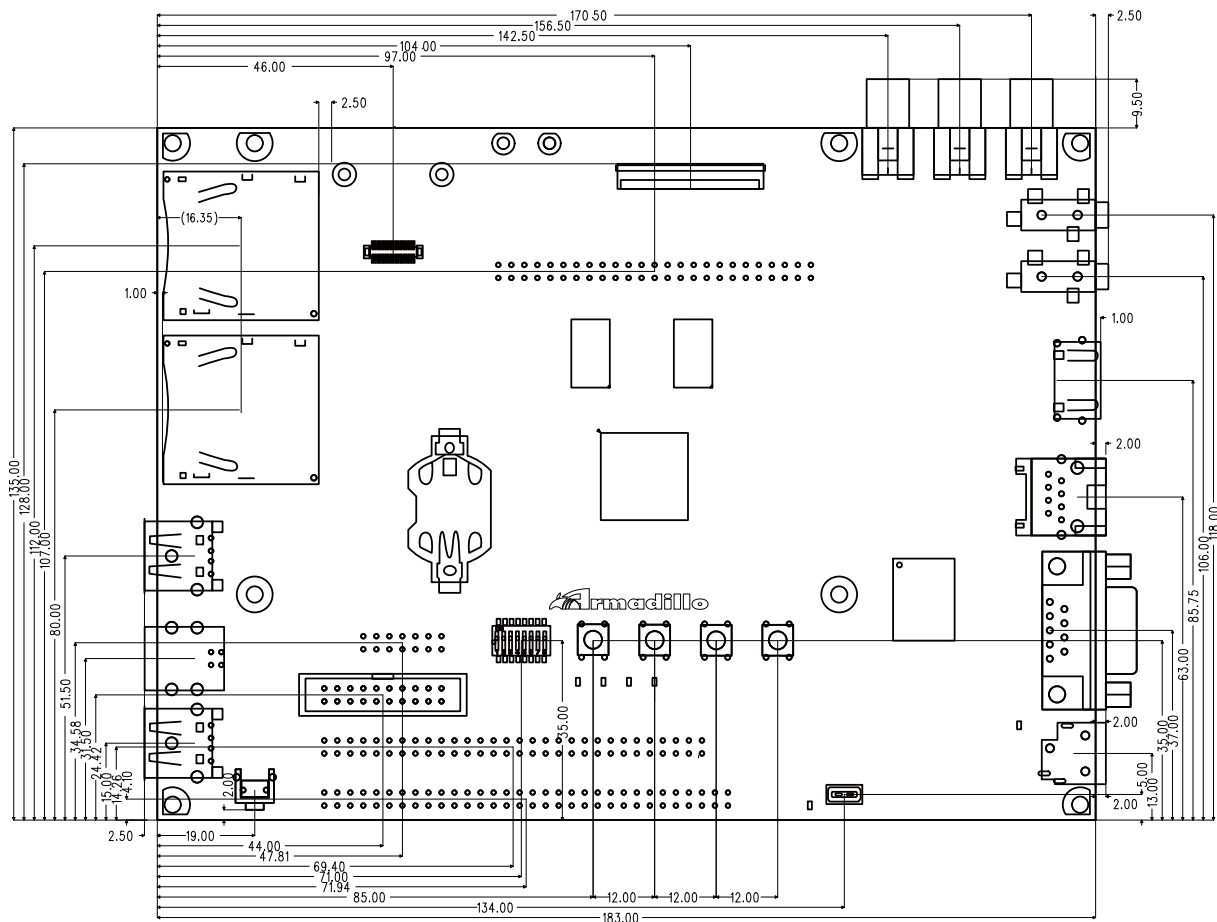
SW	説明
SW3	R-Mobile A1 の MEMC_AD6 ピンに接続
SW4	R-Mobile A1 の MEMC_AD7 ピンに接続
SW5	R-Mobile A1 の MEMC_AD4 ピンに接続
SW6	R-Mobile A1 の MEMC_AD5 ピンに接続

19. 基板形状図



[Unit : mm]

図 19.1 基板形状および固定穴寸法



[Unit : mm]

図 19.2 コネクタ中心寸法

付録 A Hermit-At ブートローダー

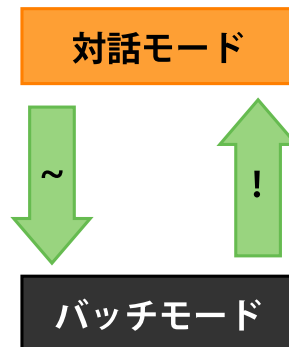
Hermit-At は、アットマークテクノ製品に採用されている高機能ダウンローダー兼ブートローダー^[1]です。Armadillo を保守モードで起動すると、Hermit-At ブートローダーのプロンプトが表示されます。プロンプトからコマンドを入力することにより、フラッシュメモリの書き換えや、Linux カーネルパラメーターの設定等 Hermit-At ブートローダーの様々な機能を使用することができます。ここでは、代表的な機能について説明します。



Hermit-AT のモード

Hermit-AT には、2つのモードがあります。コマンドプロンプトを表示して対話的に動作する「対話モード」と、Hermit-AT ダウンローダと通信するための「バッチモード」です。バッチモードではコマンドプロンプトの表示や入力した文字の表示を行いませんが、コマンドの実行は可能です。

起動直後の Hermit-AT は必ず対話モードになっています。対話モードからバッチモードに移行するにはチルダ「~」を、バッチモードから対話モードに移行するにはエクスクラメーションマーク「!」を入力します。



Hermit-AT ダウンローダと通信を行った場合は、バッチモードに移行します。これは通信を確立するために Hermit-AT ダウンローダがチルダを送信するためです。

対話モードからバッチモードに移行したり、バッチモード中に入力したコマンドが成功した場合などは以下のように表示されます。

```
+OK
```

A.1. version

バージョン情報を表示するコマンドです。

^[1]Armadillo-800 EVA では、Hermit-At のブートローダー機能のみ利用可能です。

```
構文 : version
```

図 A.1 version 構文

A.1.1. version 使用例

```
hermit> version  
Hermit-At v3.0.0 (Armadillo-800 EVA) compiled at 22:22:10, Dec 21 2011
```

図 A.2 version の使用例

A.2. info

ボード情報を表示するコマンドです。

```
構文 : info
```

図 A.3 info 構文

A.2.1. info 使用例

```
hermit> info  
Board Type: 0x08000000  
Revision: 0x00000003  
Lot: 0x00000001  
Serial Number: 0  
Boot Mode: 0x00000038 (eMMC)  
Jumper: ON  
Tact-SW: OFF, OFF, OFF, OFF  
ORIG MAC-1: 00:11:0c:00:00:00
```

図 A.4 info の使用例

A.3. mac

MAC アドレスを表示するコマンドです。

```
構文 : mac
```

図 A.5 mac 構文

A.3.1. mac 使用例

```
hermit> mac
00:11:0c:00:00:00
```

図 A.6 mac の使用例

A.4. setenv と clearenv

Linux カーネルパラメーターを設定するコマンドです。setenv で設定されたパラメータは、Linux カーネルブート時にカーネルに渡されます。clearenv を実行すると、設定がクリアされます。このパラメータは、内蔵ストレージに保存され再起動後も設定は有効となります。

構文：setenv [カーネルパラメーター]...
 説明：カーネルパラメーターを設定します。オプションを指定せずに実行すると、現在の設定を表示します。

構文：clearenv
 説明：設定されているオプションをクリアします。

図 A.7 setenv/clearenv 構文

A.4.1. setenv/clearenv 使用例

```
hermit> setenv console=ttySC1,115200
hermit> setenv
1: console=ttySC1,115200
hermit> clearenv
hermit> setenv
hermit>
```

図 A.8 setenv と clearenv の使用例

A.4.2. Linux カーネルパラメーター

Linux カーネルパラメーターの例を、「表 A.1. よく使用される Linux カーネルパラメーター」に示します。その他のオプションについては、[linux-2.6/Documentation/kernel-parameters.txt](#) を参照してください。

表 A.1 よく使用される Linux カーネルパラメーター

オプション	説明
console	カーネルコンソールとして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。
rootdelay	ルートファイルシステムをマウントする前に指定秒間待機します。
rootwait	ルートファイルシステムがアクセス可能になるまで待機します。
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。

A.5. setbootdevice

Linux カーネルを格納しているブートデバイスを指定するコマンドです。この設定は内蔵ストレージに保存され、再起動後も設定は有効となります。

構文：setbootdevice mmcblk0pN

説明：内蔵ストレージのパーティション N の /boot/ ディレクトリに置かれたカーネルイメージを RAM に展開してブートします

↩

構文：setbootdevice mmcblk1pN

説明：SD カード(CON7)のパーティション N の /boot/ ディレクトリに置かれたカーネルイメージを RAM に展開してブートします

↩

図 A.9 setbootdevice 構文

A.5.1. setbootdevice の使用例

内蔵ストレージのパーティション 4 の /boot/ ディレクトリに置かれたカーネルイメージでブートするには、「図 A.10. ブートデバイスに内蔵ストレージのパーティション 4 を指定する」のようにコマンドを実行します。

```
hermit> setbootdevice mmcblk0p4
```

図 A.10 ブートデバイスに内蔵ストレージのパーティション 4 を指定する

SD カード(CON7)のパーティション 1 に格納されたカーネルイメージでブートするには、「図 A.11. ブートデバイスに SD カードを指定する」のようにコマンドを実行します。

```
hermit> setbootdevice mmcblk1p1
```

図 A.11 ブートデバイスに SD カードを指定する

A.6. frob

指定したアドレスのデータを読み込む、または、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 A.2 frob コマンド

frob コマンド	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。
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。
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。
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。
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。
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。

A.7. boot

setbootdevice で指定されたブートデバイスから Linux カーネルをブートするコマンドです。

```
構文 : boot
```

図 A.12 boot 構文

A.7.1. boot 使用例

```
hermit> boot
mmcscd: SD card at address 0x00000001
mmcscd: M8G2FA 1048576KiB
gendisk: /dev/mmcblk0p4: start=0x000f4280, size=0x001dc0c0
gendisk: Image.bin is found. (4390496 Bytes)
Copying      kernel...done.           ❶
Doing console=ttySC1,115200
Doing noinitrd
Doing rootwait
Doing root=/dev/mmcblk0p4
Doing init=/init                       ❷
Linux version 2.6.35.7 (atmark@atde4) (gcc version 4.4.5 (Debian 4.4.5-8) )
#1 PREEMPT Wed Dec 21 22:37:47 JST 2011  ❸
CPU: ARMv7 Processor [412fc093] revision 3 (ARMv7), cr=10c53c7f
CPU: VIPT nonaliasing data cache, VIPT nonaliasing instruction cache
Machine: Armadillo-800EVA
:
:
```

図 A.13 boot の使用例

- ❶ カーネルイメージ RAM 上に展開しています。
- ❷ setenv でカーネルパラメーターを設定している場合、ここで表示されます。ここまでは Hermit-At が表示しています。
- ❸ カーネルがブートされ、カーネルの起動ログが表示されます。

付録 B LCD パネルのドット欠けについて

LCD パネルはその性質上、一定の割合でドット欠け(点欠陥)が生じます。Armadillo-800 EVA に使用されている LCD パネルの点欠陥の許容範囲は、以下の基準に従います。

B.1. 点欠陥の定義

表 B.1 点欠陥の定義

欠陥	定義
輝点欠陥	全黒表示画面において、周辺同色画素より明るいと認識される点欠陥。
黒点欠陥	全白表示画面において、周辺同色画素より暗いと認識される点欠陥。
連続点欠陥	輝点、黒点の点欠陥が複数にわたり連続して発生している物。 黒点-黒点、輝点-輝点のいずれの場合についても連続点欠陥とする。

B.2. 検査基準

表 B.2 点欠陥許容範囲

欠陥	許容範囲
輝点欠陥	5 個
黒点欠陥	8 個
2 連続点欠陥	1 組(輝点) 1 組(黒点)
連続点欠陥(3 連続以上)	0 個(輝点、黒点とも)
欠陥総数	8 個

付録 C LCD パネル固定部品の変更

Armadillo-800 EVA に同梱する LCD パネルを LCD 拡張ボードに固定する部品を、2015 年 2 月出荷分から変更しました。以前は両面テープで固定していましたが、脱着を可能にするためネジ止めに変更しました。

表 C.1 変更部品

	変更前	変更後
部品型番	7811 (発泡体両面テープ)	AT2090A (固定部品、ABS 樹脂)
部品メーカー	TERAOKA SEISAKUSHO	アットマークテクノ

C.1. 変更による影響

表 C.2 変更部品

項目	内容	製品仕様への影響	
仕様	電気的特性	変更なし	本変更に伴う対象製品本体の機能、寸法、電気的仕様、使用方法の変更等はありません
	外観仕様	固定部品の形状が変更になります(「C.2. 外観・寸法図」参照)	
	寸法仕様	固定部品の寸法が変更になります(「C.2. 外観・寸法図」参照)	
	機能・性能	変更なし	

C.2. 外観・寸法図

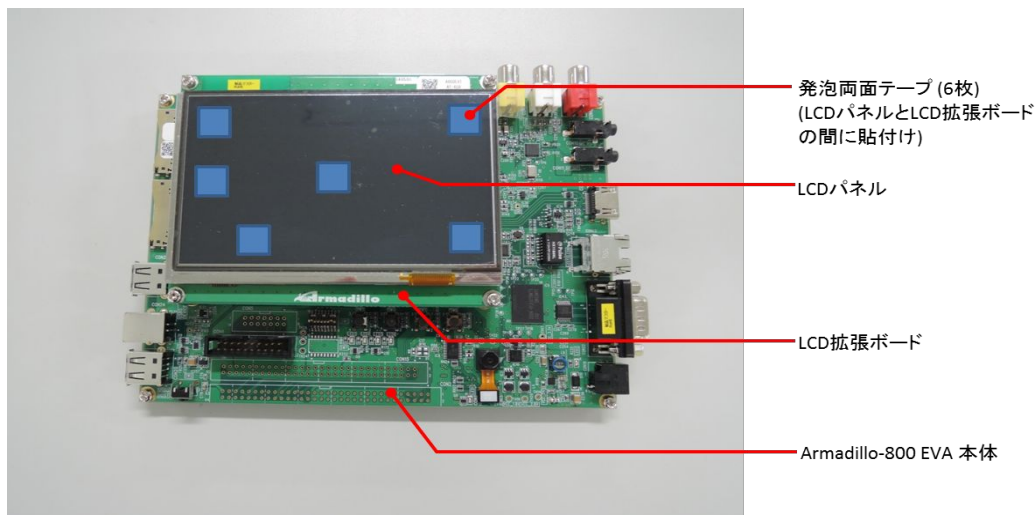


図 C.1 外観: 変更前

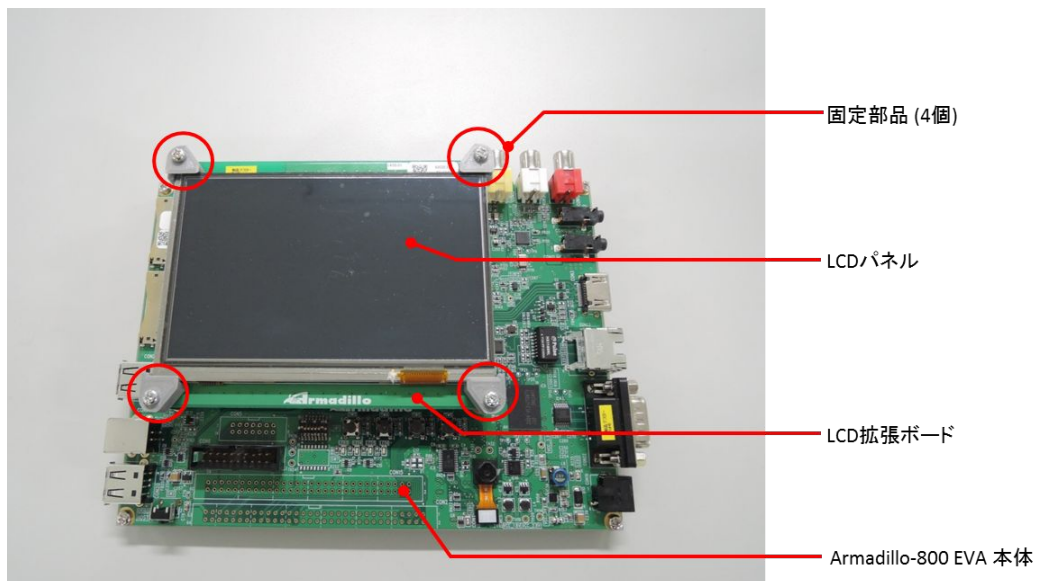


図 C.2 外観: 変更後

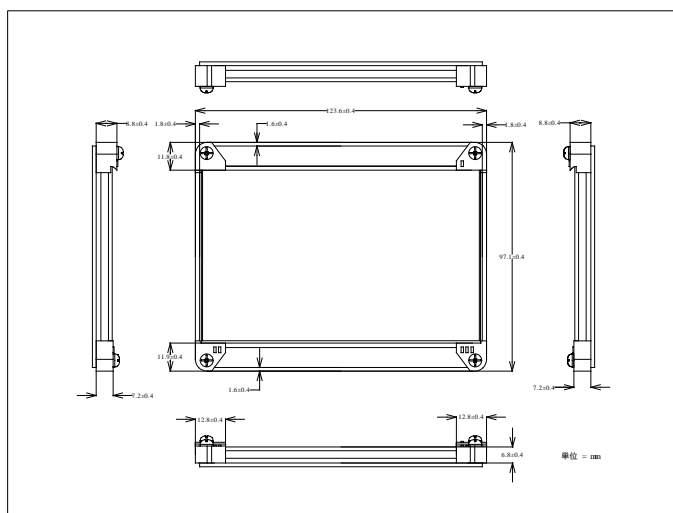


図 C.3 寸法図

改訂履歴

バージョン	年月日	改訂内容
1.0.0	2012/01/05	<ul style="list-style-type: none"> ・ 初版発行
1.1.0	2012/02/23	<ul style="list-style-type: none"> ・ linux-2.6.35-a800eva-at2 に対応 ・ 各種機能の使用方法を、「8. Debian GNU/Linux で各種機能を使用する」に集約 ・ 「8. Debian GNU/Linux で各種機能を使用する」に、CON3/12/13 に接続した機器、外部ストレージ、LED バックライト、LED、ユーザースイッチの使用方法を追記 ・ Linux カーネルの説明を、「17. Linux カーネル仕様」に集約 ・ 「17. Linux カーネル仕様」に、デフォルトコンフィギュレーション、Android 機能の説明を追記 ・ 「1.2. 本書の構成」の説明を更新 ・ 「5.2. 接続方法」に、AC アダプターを接続時の注意書き追加 ・ 「表 18.3. CON3 信号配列」の DDC_CEC 信号の機能を未接続に変更 ・ 誤記、表記ゆれ修正
1.2.0	2012/03/22	<ul style="list-style-type: none"> ・ root ユーザー権限で実行すべきコマンド入力例が、一般ユーザー権限での表示になっていたものを修正 ・ 「表 18.15. CON15 信号配列」信号名に関する誤記修正(A1_SCL を A1_SCL1 に、A1_SDA を A1_SDA1 に) ・ 「8.6.2. X サーバーの起動」で起動確認がしやすいように手順を修正 ・ 「8.6.2. X サーバーの起動」に CON4 に接続されたディスプレイを使用する方法を追記 ・ 「11.2.1. クロス開発ツールが依存する Debian パッケージのインストール」追記 ・ 付録 B LCD パネルのドット欠けについて追加
1.3.0	2012/07/30	<ul style="list-style-type: none"> ・ linux-2.6.35-a800eva-at3 に対応 ・ 「8.9. USB ガジェット」追加 ・ ブートローダーイメージファイルを Hermit-At v3.1.0 に合わせて修正 ・ 「表 18.23. CON24 信号配列」の A1_VBUS ピンの誤記修正 ・ ディップスイッチ設定の説明画像、「図 8.3. ディップスイッチの SDHI1 設定(AWL13 モジュールインターフェース有効)」、「図 8.25. ディップスイッチの USB0 設定(USB インターフェース 3 有効)」、「図 8.33. ディップスイッチの SDHI1/USB1 設定」を追加 ・ 「18.12. CON15(拡張インターフェース)」の CAM_DISABLE ピンについての説明を変更 ・ 「3.1. ボード概要」に記載していた GPIO の本数に関する誤記修正 ・ 「4.2. ディップスイッチ」に起動デバイスを拡張バス(CSO)に設定する際の注意事項を追加 ・ Android 機能についての説明を変更 ・ その他誤記、表記ゆれ修正
1.3.1	2012/10/23	<ul style="list-style-type: none"> ・ 回路図、部品表の参照先についての文言を修正 ・ 「図 19.1. 基板形状および固定穴寸法」に穴径、PAD 情報を追加
1.3.2	2014/12/24	<ul style="list-style-type: none"> ・ 横浜営業所の住所追記 ・ 「1.4. 謝辞」の記載内容を一部変更 ・ 「2.6. 輸出について」の記載内容を一部変更 ・ 「2.7. 商標について」に SDXC・microSDXC を追加

		<ul style="list-style-type: none">・ LCD モジュールの変更に伴い、対応する LCD モジュールの型番を追加・ 誤記修正
1.4.0	2015/02/06	<ul style="list-style-type: none">・ 付録 C LCD パネル固定部品の変更を追加

Armadillo-800 EVA 製品マニュアル
Version 1.4.0
2015/02/06

株式会社アットマークテクノ

札幌本社

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

横浜営業所

〒221-0835 横浜市神奈川区鶴屋町3丁目 30-4 明治安田生命横浜西口ビル 7F
TEL 045-548-5651 FAX 050-3737-4597
