Armadillo-810
製品マニュアル

A8100-U00Z
A8101-D00Z

Version 1.4.1
2014/07/31

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-810 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

Version 1.4.1
2014/07/31

目次

図目次

表目次

1. はじめに

このたびは Armadillo-810 をお求めいただき、ありがとうございます。

Armadillo-810 は、ルネサスエレクトロニクス製 Cortex-A9 プロセッサ「R-Mobile A1」、DDR3 SDRAM、フラッシュメモリを中心に、カメラインターフェース、シリアルポート、USB 2.0 デバイスポートなどを搭載し、且つ、拡張用コネクタには USB 2.0 ホストインターフェース、SD/SDIO インターフェース、SPI、GPIO などといった組み込みシステムに求められる機能を備える小型 CPU ボードです。

Armadillo-810 は、インテリジェントカメラのプラットフォームとして利用することを想定して設計されています。カメラから取得した画像を加工・解析を行い特定用途向けのデータに変換して送信することができます。Armadillo-810 カメラモデル開発セットには、シキノハイテック製カメラモジュールが搭載され、且つ必要なソフトウェアが同梱されていますので、ご購入後すぐにシステム開発をスタートすることができます。

Armadillo-800 シリーズは標準 OS に Linux を採用していますので、Linux の豊富なソフトウェア資産を利用することができます。また、C 言語などのプログラミング言語を使用し、オリジナルのプログラムを作成して動作させることも可能です。

尚、Armadillo-810 には、**ご購入ユーザーに限定して公開しているソフトウェアやハードウェア情報**があります。主な限定コンテンツを次に示します。

- FSE
- AV コーデックミドルウェア
- 拡張ボードの回路図

限定コンテンツを取得するには、「22. ユーザー登録」を参照してください。

以降、本書では他の Armadillo ブランド製品にも共通する記述については、製品名を Armadillo と表記します。

1.1. 本書および関連ファイルのバージョンについて

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。本書を読み始める前に、Armadillo サイトで最新版の情報をご確認ください。

Armadillo サイト - Armadillo-810 ドキュメント・ダウンロード

http://armadillo.atmark-techno.com/armadillo-810/downloads

1.2. 本書の構成

本書には、ご利用にあたっての注意事項や、ご購入時のソフトウェアの状態、ハードウェア・ソフトウェアをカスタマイズする場合に必要な情報などが記載されています。

◆ はじめにお読みください。

「1. はじめに」、「2. 注意事項」

◆ **Armadillo-810 の仕様を紹介します。**

「3. システム概要」

◆ **工場出荷状態のソフトウェアの使い方や、動作を確認する方法を紹介します。**

「4. 作業の前に」、「5. 起動と終了」、「6. 動作確認方法」、「7. コンフィグ領域 – 設定ファイルの保存領域」

◆ **工場出荷状態のソフトウェア仕様について紹介します。**

「8. Linux カーネル仕様」、「9. ユーザーランド仕様」、「10. ブートローダー仕様」

◆ **システム開発に必要な情報を紹介します。**

「11. ビルド手順」、「12. フラッシュメモリの書き換え方法」、「13. 開発の基本的な流れ」、「14. プログラミングガイド」、「15. AV コーデックミドルウェア」、「16. SD ブートの活用」、「17. JTAG ICE を利用する」

◆ **ハードウェアをカスタマイズする場合に必要な情報を紹介します。**

「19. ハードウェア仕様」、「20. 基板形状図」、「21. 拡張ボード/オプションモジュール」

◆ **ご購入ユーザーに限定して公開しているソフトウェアの紹介やユーザー登録について紹介します。**

「18. 顔認識ミドルウェア「FSE」」、「22. ユーザー登録」

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りたっています。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. 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。

- ご使用の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。

2.3. 取扱い上の注意事項

本製品に恒久的なダメージをあたえないよう、取扱い時には以下のような点にご注意ください。

破損しやすい箇所	カメラモジュールは、破損しやすい部品になっています。無理に力を加えて破損することのないよう十分注意してください。
本製品の改造	本製品に改造[1]を行った場合は保証対象外となりますので十分ご注意ください。また、改造やコネクタ等の増設[2]を行う場合は、作業前に必ず動作確認を行ってください。
電源投入時のコネクタ着脱	本製品や周辺回路に電源が入っている状態で、活線挿抜対応インターフェース(SD/SDIO, USB)以外へのコネクタ着脱は、絶対に行わないでください。
静電気	本製品には CMOS デバイスを使用していますので、ご使用になる時までは、帯電防止対策された出荷時のパッケージ等にて保管してください。
ラッチアップ	電源および入出力からの過大なノイズやサージ、電源電圧の急激な変動等により、使用している CMOS デバイスがラッチアップを起こす可能性があります。いったんラッチアップ状態となると、電源を切断しないかぎりこの状態が維持されるため、デバイスの破損につながることがあります。ノイズの影響を受けやすい入出力ラインには、保護回路を入れることや、ノイズ源となる装置と共通の電源を使用しない等の対策をとることをお勧めします。
衝撃	落下や衝撃などの強い振動を与えないでください。

2.4. ソフトウェア使用に関しての注意事項

本製品に含まれるソフトウェアについて	本製品の標準出荷状態でプリインストールされている Linux 対応ソフトウェアは、個別に明示されている（書面、電子データでの通知、口頭での通知を含む）場合を除き、オープンソースとしてソースコードが提供されています。再配布等の権利については、各ソースコードに記載のライセンス形態にしたがって、お客様の責任において行使してください。また、本製品に含まれるソフトウェア（付属のドキュメント等も含む）は、現状有姿（AS IS）にて提供します。お客様ご自身の責任において、使用用途・目的の適合について事前に十分な検討と試験を実施した上でお使いください。

[1] コネクタ非搭載箇所へのコネクタ等の増設は除く。
[2] コネクタを増設する際にはマスキングを行い、周囲の部品に半田くず、半田ボール等付着しないよう十分にご注意ください。

アットマークテクノは、当該ソフトウェアが特定の目的に適合すること、ソフトウェアの信頼性および正確性、ソフトウェアを含む本製品の使用による結果について、お客様に対し何らの保証も行いません。

パートナー等の協力により Armadillo ブランド製品向けに提供されているミドルウェア、その他各種ソフトウェアソリューションは、ソフトウェア毎にライセンスが規定されています。再頒布権等については、各ソフトウェアに付属する readme ファイル等をご参照ください。その他のバンドルソフトウェアについては、各提供元にお問い合わせください。

2.5. 書込み禁止領域について

EEPROM のデータは、本製品に含まれるソフトウェアで使用しています。正常に動作しなくなる可能性があるため、書込みを行わないでください。また、意図的に書込みを行った場合は保証対象外となります。

2.6. 電波障害について

Armadillo-810 は、クラス A 情報技術装置です。この装置を家庭環境で使用すると電波妨害を引き起こすことがあります。この場合には使用者が適切な対策を講ずるよう要求されることがあります。VCCI-A

2.7. 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

2.8. 輸出について

本製品の開発・製造は、原則として日本国内での使用を想定して実施しています。本製品を輸出する際は、輸出者の責任において、輸出関連法令等を遵守し、必要な手続きを行ってください。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。本製品および関連技術は、大量破壊兵器の開発目的、軍事利用その他軍事用途の目的、その他国内外の法令および規則により製造・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。

2.9. 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. システム概要

3.1. 製品の特長

3.1.1. Armadillo とは

「Armadillo (アルマジロ)」は、ARM コアプロセッサ搭載・Linux 対応の組み込み機器プラットフォームのブランドです。Armadillo ブランド製品には以下の特長があります。

◆ ARM プロセッサ搭載・省電力設計

ARM コアプロセッサを搭載しています。1 〜数ワット程度で動作する省電力設計で、発熱が少なくファンを必要としません。

◆ 小型・手のひらサイズ

CPU ボードは名刺サイズ程度の手のひらサイズが主流です。名刺 1 / 3 程度の小さな CPU モジュールや無線 LAN モジュール等、超小型のモジュールもラインアップしています。

◆ 標準 OS として Linux をプリインストール

標準 OS に Linux を採用しており、豊富なソフトウェア資産と実績のある安定性を提供します。ソースコードをオープンソースとして公開しています。

◆ 開発環境

Armadillo の開発環境として、「Atmark Techno Development Environment (ATDE)」を無償で提供しています。ATDE は、VMware など仮想マシン向けのデータイメージです。このイメージには、Linux デスクトップ環境をベースに GNU 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。ATDE を使うことで、開発用 PC の用意やツールのインストールなどといった開発環境を整える手間を軽減することができます。

3.1.2. Armadillo-810 とは

Armadillo-810 は、ルネサスエレクトロニクス製 Cortex-A9 プロセッサ「R-Mobile A1」、DDR3 SDRAM、フラッシュメモリを中心に構成された低消費電力なインテリジェントカメラ向け組み込みプラットフォームです。

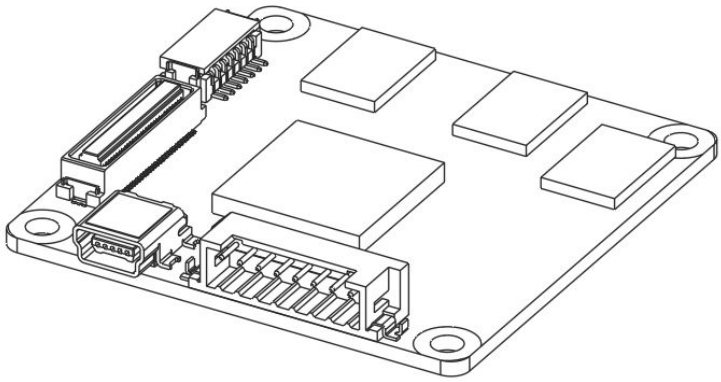


図 3.1 Armadillo-810

◆ 小型・省電力、-20℃～+ 70℃まで動作

50mm×50mm の手のひらサイズで実現した、組み込み CPU ボードです。動作温度範囲は−20℃～+ 70℃まで対応しており、使用環境を選びません。また、試作から量産まで安心して使うことができます。

◆ カメラ入力と画像処理に特化

YUV422 8/16bit、最大 8188×8188 ピクセルのカメラを接続可能なカメラインターフェースを搭載。高性能 Cortex-A9 プロセッサや NEON SIMD エンジンを活用し、取り込んだ画像を Armadillo-810 自身で画像処理できます。

◆H.264/AVC、AAC、JPEG 対応エンコーダー

H.264/AVC、AAC、JPEG のエンコード・デコードに対応する「AV コーデックミドルウェア」をボード本体に標準でバンドル。Full HD サイズ(1920×1080 ピクセル)での H.264/AVC エンコードなどにも対応可能です[1]。 「AV コーデックミドルウェア」はボード本体とアプリケーションの間を補完する、マルチメディア機能に特化したミドルウェアです。R-Mobile A1 に搭載されたリアルタイム制御用のサブ CPU(SH-4A)やアクセラレータ (VCP1、SPU など)によるハードウェア支援を最大限に活用することで、メイン CPU(ARM Cortex-A9)に大きな負荷をかけずに動画エンコードなどの機能を実現することができ、効率的なシステム設計に役立ちます。

AV コーデックミドルウェアは、「アットマークテクノユーザーズサイト [https://users.atmark-techno.com/]」にて、購入者向けに提供しています。AV コーデックミドルウェアをダウンロードするには、前述のユーザーズサイトでユーザーアカウントの作成および購入製品登録を行う必要があります。

[1]量産時は、使用条件によりライセンス料の請求対象となる場合があります。

図 3.2 Armadillo-810 の特長

3.2. ボード概要

Armadillo-810 の主な仕様は次の通りです。

表 3.1 Armadillo-810 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	メイン: ARM Cortex-A9 - 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 リアルタイム制御用: SH-4A
システムクロック	CPU コアクロック(ARM Cortex-A9): 792MHz CPU コアクロック(SH-4A): 594MHz DDR クロック: 396MHz 源発振クロック: 24MHz
RAM	DDR3 SDRAM: 512MByte バス幅 32bit (DDR3-800) Micron Technology MT41K128M16JT-125 IT:K もしくは同等品
フラッシュメモリ	NOR フラッシュメモリ: 64MByte バス幅 16bit Micron Technology PC28F512P33BFD もしくは同等品
シリアル(UART)	RS232C x 1、3.3V CMOS x 1
拡張インターフェース 1	USB、UART、SPI、I2S、I2C、SD、MMC、PWM、IrDA、JTAG、GPIO
拡張インターフェース 2	Camera、I2C、UART、PWM、GPIO
USB	USB2.0 Device (High Speed 対応)
LED	黄色(面実装) x 4
電源電圧	DC 4.35～5.25V
消費電力	1.3W(Typ.)、待機時 0.9W(Typ.)
使用周囲温度	-20～70℃(ただし結露なきこと)
基板サイズ	50 x 50mm(突起部を除く)

3.3. ブロック図

Armadillo-810 のブロック図は次の通りです。

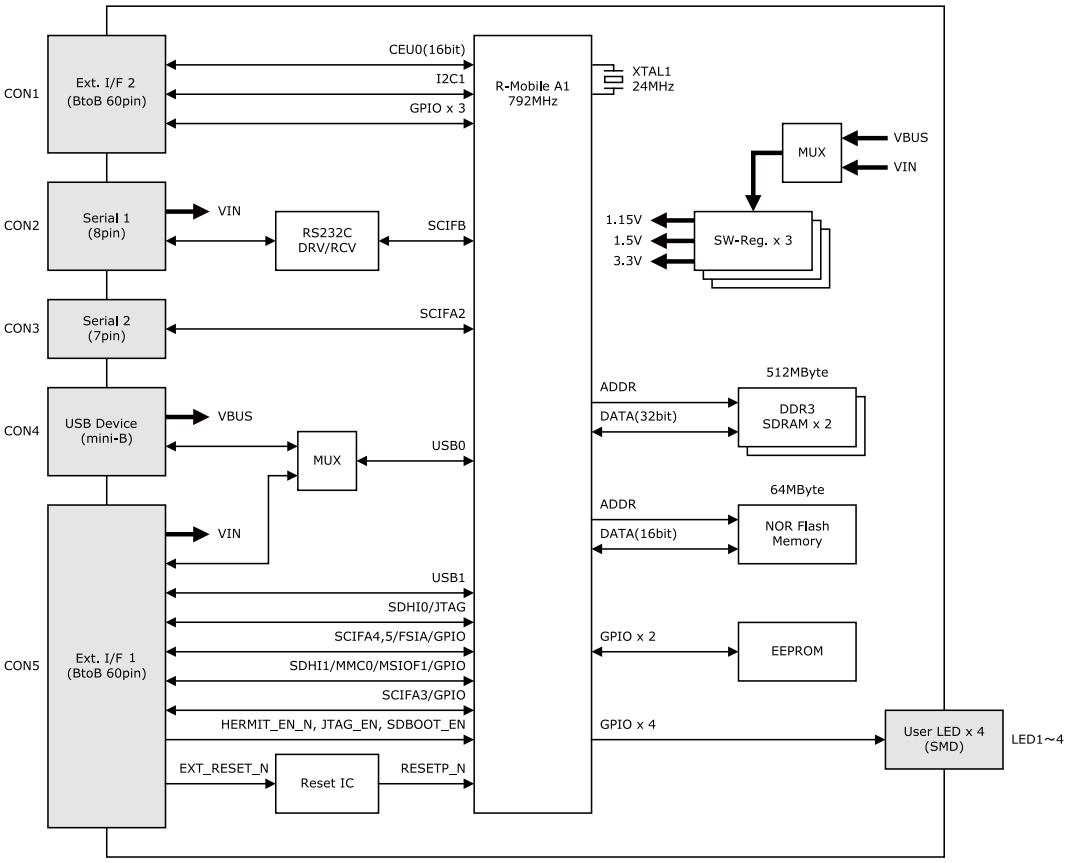


図 3.3 Armadillo-810 のブロック図

3.4. ソフトウェア構成

本章では Armadillo-810 で動作するソフトウェアの構成について説明します。

Armadillo-810 で利用可能なソフトウェアを「表 3.2. Armadillo-810 で利用可能なソフトウェア」に示します。

表 3.2 Armadillo-810 で利用可能なソフトウェア

ソフトウェア	説明
Hermit-At	ブートローダーです。Linux カーネルを起動させる機能の他に、ダウンローダーと協調動作を行いフラッシュメモリを書き替える機能など様々な機能を持っています。工場出荷状態ではブートローダーイメージはフラッシュメモリに配置されていますが、プロセッサ(R-Mobile A1)の機能により SD カードに配置することもできます。
Linux カーネル	バージョン 3.x 系の Linux カーネルです。工場出荷状態では Linux カーネルイメージはフラッシュメモリに配置されていますが、Hermit-At の機能により SD カードに配置することもできます。
Atmark Dist	uClinux-dist をベースにしたアットマークテクノ製品向けの Linux ディストリビューションです。フラッシュメモリ向けのユーザーランドを提供します。工場出荷状態では Atmark Dist ユーザーランドイメージはフラッシュメモリに配置されていますが、SD カードなどのストレージに配置することもできます。
Debian GNU/Linux	Debian Project によって作成された Linux ディストリビューションです。パッケージ管理システムを備えているため、Debian Project が提供する豊富なソフトウェアパッケージを簡単に追加することができます。利用する場合は、SD カードなどのストレージデバイスに構築する必要があります。
AV コーデックミドルウェア	H.264/AVC、AAC デコード及び H.264/AVC、AAC、JPEG エンコードに対応したミドルウェアです。
FSE (Face Sensing Engine)	顔検出や特徴点抽出などの機能を持つ OKI (沖電気工業株式会社)製の顔認識エンジンです。工場出荷状態では使用することができません。アットマークテクノ ユーザーズサイトから評価用デモアプリをダウンロードすることができます。

ソフトウェア	説明
OpenCV (Open Source Computer Vision)	オープンソースのコンピュータービジョン向けライブラリです。工場出荷状態では使用することができません。Armadillo サイトに、OpenCV を用いて画像処理を行う方法について説明した Howto ページが公開されています。

Armadillo-810 のフラッシュメモリのメモリマップを「表 3.3. フラッシュメモリ メモリマップ」に示します。

表 3.3 フラッシュメモリ メモリマップ

物理アドレス	パーティション名	サイズ	工場出荷状態で書き込まれているソフトウェア
0x00000000 | 0x0003FFFF	bootloader	256kByte	Hermit-At ブートローダーイメージ
0x00040000 | 0x0007FFFF	config	256kByte	アプリケーションの設定情報など
0x00080000 | 0x000BFFFF	license	256kByte	AV コーデックミドルウェアライセンス
0x000C0000 | 0x004BFFFF	firmware	4MByte	armhf アーキテクチャ用 OpenGL ES2 ライブラリ AV コーデックミドルウェア
0x004C0000 | 0x008BFFFF	kernel	4MByte	Linux カーネルイメージ
0x008C0000 | 0x03FFFFFF	userland	55.25Mbyte	Atmark Dist ユーザーランドイメージ

4. 作業の前に

4.1. 準備するもの

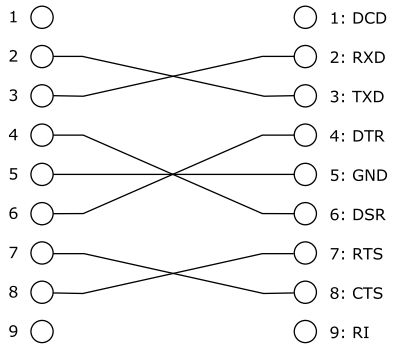
Armadillo を使用する前に、次のものを準備してください。

作業用 PC	Linux または Windows が動作し、2 ポート以上[1]の USB インターフェースを持つ PC です。作業用 PC には、「4.2. 開発/動作確認環境の構築」を参照して開発/動作確認環境を構築してください。
SD カード	SD スロットの動作を確認する場合などに利用します。
USB メモリ	USB ホストの動作を確認する場合などに利用します。
シリアルクロスケーブル	D-Sub9 ピン(メス - メス)のクロス接続用ケーブルです。シリアルポートの動作を確認する場合に利用します。
tar.xz 形式のファイルを展開するソフトウェア	開発/動作確認環境を構築するために利用します。Linux では、tar[2]で展開できます。Windows では、7-Zip や Lhaz などが対応しています。7-Zip は、開発用 DVD に収録されています。

シリアルクロスケーブルの結線

シリアルクロスケーブルには様々な結線のものが存在します。本書ではハードウェアフロー(RTS/CTS)を使用しないため、互いのコネクタの TxD と RxD 同士および GND 同士が結線されているケーブルであれば利用可能です。

ハードウェアフローを使用する場合は、以下のようなインタリンク結線のシリアルケーブルをご利用ください。



[1]USB HUB を利用することもできます。
[2]tar.xz 形式のファイルを展開するには Jxf オプションを指定します。

4.2. 開発/動作確認環境の構築

アットマークテクノ製品のソフトウェア開発や動作確認を簡単に行うために、VMware 仮想マシンのデータイメージを提供しています。この VMware 仮想マシンのデータイメージを ATDE(Atmark Techno Development Environment)と呼びます。ATDE の起動には仮想化ソフトウェアである VMWare を使用します。ATDE のデータは、tar.xz 圧縮されています。環境に合わせたツールで展開してください。

仮想化ソフトウェアとして、VMware の他に Oracle VM VirtualBox が有名です。Oracle VM VirtualBox には以下の特徴があります。

- GPL v2(General Public License version 2)で提供されている[3]
- VMware 形式の仮想ディスク(.vmdk)ファイルに対応している

Oracle VM VirtualBox から ATDE を起動し、ソフトウェア開発環境として使用することができます。ただし、UVC ガジェットの動作確認など、本書に記載されている内容のうち一部について適用できない場合があります。十分ご注意の上で作業してください。

ATDE は、バージョンにより対応するアットマークテクノ製品が異なります。Armadillo-810 に対応している ATDE は、ATDE5 (ATDE バージョン 5)です。

ATDE5 は Debian GNU/Linux 7.0(コードネーム wheezy)をベースに、Armadillo-810 のソフトウェア開発を行うために必要なクロス開発ツールや、Armadillo-810 の動作確認を行うために必要なツールが事前にインストールされています。

4.2.1. ATDE5 セットアップ

4.2.1.1. VMware のインストール

ATDE5 を使用するためには、作業用 PC に VMware がインストールされている必要があります。VMware 社 Web ページ(http://www.vmware.com/)を参照し、利用目的に合う VMware 製品をインストールしてください。また、ATDE5 は tar.xz 圧縮されていますので、環境に合せたツールで展開してください。

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。

VMware や ATDE5 が動作しないことを未然に防ぐため、使用する VMware のドキュメントから以下の項目についてご確認ください。

- ホストシステムのハードウェア要件
- ホストシステムのソフトウェア要件
- ゲスト OS のプロセッサ要件

[3]バージョン 3.x までは PUEL(VirtulBox Personal Use and Evaluation License)が適用されている場合があります。

VMware のドキュメントは、VMware 社 Web ページ(http://www.vmware.com/)から取得することができます。

4.2.1.2. ATDE5 の取得

「表 4.1. ATDE5 の種類」に示す ATDE5 のアーカイブのうちいずれか 1 つを作業用 PC にコピーします。ATDE5 のアーカイブは Armadillo サイト(http://armadillo.atmark-techno.com)または、開発セット付属の DVD から取得可能です。

表 4.1 ATDE5 の種類

ATDE5 アーカイブ	ベースの Debian GNU/Linux
atde5-[version]-amd64.zip	64-bit PC(「amd64」)アーキテクチャ用 Debian GNU/Linux 7.0
atde5-[version]-i386.zip	32-bit PC(「i386」)アーキテクチャ用 Debian GNU/Linux 7.0

作業用 PC の動作環境(ハードウェア、VMware、ATDE5 の種類など)により、ATDE5 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。

4.2.1.3. ATDE5 の起動

ATDE5 のアーカイブを展開したディレクトリに存在する仮想マシン構成(.vmx)ファイルを VMware 上で開くと、ATDE5 を起動することができます。ATDE5 にログイン可能なユーザーを、「表 4.2. ユーザー名とパスワード」に示します[4]。

表 4.2 ユーザー名とパスワード

ユーザー名	パスワード	権限
atmark	atmark	一般ユーザー
root	root	特権ユーザー

ATDE に割り当てるメモリおよびプロセッサ数を増やすことで、ATDE をより快適に使用することができます。仮想マシンのハードウェア設定の変更方法については、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ください。

4.2.2. 取り外し可能デバイスの使用

VMware は、ゲスト OS (ATDE)による取り外し可能デバイス(USB デバイスや DVD など)の使用をサポートしています。デバイスによっては、ホスト OS (VMware を起動している OS)とゲスト OS で同時に使用することができません。そのようなデバイスをゲスト OS で使用するためには、ゲスト OS にデバイスを接続する操作が必要になります。

[4]特権ユーザーで GUI ログインを行うことはできません。

取り外し可能デバイスの使用方法については、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ください。

Armadillo-810 の動作確認をおこなうためには、「表 4.3. 動作確認に使用する取り外し可能デバイス」に示すデバイスをゲスト OS に接続する必要があります。

表 4.3 動作確認に使用する取り外し可能デバイス

デバイス	デバイス名
開発用 USB シリアル変換アダプタ(Armadillo-800 シリーズ対応)	Future Devices FT232R USB UART
Armadillo-810 USB インターフェース(CON4)	g_uvc_acm_ether
作業用 PC の物理シリアルポート	シリアルポート

4.2.3. コマンドライン端末(GNOME 端末)の起動

ATDE5 で、CUI (Character-based User Interface)環境を提供するコマンドライン端末を起動します。ATDE5 で実行する各種コマンドはコマンドライン端末に入力し、実行します。コマンドライン端末にはいくつかの種類がありますが、ここでは GNOME デスクトップ環境に標準インストールされている GNOME 端末を起動します。

GNOME 端末を起動するには、「図 4.1. GNOME 端末の起動」のようにデスクトップ左上のメニューから「端末」を選択してください。

図 4.1 GNOME 端末の起動

「図 4.2. GNOME 端末のウィンドウ」のようにウィンドウが開きます。

図 4.2 GNOME 端末のウィンドウ

4.2.4. シリアル通信ソフトウェア(minicom)の使用

シリアル通信ソフトウェア(minicom)のシリアル通信設定を、「表 4.4. シリアル通信設定」のように設定します。また、minicom を起動する端末の横幅を 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。

表 4.4 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

minicom の設定を開始するには、「図 4.3. minicom 設定方法」のようにしてください。設定完了後、デフォルト設定(dfl)に保存して終了します。

```
[ATDE ~]$ LANG=C minicom --setup
```

図 4.3 minicom 設定方法

minicom を起動させるには、「図 4.4. minicom 起動方法」のようにしてください。

```
[ATDE ~]$ LANG=C minicom --noinit --wrap --device /dev/ttyUSB0
```

図 4.4 minicom 起動方法

minicom を終了させるには、まず Ctrl+a に続いて q キーを入力します。その後、以下のように表示されたら「Yes」にカーソルを合わせて Enter キーを入力すると minicom が終了します。

```
+----------------------+
| Leave without reset? |
|     Yes       No     |
+----------------------+
```

図 4.5 minicom 終了確認

Ctrl+a に続いて z キーを入力すると、minicom のコマンドヘルプが表示されます。

4.3. インターフェースレイアウト

Armadillo-810 及びカメラモデル開発セットに含まれる各基板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。

4.3.1. Armadillo-810 インターフェースレイアウト

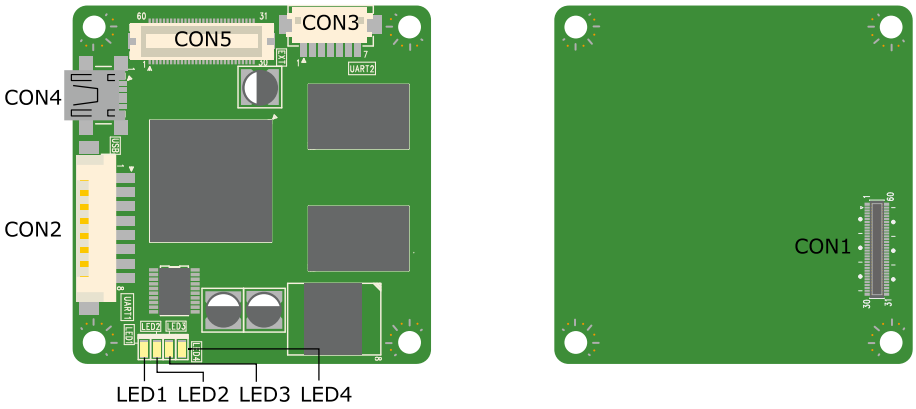


図 4.6 Armadillo-810 のインターフェースレイアウト図

表 4.5 Armadillo-810 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	拡張インターフェース 2 (B コネクタ)	BtoB コネクタ 60P(0.4mm ピッチ) DF40C-60DP-0.4V(51)/HIROSE ELECTRIC	対向コネクタ例: DF40HC(4.0)-60DS-0.4V(51)/HIROSE ELECTRIC 挿抜寿命:30 回
CON2	シリアルインターフェース 1	ピンヘッダ 8P(2mm ピッチ) DF3DZ-8P-2H(51)/HIROSE ELECTRIC	信号レベル: RS232C 対向コネクタ例: DF3-8S-2C/HIROSE ELECTRIC 挿抜寿命:50 回
CON3	シリアルインターフェース 2	ピンヘッダ 7P(1.25mm ピッチ) DF13A-7P-1.25H(51)/HIROSE ELECTRIC	信号レベル: 3.3V CMOS 対向コネクタ例: DF13-7S-1.25C/HIROSE ELECTRIC 挿抜寿命:50 回
CON4	USB インターフェース	USB mini B コネクタ 54819-0572/Molex[a]	USB2.0 Device(High Speed 対応)

部品番号	インターフェース	形状	備考
CON5	拡張インターフェース 1 (A コネクタ)	BtoB コネクタ 60P(0.5mm ピッチ) DF17(4.0)-60DS-0.5V(57)/HIROSE ELECTRIC	対向コネクタ例: DF17(4.0)-60DP-0.5V(57)/HIROSE ELECTRIC 挿抜寿命:50 回
LED1～LED4	ユーザー LED	LED(黄色、面実装)	

[a]製品リビジョン Rev.3.2 以前は、UB-M5BR-G14-4S(LF)(SN)/J.S.T. Mfg.が搭載されています。

4.3.2. Armadillo-810 拡張ボード 01 (A コネクタ用) インターフェースレイアウト

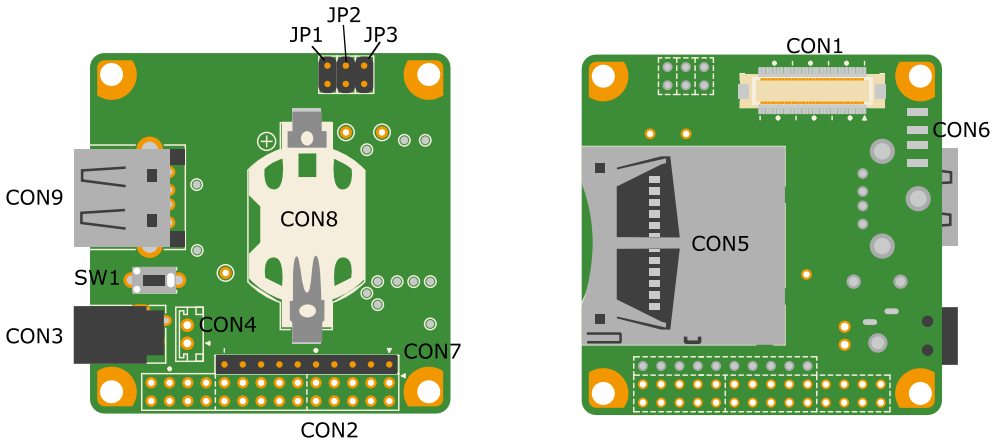


図 4.7 Armadillo-810 拡張ボード 01 (A コネクタ用)のインターフェースレイアウト図

表 4.6 Armadillo-810 拡張ボード 01 (A コネクタ用)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	Armadillo-810 接続インターフェース	BtoB コネクタ 60P(0.5mm ピッチ) DF17(4.0)-60DP-0.5V(57)/HIROSE ELECTRIC	挿抜寿命:50 回 Armadillo-810 の拡張インターフェース 1 (A コネクタ)と接続
CON2	拡張インターフェース	ピンヘッダ 28P(2.54mm ピッチ)	コネクタ非搭載(搭載コネクタ例：A1-28PA-2.54DSA(71)/HIROSE ELECTRIC)
CON3	電源入力 1	DC ジャック HEC3600-016110/HOSIDEN	対応プラグ： EIAJ#2
CON4	電源入力 2	ピンヘッダ 2P(2.5mm ピッチ)	コネクタ非搭載(搭載コネクタ例：B2B-EH/J.S.T. Mfg.)
CON5	SD インターフェース	SD スロット SCDA9A0400/ALPS ELECTRIC	信号線は CON7 と共通
CON6	Reserved	Pad	このインターフェースを使用する場合の動作は保証されていません
CON7	JTAG インターフェース	ピンヘッダ 10P(2.54mm ピッチ) A2-10PA-2.54DSA(71)/HIROSE ELECTRIC	信号線は CON5 と共通
CON8	RTC 外部バックアップインターフェース	電池ボックス SMTU2032-LF.TR/Renata SA	対応電池: CR2032
CON9	USB インターフェース	USB Type A コネクタ UBA-4R-D14T-4D(LF)(SN)/J.S.T. Mfg.	USB2.0 Host(High Speed 対応)

部品番号	インターフェース	形状	備考
JP1	起動モード設定ジャンパ	ピンヘッダ 6P(2.54mm ピッチ) A1-6PA-2.54DSA(71)/HIROSE ELECTRIC	オープン: OS 自動起動モード ショート: 保守モード
JP2	SD/JTAG 設定ジャンパ		オープン: SD(CON5)有効/JTAG(CON7)無効 ショート: SD(CON5)無効/JTAG(CON7)有効
JP3	起動デバイス設定ジャンパ	オープン: オンボードフラッシュメモリブート ショート: SD(CON5)ブート	
SW1	リセットスイッチ	タクトスイッチ SKHLACA010/ALPS ELECTRIC	

4.3.3. Armadillo-810 カメラモジュール 01 (B コネクタ用) インターフェースレイアウト

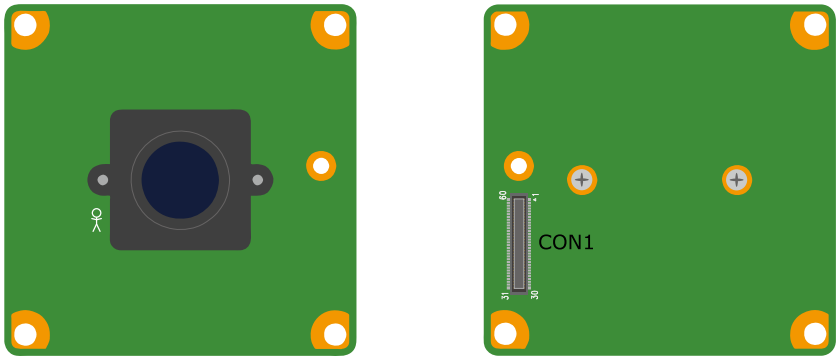


図 4.8 Armadillo-810 カメラモジュール 01 (B コネクタ用)のインターフェースレイアウト図

表 4.7 Armadillo-810 カメラモジュール 01 (B コネクタ用)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	Armadillo-810 接続インターフェース	ピンソケット 60P(0.4mm ピッチ) DF40HC(4.0)-60DS-0.4V(51)/HIROSE ELECTRIC	対向コネクタ例: DF40C-60DP-0.4V(51)/HIROSE ELECTRIC 挿抜寿命:30 回 Armadillo-810 の拡張インターフェース 2 (B コネクタ)と接続

4.4. 組み立て

4.4.1. Armadillo-810 カメラモデルの組み立て

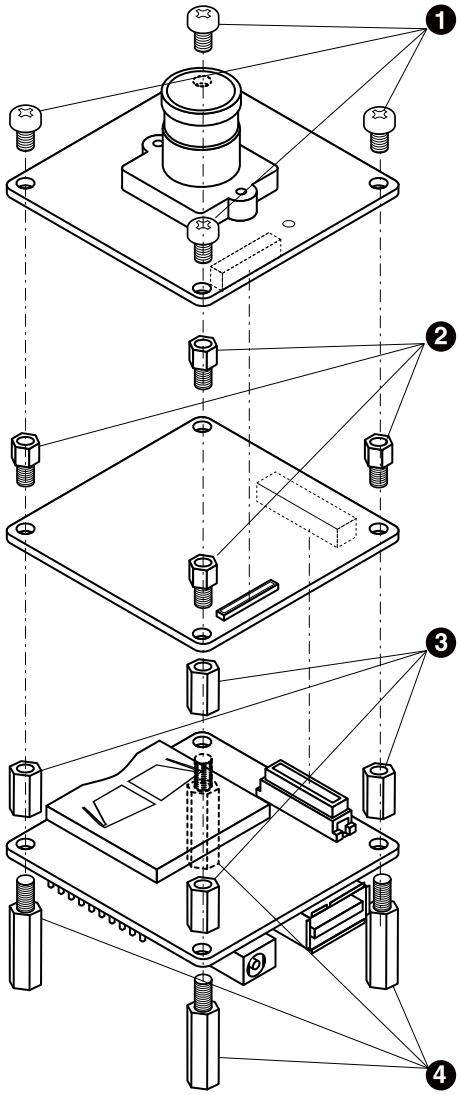


図 4.9 Armadillo-810 カメラモデルの組み立て

❶ なべ小ねじ(M3、L=4mm)

❷ 金属スペーサ(M3、L=8mm)

❸ 金属スペーサ(M3、L=8mm)

❹ 金属スペーサ(M3、L=15mm)

コネクタ嵌合時の取扱い上の注意

嵌合する際は、コネクタの中心をきっちり合わせてください。

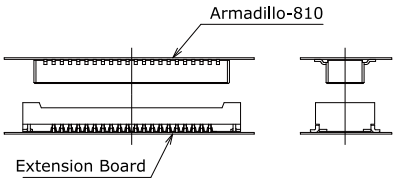


図 4.10 コネクタ嵌合時の取扱い上の注意 1

位置合わせをする際は、無理な力を加えることなく誘い込み口を探してください。無理な力を加えると、モールドの破損、削れが発生し、接触抵抗の不具合等に繋る場合があります。

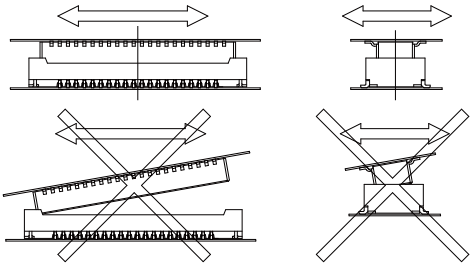


図 4.11 コネクタ嵌合時の取扱い上の注意 2

コネクタが誘い込まれると、コネクタ間の距離が近くなり、平行になって前後左右に動かなくなります。この状態からまっすぐに嵌合してください。

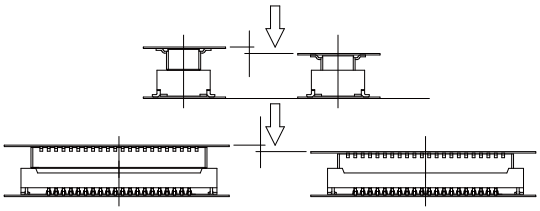


図 4.12 コネクタ嵌合時の取扱い上の注意 3

コネクタ抜去時の取扱い上の注意

コネクタは平行に抜去してください。

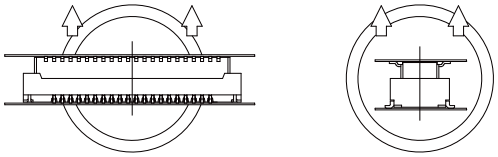


図 4.13 コネクタ抜去時の取扱い上の注意 1

平行に抜去することが困難な場合、コネクタ幅の狭い方向から斜めに抜去してください。

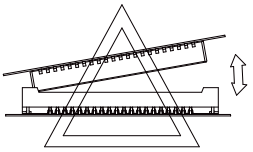


図 4.14 コネクタ抜去時の取扱い上の注意 2

コネクタが損傷する可能性が高いため、コネクタのコーナー方向や幅の広い方向から斜めに抜去しないでください。

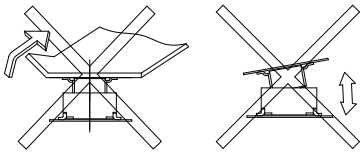


図 4.15 コネクタ抜去時の取扱い上の注意 3

ねじ締め時の注意事項

ねじは図のようにコネクタから遠い箇所から順に締めてください。先にコネクタに近い箇所のねじを締めると、コネクタに無理な力が加わり、コネクタ破損の原因となることがあります。

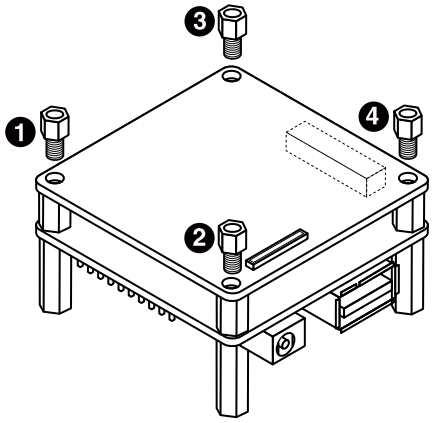


図 4.16 ねじ締め時の注意事項 1

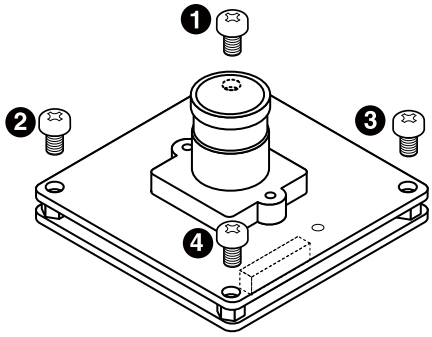


図 4.17 ねじ締め時の注意事項 2

4.4.2. レンズの交換方法

Armadillo-810 カメラモジュール 01 (B コネクタ用)には出荷時にカメラモジュール用レンズ(水平画角 79°)が取り付けられています。付属のカメラモジュール用レンズ(水平画角 120°)をご使用になりたい場合は、カメラレンズを交換して使用する事が可能です。

交換時には、カメラモジュールを下向きにし、レンズを矢印の方向に回してホルダから外し、交換したいレンズを装着して下さい。

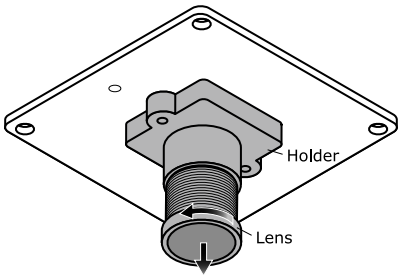


図 4.18 カメラレンズの交換

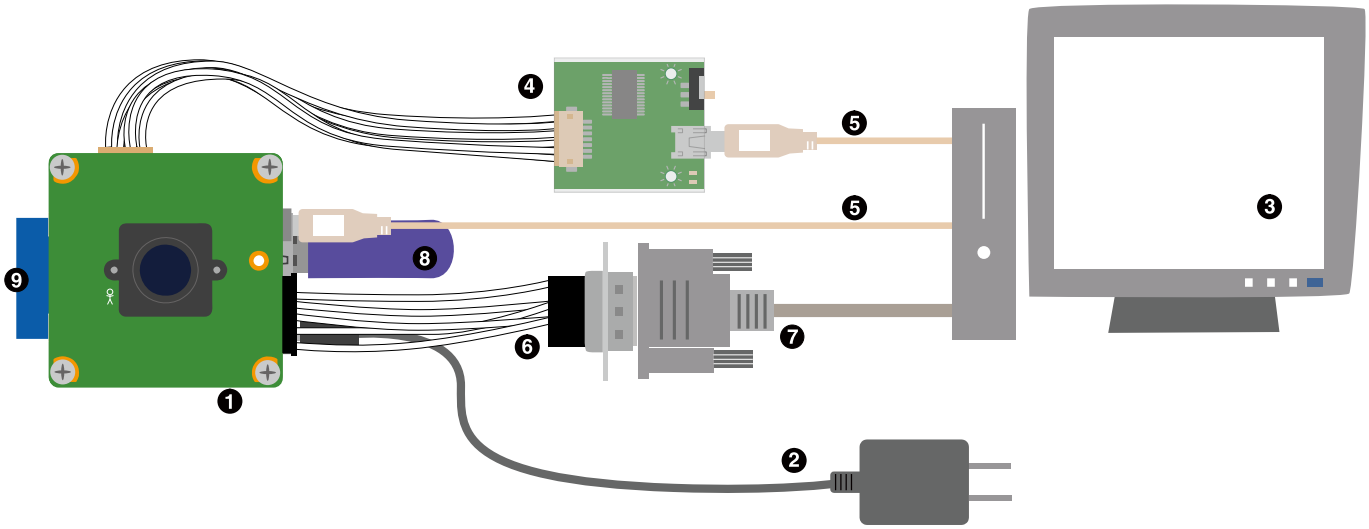
カメラレンズ交換時の注意事項

下記の項目に十分留意して作業を行ってください。下記の項目を守らず作業を行った場合、撮像素子やレンズが破損、またはゴミ等が付着し、取得画像にノイズとなって残る可能性があります。

- レンズに直接触らないで下さい。
- レンズホルダはカメラモジュールから外さないで下さい。
- ピント調整する際はカメラの映像を確認しながら行ってください。
- レンズ交換の際はゴミや埃が撮像素子に付着しないよう注意してください。
- 埃が立たない場所で作業を行って下さい。
- レンズ交換は埃やゴミが侵入しないよう短時間で作業を完了させて下さい。
- 基板からのレンズ先端の高さは 18mm 以下にしないでください。18mm 以下までレンズを入れた場合、レンズがイメージセンサーに接触し、レンズおよびイメージセンサーが故障する恐れがあります。
- レンズ交換作業による破損または故障につきましては、保証対象外となります。

4.5. 接続方法

Armadillo-810 カメラモデルと作業用 PC との接続例を「図 4.19. 接続図」に示します。



1. Armadillo-810 カメラモデル
2. AC アダプタ(5V/2.0A EIAJ#2)[5]
3. 作業用 PC
4. 開発用 USB シリアル変換アダプタ(Armadillo-800 シリーズ対応)[5]
5. USB2.0 ケーブル(A-miniB タイプ, 1.8m)[5]
6. 7 ピン シリアルケーブル(Armadillo-800 シリーズ対応)[5]
7. シリアルクロスケーブル
8. USB メモリ
9. SD カード

図 4.19 接続図

開発用 USB シリアル変換アダプタ(Armadillo-800 シリーズ対応)の取扱い上の注意

USB シリアル変換アダプタには電源投入順序があります。Armadillo-810 に接続する際は、以下の手順に従ってご使用ください。接続手順に従わない場合は、USB シリアル変換アダプタが故障する可能性がありますのでご注意ください。

1. 起動中の作業用 PC と USB シリアル変換アダプタを USB2.0 ケーブルで接続します。
2. Armadillo-810 のシリアルインターフェース 2 (CON3)に USB シリアル変換アダプタを接続します。
3. 上記接続を確認後、Armadillo-810 に電源を投入します。

[5]Armadillo-810 カメラモデル開発セット付属品

また、Armadillo-810 に USB シリアル変換アダプタを接続した状態のまま、作業用 PC または USB シリアル変換アダプタから USB2.0 ケーブルを抜く場合や作業用 PC をシャットダウンする場合は、Armadillo-810 の電源が切断されていることを確認してから行ってください。

4.6. ジャンパピンの設定について

ジャンパの設定を変更することで、Armadillo-810 の動作を変更することができます。ジャンパの機能を「表 4.8. ジャンパの機能」に示します。

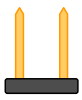
表 4.8 ジャンパの機能

ジャンパ	機能	動作
JP1	起動モード設定	オープン: OS を自動起動します。 ショート: ブートローダーを保守モードにします。
JP2	SD/JTAG 設定	オープン: SD インターフェースを有効化します。JTAG インターフェースは無効化されます。 ショート: JTAG インターフェースを有効化します。SD インターフェースは無効化されます。
JP3	起動デバイス設定	オープン: オンボードフラッシュメモリのブートローダーを起動します。 ショート: SD カードのブートローダーを起動します。

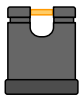
各ジャンパは必要に応じて切り替えの指示があります。ここでは、全てのジャンパをオープンに設定しておきます。

ジャンパピンの位置は「図 4.7. Armadillo-810 拡張ボード 01 (A コネクタ用)のインターフェースレイアウト図」で確認することができます。

ジャンパのオープン、ショートとは



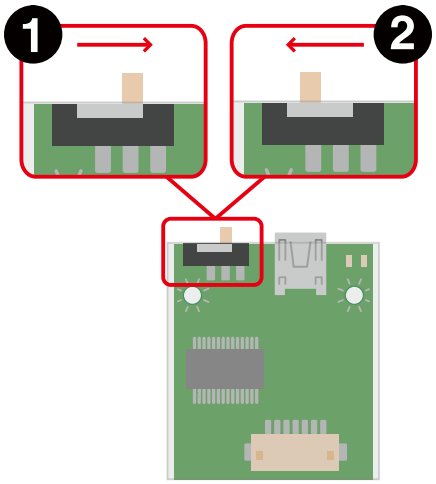
「オープン」とはジャンパピンにジャンパソケットを接続していない状態です。



「ショート」とはジャンパピンにジャンパソケットを接続している状態です。

4.7. スライドスイッチの設定について

開発用 USB シリアル変換アダプタ (Armadillo-800 シリーズ対応) のスライドスイッチには、Armadillo-810 拡張ボード 01 (A コネクタ用)の JP1 と同じ機能が割り当てられています。



❶ OS 自動起動モード

❷ 保守モード

図 4.20 スライドスイッチの設定

Armadillo-810 に Armadillo-810 拡張ボード 01 (A コネクタ用)を接続している場合は、常に「OS 自動起動モード」に設定してください。

4.8. vi エディタの使用方法

vi エディタは、Armadillo に標準でインストールされているテキストエディタです。本書では、Armadillo の設定ファイルの編集などに vi エディタを使用します。

vi エディタは、ATDE にインストールされてる gedit や emacs などのテキストエディタとは異なり、モードを持っていることが大きな特徴です。vi のモードには、コマンドモードと入力モードがあります。コマンドモードの時に入力した文字はすべてコマンドとして扱われます。入力モードでは文字の入力ができます。

本章で示すコマンド例は ATDE で実行するよう記載していますが、Armadillo でも同じように実行することができます。

4.8.1. vi の起動

vi を起動するには、以下のコマンドを入力します。

```
[ATDE ~]# vi [file]
```

図 4.21 vi の起動

file にファイル名のパスを指定すると、ファイルの編集(*file* が存在しない場合は新規作成)を行ないます。vi はコマンドモードの状態で起動します。

4.8.2. 文字の入力

文字を入力するにはコマンドモードから入力モードへ移行する必要があります。コマンドモードから入力モードに移行するには、「表 4.9. 入力モードに移行するコマンド」に示すコマンドを入力します。入力モードへ移行後は、キーを入力すればそのまま文字が入力されます。

表 4.9 入力モードに移行するコマンド

コマンド	動作
i	カーソルのある場所から文字入力を開始
a	カーソルの後ろから文字入力を開始

入力モードからコマンドモードに戻りたい場合は、ESC キーを入力することで戻ることができます。現在のモードが分からなくなった場合は、ESC キーを入力し、一旦コマンドモードへ戻ることにより混乱を防げます。

日本語変換機能を OFF に

vi のコマンドを入力する時は ATDE の日本語入力システム(Mozc)を OFF にしてください。日本語入力システムの ON/OFF は、半角/全角キーまたは、Shift+Space キーで行うことができます。

「i」、「a」それぞれのコマンドを入力した場合の文字入力の開始位置を「図 4.22. 入力モードに移行するコマンドの説明」に示します。

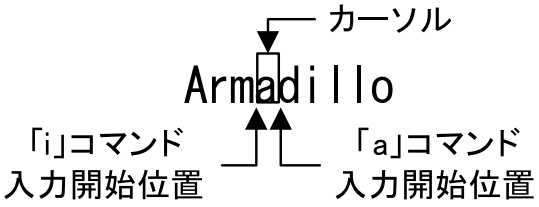


図 4.22 入力モードに移行するコマンドの説明

vi での文字削除

コンソールの環境によっては BS(Backspace)キーで文字が削除できず、「^H」文字が入力される場合があります。その場合は、「4.8.4. 文字の削除」で説明するコマンドを使用し、文字を削除してください。

4.8.3. カーソルの移動

方向キーでカーソルの移動ができますが、コマンドモードで「表 4.10. カーソルの移動コマンド」に示すコマンドを入力することでもカーソルを移動することができます。

表 4.10 カーソルの移動コマンド

コマンド	動作
h	左に 1 文字移動

コマンド	動作
j	下に 1 文字移動
k	上に 1 文字移動
l	右に 1 文字移動

4.8.4. 文字の削除

文字を削除する場合は、コマンドモードで「表 4.11. 文字の削除コマンド」に示すコマンドを入力します。

表 4.11 文字の削除コマンド

コマンド	動作
x	カーソル上の文字を削除
dd	現在行を削除

「x」コマンド、「dd」コマンドを入力した場合に削除される文字を「図 4.23. 文字を削除するコマンドの説明」に示します。

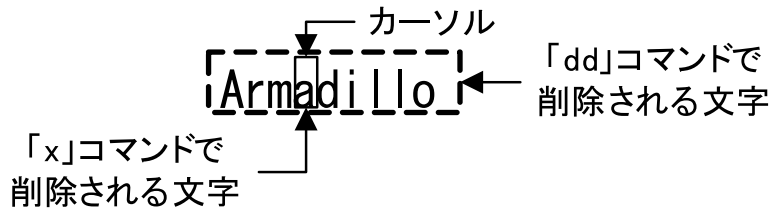


図 4.23 文字を削除するコマンドの説明

4.8.5. 保存と終了

ファイルの保存、終了を行うコマンドを「表 4.12. 保存・終了コマンド」に示します。

表 4.12 保存・終了コマンド

コマンド	動作
:q!	変更を保存せずに終了
:w *[file]*	ファイル名を *file* に指定して保存
:wq	ファイルを上書き保存して終了

保存と終了を行うコマンドは「:」(コロン)からはじまるコマンドを使用します。":"キーを入力すると画面下部にカーソルが移り入力したコマンドが表示されます。コマンドを入力した後 Enter キーを押すことで、コマンドが実行されます。

5. 起動と終了

5.1. 起動

Armadillo の電源を投入してください。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。

```
Hermit-At v3.2.4 (Armadillo-810/nor) compiled at 21:52:10, Jan 27 2014
Uncompressing  kernel.....................................................
.....................................................................
.....................done.
Uncompressing ramdisk.....................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................done.
Booting Linux on physical CPU 0
Initializing cgroup subsys cpuset
Initializing cgroup subsys cpu
Linux version 3.4-at6 (atmark@atde5) (gcc version 4.6.3 (Debian 4.6.3-14atmark1)
 ) #2 PREEMPT Mon Jan 27 23:24:22 JST 2014
CPU: ARMv7 Processor [412fc093] revision 3 (ARMv7), cr=10c53c7d
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine: armadillo810
cma: CMA: reserved 128 MiB at 50000000
Memory policy: ECC disabled, Data cache writeback
bootconsole [early_ttySC2] enabled
Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 97536
Kernel command line: console=ttySC2,115200 earlyprintk=sh-sci.2,115200 mem=384M
PID hash table entries: 2048 (order: 1, 8192 bytes)
Dentry cache hash table entries: 65536 (order: 6, 262144 bytes)
Inode-cache hash table entries: 32768 (order: 5, 131072 bytes)
allocated 786432 bytes of page_cgroup
please try 'cgroup_disable=memory' option if you don't want memory cgroups
Memory: 384MB = 384MB total
Memory: 212976k/212976k available, 180240k reserved, 0K highmem
Virtual kernel memory layout:
    vector  : 0xffff0000 - 0xffff1000   (   4 kB)
    fixmap  : 0xfff00000 - 0xfffe0000   ( 896 kB)
    vmalloc : 0xd8800000 - 0xff000000   ( 616 MB)
    lowmem  : 0xc0000000 - 0xd8000000   ( 384 MB)
    pkmap   : 0xbfe00000 - 0xc0000000   (   2 MB)
```

```
    modules : 0xbf000000 - 0xbfe00000   (  14 MB)
      .text : 0xc0008000 - 0xc04d2000   (4904 kB)
      .init : 0xc04d2000 - 0xc04f5000   ( 140 kB)
      .data : 0xc04f6000 - 0xc052d620   ( 222 kB)
       .bss : 0xc052d644 - 0xc0556a74   ( 166 kB)
NR_IRQS:16 nr_irqs:16 16
sched_clock: 32 bits at 128 Hz, resolution 7812500ns, wraps every 3489660920ms
Console: colour dummy device 80x30
 sh_cmt_simple.10: used as clock source
 sh_cmt_simple.14: used for clock events
 sh_cmt_simple.14: used for periodic clock events
Calibrating delay loop... 1576.53 BogoMIPS (lpj=6156288)
pid_max: default: 32768 minimum: 301
Mount-cache hash table entries: 512
Initializing cgroup subsys cpuacct
Initializing cgroup subsys memory
Initializing cgroup subsys devices
Initializing cgroup subsys freezer
Initializing cgroup subsys blkio
CPU: Testing write buffer coherency: ok
hw perfevents: enabled with ARMv7 Cortex-A9 PMU driver, 7 counters available
Setting up static identity map for 0x403b32b8 - 0x403b32ec
dummy:
NET: Registered protocol family 16
DMA: preallocated 256 KiB pool for atomic coherent allocations
pfc: r8a7740_pfc handling gpio 0 -> 858
gpiochip_add: registered GPIOs 0 to 858 on device: r8a7740_pfc
CON5: STANDARD extension board found.
L310 cache controller enabled
l2x0: 8 ways, CACHE_ID 0x410000c7, AUX_CTRL 0x42440000, Cache size: 262144 B
hw-breakpoint: found 5 (+1 reserved) breakpoint and 1 watchpoint registers.
hw-breakpoint: maximum watchpoint size is 4 bytes.
bio: create slab <bio-0> at 0
sdhi0: 3300 mV
SCSI subsystem initialized
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
i2c-gpio i2c-gpio.2: using pins 99 (SDA) and 98 (SCL)
i2c-sh_mobile i2c-sh_mobile.0: Runtime PM disabled, clock forced on.
i2c-sh_mobile i2c-sh_mobile.0: I2C adapter 0 with bus speed 100000 Hz
i2c-sh_mobile i2c-sh_mobile.1: Runtime PM disabled, clock forced on.
i2c-sh_mobile i2c-sh_mobile.1: I2C adapter 1 with bus speed 100000 Hz
Linux video capture interface: v2.00
Advanced Linux Sound Architecture Driver Version 1.0.25.
Switching to clocksource sh_cmt_simple.10
 sh_cmt_simple.14: used for oneshot clock events
NET: Registered protocol family 2
IP route cache hash table entries: 4096 (order: 2, 16384 bytes)
TCP established hash table entries: 16384 (order: 5, 131072 bytes)
TCP bind hash table entries: 16384 (order: 4, 65536 bytes)
TCP: Hash tables configured (established 16384 bind 16384)
TCP: reno registered
UDP hash table entries: 256 (order: 0, 4096 bytes)
UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
NET: Registered protocol family 1
RPC: Registered named UNIX socket transport module.
RPC: Registered udp transport module.
```

```
RPC: Registered tcp transport module.
RPC: Registered tcp NFSv4.1 backchannel transport module.
Trying to unpack rootfs image as initramfs...
rootfs image is not initramfs (junk in compressed archive); looks like an initrd
Freeing initrd memory: 39152K
audit: initializing netlink socket (disabled)
type=2000 audit(0.625:1): initialized
VFS: Disk quotas dquot_6.5.2
Dquot-cache hash table entries: 1024 (order 0, 4096 bytes)
squashfs: version 4.0 (2009/01/31) Phillip Lougher
NFS: Registering the id_resolver key type
nfs4filelayout_init: NFSv4 File Layout Driver Registering...
msgmni has been set to 748
Block layer SCSI generic (bsg) driver version 0.4 loaded (major 253)
io scheduler noop registered
io scheduler deadline registered
io scheduler cfq registered (default)
sh-dma-engine sh-dma-engine.0: Runtime PM disabled, clock forced on.
sh-dma-engine sh-dma-engine.1: Runtime PM disabled, clock forced on.
sh-dma-engine sh-dma-engine.2: Runtime PM disabled, clock forced on.
sh-dma-engine sh-dma-engine.3: Runtime PM disabled, clock forced on.
SuperH SCI(F) driver initialized
sh-sci sh-sci.0: Runtime PM disabled, clock forced on.
sh-sci.0: ttySC0 at MMIO 0xe6c40000 (irq = 132) is a scifa
console [ttySC2] enabled, bootconsole disabled
console [ttySC2] enabled, bootconsole disabled
sh-sci sh-sci.1: Runtime PM disabled, clock forced on.
sh-sci.1: ttySC1 at MMIO 0xe6c50000 (irq = 133) is a scifa
sh-sci sh-sci.2: Runtime PM disabled, clock forced on.
sh-sci.2: ttySC2 at MMIO 0xe6c60000 (irq = 134) is a scifa
sh-sci sh-sci.3: Runtime PM disabled, clock forced on.
sh-sci.3: ttySC3 at MMIO 0xe6c70000 (irq = 135) is a scifa
sh-sci sh-sci.4: Runtime PM disabled, clock forced on.
sh-sci.4: ttySC4 at MMIO 0xe6c80000 (irq = 136) is a scifa
sh-sci sh-sci.5: Runtime PM disabled, clock forced on.
sh-sci.5: ttySC5 at MMIO 0xe6cb0000 (irq = 137) is a scifa
sh-sci sh-sci.6: Runtime PM disabled, clock forced on.
sh-sci.6: ttySC6 at MMIO 0xe6cc0000 (irq = 138) is a scifa
sh-sci sh-sci.7: Runtime PM disabled, clock forced on.
sh-sci.7: ttySC7 at MMIO 0xe6cd0000 (irq = 139) is a scifa
sh-sci sh-sci.8: Runtime PM disabled, clock forced on.
sh-sci.8: ttySC8 at MMIO 0xe6c30000 (irq = 140) is a scifb
brd: module loaded
loop: module loaded
physmap platform flash device: 08000000 at 00000000
physmap-flash.0: Found 1 x16 devices at 0x0 in 16-bit bank. Manufacturer ID 0x00
0089 Chip ID 0x008965
Intel/Sharp Extended Query Table at 0x010A
Intel/Sharp Extended Query Table at 0x010A
Intel/Sharp Extended Query Table at 0x010A
Intel/Sharp Extended Query Table at 0x010A
Intel/Sharp Extended Query Table at 0x010A
Using buffer write method
Using auto-unlock on power-up/resume
cfi_cmdset_0001: Erase suspend on write enabled
Creating 6 MTD partitions on "physmap-flash.0":
0x000000000000-0x000000040000 : "bootloader"
0x000000040000-0x000000080000 : "config"
```

```
0x000000080000-0x0000000c0000 : "license"
0x0000000c0000-0x0000004c0000 : "firmware"
0x0000004c0000-0x0000008c0000 : "kernel"
0x0000008c0000-0x000004000000 : "userland"
pegasus: v0.6.14 (2006/09/27), Pegasus/Pegasus II USB Ethernet driver
usbcore: registered new interface driver pegasus
usbcore: registered new interface driver asix
usbcore: registered new interface driver smsc95xx
ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
rmobile-ehci-driver rmobile-ehci-driver: R-Mobile EHCI
rmobile-ehci-driver rmobile-ehci-driver: new USB bus registered, assigned bus nu
mber 1
rmobile-ehci-driver rmobile-ehci-driver: irq 266, io mem 0xc6701000
rmobile-ehci-driver rmobile-ehci-driver: USB 2.0 started, EHCI 1.00
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 1 port detected
ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
rmobile-ohci-driver rmobile-ohci-driver: R-Mobile OHCI
rmobile-ohci-driver rmobile-ohci-driver: new USB bus registered, assigned bus nu
mber 2
rmobile-ohci-driver rmobile-ohci-driver: irq 266, io mem 0xc6700000
hub 2-0:1.0: USB hub found
hub 2-0:1.0: 1 port detected
Initializing USB Mass Storage driver...
usbcore: registered new interface driver usb-storage
USB Mass Storage support registered.
renesas_usbhs renesas_usbhs: Runtime PM disabled, clock forced on.
renesas_usbhs renesas_usbhs: gadget probed
renesas_usbhs renesas_usbhs: probed
 gadget: using random self ethernet address
 gadget: using random host ethernet address
usb0: MAC ca:cc:09:2a:c4:00
usb0: HOST MAC 1a:e7:10:54:42:91
 gadget: UVC Composite Gadget, version: 0.9.0
 gadget: userspace failed to provide iSerialNumber
 gadget: g_uvc_acm_ether ready
mousedev: PS/2 mouse device common for all mice
hub 1-0:1.0: over-current condition on port 1
rtc-s35390a 2-0030: rtc core: registered rtc-s35390a as rtc0
i2c /dev entries driver
sh_mobile_ceu sh_mobile_ceu.0: Runtime PM disabled, clock forced on.
soc-camera-pdrv soc-camera-pdrv.0: Probing soc-camera-pdrv.0
sh_mobile_ceu sh_mobile_ceu.0: SuperH Mobile CEU driver attached to camera 0
ov772x 1-0021: ov7725 Product ID 77:21 Manufacturer ID 7f:a2
sh_mobile_ceu sh_mobile_ceu.0: SuperH Mobile CEU driver detached from camera 0
uvcvideo: Unable to create debugfs directory
usbcore: registered new interface driver uvcvideo
USB Video Class driver (1.1.1)
sh_mobile_wdt sh_mobile_wdt.0: Runtime PM disabled, clock forced on.
device-mapper: ioctl: 4.22.0-ioctl (2011-10-19) initialised: dm-devel@redhat.com
sh_mobile_sdhi sh_mobile_sdhi.0: Runtime PM disabled, clock forced on.
sh_mobile_sdhi sh_mobile_sdhi.0: Platform OCR mask is ignored
sh_mobile_sdhi sh_mobile_sdhi.0: mmc0 base at 0xe6850000 clock rate 99 MHz
usbcore: registered new interface driver usbhid
usbhid: USB HID core driver
usbcore: registered new interface driver snd-usb-audio
ip_tables: (C) 2000-2006 Netfilter Core Team
TCP: cubic registered
```

```
NET: Registered protocol family 17
VFP support v0.3: implementor 41 architecture 3 part 30 variant 9 rev 3
registered taskstats version 1
rtc-s35390a 2-0030: setting system clock to 2000-01-01 00:00:00 UTC (946684800)
ALSA device list:
  No soundcards found.
RAMDISK: ext2 filesystem found at block 0
RAMDISK: Loading 39152KiB [1 disk] into ram disk... done.
VFS: Mounted root (ext2 filesystem) on device 1:0.
Freeing init memory: 140K
Mounting proc: done
Starting fsck for root filesystem.
fsck 1.25 (20-Sep-2001)
/dev/ram0: clean, 1296/1600 files, 35212/39152 blocks
Checking root filesystem: done
Remounting root rw: done
Mounting usbfs: done
Mounting sysfs: done
Mounting tmpfs on /dev: done
Cleaning up system: done
Running local start scripts.
Creating mtd devnode: done
Loading /etc/config: done
Starting udevd: done
Mounting devpts: done
Changing file permissions: done
Configure /home/ftp: done
Starting syslogd: done
Starting klogd: done
Mounting firmware on /opt/firmware: done
Mounting license on /opt/license: done
Mounting tmpfs on /tmp, /var/tmp: done
Mounting ramfs on /home/ftp/pub: done
Creating decoder firmware symlink: done
Creating encoder firmware symlink: done
Setting hostname: done
Starting basic firewall: done
Configuring network interfaces: done
Starting inetd: done
Creating avahi.services: done
Starting avahi.daemon: done
Starting lighttpd: done
Starting sshd: failed
  (sshd: you will be available to use after run '/etc/init.d/sshd keygen')
Running local start script (/etc/config/rc.local).
Starting uvc-gadget: done
load encoder firmware: done
acm_h264enc: H.264 Encoder of AV Codec Middleware
acm_aacenc: AAC Encoder of AV Coenc Middleware
acm_jpegenc: JPEG Encoder of AV Codec Middleware

atmark-dist v1.33.0 (AtmarkTechno/Armadillo-810)
Linux 3.4-at6 [armv7l arch]

armadillo810-0 login:
```

図 5.1 起動ログ

5.2. 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 5.1. シリアルコンソールログイン時のユーザ名とパスワード」に示すユーザでログインすることができます。

表 5.1 シリアルコンソールログイン時のユーザ名とパスワード

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

5.3. 終了方法

安全に終了させる場合は、次のようにコマンドを実行し、「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
[armadillo ~]#
System is going down for system reboot now.

Starting local stop scripts.
Syncing all filesystems: done
Unmounting all filesystems: done
The system is going down NOW!
Sent SIGTERM to all processes
Sent SIGKILL to all processes
Requesting system halt
System halted.
```

図 5.2 終了方法

SD カードなどのストレージをマウントしていない場合は、電源を切断し終了させることもできます。

ストレージにデータを書き込んでいる途中に電源を切断した場合、ファイルシステム、及び、データが破損する恐れがあります。ストレージをアンマウントしてから電源を切断するようにご注意ください。

6. 動作確認方法

6.1. USB ガジェット

Armadillo-810 を USB デバイスとして使用することができます。ここでは動作確認を ATDE5 で行うため、「4.2.2. 取り外し可能デバイスの使用」を参照して ATDE5 と Armadillo-810 の CON4 を USB2.0 ケーブルで接続する必要があります。

USB デバイスの機能は、Linux カーネルの USB ガジェットドライバによって提供されます。USB ガジェットドライバは様々な種類のものが用意されていますが、工場出荷イメージでは Armadillo-810 を USB 複合デバイス[1]として使用することができる「UVC Composite Gadget」が有効になっています。

UVC Composite Gadget は以下に示す 3 種類の USB デバイス機能を持っています。各機能は同時に利用することができます。

UVC ガジェット

Armadillo をビデオ出力デバイス、ATDE をビデオ入力デバイスとして扱うことができます。V4L2 インターフェースを利用するキャプチャーアプリケーションなどで利用することができます。

シリアルガジェット

Armadillo と ATDE を、互いにシリアルデバイスとして扱うことができます。シリアル(tty)デバイスで通信するアプリケーションなどで利用することができます。

イーサネットガジェット

Armadillo と ATDE を、互いにネットワークデバイスとして扱うことができます。ソケットで通信するアプリケーションで利用することができます。

6.1.1. UVC ガジェット

Armadillo-810 の UVC ガジェットは、「UVC(USB Video Class)」として実装されています。ATDE が UVC ガジェットを認識すると、/dev/video0 というデバイスファイルが作成されます。アプリケーションは、このデバイスを介して画像を取得することができます。ここでは、「guvcview」というアプリケーションで動作の確認を行います。

6.1.1.1. 起動方法

「図 6.1. guvcview を起動」を実行すると、「図 6.2. guvcview のビデオウィンドウ」と「図 6.3. guvcview のコントロールウィンドウ」の 2 つのウィンドウが立ち上がります。ビデオウィンドウには、UVC ガジェットから取得した映像が表示され、コントロールウィンドウで解像度の切り替えなどを行うことが出来ます。

```
[ATDE ~]$ guvcview -w 0
```

図 6.1 guvcview を起動

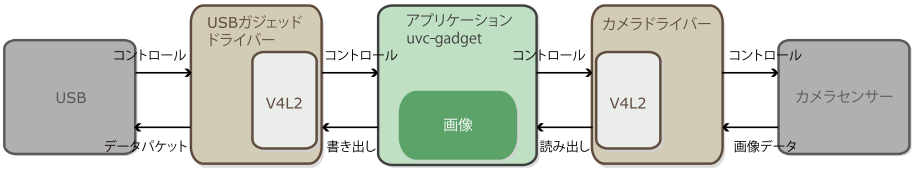
[1]複数の USB 機能を持ったデバイスのこと。

図 6.2 guvcview のビデオウィンドウ[2]

図 6.3 guvcview のコントロールウィンドウ

UVC ガジェットの機能は、UVC ガジェットドライバとカメラドライバで画像を転送するような形で実現しています。2 つのドライバは「V4L2(Video for Linux 2)」という共通のインターフェースを持っています。V4L2 を利用して画像を転送するのが「uvc-gadget」というアプリケーションです。UVC ガジェットの概念図を次に示します。

[2]画面は、ハメコミ合成しています。



6.1.1.2. カメラ設定

「6.1.1.1. 起動方法」に従って **guvcview** を起動した状態で、リアルタイムにカメラの設定を変更することができます。カメラの設定には、「**camctrl**」というアプリケーションを使用します。

camctrl は、動作中のカメラ設定を変更および取得するサンプルアプリケーションです。ソースコードは MIT ライセンスで配布されています。

camctrl のソースコードは、Atmark Dist のソースコード(`user/camctrl/`)に含まれています。

camctrl コマンドは、ユーザーランドイメージ v1.03(Atmark Dist v20131018)で追加されました。ユーザーランドイメージ v1.02(Atmark Dist v20130704)以前には含まれていません。

camctrl コマンドのフォーマットは、次の通りです。

```
[armadillo ~]# camctrl [アプリケーションオプション] [Set オプション|Get オプション] [ヘルプオプション]
```

図 6.4 camctrl コマンド書式

「アプリケーションオプション」の"--set"を指定した場合はカメラ設定の変更を、"--get"を指定した場合は現在のカメラ設定の取得を行います。"--set"と"--get"のどちらも指定しなかった場合は、"--get"が指定された場合と同じ挙動を行います。

「アプリケーションオプション」を次に示します。

表 6.1 camctrl のアプリケーションオプション

オプション	説明
--set	「Set オプション」に従って、カメラ設定を変更します。"--get"オプションと同時に指定することはできません。
--get	「Get オプション」に従って、現在のカメラ設定を取得します。"--set"オプションと同時に指定することはできません。
--version	バージョン番号を表示します。

カメラ設定の変更を行う際に、どの設定を変更するかを決定するための「Set オプション」を次に示します。「アプリケーションオプション」には"--set"を指定する必要があります。

表 6.2 camctrl の Set オプション

オプション	設定値	説明
--gain	16～496 までの整数(10 進数)	ゲイン値を設定します(自動ゲイン調整機能を無効化します)。この設定は画面全体の明さに影響します。分解能または計算誤差により、設定値と実際の設定が異なる場合があります。"--auto-gain"オプションと同時に指定することはできません。
--exposure	0～1081217 までの整数(10 進数)	露光時間をマイクロ秒単位で設定します(自動露光時間調整機能を無効化します)。この設定は画面全体の明さや動体検出時の時間分解能に影響します。分解能または計算誤差により、設定値と実際の設定が異なる場合があります。"--auto-exposure"オプションと同時に指定することはできません。
--red	128～510 までの整数(10 進数)	青、赤、緑色それぞれのゲイン値を設定します(自動ホワイトバランス調整機能を無効化します)。この設定は色合いに影響します。青、赤、緑色全てのゲイン値が 255 以下または 256 以上となるよう設定してください[a]。分解能または計算誤差により、設定値と実際の設定が異なる場合があります。"--auto-white-balance"オプションと同時に指定することはできません。
--green		
--blue		
--auto-gain	"on" または "off"	自動ゲイン調整機能の有効/無効を設定します。"--gain"オプションと同時に指定することはできません。
--auto-exposure	"on" または "off"	自動露光時間調整機能の有効/無効を設定します。"--exposure"オプションと同時に指定することはできません。
--auto-white-balance	"on" または "off"	自動ホワイトバランス調整機能の有効/無効を設定します。"--red"、"--green"および"--blue"オプションと同時に指定することはできません。
--colorbar	"on" または "off"	カラーバー出力の有効/無効を設定します。有効化すると、現在取得中の画像にカラーバーが重ね合わせて表示されます。
--dsp-colorbar	"on" または "off"	DSP[b]カラーバー出力の有効/無効を設定します。有効化すると、カラーバーのみが表示されます。

[a]"--red=100 --green=100 --blue=300"のように 255 以下と 256 以上のゲイン値を混ぜて設定した場合は、正しく設定されません。
[b]Digital Signal Processor

カメラ設定の取得を行う際に、どの設定を取得するかを決定するための「Get オプション」を次に示します。「アプリケーションオプション」には"--get"を指定するか、未指定である必要があります。

表 6.3 camctrl の Get オプション

オプション	説明
--all	全設定の設定値を表示します。
--gain	ゲイン値を表示します。表示される数値は 10 進数の整数です。
--exposure	露光時間を表示します。表示される数値はマイクロ秒単位です。
--red	赤、緑、青色それぞれのゲイン値を表示します。表示される数値は 10 進数の整数です。
--green	
--blue	
--auto-gain	自動ゲイン調整機能の有効/無効を表示します。有効時には"on"が、無効時には"off"が表示されます。
--auto-exposure	自動露光時間調整機能の有効/無効を表示します。有効時には"on"が、無効時には"off"が表示されます。
--auto-white-balance	自動ホワイトバランス調整機能の有効/無効を表示します。有効時には"on"が、無効時には"off"が表示されます。
--colorbar	カラーバー出力の有効/無効を表示します。有効時には"on"が、無効時には"off"が表示されます。
--dsp-colorbar	DSP カラーバー出力の有効/無効を表示します。有効時には"on"が、無効時には"off"が表示されます。

camctrl コマンドの使用例を次に示します。

```
[armadillo ~]# camctrl --get --all ❶
                gain: 20
            exposure: 8381
                 red: 164
               green: 128
                blue: 312
           auto-gain: on
```

```
       auto-exposure: on
  auto-white-balance: on
           colorbar: off
        dsp-colorbar: off
[armadillo ~]# camctrl --set --exposure=5000 ❷
[armadillo ~]# camctrl --get --all ❸
               gain: 20
           exposure: 4998 ❹
                red: 164
              green: 128
               blue: 312
          auto-gain: on
       auto-exposure: off ❺
  auto-white-balance: on
           colorbar: off
        dsp-colorbar: off
```

❶ 全設定の設定値を表示します。

❷ 露光時間を 5000 マイクロ秒に設定します。自動露光時間調整機能は無効化されます。

❸ 設定が反映されていることを確認するため、再度全設定の設定値を表示します。

❹ 露光時間が設定されていることが確認できます。分解能により誤差が生じています。

❹ 自動露光時間調整機能が無効に設定されていることが確認できます。

図 6.5 camctrl コマンドの使用例

guvcview の再起動やコントロールウィンドウから解像度の切り替えなどの制御を行った場合には、**camctrl** で設定した設定値は初期化されます。

各オプションの使用方法は、「ヘルプオプション」を使用することで表示可能です。「ヘルプオプション」を次に示します。

表 6.4 camctrl のヘルプオプション

オプション	説明
--help	「ヘルプオプション」と「アプリケーションオプション」の使用方法を表示します。
--help-all	全オプションの使用方法を表示します。
--help-get	「Get オプション」の使用方法を表示します。
--help-set	「Set オプション」の使用方法を表示します。

6.1.2. シリアルガジェット

Armadillo-810 のシリアルガジェットは、「CDC-ACM(USB Communication Device Class - Abstract Control Model)」として実装されています。Armadillo-810 では/dev/ttyGS0 を、ATDE では/dev/ttyACM0 を使用したシリアル通信を行うことができます。

シリアルガジェットの動作を確認するには、Armadillo-810 の/dev/ttyGS0 にシリアルコンソールを起動させます。ATDE のシリアル通信ソフトウェア(minicom)を用いることで、シリアルガジェット経由で Armadillo-810 にログインすることができます。

手順 6.1 CDC-ACM 通信確認手順

1. ATDE で minicom を起動します。シリアルデバイスには/dev/ttyACM0 を指定します。

```
[ATDE ~]$ minicom -o -w -D /dev/ttyACM0
```

図 6.6 /dev/ttyACM0 を指定してシリアルターミナルを起動

2. Armadillo で getty を起動します。シリアルデバイスには ttyGS0[3]を指定します。/etc/inittab の設定を有効にするためには、プロセス ID が 1 である init プロセスに SIGHUP シグナルを送る必要があります。

```
[armadillo ~]# echo ::respawn:/sbin/getty -L 115200 ttyGS0 >> /etc/inittab
[armadillo ~]# kill -SIGHUP 1
```

図 6.7 /dev/ttyGS0 上でシリアルコンソールを起動

ATDE の minicom にログインプロンプトが表示されます。ユーザー「guest」でログインすることができます。

以下のように/etc/securetty に端末(シリアルデバイス)を登録すると、特権ユーザー「root」でログインすることが可能になります。

```
[armadillo ~]# echo ttyGS0 >> /etc/securetty
```

6.1.3. イーサネットガジェット

Armadillo-810 のイーサネットガジェットは、「RNDIS(Remote NDIS)」として実装されています。Armadillo-810 と ATDE は、互いにネットワークインターフェース usb0 を使用したネットワーク通信を行うことができます。

Armadillo-810 と ATDE をイーサネットガジェットで接続すると、IPv4LL という機構を使ってリンクローカルアドレス[4]が設定されます。

ATDE5 のネットワーク設定は、ネットワークマネージャーを利用せずに "/etc/network/interfaces" に基づいて設定されています。

ATDE でイーサネットガジェットを認識したかどうかは、"ifconfig"コマンドの出力結果をみると判断することができます。

```
[ATDE ~]$ LANG=C sudo ifconfig
(省略)
```

[3]/dev/を指定する必要はありません。
[4]IPv4LL によって割り当てられる特定のアドレス範囲(169.254.0.1～169.254.255.254)の IP アドレス。

```
usb0      Link encap:Ethernet  HWaddr a2:e4:92:6f:c6:54 ❶
          inet6 addr: fe80::a0e4:92ff:fe6f:c654/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:12 errors:0 dropped:0 overruns:0 frame:0
          TX packets:77 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:905 (905.0 B)  TX bytes:13943 (13.6 KiB)

usb0:avahi Link encap:Ethernet  HWaddr a2:e4:92:6f:c6:54 ❷
          inet addr:169.254.9.18  Bcast:169.254.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

❶ イーサネットガジェットを認識すると"usb0"が表示されます

❷ リンクローカルアドレスが割り当てられた場合に"usb0:avahi"が表示されます

図 6.8 イーサネットガジェット認識時の ifconfig の出力例

イーサネットガジェットのネットワークインターフェース(usb0)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。

Armadillo-810 と ATDE をイーサネットガジェットで接続すると、マルチキャスト DNS(mDNS)という技術を利用してローカルネットワークで利用することができるホスト名が設定されます。mDNS により設定されるホスト名を次に示します。

表 6.5 mDNS で設定されるホスト名

ホスト	ホスト名
Armadillo-810	armadillo810-0.local
ATDE5	atde5.local

ATDE5 が所属するネットワーク内に複数の ATDE5 が存在する場合は、mDNS で設定されるホスト名が重複しないように"atde5-2.local"のようなホスト名が設定されます。

ATDE5 に mDNS で設定されたホスト名を確認するには、次のようにコマンドを実行します。

```
[ATDE ~]$ ps axu | grep avahi-daemon
avahi     2464  0.0  0.0  34156  1700 ?        S    12:40   0:00 avahi-
daemon: running [atde5-2.local]
(省略)
```

ネットワーク設定が完了すると、ネットワーク通信ができる状態となります。ping で通信させてみましょう。

```
[ATDE ~]$ ping armadillo810-0.local
```

図 6.9 イーサネットガジェットの通信確認

6.2. ネットワーク

イーサネットガジェットを利用することにより、ネットワーク機能を利用することができます。ここでは、ネットワークの設定方法やネットワークを利用するアプリケーションについて説明します。

6.2.1. ネットワーク設定の変更方法

Armadillo のネットワーク設定の変更方法について説明します。

ネットワーク接続に関する不明な点については、ネットワークの管理者へ相談してください。

6.2.1.1. Armadillo にログインしてネットワーク設定を変更する

Armadillo 上の「/etc/config」以下にあるファイルを編集し、コンフィグ領域に保存することにより起動時のネットワーク設定を変更することができます。コンフィグ領域の保存については、「7. コンフィグ領域 – 設定ファイルの保存領域」を参照してください。

6.2.1.1.1. デフォルト状態のネットワーク設定

ネットワーク設定は、/etc/config/interfaces に記述されています。デフォルト状態では、次のように設定されています。

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

auto lo usb0
iface lo inet loopback
iface eth0 inet dhcp
iface usb0 inet manual ❶
    up ifconfig usb0 up
    post-up zcip usb0 /etc/zcip.script > /dev/null
    down ifconfig usb0 down
```

❶ イーサネットガジェット(usb0)用の設定は、リンクローカルアドレスを利用

図 6.10 デフォルト状態の/etc/config/interfaces

6.2.1.1.2. 固定 IP アドレスに設定する

「表 6.6. 固定 IP アドレス設定例」に示す内容に設定変更するには、vi エディタで/etc/config/interfaces を、「図 6.11. 固定 IP アドレス設定」のように編集します。

表 6.6 固定 IP アドレス設定例

項目	設定
IP アドレス	192.168.10.10
ネットマスク	255.255.255.0
ネットワークアドレス	192.168.10.0
ブロードキャストアドレス	192.168.10.255
デフォルトゲートウェイ	192.168.10.1

```
[armadillo ~]# vi /etc/config/interfaces
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

auto lo usb0
iface lo inet loopback
iface eth0 inet dhcp
iface usb0 inet static
      address 192.168.10.10
      netmask 255.255.255.0
      network 192.168.10.0
      broadcast 192.168.10.255
      gateway 192.168.10.1
```

図 6.11 固定 IP アドレス設定

6.2.1.1.3. DHCP に設定する

イーサネットガジェットは、IPv4LL によってリンクローカルアドレスが設定されるため、通常 DHCP に設定する必要はありません。

DHCP に設定するには、vi エディタで/etc/config/interfaces を、次のように編集します。

```
[armadillo ~]# vi /etc/config/interfaces
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

auto lo usb0
iface lo inet loopback
iface eth0 inet dhcp
iface usb0 inet dhcp
```

図 6.12 DHCP 設定

6.2.1.1.4. DNS サーバーを指定する

DNS サーバーを指定する場合は、vi エディタで/etc/config/resolv.conf を編集します。

```
[armadillo ~]# vi /etc/config/resolv.conf
nameserver 192.168.10.1
```

図 6.13 DNS サーバーの設定

DHCP を利用している場合には、DHCP サーバーが DNS サーバーを通知する場合があります。この場合、/etc/config/resolv.conf は自動的に更新されます。

6.2.1.2. 接続を確認する

ここでは、変更した IP 設定で正常に通信が可能か確認します。次のように設定を反映させます。

```
[armadillo ~]# ifdown -a
[armadillo ~]# ifup -a
```

図 6.14 設定を反映させる

ネットワークの設定をコンフィグ領域に保存し、Armadillo を再起動している場合には「図 6.14. 設定を反映させる」に示す操作は不要です。

同じネットワーク内にある通信機器と PING 通信を行います。

```
[armadillo ~]# ping 192.168.10.20
```

図 6.15 PING 確認

6.2.2. ファイアーウォール

Armadillo では、簡易ファイアーウォールが動作しています。設定されている内容を参照するには、「図 6.16. iptables」のようにコマンド実行してください。

```
[armadillo ~]# iptables --list
```

図 6.16 iptables

6.2.3. ネットワークアプリケーション

工場出荷イメージで利用することができるネットワークアプリケーションについて説明します。

ATDE と Armadillo のネットワーク設定がデフォルト状態であることを想定して記述しています。ネットワーク設定を変更している場合は適宜読み換えてください。

6.2.3.1. TELNET

ATDE などの PC からネットワーク経由でログインし、リモート操作することができます。ログイン可能なユーザを次に示します。

表 6.7 TELNET でログイン可能なユーザ

ユーザ名	パスワード
guest	(なし)

TELNET を使用して ATDE から Armadillo にリモートログインする場合の例を、次に示します。

```
[ATDE ~]$ telnet armadillo810-0.local
Trying 169.254.245.125...
Connected to armadillo810-0.local.
Escape character is '^]'.

atmark-dist v1.31.0 (AtmarkTechno/Armadillo-810)
Linux 3.4-at1 [armv7l arch]

armadillo810-0 login: guest ❶
[guest@armadillo ~]$
[guest@armadillo ~]$ su ❷
Password: ❸
[root@armadillo ~]#
[root@armadillo ~]# exit ❹
[guest@armadillo ~]$ exit ❺
Connection closed by foreign host.
[ATDE ~]$
```

❶ "guest"と入力するとログインすることができます。パスワードの入力は不要です。

❷ 特権ユーザーとなる場合には"su"コマンドを実行します。

❸ 特権ユーザーのデフォルトパスワードは"root"です。

❹ 特権トユーザーから guest ユーザーに戻る場合は、"exit"と入力します

❺ telnet を終了するにはもう一度"exit"を入力します

図 6.17 telnet でリモートログイン

6.2.3.2. FTP

ATDE などの PC からネットワーク経由でファイル転送することができます。次に示すユーザでログインすることができます。

表 6.8 ftp でログイン可能なユーザ

ユーザ名	パスワード
ftp	(なし)

ftp を使用して ATDE から Armadillo にファイルを転送する場合の例を、次に示します。

```
[ATDE ~]$ ls -l file
-rw-r--r-- 1 atmark atmark 1048576 Jan 1 12:00 file
[ATDE ~]$ ftp armadillo810-0.local
Connected to armadillo810-0.local.
220 localhost FTP server (GNU inetutils 1.4.1) ready.
Name (armadillo810-0.local:atmark): ftp
331 Guest login ok, type your name as password.
Password: ❶
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd pub ❷
250 CWD command successful.
ftp> put file ❸
local: file remote: file
200 PORT command sucessful.
150 Opening BINARY mode data connection for 'file'.
226 Transfer complete.
1048576 bytes sent in 0.14 secs (7399.5 kB/s)
ftp> quit ❹
221 Goodbye.
[ATDE ~]$
```

❶ ftp ユーザにパスワードが設定されていないため Enter キーを入力します。

❷ ファイル転送することができる pub ディレクトリに移動します。

❸ ファイルをアップロードします。ダウンロードする場合は"get"コマンドを使用します。

❹ ftp を終了する場合は"quit"と入力します。

図 6.18 ftp でファイル転送

ATDE から Armadillo にファイルをアップロードすると、/home/ftp/pub/ディレクトリ以下にファイルが作成されています。ダウンロードする場合も、同じディレクトリにファイルを配置してください。

```
[armadillo ~]# cd /home/ftp/pub/
[armadillo /home/ftp/pub]# ls
file
```

図 6.19 Armadillo 上でアップロードされたファイルを確認

6.2.3.3. HTTP サーバー

Armadillo では、HTTP サーバーが動作しています。ATDE などの PC の Web ブラウザから Armadillo の URL (http://armadillo810-0.local/ または、http://[Armadillo の IP アドレス]/ [5])にアクセスすると、Armadillo のトップページ(index.html)が表示されます。

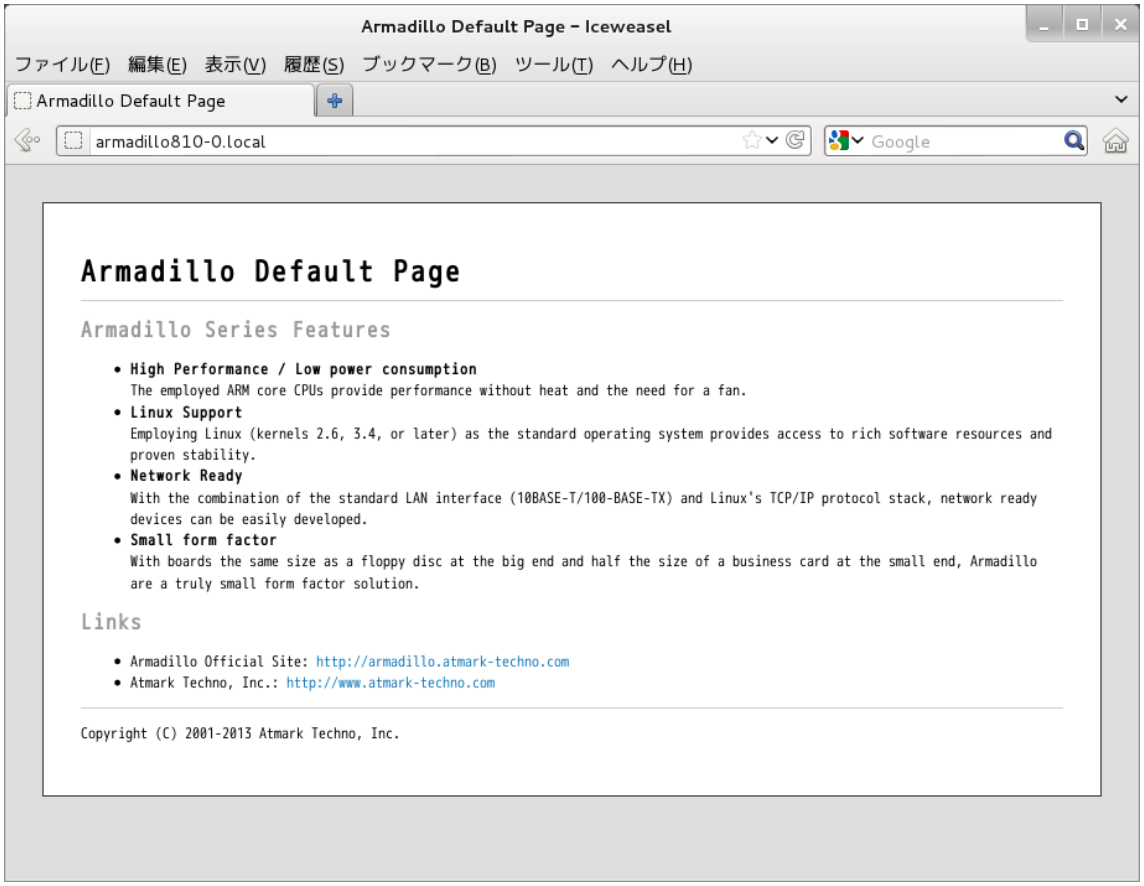


図 6.20 Armadillo トップページ

6.3. シリアル

Armadillo-810 でシリアルポートとして使用可能なデバイスを次に示します。

表 6.9 シリアルデバイス

インターフェース	デバイスファイル(ATDE5)	デバイスファイル(Armadillo-810)
シリアルインターフェース 1(CON2)	/dev/ttyS0	/dev/ttySC8
シリアルインターフェース 2(CON3)	/dev/ttyUSB0	/dev/ttySC2
USB インターフェース(CON4)[a]	/dev/ttyACM0	/dev/ttyGS0

[a] 「6.1. USB ガジェット」に示すシリアルガジェットを使用します。

「表 6.9. シリアルデバイス」に示すデバイスファイルは、「4.5. 接続方法」に従って接続された場合のものです。異なる接続をしている場合は適宜読み替えてください。

[5] Armadillo の IP アドレスが 192.168.10.10 の場合、http://192.168.10.10/ となります。

6.3.1. シリアルコンソールとして使用する

シリアルの動作を確認するには、シリアルコンソールを起動させます。ATDE のシリアル通信ソフトウェア(minicom)を用いることで、シリアル経由で Armadillo にログインすることができます。

Armadillo-810 の工場出荷イメージでは、シリアルインターフェース 2(CON3)が標準でシリアルコンソールとして使用できるようになっています。

シリアルインターフェース 1(CON2)をシリアルコンソールとして使用する手順を次に示します。

手順 6.2 シリアルコンソールとして使用

1. ATDE で minicom を起動します。シリアルデバイスには/dev/ttyS0 を指定します。

```
[ATDE ~]$ minicom -o -w -D /dev/ttyS0
```

2. Armadillo で getty を起動します。シリアルデバイスには ttySC8[6]を指定します。/etc/inittab の設定を有効にするためには、プロセス ID が 1 である init プロセスに SIGHUP シグナルを送る必要があります。

```
[armadillo ~]# echo ::respawn:/sbin/getty -L 115200 ttySC8 >> /etc/inittab
[armadillo ~]# kill -SIGHUP 1
```

ATDE の minicom にログインプロンプトが表示されます。ユーザー「guest」でログインすることができます。

以下のように/etc/securetty に端末(シリアルデバイス)を登録すると、特権ユーザー「root」でログインすることが可能になります。

```
[armadillo ~]# echo ttySC8 >> /etc/securetty
```

6.4. ストレージ

Armadillo-810 でストレージとして使用可能なデバイスを次に示します。

表 6.10 ストレージデバイス

デバイス種類	ディスクデバイス	先頭パーティション
USB フラッシュメモリ	/dev/sd*[a]	/dev/sd*1
SD カード	/dev/mmcblk0	/dev/mmcblk0p1

[a]USB ハブを利用して複数の USB メモリを接続した場合は、認識された順に sda sdb sdc … となります。

[6]/dev/を指定する必要はありません。

6.4.1. ストレージの使用方法

ここでは、SD カードを例にストレージの使用方法を説明します。

Linux では、アクセス可能なファイルやディレクトリは、一つの木構造にまとめられています。あるストレージデバイスのファイルシステムを、この木構造に追加することを、マウントするといいます。マウントを行うコマンドは、mount です。

mount コマンドの典型的なフォーマットは、次の通りです。

```
mount -t fstype device dir
```

図 6.21 mount コマンド書式

-t オプションに続く device には、ファイルシステムタイプを指定します[7]。FAT32 ファイルシステムの場合は vfat[8]、EXT3 ファイルシステムの場合は ext3 を指定します。

device には、ストレージデバイスのデバイスファイル名を指定します。SD カードのパーティション 1 の場合は/dev/mmcblk0p1、パーティション 2 の場合は/dev/mmcblk0p2 となります。

dir には、ストレージデバイスのファイルシステムをマウントするディレクトリを指定します。

SD スロットに SD カードを挿入した状態で「図 6.22. ストレージのマウント」に示すコマンドを実行すると、/mnt ディレクトリに SD カードのファイルシステムをマウントします。SD カード内のファイルは、/mnt ディレクトリ以下に見えるようになります。

```
[armadillo ~]# mount -t vfat /dev/mmcblk0p1 /mnt
```

図 6.22 ストレージのマウント

FAT32 ファイルシステムをマウントした場合、次の警告メッセージが表示される場合があります。

```
FAT-fs (mmcblk0p1): utf8 is not a recommended IO charset for
FAT filesystems, filesystem will be case sensitive!
```

これは無視して構いません。 UTF-8 ロケールでは結局はファイル名の表示を正しく処理できないためです。

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。オプションとして、アンマウントしたいデバイスがマウントされているディレクトリを指定します。

[7]ファイルシステムタイプの指定は省略可能です。省略した場合、mount コマンドはファイルシステムタイプを推測します。この推測は必ずしも適切なものとは限りませんので、事前にファイルシステムタイプが分かっている場合は明示的に指定してください。
[8]通常、購入したばかりの SD カードは FAT32 ファイルシステムでフォーマットされています。

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

図 6.23 ストレージのアンマウント

6.4.2. ストレージのパーティション変更とフォーマット

通常、購入したばかりの SD カードや USB メモリは、一つのパーティションを持ち、FAT32 ファイルシステムでフォーマットされています。

パーティション構成を変更したい場合、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 6.24. fdisk コマンドによる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二つ作成しています。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。先頭のパーティションは/dev/mmcblk0p1、二つめは/dev/mmcblk0p2 となります。fdisk コマンドの詳細な使い方は、man ページ等をご参照ください。

```
[armadillo ~]# fdisk /dev/mmcblk0

The number of cylinders for this disk is set to 62528.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): d
Selected partition 1

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-62528, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-62528, default 62528): +100M

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (3054-62528, default 3054):
Using default value 3054
Last cylinder or +size or +sizeM or +sizeK (3054-62528, default 62528):
Using default value 62528

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

Calling ioctl() to re-read partition table.
 mmcblk0: p1 p2
```

```
 mmcblk0: p1 p2
Syncing disks.
```

図 6.24 fdisk コマンドによるパーティション変更

FAT32 ファイルシステムでストレージデバイスをフォーマットするには、mkfs.vfat コマンドを使用します。また、EXT2 や EXT3 ファイルシステムでフォーマットするには、mke2fs コマンドを使用します。SD カードのパーティション 1 を EXT3 ファイルシステムでフォーマットするコマンド例を、次に示します。

```
[armadillo ~]# mke2fs -j /dev/mmcblk0p1
```

図 6.25 EXT3 ファイルシステムの構築

6.5. AV コーデックミドルウェア

AV コーデックミドルウェアを使い、Armadillo-810 カメラモジュール 01 (B コネクタ用)のカメラデバイスから取り込んだ画像を H.264/AVC でエンコードし、MP4(MPEG-4 Part 14)コンテナに格納する方法を説明します。

AV コーデックミドルウェアには、Atmark Dist v20140131 以降(ユーザーランドイメージ romfs-a810-v1.04.img 以降)、Linux カーネル v3.4-at6 以降(カーネルイメージ linux-a810-v1.05.img.gz 以降)で対応しています。それ以前のものを使用されている場合、本節で説明する動作確認を行う前にイメージを対応バージョンに書き換えてください。

標準状態では uvc-gadget が起動しており、それがカメラデバイスを使用しています。AV コーデックミドルウェアでカメラデバイスを扱う前に、uvc-gadget を停止させてください。

```
[armadillo ~]# killall uvc-gadget
```

図 6.26 uvc-gadget の停止

また、エンコード結果のファイルはサイズが大きくなるので、ストレージに保存します。「6.4. ストレージ」を参照して、USB メモリや SD カード等のストレージデバイスを/mnt にマウントしておいてください。

Armadillo-810 カメラモデルの場合、Armadillo-810 カメラモジュール 01 (B コネクタ用)のデバイスファイルは/dev/video1 となります。下記のコマンドを実行すると、カメラデバイスから取り込んだ画像をエンコードし、MP4 ファイルに保存します。エンコードを停止する場合は、Ctrl+c を入力してください。

```
[armadillo ~]# gst-launch-1.0 -e v4l2src device=/dev/video1 \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmh264enc ! queue \
 ! qtmux ! filesink location=/mnt/output.mp4
Setting pipeline to PAUSED ...
```

```
sh_mobile_ceu sh_mobile_ceu.0: SuperH Mobile CEU driver attached to camera 0
Pipeline is live and does not need PREROLL ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
WARNING: from element /GstPipeline:pipeline0/GstV4l2Src:v4l2src0: Could not get
parameters on device '/dev/video1'
Additional debug info:
gstv4l2object.c(2445): gst_v4l2_object_set_format (): /GstPipeline:pipeline0/Gst
V4l2Src:v4l2src0:
system error: Inappropriate ioctl for device
Redistribute latency...
```

図 6.27 H.264/AVC 動画のエンコード

次のように num-buffers プロパティを指定すると、カメラデバイスから取り込む画像の枚数を指定することできます。

```
[armadillo ~]# gst-launch-1.0 -e v4l2src device=/dev/video1 \
 num-buffers=300 \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmh264enc ! queue \
 ! qtmux ! filesink location=/mnt/output.mp4
```

6.6. LED

Armadillo-810 の LED は、LED クラスとして実装されています。LED クラスディレクトリ以下のファイルによって LED の制御を行うことができます。LED クラスディレクトリと LED の対応については、「表 6.11. LED クラスディレクトリと LED の対応」を参照してください。

表 6.11 LED クラスディレクトリと LED の対応

LED クラスディレクトリ	説明	デフォルトトリガ
/sys/class/leds/LED1/	LED1 (黄)	none
/sys/class/leds/LED2/	LED2 (黄)	none
/sys/class/leds/LED3/	LED3 (黄)	none
/sys/class/leds/LED4/	LED4 (黄)	none

以降の説明では、任意の LED を示す LED クラスディレクトリを"/sys/class/leds/[LED]"のように表記します。

6.6.1. LED を点灯/消灯する

LED クラスディレクトリ以下の brightness ファイルへ値を書き込むことによって、LED の点灯/消灯を行うことができます。brightness に書き込む有効な値は 0～255 です。

brightness に 0 以外の値を書き込むと LED が点灯します。

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

図 6.28 LED を点灯させる

Armadillo-810 の LED には輝度制御の機能が無いため、0 (消灯)、1〜255 (点灯)の 2 つの状態のみ指定することができます。

brightness に 0 を書き込むと LED が消灯します。

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

図 6.29 LED を消灯させる

brightness を読み出すと LED の状態が取得できます。

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

図 6.30 LED の状態を表示する

6.6.2. トリガを使用する

LED クラスディレクトリ以下の trigger ファイルへ値を書き込むことによって LED の点灯/消灯にトリガを設定することができます。trigger に書き込む有効な値を次に示します。

表 6.12 trigger の種類

設定	説明
none	トリガを設定しません。
mmc0	SD カード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。
heartbeat	心拍のように点灯/消灯を行います。工場出荷イメージでは設定することができません。
default-on	主にカーネルから使用します。起動時に LED が点灯します。工場出荷イメージでは設定することができません。

以下のコマンドを実行すると、LED が 1 秒点灯、500 ミリ秒消灯を繰り返します。

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

図 6.31 LED のトリガに timer を指定する

trigger を読み出すと LED のトリガが取得できます。"[ ]"が付いているものが現在のトリガです。

```
[armadillo ~]# cat /sys/class/leds/[LED]/trigger
[none] mmc0 timer
```

図 6.32 LED のトリガを表示する

6.7. GPIO

Armadillo-810 の GPIO は、generic GPIO として実装されています。GPIO クラスディレクトリ以下のファイルによって GPIO の制御を行うことができます。GPIO クラスディレクトリと GPIO の対応を次に示します。

表 6.13 Armadillo-810 拡張ボード 01 (A コネクタ用)の CON2 の GPIO ディレクトリ

ピン番号	GPIO ディレクトリ
CON2 1 ピン	/sys/class/gpio/gpio66/
CON2 2 ピン	/sys/class/gpio/gpio67/
CON2 3 ピン	/sys/class/gpio/gpio68/
CON2 4 ピン	/sys/class/gpio/gpio69/
CON2 5 ピン	/sys/class/gpio/gpio70/
CON2 6 ピン	/sys/class/gpio/gpio71/
CON2 7 ピン	/sys/class/gpio/gpio72/
CON2 8 ピン	/sys/class/gpio/gpio73/
CON2 9 ピン	/sys/class/gpio/gpio74/
CON2 10 ピン	/sys/class/gpio/gpio75/
CON2 13 ピン	/sys/class/gpio/gpio160/
CON2 14 ピン	/sys/class/gpio/gpio161/
CON2 15 ピン	/sys/class/gpio/gpio159/
CON2 16 ピン	/sys/class/gpio/gpio162/
CON2 18 ピン	/sys/class/gpio/gpio7/
CON2 19 ピン	/sys/class/gpio/gpio8/
CON2 20 ピン	/sys/class/gpio/gpio10/
CON2 21 ピン	/sys/class/gpio/gpio11/
CON2 22 ピン	/sys/class/gpio/gpio20/
CON2 23 ピン	/sys/class/gpio/gpio12/
CON2 24 ピン	/sys/class/gpio/gpio5/
CON2 25 ピン	/sys/class/gpio/gpio13/
CON2 26 ピン	/sys/class/gpio/gpio9/

以降の説明では、任意の GPIO を示す GPIO クラスディレクトリを"/sys/class/gpio/[GPIO]"のように表記します。

6.7.1. 入出力方向を変更する

GPIO ディレクトリ以下の direction ファイルへ値を書き込むことによって、入出力方向を変更することができます。direction に書き込む有効な値を次に示します。

表 6.14 direction の設定

設定	説明
high	入出力方向を OUTPUT に設定します。出力レベルの取得/設定を行うことができます。出力レベルは HIGH レベルになります。
out	入出力方向を OUTPUT に設定します。出力レベルの取得/設定を行うことができます。出力レベルは LOW レベルになります。
low	out を設定した場合と同じです。
in	入出力方向を INPUT に設定します。入力レベルの取得を行うことができますが設定はできません。

6.7.2. 入力レベルを取得する

GPIO ディレクトリ以下の value ファイルから値を読み出すことによって、入力レベルを取得することができます。"0"は LOW レベル、"1"は HIGH レベルを表わします。入力レベルの取得は入出力方向が INPUT, OUTPUT のどちらでも行うことができます。入出力方向が OUTPUT の時に読み出される値は、GPIO ピンの状態ではなく、自分が value ファイルに書き込んだ値となります。

```
[armadillo ~]# cat /sys/class/gpio/[GPIO]/value
0
```

図 6.33 GPIO の入力レベルを取得する

6.7.3. 出力レベルを設定する

GPIO ディレクトリ以下の value ファイルへ値を書き込むことによって、出力レベルを設定することができます。"0"は LOW レベル、"0"以外は HIGH レベルを表わします。出力レベルの設定は入出力方向が OUTPUT でなければ行うことはできません。

```
[armadillo ~]# echo 1 > /sys/class/gpio/[GPIO]/value
```

図 6.34 GPIO の出力レベルを設定する

6.8. RTC

Armadillo-810 拡張ボード 01 (A コネクタ用)には、カレンダ時計(Real Time Clock)が実装されています。電源を切断しても一定時間(平均 300 秒間、最小 60 秒間)時刻を保持することができます

長時間電源が切断されても時刻データを保持させたい場合は、Armadillo-810 拡張ボード 01 (A コネクタ用)の CON8(RTC 外部バックアップ)にバッテリ(対応電池: CR2032)を接続します。

6.8.1. RTC に時刻を設定する

Linux の時刻には、Linux カーネルが管理するシステムクロックと、RTC が管理するハードウェアクロックの 2 種類があります。RTC 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。

システムクロックは、date コマンドを用いて設定します。date コマンドの引数には、設定する時刻を[MMDDhhmmCCYY.ss]というフォーマットで指定します。時刻フォーマットの各フィールドの意味を次に示します。

表 6.15 時刻フォーマットのフィールド

フィールド	意味
MM	月
DD	日(月内通算)
hh	時
mm	分
CC	年の最初の 2 桁(省略可)
YY	年の最後の 2 桁(省略可)
ss	秒(省略可)

2013 年 1 月 23 日 4 時 56 分 00 秒に設定する例を次に示します。

```
[armadillo ~]# date ❶
Sat Jan  1 09:00:00 JST 2000
[armadillo ~]# date 012304562013.00 ❷
Wed Jan 23 04:56:00 JST 2013
[armadillo ~]# date ❸
Wed Jan 23 04:56:00 JST 2013
```

❶ 現在のシステムクロックを表示します。

❷ システムクロックを設定します。

❸ システムクロックが正しく設定されていることを確認します。

図 6.35 システムクロックを設定

システムクロックを設定後、ハードウェアクロックを hwclock コマンドを用いて設定します。

```
[armadillo ~]# hwclock ❶
Sat Jan  1 00:00:00 2000  0.000000 seconds
[armadillo ~]# hwclock --utc --systohc ❷
[armadillo ~]# hwclock --utc ❸
Wed Jan 23 04:56:10 2013  0.000000 seconds
```

❶ 現在のハードウェアクロックを表示します。

❷ ハードウェアクロックを協定世界時(UTC)で設定します。

❸ ハードウェアクロックが UTC で正しく設定されていることを確認します。

図 6.36 ハードウェアクロックを設定

7. コンフィグ領域 – 設定ファイルの保存領域

コンフィグ領域は、設定ファイルなどを保存しハードウェアのリセット後にもデータを保持することができるフラッシュメモリ領域です。コンフィグ領域からのデータの読出し、またはコンフィグ領域への書込みは、flatfsd コマンドを使用します。

7.1. コンフィグ領域の読出し

コンフィグ領域を読み出すには以下のコマンドを実行します。読み出されたファイルは、「/etc/config」ディレクトリに作成されます。

```
[armadillo ~]# flatfsd -r
```

図 7.1 コンフィグ領域の読出し方法

デフォルトのソフトウェアでは、起動時に自動的にコンフィグ領域の読出しを行うように設定されています。コンフィグ領域の情報が壊れている場合、「/etc/default」ディレクトリの内容が反映されます。

7.2. コンフィグ領域の保存

コンフィグ領域を保存するには以下のコマンドを実行します。保存されるファイルは、「/etc/config」ディレクトリ以下のファイルです。

```
[armadillo ~]# flatfsd -s
```

図 7.2 コンフィグ領域の保存方法

コンフィグ領域の保存をおこなわない場合、「/etc/config」ディレクトリ以下のファイルへの変更は電源遮断時に失われます。

7.3. コンフィグ領域の初期化

コンフィグ領域を初期化するには以下のコマンドを実行します。初期化時には、「/etc/default」ディレクトリ以下のファイルがコンフィグ領域に保存され、且つ「/etc/config」ディレクトリにファイルが複製されます。

```
[armadillo ~]# flatfsd -w
```

図 7.3 コンフィグ領域の初期化方法

8. Linux カーネル仕様

本章では、工場出荷状態の Armadillo-810 の Linux カーネルの仕様について説明します。

8.1. デフォルトコンフィグレーション

工場出荷状態のフラッシュメモリに書き込まれている Linux カーネルイメージをビルドする場合には、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-810 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(linux-3.4-[VERSION].tar.gz)に含まれる arch/arm/configs/armadillo810_defconfig です。

armadillo810_defconfig で有効になっている主要な設定を「表 8.1. Linux カーネル主要設定」に示します。

表 8.1 Linux カーネル主要設定

コンフィグ	説明
NO_HZ	Tickless System (Dynamic Ticks)
HIGH_RES_TIMERS	High Resolution Timer Support
PREEMPT	Preemptible Kernel (Low-Latency Desktop)
AEABI	Use the ARM EABI to compile the kernel
VFP	VFP-format floating point maths
NEON	Advanced SIMD (NEON) Extension support
BINFMT_ELF	Kernel support for ELF binaries

8.2. デフォルト起動オプション

工場出荷状態の Armadillo-810 の Linux カーネルの起動オプションについて説明します。デフォルト状態では、次のように設定されています。

表 8.2 Linux カーネルのデフォルト起動オプション

起動オプション	説明
console=ttySC2,115200	起動ログなどが出力されるイニシャルコンソールに ttySC2(Armadillo-810:CON3)を、ボーレートに 115200bps を指定します。
earlyprintk=sh-sci.2,115200	可能な限り早い段階で起動ログを出力するデバイスとして sh-sci.2(Armadillo-810:CON3)を、ボーレートに 115200bps を指定します。Linux カーネルが起動しないような不具合のデバッグに役立ちます。
mem=384M	Linux カーネルが利用可能なメモリを 384MByte に制限します。AV コーデックミドルウェアを使用する場合に必須の設定です。

Linux カーネルイメージ linux-a810-v1.05.bin.gz 以降(linux-3.4-at6 以降)のデフォルト起動オプションです。linux-a840-v1.04.bin.gz 以前(linux-3.4-at5 以前)では、AV コーデックミドルウェアが使用できないため次のように設定されていました。

```
console=ttySC2,115200 earlyprintk=sh-sci.2,115200
```

8.3. Linux ドライバー覧

Armadillo-810 を制御する Linux ドライバ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-armadillo810.c

SoC(R-Mobile A1)固有ドライバー

ソースコード	arch/arm/mach-shmobile/setup-r8a7740.c arch/arm/mach-shmobile/pfc-r8a7740.c arch/arm/mach-shmobile/intc-r8a7740.c arch/arm/mach-shmobile/clock-r8a7740.c

割り込みコントローラードライバー

ソースコードディレクトリ	drivers/sh/intc/

タイマードライバー

ソースコード	drivers/clocksource/sh_cmt_simple.c

MTD マップドライバー

ソースコード	drivers/mtd/maps/physmap.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/ttySC2 (Armadillo-810: CON3) /dev/ttySC8 (Armadillo-810: CON2)

SD ホストドライバー

ソースコード	drivers/mmc/host/sh_mobile_sdhi.c
デバイス	/dev/mmcblk0

USB ホストドライバー

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ー

ソースコードディレクトリ	drivers/usb/renesas_usbhs/

USB ガジェット - UVC コンポジットドライバー

ソースコード	drivers/usb/gadget/uvc_acm_ether.c drivers/usb/gadget/webcam-armadillo810.c
デバイス	/dev/video0 (UVC gadget) /dev/ttyGS0 (CDC-ACM gadget)

ソケット	usb0 (RNDIS/CDC-ECM gadget)

キャプチャーインターフェースドライバー

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

カメラドライバー

ソースコード	drivers/media/video/ov772x.c
デバイス	/dev/video1 (OV7725)

リアルタイムクロックドライバー

ソースコード	drivers/rtc/rtc-s35390a.c
デバイス	/dev/rtc0

LED ドライバー

ソースコード	drivers/leds/leds-gpio.c
デバイス	/sys/class/leds/LED1 (LED1) /sys/class/leds/LED2 (LED2) /sys/class/leds/LED3 (LED3) /sys/class/leds/LED4 (LED4)

オーディオドライバー

ソースコード	sound/soc/sh/fsi.c

I2C バスドライバー

ソースコード	drivers/i2c/busses/i2c-sh_mobile.c (i2c-0, i2c-1) drivers/i2c/busses/i2c-gpio.c (i2c-2)

SPI マスタードライバー

ソースコード	drivers/spi/spi-sh-msiof.c

PWM - バックライトドライバー

ソースコード	drivers/misc/rmob-tpu-pwm.c drivers/video/backlight/backlight.c drivers/video/backlight/pwm_bl.c

AV コーデックミドルウェアドライバー

ソースコードディレクトリ	drivers/media/video/acm/
デバイス	/dev/video2 (H.264/AVC エンコーダー)[1][2] /dev/video3 (AAC エンコーダー)[1][2] /dev/video4 (JPEG エンコーダー)[1][2]

[1]AV コーデックミドルウェアのエンコーダー有効時。
[2]デフォルトカーネルで外部デバイスを何も接続しなかった場合。V4L2 ドライバ対応の USB カメラなどを接続すると、デバイスファイル名が変わる事があります。

/dev/video2 (H.264/AVC デコーダー)[3][2]
/dev/video3 (AAC デコーダー)[3][2]

[3]AV コーデックミドルウェアのデコーダー有効時。

9. ユーザーランド仕様

本章では、工場出荷状態の Armadilloo-810 のユーザーランドの基本的な仕様について説明します。

9.1. 起動処理

Armadillo-810 のユーザーランドの起動処理について説明します。ユーザーランドの起動処理は大きく分けて次の手順で初期化が行われています。

1. Linux カーネルが/sbin/init を実行し/etc/inittab の sysinit に登録されている/etc/init.d/rc スクリプトを実行
2. rc スクリプトの中で、/etc/rc.d/ディレクトリの起動スクリプトを順次実行
3. ローカル起動スクリプト(/etc/config/rc.local)を実行
4. /etc/inittab の respawn タブに登録されたものを実行

9.1.1. inittab

Linux カーネルは、ルートファイルシステムをマウントすると、/sbin/init を実行します。init プロセスは、コンソールの初期化を行い/etc/inittab に記載された設定にしたがってコマンドを実行します。

デフォルト状態の Armadillo-810 の/etc/inittab は次のように設定されています。

```
::sysinit:/etc/init.d/rc

::respawn:/sbin/getty -L 115200 ttySC2 vt102

::shutdown:/etc/init.d/reboot
::ctrlaltdel:/sbin/reboot
```

図 9.1 デフォルト状態の/etc/inittab

inittab の書式は、次のようになっています。

```
id:runlevel:action:process
```

図 9.2 inittab の書式

Armadillo-810 の init では、"id"フィールドに起動されるプロセスが使用するコンソールを指定することができます。省略した場合は、システムコンソールが使用されます。"runlevel"フィールドは未対応のため利用できません。

"action"フィールド及び"process"フィールドは、どのような状態(action)のときに何(process)を実行するかを設定することができます。action フィールドに指定可能な値を「表 9.1. inittab の action フィールドに設定可能な値」に示します。

表 9.1 inittab の action フィールドに設定可能な値

値	process を実行するタイミング
sysinit	init プロセス起動時
respawn	sysinit 終了後。このアクションで起動されたプロセスが終了すると、再度 process を実行する
shutdown	シャットダウンする時
ctrlaltdel	Ctrl-Alt-Delete キーの組み合わせが入力された時

9.1.2. /etc/init.d/rc

rc スクリプトでは、システムの基礎となるファイルシステムをマウントしたり、/etc/rc.d/ディレクトリ以下にある S から始まるスクリプト(初期化スクリプト)が実行できる環境を構築します。その後、初期化スクリプトを実行していきます。初期化スクリプトは、S の後に続く 2 桁の番号の順番で実行します。

9.1.3. /etc/rc.d/S スクリプト(初期化スクリプト)

初期化スクリプトでは、システムの環境を構築するもの、デーモン(サーバー)を起動するものの 2 つの種類があります。Armadillo-810 のデフォルト状態で登録されている初期化スクリプトを「表 9.2. /etc/rc.d ディレクトリに登録された初期化スクリプト」に示します。

表 9.2 /etc/rc.d ディレクトリに登録された初期化スクリプト

スクリプト	初期化内容
S01mtd	フラッシュメモリのパーティション名に従って MTD のデバイスファイルへのシンボリックリンクを作成します
S03flatfsd	flatfsd を使いコンフィグ領域(/etc/config/)を復元します
S05udevd	udevd を起動し、Linux カーネルから発行された uevent をハンドリングします
S06mountdevsubfs	udevd 起動後にマウントする必要のあるファイルシステムをマウントします
S20checkroot	システム関連のファイルのパーミッション設定や、オーナーを設定します
S21checkftp	FTP が利用するファイルやライブラリの配置、パーミッションの設定をします
S30syslogd, S31klogd	ログデーモンを起動します
S40mount	その他のファイルシステムをマウントします
S45firmware	/opt/firmware に配置されたファームウェアファイルから/lib/firmware/ディレクトリにシンボリックリンクを作成します
S45module-init-tools	/etc/modules に記載されたカーネルモジュールをロードします
S50hostname	hostname を設定します
S55firewall, S56networking, S57inetd	ネットワーク関連の初期化を行い、インターネットスーパーサーバー(inetd)を起動します
S60avahi, S60lighttpd, S60sshd	ネットワークデーモンを起動します
S90rc.local	コンフィグ領域(/etc/config/)に保存された rc.local を実行します

9.1.4. /etc/config/rc.local

コンフィグ領域に保存された rc.local は、ユーザーランドイメージを変更することなく、起動時に特定の処理を行うことができるようになっています。

Armadillo-810 では、システム起動時に自動的に uvc-gadget アプリケーションを起動させたり、AV コーデックミドルウェアのファームウェアをロードするために利用しています。カメラデバイス(/dev/video1)を他のアプリケーションで利用する場合は、uvc-gadget を自動起動させないように設定したり、AV コーデックミドルウェアを使わない場合はファームウェアをロードしないように設定できます。

デフォルト状態の/etc/config/rc.local は次のように記載されています。

```
#!/bin/sh

. /etc/init.d/functions

PATH=/bin:/sbin:/usr/bin:/usr/sbin

#
# for USB Gadget "UVC Composite with ACM and RNDIS"
#   - activate UVC
#
USB_GADGET_IS_UVC_COMPOSITE=y ❶
if [ "${USB_GADGET_IS_UVC_COMPOSITE}" = "y" ]; then
   /etc/init.d/uvc-gadget
fi

#
# for AV Codec Midleware
#   - load firmware
#
ACM_CODEC=encoder ❷
if [ "${ACM_CODEC}" = "encoder" -o "${ACM_CODEC}" = "decoder" ]; then
   echo -n "load ${ACM_CODEC} firmware: "
   echo "${ACM_CODEC}" > /sys/devices/platform/acm.0/codec
   for i in 1 2 3 4 5; do
      sleep 1
      grep "\[${ACM_CODEC}\]" /sys/devices/platform/acm.0/codec > /dev/null
      if [ $? -eq 0 ]; then
         break
      else
         false
      fi
   done
   check_status
fi
```

❶ "y"から"n"に設定を変更しコンフィグ領域を保存すると、次回起動時に uvc-gadget が自動起動されないようになります

❷ "decoder"を指定しコンフィグ領域を保存すると、次回起動時にデコーダー用ファームウェアをロードするようになります。また、"encoder"か"decoder"以外の文字列を指定した場合は、ファームウェアをロードしません。

図 9.3 デフォルト状態の/etc/config/rc.local

AV コーデックミドルウェアのファームウェアロード処理は、romfs-a810-v1.04.img(Atmark Dist v20140131)で追加されました。ユーザーランドイメージ romfs-a810-v1.03.img 以前(Atmark Dist v20131018 以前)で Armadillo-840 を起動したことがある場合は、次のようにコンフィグ領域を初期化すると次回起動時からファームウェアロード処理が行われるようになります。

```
[armadillo ~]# flatfsd -w
```

9.2. プリインストールアプリケーション

デフォルトのユーザーランドにインストールされているアプリケーションを一覧します。

・ /bin

```
addgroup          expect            linux64           reformime
adduser           false             ln                rev
amixer            fdflush           login             rm
aplay             fgrep             lrz               rmdir
arecord           flatfsd           ls                rpm
ash               fsck              lsattr            run-parts
base64            fsck.ext2         lsz               scriptreplay
busybox           fsync             lzop              sed
cat               ftp               mail              setarch
catv              ftpd              makemime          setserial
chattr            getopt            mkdir             sh
chgrp             grep              mke2fs            sleep
chmod             gst-inspect-1.0   mknod             ssh
chown             gst-launch-1.0    mktemp            ssh-keygen
conspy            gst-typefind-1.0  more              stat
cp                gunzip            mount             stty
cpio              gzip              mountpoint        su
cttyhack          hostname          mpstat            sync
date              htpasswd          mt                tar
dd                hush              mv                tftp
delgroup          ionice            netflash          tip
deluser           iostat            netstat           touch
df                ip                nice              true
dmesg             ipaddr            ntpclient         tune2fs
dnsdomainname     ipcalc            pidof             umount
dumpkmap          iplink            ping              uname
e2fsck            iproute           ping6             usleep
echo              iprule            pipe_progress     vi
ed                iptables          powertop          watch
egrep             iptunnel          printenv          wget
ethtool           kill              ps                zcat
evtest            linux32           pwd
```

・ /usr/bin

```
[               envdir          lsusb           runsv           traceroute6
[[              envuidgid       lzcat           runsvdir        tty
add-shell       ether-wake      lzma            rx              ttysize
ar              expand          lzopcat         script          udevinfo
arping          expr            md5sum          seq             udpsvd
awk             fdformat        mesg            setkeycodes     unexpand
basename        fgconsole       microcom        setsid          uniq
beep            find            mjpg_streamer   setuidgid       unix2dos
bunzip2         flock           mkfifo          sha1sum         unlzma
bzcat           fold            mkpasswd        sha256sum       unlzop
```

```
bzip2           free            mksquashfs      sha512sum       unsquashfs
cal             ftpget          nc              showkey         unxz
camctrl         ftpput          nmeter          smemcap         unzip
chat            fuser           nohup           softlimit       uptime
chpst           groups          nslookup        sort            users
chrt            hd              od              spawn-fcgi      uudecode
chvt            head            openvt          split           uuencode
cksum           hexdump         passwd          strings         vi
clear           hostid          patch           sudo            vlock
cmp             id              pgrep           sudoedit        volname
comm            ifplugd         pkill           sum             wall
convert_pubkey  install         pmap            sv              wc
crontab         ipcrm           printf          tac             wget
cryptpw         ipcs            pscan           tail            which
cut             kbd_mode        pstree          tcpsvd          who
dc              killall         pwdx            tee             whoami
deallocvt       killall5        readahead       telnet          whois
diff            last            readlink        test            xargs
dirname         less            realpath        tftp            xz
dos2unix        logger          remove-shell    tftpd           xzcat
dpkg-deb        logname         renice          time            yes
du              lpq             reset           timeout
dumpleases      lpr             resize          top
eject           lsof            rpm2cpio        tr
env             lspci           rtcwake         traceroute
```

・/sbin

```
acpid           fsck.msdos      mkdosfs         route
adjtimex        fsck.vfat       mke2fs          runlevel
arp             getty           mkfs.ext2       setconsole
avahi-daemon    halt            mkfs.minix      slattach
blkid           hdparm          mkfs.msdos      sshd
blockdev        hwclock         mkfs.vfat       start-stop-daemon
bootchartd      ifconfig        mkswap          sulogin
chat            ifdown          modinfo         swapoff
depmod          ifenslave       modprobe        swapon
devmem          ifup            nameif          switch_root
dosfsck         init            nanddump        sysctl
dosfslabel      insmod          nandwrite       syslogd
fbsplash        iwconfig        nftl_format     tunctl
fdisk           iwlist          nftldump        tune2fs
findfs          iwpriv          pivot_root      udevcontrol
flash_erase     klogd           poweroff        udevd
flash_eraseall  loadkmap        pppd            udevsettle
flash_info      logread         pppdump         udevtrigger
flash_lock      losetup         pppoe-discovery udhcpc
flash_unlock    lsmod           pppstats        vconfig
freeramdisk     makedevs        raidautorun     watchdog
fsck            man             reboot          zcip
fsck.minix      mdev            rmmod
```

・/usr/sbin

```
brctl           i2cset          setfont
chpasswd        inetd           setlogcons
chroot          lighttpd        svlogd
```

```
crond                loadfont            telnetd
dhcprelay            lpd                 ubiattach
dnsd                 nanddump            ubidetach
fakeidentd           nandwrite           ubimkvol
fbset                nbd-client          ubirmvol
ftpd                 ntpd                ubirsvol
get-board-info-a810  popmaildir          ubiupdatevol
httpd                rdate               udevmonitor
i2cdetect            rdev                udhcpd
i2cdump              readprofile         uvc-gadget
i2cget               sendmail            visudo
```

10. ブートローダー仕様

この章では、ブートローダーの起動モードや利用することができる機能について説明します。

10.1. ブートローダーイメージの選択

電源投入時の"SDBOOT_EN"ピンの状態によりブートローダーイメージを選択することができます。"SDBOOT_EN"ピンの状態が Low であればフラッシュメモリの bootloader パーティションに書き込まれているブートローダーが起動し、"SDBOOT_EN"ピンの状態が High であれば SD カードの第 1 パーティションのブートローダーイメージ(/sdboot.bin)が起動します。Armadillo-810 のデフォルト状態では、"SDBOOT_EN"ピンは Low(GND に 10kΩ プルダウン)となっており、フラッシュメモリに書き込まれているブートローダーが起動します。

表 10.1 SDBOOT_EN ピンとブートローダーイメージの対応

SDBOOT_EN	ブートローダーイメージ
Low(0V)	フラッシュメモリの bootloader パーティション
High(3.3V)	SD カードの第 1 パーティションの/sdboot.bin

Armadillo-810 カメラモデルでは、"SDBOOT_EN"ピンは拡張ボード 01 (A コネクタ用)の JP3 に接続されており、ジャンパーのオープン/ショートによりブートローダーイメージを選択することができます。

表 10.2 拡張ボード 01 (A コネクタ用)の JP3 によるブートローダーイメージの選択

JP3	ブートローダーイメージ
オープン	フラッシュメモリの bootloader パーティション
ショート	SD カードの第 1 パーティションの/sdboot.bin

10.2. ブートローダー起動モード

ブートローダーが起動すると"HERMIT_EN_N"ピンの状態により 2 つのモードのどちらかに遷移します。

表 10.3 ブートローダー起動モード

起動モードの種別	HERMIT_EN_N	説明
OS 自動起動モード	High(3.3V)	電源投入後、自動的に Linux カーネルを起動させます。
保守モード	Low(0V)	各種設定が可能な Hermit-At コマンドプロンプトが起動します。

Armadillo-810 カメラモデルでは、"HERMIT_EN_N"ピンは拡張ボード 01 (A コネクタ用)と開発用 USB シリアル変換アダプタに接続されています。拡張ボード 01 (A コネクタ用)では、JP1 に接続されており、JP1 をオープンすると"HERMIT_EN_N"ピンの状態は High、JP1 をショートとすると"HERMIT_EN_N"ピンの状態は Low となります。開発用 USB シリアル変換アダプタではスライドスイッチに接続されており、基板内側にスライドさせると"HERMIT_EN_N"ピンの状態は High、外側にスライドさせると"HERMIT_EN_N"ピンの状態は Low となります。

拡張ボード 01 (A コネクタ用)の JP1 と開発用 USB シリアル変換アダプタのスライドスイッチの組み合わせにより、どの起動モードとなるかを「表 10.4. ブートローダー起動モードスイッチ」に示します。

表 10.4 ブートローダー起動モードスイッチ

拡張ボード 01 (A コネクタ用) JP1	開発用 USB シリアル変換アダプタ スライドスイッチ	起動モード
オープン	内側	OS 自動起動モード
オープン	外側	保守モード
ショート	内側	保守モード
ショート	外側	保守モード

10.3. ブートローダーの機能

Hermit-At の保守モードでは、Linux カーネルの起動オプションの設定やフラッシュメモリの書き換えなどを行うことできます。

保守モードで利用できるコマンドは、「表 10.5. 保守モードコマンド一覧」に示します。

表 10.5 保守モードコマンド一覧

コマンド	説明
erase program download	フラッシュメモリを書き換える場合に使用します
memmap	フラッシュメモリのメモリマップを表示します
setbootdevice setenv clearenv	OS の起動設定をする場合に使用します
boot	OS を起動する場合に使用します
frob	簡易的にメモリアクセスする場合に使用します
md5sum	メモリ空間の MD5 サム値を表示する場合に使用します
info	ハードウェアの情報を表示します
version	ブートローダーのバージョンを表示します

各コマンドのヘルプを表示するには「図 10.1. hermit コマンドのヘルプを表示」のようにします。

```
hermit> help [コマンド]
```

図 10.1 hermit コマンドのヘルプを表示

10.3.1. コンソールの指定方法

ブートローダーおよび Linux カーネルのコンソールを指定するには、後述する Linux カーネル起動オプションを設定する場合の setenv コマンドで行います。Linux カーネル起動オプションの console パラメータは、ブートローダーのコンソールにも影響する仕組みとなっています。

コンソール指定子とそれに対応するログ表示先/保守モードプロンプト出力先を「表 10.6. コンソール指定子とログ出力先」に示します。

表 10.6 コンソール指定子とログ出力先

コンソール指定子	OS 自動起動モード時のログ出力先	保守モードプロンプト出力先[a]
ttySC2	CON3	CON3
ttySC8	CON2	CON2
none	なし	CON3
その他(tty1 等)	指定するコンソール[b]	CON3

[a]ブートローダーの再起動後に反映されます

[b]ブートローダーのログは出力されません

10.3.2. Linux カーネルイメージの指定方法

ブートローダーが OS を起動させる場合、フラッシュメモリに書き込まれた Linux カーネルイメージか、SD カード内に保存されているイメージファイルを指定することができます。

Linux カーネルイメージを指定するには、"setbootdevice"コマンドを使用します。「表 10.7. Linux カーネルイメージ指定子」に示す指定子を設定することができます。

表 10.7 Linux カーネルイメージ指定子

指定子	Linux カーネルイメージの配置場所
flash	フラッシュメモリの kernel パーティションに書き込まれたイメージ
mmcblk0p1	SD カードのパーティション 1 に保存されている/boot/linux.bin.gz ファイル "p1"はパーティションを示しており、"p2"とするとパーティション 2 のファイルを指定可能

10.3.3. Linux カーネル起動オプションの指定方法

Linux カーネルには様々な起動オプションがあります。詳しくは、Linux の解説書や、Linux カーネルのソースコードに含まれているドキュメント(Documentation/kernel-parameters.txt)を参照してください。

ここでは Armadillo-810 で使用することができる、代表的な起動オプションを「表 10.8. Linux カーネルの起動オプションの一例」に紹介します。

表 10.8 Linux カーネルの起動オプションの一例

オプション指定子	説明
console=	起動ログなどが出力されるイニシャルコンソールを指定します。 次の例では、コンソールに ttySC2 を、ボーレートに 115200 を指定しています。 `console=ttySC2,115200`
root=	ルートファイルシステムが構築されているデバイスを指定します。 デバイスには Linux カーネルが認識した場合のデバイスを指定します。 次の例では、デバイスに SD カードの第 2 パーティションを指定しています。 `root=/dev/mmcblk0p2`
rootwait	"root="で指定したデバイスが利用可能になるまでルートファイルシステムのマウントを遅らせます。
noinitrd	initrd を利用しないことを明示します。
mem	Linux カーネルが利用可能なメモリの量を指定します。 RAM の一部を専用メモリとして利用したい場合などに設定します。

11. ビルド手順

本章では、ソースコードから工場出荷イメージと同じイメージを作成する手順について説明します。

使用するソースコードは、開発セット付属の DVD に収録されています。最新版のソースコードは、Armadillo サイトからダウンロードすることができます。新機能の追加や不具合の修正などが行われているため、DVD に収録されているものよりも新しいバージョンがリリースされているかを確認して、最新バージョンのソースコードを利用することを推奨します。

Armadillo サイト - Armadillo-810 ドキュメント・ダウンロード

https://armadillo.atmark-techno.com/armadillo-810/downloads

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

11.1. Linux カーネル/ユーザーランドをビルドする

ここでは、ソースコードディストリビューションである「Atmark Dist」と、「Linux カーネル」のソースコードからイメージファイルを作成する手順を説明します。

手順 11.1 Linux カーネル/ユーザーランドをビルド

1. ソースコードを準備します。Atmark Dist と Linux カーネルのソースコードアーカイブを準備し展開します。展開後、Atmark Dist に Linux カーネルのソースコードを登録するために、シンボリックリンクを作成します。

```
[ATDE ~]$ ls
atmark-dist.tar.gz  linux-3.4-at.tar.gz
[ATDE ~]$ tar zxf atmark-dist.tar.gz
[ATDE ~]$ tar zxf linux-3.4-at.tar.gz
[ATDE ~]$ ls
atmark-dist  atmark-dist.tar.gz  linux-3.4-at  linux-3.4-at.tar.gz
[ATDE ~]$ ln -s ../linux-3.4-at atmark-dist/linux-3.x
```

2. Atmark Dist ディレクトリに入り、コンフィグレーションを行います。ここでは、menuconfig を利用します。

```
[ATDE ~]$ cd atmark-dist
[ATDE ~/atmark-dist]$ make menuconfig
```

```
 atmark-dist v1.31.0 Configuration
--------------------------------------------------------------------------------
                                Main Menu
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ------------------------------------------------------------------------
               Vendor/Product Selection  --->
               Kernel/Library/Defaults Selection  --->
               ---
               Load an Alternate Configuration File
               Save Configuration to an Alternate File


  ------------------------------------------------------------------------

                   <Select>    < Exit >    < Help >
```

3. メニュー項目は、上下キーで移動することができます。下部の Select/Exit/Help は左右キーで移動することができます。選択するには Enter キーを押下します。 "Vendor/Product Selection --->"に移動して Enter キーを押下します。 Vendor には "AtmarkTechno" を選択し、AtmarkTechno Products には "Armadillo-810" を選択します。

```
 atmark-dist v1.31.0 Configuration
--------------------------------------------------------------------------------
                        Vendor/Product Selection
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ------------------------------------------------------------------------
               --- Select the Vendor you wish to target
               (AtmarkTechno) Vendor
               --- Select the Product you wish to target
               (Armadillo-810) AtmarkTechno Products


  ------------------------------------------------------------------------

                   <Select>    < Exit >    < Help >
```

4. 前のメニューに戻るには、"Exit"に移動して Enter キーを押下します。 続いて、"Kernel/Library/Defaults Selection --->"に移動して Enter キーを押下します。"Default all settings (lose changes)"に移動して"Y"キーを押下します。押下すると"[*]"のように選択状態となります。

```
 atmark-dist v1.31.0 Configuration
------------------------------------------------------------------------------
                    Kernel/Library/Defaults Selection
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ---------------------------------------------------------------------
             --- Kernel is linux-3.x
             (default) Cross-dev
             (None) Libc Version
             [*] Default all settings (lose changes) (NEW)
             [ ] Customize Kernel Settings (NEW)
             [ ] Customize Vendor/User Settings (NEW)
             [ ] Update Default Vendor Settings (NEW)

  ---------------------------------------------------------------------

                    <Select>    < Exit >    < Help >
```

5. 前のメニューに戻るため、"Exit"に移動して Enter キーを押下します。コンフィグレーションを抜けるためにもう一度"Exit"に移動して Enter キーを押下します。

6. コンフィグレーションを確定させるために"Yes"に移動して Enter キーを押下します。

```
 atmark-dist v1.31.0 Configuration
------------------------------------------------------------------------------


        ---------------------------------------------------------
            Do you wish to save your new kernel configuration?

                           < Yes >      <  No  >

        ---------------------------------------------------------
```

7. コンフィグレーションが完了するので、続いてビルドを行います。 ビルドは"make"コマンドを実行します。

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

ビルドログが表示されます。ビルドする PC のスペックにもよりますが、数分から十数分程度かかります。

8. ビルドが終了すると、atmark-dist/images/ディレクトリ以下にイメージファイルが作成されています。Armadillo-810 では圧縮済みのイメージ(拡張子が".gz"のもの)を利用します。

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

11.1.1. ツールチェーンを変更するには

Armadillo-810 では、ARM の 2 つのアーキテクチャに対応しています。 "armhf" (デフォルト) では、浮動小数点演算に VFP コプロセッサを利用します。 "armel"では、浮動小数点演算に専用のソフトウェアライブラリを利用します。 基本的には"armhf"の方が性能が高く、特に"armel"でなければならない場合以外は"armhf"を利用してください。

ATDE には、上記 2 つのアーキテクチャ用のツールチェーン(コンパイラやリンカ、クロスライブラリなど)を用意してあります。

Linux カーネル及びユーザーランドのアーキテクチャを変更するには、Atmark Dist のコンフィグレーション時に、"Cross-dev"に利用したいアーキテクチャを選択します。次の例では、"armel"を指定している状態となります。"default"となっている場合は、Armadillo-810 の場合では"armhf"が選択されます。

```
 atmark-dist v1.31.0 Configuration
------------------------------------------------------------------------------
                    Kernel/Library/Defaults Selection
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ---------------------------------------------------------------------
            --- Kernel is linux-3.x
            (armel) Cross-dev
            (None) Libc Version
            [*] Default all settings (lose changes) (NEW)
            [ ] Customize Kernel Settings (NEW)
            [ ] Customize Vendor/User Settings (NEW)
            [ ] Customize Vendor/User Settings (NEW)

  ---------------------------------------------------------------------

                    <Select>    < Exit >    < Help >
```

11.2. ブートローダーをビルドする

ここでは、ブートローダーである「Hermit-At」のソースコードからイメージファイルを作成する手順を説明します。

手順 11.2 ブートローダーをビルド

1. Hermit-At のソースコードアーカイブを準備し展開します。展開後、hermit-at ディレクトリに移動します。

```
[ATDE ~]$ ls
hermit-at.tar.gz
[ATDE ~]$ tar zxf hermit-at.tar.gz
[ATDE ~]$ ls
hermit-at  hermit-at.tar.gz
[ATDE ~]$ cd hermit-at
```

2. Armadillo-810 用にコンフィグレーションを行います。ここでは例としてフラッシュメモリ起動用イメージを作成します。デフォルトコンフィグには"armadillo810_nor_defconfig"

を指定します。SD カード起動用イメージを作成する場合は、"armadillo810_mmcsd_defconfig"を指定してください。

```
[ATDE ~/hermit-at]$ make armadillo810_nor_defconfig
```

3. ビルドには"make"コマンドを利用します。

```
[ATDE ~/hermit-at]$ make
```

4. ビルドが終了すると、hermit-at/src/target/armadillo8x0/ディレクトリ以下にイメージファイルが作成されています。

```
[ATDE ~/hermit-at]$ ls src/target/armadillo8x0/loader-armadillo810*.bin
src/target/armadillo8x0/loader-armadillo810-nor-v3.2.0.bin
```

11.2.1. ツールチェーンを変更するには

Linux カーネルとユーザーランドのアーキテクチャを変更するのと同様に、ブートローダーもアーキテクチャを変更することができます。ただし、特に動作に影響を与えないため、変更する必要はありません。

ブートローダーのビルド時にアーキテクチャを変更するには、CROSS_COMPILE オプションを利用します。"armel"を指定する場合は、ビルド時に "CROSS_COMPILE=arm-linux-gnueabi-"をつけてビルドしてください。

```
[ATDE ~/hermit-at]$ make CROSS_COMPILE=arm-linux-gnueabi-
```

12. フラッシュメモリの書き換え方法

本章では、Armadillo-810 のフラッシュメモリに書き込まれているイメージファイルを更新する手順について説明します。

フラッシュメモリの書き換え方法には、大きく分けて以下の 2 種類の方法があります。

表 12.1 フラッシュメモリの書き換え方法

方法	特徴
netflash を使用する	・イメージファイルをネットワークまたはストレージで転送するため書き換えが高速 ・Armadillo で Linux にログインできる必要がある
ダウンローダー使用する	・イメージファイルをシリアルで転送するため書き換えが低速 ・Armadillo でブートローダーが起動できればよい

フラッシュメモリを書き換えるためには、Linux またはブートローダーが起動している必要があります。フラッシュメモリに書き込まれているブートローダーが起動しない状態になってしまった場合は、「16. SD ブートの活用」を参照して SD カードからソフトウェアを起動させてください。

ダウンローダーを使用してユーザーランドイメージなどサイズの大きなイメージファイルを書き換えると非常に時間がかかります。これは、イメージファイルを Armadillo に転送する際にシリアルの転送速度がボトルネックとなるためです。サイズの大きなイメージファイルを書き換える場合は netflash を使用する方法を推奨します。

12.1. フラッシュメモリのパーティションについて

フラッシュメモリの書き換えは、パーティション毎に行います。パーティションは"リージョン"とも呼ばれます。

各パーティションのサイズはフラッシュメモリ内には保存されていません。ブートローダーと Linux カーネルそれぞれが同じパーティションテーブルを保持することにより、一意的に扱うことができるようになっています。

各パーティションは、書き込みを制限することが可能です。書き込みを制限する理由は、誤動作や予期せぬトラブルにより、フラッシュメモリ上のデータが不意に破壊または消去されることを防ぐためです。

読み込みは、常時可能です。読み込みに制限を付けることはできません。

各パーティションのデフォルト状態での書き込み制限の有無と、対応するイメージファイル名を「表 12.2. パーティションのデフォルト状態での書き込み制限の有無と対応するイメージファイル名」に示します。

表 12.2 パーティションのデフォルト状態での書き込み制限の有無と対応するイメージファイル名

パーティション	書き込み制限	イメージファイル名	備考
bootloader	あり	loader-armadillo810-nor-*[version]*.bin	ブートローダーイメージを配置するパーティションです。
config	なし	なし	ユーザーランドアプリケーション"flatfsd"がFlat file-system(フラッシュメモリ向けファイルシステム)を構築するパーティションです。使用方法については「7. コンフィグ領域 – 設定ファイルの保存領域」を参照してください。
license	あり	なし	有償ミドルウェアなどのライセンスファイルを配置するパーティションです。
firmware	あり	squashfs-a810-firmware-*[version]*.img	有償ミドルウェアなどのファームウェアを配置するパーティションです。
kernel	なし	linux-a810-*[version]*.bin.gz	Linux カーネルイメージを配置するパーティションです。
userland	なし	romfs-a810-*[version]*.img.gz	ユーザーランドイメージを配置するパーティションです。

Linux v3.4-at3 から、書き込みの制限を外す仕組みを導入しています。また、これに合わせ、hermit-at v3.2.2 および Linux v3.4-at3 から、デフォルト状態で、"license"と"firmware"の両パーティションへの書き込みを制限することにしました。

license パーティションにはボードごとに固有なイメージが出荷時に書き込まれています。本パーティションを書き換えてライセンスファイルが消えてしまった場合、AV コーデックミドルウェアを使用できなくなります。特別な理由がない限り、license パーティションは書き換えないでください。万一、ライセンスファイルが消えてしまった場合、弊社営業部へご相談ください。

工場出荷状態でフラッシュメモリに書き込まれているイメージファイルは、最新版ではない可能性があります。最新版のブートローダー、Linux カーネルおよびユーザーランドイメージファイルは Armadillo サイトから、ファームウェアイメージファイルはユーザーズサイトからダウンロード可能です。最新版のイメージファイルに書き換えてからのご使用を推奨します。

ダウンローダーでは、書き込みが制限されているパーティションを"ロック(locked)されている"と呼びます。このパーティションを強制的に書き換える場合は、"--force-locked"というオプションを付けます。他のオプションについては、「12.3. ダウンローダーを使用してフラッシュメモリを書き換える」を参照してください。

Linux が動いている場合、パーティションの書き込み制限をコマンドで外すことが可能です。Sysfs の MTD クラスディレクトリ以下にある"ro"というファイルに 0 を書き込むことで制限を外すことが可能です。逆に 1 を書き込めば、パーティションへの書き込みを制限する事が可能です。

MTD クラスディレクトリとパーティションの対応については、「表 12.3. パーティションと MTD クラスディレクトリの対応」を参照してください。

表 12.3 パーティションと MTD クラスディレクトリの対応

パーティション	MTD クラスディレクトリ
bootloader	/sys/class/mtd/mtd0
config	/sys/class/mtd/mtd1
license	/sys/class/mtd/mtd2
firmware	/sys/class/mtd/mtd3
kernel	/sys/class/mtd/mtd4
userland	/sys/class/mtd/mtd5

以降の説明では、任意のパーティションを示す MTD クラスディレクトリを"/sys/class/mtd/[MTD]"のように表記します。

書き込み制限を外すには、ro ファイルに 0 を書き込みます。

```
[armadillo ~]# echo 0 > /sys/class/mtd/[MTD]/ro
```

図 12.1 書き込み制限を外す

書き込みを制限するには、ro ファイルに 1 を書き込みます。

```
[armadillo ~]# echo 1 > /sys/class/mtd/[MTD]/ro
```

図 12.2 書き込みを制限する

12.2. netflash を使用してフラッシュメモリを書き換える

Linux が動作している状態では、Linux アプリケーションの netflash を利用することでフラッシュメモリを書き換えることができます。ここでは、netflash を利用して次に示す場所に存在するイメージファイルをフラッシュメモリに書き込む手順を紹介します。

- Web サーバー上のイメージファイル
- ストレージ上のイメージファイル

netflash コマンドのヘルプは次のとおりです。

```
[armadillo ~]# netflash -h
usage: netflash [-bCfFhijklntuv?] [-c console-device] [-d delay] [-o offset] [-r flash-device]
                [net-server] file-name

    -b      don't reboot hardware when done
    -C      check that image was written correctly
    -f      use FTP as load protocol
    -F      force overwrite (do not preserve special regions)
    -h      print help
    -i      ignore any version information
    -H      ignore hardware type information
    -j      image is a JFFS2 filesystem
    -k      don't kill other processes (or delays kill until
        after downloading when root filesystem is inside flash)
    -K      only kill unnecessary processes (or delays kill until
        after downloading when root filesystem is inside flash)
    -l      lock flash segments when done
    -n      file with no checksum at end (implies no version information)
    -p      preserve portions of flash segments not actually written.
    -s      stop erasing/programming at end of input data
    -t      check the image and then throw it away
    -u      unlock flash segments before programming
    -v      display version number
```

図 12.3 netflash コマンドのヘルプ

"-r"オプションに指定するフラッシュメモリのデバイスファイルとパーティションの対応を次に示します。

表 12.4 フラッシュメモリのパーティションとデバイスファイル

パーティション	デバイスファイル
bootloader[a]	/dev/flash/bootloader
config	/dev/flash/config
license[a]	/dev/flash/license
firmware[a]	/dev/flash/firmware
kernel	/dev/flash/kernel
userland	/dev/flash/userland

[a]書き込みが制限されています。詳細については、「12.1. フラッシュメモリのパーティションについて」を参照してください。

12.2.1. Web サーバー上のイメージファイルを書き込む

ATDE では、標準で Web サーバー(lighttpd)が動作しており、/var/www/ディレクトリ以下に置かれたファイルはネットワーク経由でダウンロードすることができます。netflash は、HTTP によるファイルのダウンロードをサポートしています。

イーサネットガジェット[1]を利用すると、ATDE とネットワーク通信することができます。ここでは、ATDE とネットワーク通信ができることを前提に、ATDE からイメージファイルをダウンロードして kernel パーティションに書き込む手順を説明します。

手順 12.1 Web サーバー上のイメージファイルを書き込む

1. ATDE の/var/www/ディレクトリに Linux カーネルイメージファイルを置きます。

[1]イーサネットガジェットの詳細については、「6.1.3. イーサネットガジェット」を参照してください。

```
[ATDE ~]$ ls
linux-a810-v1.00.bin.gz
[ATDE ~]$ cp linux-a810-v1.00.bin.gz /var/www/
```

2. Web サーバー上のイメージファイルの URL(http://atde5.local/linux-a810-v1.00.bin.gz)を指定して netflash コマンドを実行します。

```
[armadillo ~]# netflash -b -k -n -u -s -r /dev/flash/kernel http://atde5.local/linux-
a810-v1.00.bin.gz
.....................................................................
.....................................................................
(省略)
.....................................
netflash: got "http://atde5.local/linux-a810-v1.00.bin.gz", length=2564696
netflash: programming FLASH device /dev/flash/kernel
.................
```

3. Armadillo のプロンプトが表示されるとフラッシュメモリの書き換えは完了です。次回起動時から書き換えた Linux カーネルイメージで起動します。

```
[armadillo ~]#
```

12.2.2. ストレージ上のイメージファイルを書き込む

ストレージ(SD カードや USB メモリ)をマウントすることで、ストレージに保存されたイメージファイルをフラッシュメモリに書き込むことができます。

ここでは SD カードに保存されているイメージファイルを userland パーティションに書き込む手順を説明します。

手順 12.2 SD カード上のイメージファイルを書き込む

1. SD カードを/mnt/ディレクトリにリードオンリーでマウントします。

```
[armadillo ~]# mount -o ro /dev/mmcblk0p1 /mnt
kjournald starting.  Commit interval 5 seconds
EXT3-fs (mmcblk0p1): using internal journal
EXT3-fs (mmcblk0p1): mounted filesystem with ordered data mode
[armadillo ~]# ls /mnt
romfs-a810-v1.00.img.gz
```

2. SD カード上のイメージファイルのパス(/mnt/romfs-a810-v1.00.img.gz)を指定して netflash コマンドを実行します。

```
[armadillo ~]# netflash -b -k -n -u -s -r /dev/flash/userland /mnt/romfs-a810-
v1.00.img.gz
.....................................................................
.....................................................................
(省略)
```

```
.....................................................
netflash: got "/mnt/romfs-a810-v1.00.img.gz", length=10316650
netflash: programming FLASH device /dev/flash/userland
.....................................................................
```

3. Armadillo のプロンプトが表示されるとフラッシュメモリの書き換えは完了です。次回起動時から書き換えたユーザーランドイメージで起動します。

```
[armadillo ~]#
```

4. SD カードをアンマウントします。

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

12.3. ダウンローダーを使用してフラッシュメモリを書き換える

Linux を起動できない場合やブートローダーを更新する場合は、ダウンローダー(hermit)を使用してフラッシュメモリを書き換える必要があります。hermit は ATDE に標準でインストールされています。

hermit は Armadillo のブートローダーと協調動作を行いフラッシュメモリを書き換えることができます。hermit とブートローダー間の通信には、シリアル[2]が使用されます。

hermit のヘルプは次のとおりです。

[2]通信速度(ボーレート)は、115200bps です

```
[ATDE ~]# hermit
Usage: hermit [options] command [command options]
Available commands: download, erase, help, go, map, terminal, upload, md5sum
Armadillo-J command: firmupdate
Multiple commands may be given.
General options (defaults) [environment]:
    -e, --ethernet
    -i, --input-file <path>
    --netif <ifname> (eth0) [HERMIT_NETIF]
    --memory-map <path>
    --port <dev> (/dev/ttyS0) [HERMIT_PORT]
    -o, --output-file <path>
    --remote-mac <MAC address>
    -v, --verbose
    -V, --version
Download/Erase options:
    -a, --address <addr>
    -b, --baudrate <baudrate>
    --force-locked
    -r, --region <region name>
Memory map options:
    --anonymous-regions
Md5sum  options:
    -a, --address <addr>
    -r, --region <region name>
    -s, --size <size>
```

図 12.4 hermit コマンドのヘルプ

ここでは、bootloader パーティションを書き換える手順について説明します。

手順 12.3 ダウンローダーを使用して書き換える

1. ブートローダーが保守モードで起動するように設定します。設定方法については、「10.2. ブートローダー起動モード」を参照してください。

2. Armadillo が保守モードで起動したことを確認するために、ATDE で minicom を起動しておきます。

```
[ATDE ~]$ LANG=C minicom --noinit --wrap --device /dev/ttyUSB0
```

3. Armadillo に電源を投入します。ブートローダーが保守モードで起動すると、次のように保守モードのプロンプトが表示されます。

```
hermit>
```

4. minicom を終了させシリアルポート(/dev/ttyUSB0)を開放します。

5. bootloader パーティションと書き込むイメージファイル(loader-armadillo810-nor-v3.2.0.bin)を指定して hermit コマンドを実行します。bootloader パーティションを更新する場合は、必ず"--force-locked"オプションを指定する必要があります。

```
[ATDE ~]$ hermit download --input-file loader-armadillo810-nor-v3.2.0.bin --region
bootloader --force-locked --port /dev/ttyUSB0
serial: completed 0x0000a92c (43308) bytes.
```

書き込みが制限されているパーティションを書き換える場合、"--force-locked"オプションを指定する必要があります。

6. ATDE のプロンプトが表示されるとフラッシュメモリの書き換えは完了です。次回起動時から書き換えたブートローダーイメージで起動します。

```
[ATDE ~]$
```

12.4. ブートローダーが起動しなくなった場合の復旧作業

フラッシュメモリの bootloader パーティションを誤ったイメージファイルで書き換えたり、書き換え中に Armadillo の電源を切断してしまった場合、ブートローダーが起動しなくなる場合があります。フラッシュメモリのブートローダーが起動しなくなった場合は、SD ブートを利用して復旧する必要があります。

ブートローダーの復旧手順を次に示します。

手順 12.4 ブートローダーの復旧

1. SD ブートを行うためのブートディスクを作成します。ブートディスクの作成方法については「16. SD ブートの活用」を参照してください。

2. Armadillo にブートディスクを接続し、ブートローダーが SD カードから起動し、且つ保守モードとなるように設定します。拡張ボード 01 (A コネクタ用)の JP1 および JP3 をショートに設定してください。

3. Armadillo が保守モードで起動したことを確認するために、ATDE で minicom を起動しておきます。

```
[ATDE ~]$ LANG=C minicom --noinit --wrap --device /dev/ttyUSB0
```

4. Armadillo に電源を投入します。ブートローダーが保守モードで起動すると、次のように保守モードのプロンプトが表示されます。

```
hermit>
```

5. minicom を終了させシリアルポート(/dev/ttyUSB0)を開放します。

6. bootloader パーティションと書き込むイメージファイル(loader-armadillo810-nor-[version].bin)を指定して hermit コマンドを実行します。bootloader パーティションを更新する場合は、必ず"--force-locked"オプションを指定する必要があります。

```
[ATDE ~]$ hermit download --input-file loader-armadillo810-nor-[version].bin --region ⏎
bootloader --force-locked --port /dev/ttyUSB0
serial: completed 0x0000a92c (43308) bytes.
```

7. ATDE のプロンプトが表示されるとフラッシュメモリの書き換えは完了です。次回起動時から書き換えたブートローダーイメージで起動します。

```
[ATDE ~]$
```

13. 開発の基本的な流れ

本章では、Armadillo を用いたシステム開発の一連の流れについて説明します。

1. ユーザーオリジナルアプリケーションを作成する
2. Atmark Dist にユーザーオリジナルアプリケーションを組み込む
3. システムの最適化を行う
4. オリジナルプロダクトのコンフィグレーションを更新する

以降では、上記ステップについて順を追って説明します。

13.1. ユーザーオリジナルアプリケーションを作成する

ここでは、システムのメイン機能となるアプリケーションプログラムを作成する方法を説明します。ほとんどのシステムでは、ユーザーオリジナルなアプリケーションを実装するものと思います。本章では定番である「Hello world!」を例に、C 言語でアプリケーションプログラムのソースコードを作成し、コンパイル、動作確認する方法について説明します。

まずは、ATDE 上で動作する「Hello World!」を作成してみましょう。テキストエディタ[1]には gedit を利用します。

```
[ATDE ~]$ mkdir hello
[ATDE ~]$ cd hello
[ATDE ~/hello]$ gedit main.c &
```

図 13.1 ディレクトリを作成後、テキストエディタ(gedit)を起動

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

int main(void)
{
    printf("Hello World!\n");

    return EXIT_SUCCESS;
}
```

図 13.2 「Hello World!」のソース例(main.c)

作成したソースコードが意図したとおりに動作するか、ATDE 上で動作するようにコンパイルして実行し、動作の確認をしましょう。

[1]ATDE には、gedit、emacs や vi などのテキストエディタがあらかじめインストールされています。

```
[ATDE ~/hello]$ gcc main.c -o hello ❶
[ATDE ~/hello]$ ls
hello  main.c
[ATDE ~/hello]$ ./hello ❷
Hello World!
```

❶ ATDE 上で動作するようにコンパイルするには「gcc」コマンドを使用します。

❷ コンパイルされた実行ファイル(hello)を実行

図 13.3 ATDE 上で動作するように main.c をコンパイルし実行

意図したとおりに実行できましたね。では次に Armadillo が実行できるようにコンパイルを行います。 Armadillo のアプリケーションを作成するには、クロスコンパイルが基本的な手法となります。先に示している、ブートローダー、Linux カーネル、ユーザランドイメージもクロスコンパイルされています。

クロスコンパイルとは、別のアーキテクチャで動作する実行ファイルを作成することです。ATDE など、通常の PC は、i386 または amd64 と言われるアーキテクチャとなっています。Armadillo-810 では armhf というアーキテクチャが使われています。Armadillo-810 で実行することができる実行ファイルを ATDE 上で作成する方法を説明します。

Armadillo-810 上で動作するようにコンパイルする場合は、コンパイラ(gcc)を armhf アーキテクチャ用のもの(arm-linux-gnueabihf-gcc)を利用します。

```
[ATDE ~/hello]$ arm-linux-gnueabihf-gcc main.c -o hello
[ATDE ~/hello]$ ls
hello  main.c
```

図 13.4 Armadillo-810 上で動作するように main.c をクロスコンパイル

Armadillo-810 に実行ファイルを転送して動作の確認を行います。ここではファイル転送に USB ガジェット経由の FTP を利用します。ATDE と Armadillo-810 が USB ケーブルで接続され、且つ Armadillo-810 の起動ができていると、「図 13.5. Armadillo に FTP で hello を転送」のようにファイルを転送することができます。

```
[ATDE ~/hello]$ ftp armadillo810-0.local
Connected to armadillo810-0.local.
220 localhost FTP server (GNU inetutils 1.4.1) ready.
Name (armadillo810-0.local:atmark): ftp
331 Guest login ok, type your name as password.
Password:
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd pub
250 CWD command successful.
ftp> put hello
local: hello remote: hello
200 PORT command sucessful.
150 Opening BINARY mode data connection for 'hello'.
226 Transfer complete.
5087 bytes sent in 0.00 secs (112903.9 kB/s)
ftp> quit
221 Goodbye.
```

図 13.5 Armadillo に FTP で hello を転送

minicom などを利用して Armadillo にログインすると/home/ftp/pub に hello が転送されています。転送されたばかりのファイルには実行権限がついていないため、chmod コマンドで実行権限を付与して実行してみましょう。

```
[armadillo ~]# cd /home/ftp/pub/
[armadillo ~/home/ftp/pub]# ls
hello
[armadillo ~/home/ftp/pub]# chmod +x hello
[armadillo ~/home/ftp/pub]# ./hello
Hello World!
```

図 13.6 Armadillo 上で hello を実行

13.2. Atmark Dist にユーザーオリジナルアプリケーションを組み込む

「13.1. ユーザーオリジナルアプリケーションを作成する」では、Armadillo 上で動作することができる実行ファイルを作成することができました。続いて、Atmark Dist にそのアプリケーションを組み込み、ユーザーランドのイメージファイル(romfs.img.gz)に自動的にインストールされるように作業を行います。

はじめに hello アプリケーションをビルドするための Makefile を作成します。この Makefile は、Atmark Dist のビルドシステムに hello を組み込むために必要となります。テキストエディタで作成します。

```
TARGET = hello

CROSS_COMPILE ?= arm-linux-gnueabihf-
CC = $(CROSS_COMPILE)gcc
CFLAGS = -Wall -Wextra -O3

all: $(TARGET)

hello: main.o
    $(CC) $(LDFLAGS) $^ $(LDLIBS) -o $@

%.o: %.c
    $(CC) $(CFLAGS) -c -o $@ $<

clean:
    $(RM) *~ *.o hello
```

図 13.7 hello 用の Makefile

Makefile が正しく作成できたかを確認するために、一度ビルドしてみましょう。ビルドには make コマンドを利用します。

```
[ATDE ~/hello]$ make
arm-linux-gnueabihf-gcc -Wall -Wextra -O3 -c -o main.o main.c
arm-linux-gnueabihf-gcc  main.o  -o hello
[ATDE ~/hello]$ ls
Makefile  hello  main.c  main.o
```

図 13.8 hello を make

makefile の記述ルールは次のようになります。

```
ターゲット: 依存ファイル 1 依存ファイル 2
        コマンド 1
        コマンド 2
```

make コマンドに続けて入力することによりターゲットを指定することができます。ターゲットを指定しない場合は、makefile のルールで最初に記述されているターゲットが実行されます。

「図 13.7. hello 用の Makefile」では、ターゲット指定をしない場合は、"all"ターゲットが実行されます。clean ターゲットを指定し make すると、一時ファイルなどが消去されます。

```
[ATDE ~/hello]$ make clean
rm -f *~ *.o hello
```

図 13.9 clean ターゲット指定した例

Atmark Dist では、製品(システム)固有の設定やファイルなどを製品毎にディレクトリに分けて管理されています。このディレクトリをプロダクトディレクトリといいます。アットマークテクノ製品の場合、開発セット用の標準イメージに対応するプロダクトディレクトリが製品毎に用意されています。

ここでは、Armadillo-810 のプロダクトディレクトリをコピーしてオリジナルプロダクトを作成し、そのオリジナルプロダクトに hello を組み込みます。オリジナルプロダクトの名前は、"my-product"とします。

```
[ATDE ~/atmark-dist]$ cp -a vendors/AtmarkTechno/Armadillo-810/ vendors/AtmarkTechno/my-product
[ATDE ~/atmark-dist]$ cp -a ../hello/ vendors/AtmarkTechno/my-product/
```

図 13.10 オリジナルプロダクトを作成し hello ディレクトリをコピー

続いて、hello を Atmark Dist のビルドシステムに組み込みます。プロダクトディレクトリ(atmark-dist/vendors/AtmarkTechno/my-product/)にある Makefile をテキストエディタで開き、次のように 27 行目を追加します。

```
22 comma := ,
23 empty :=
24 space := $(empty) $(empty)
25
26 SUBDIR_y =
27 SUBDIR_y += hello/
28 SUBDIR_$(CONFIG_VENDOR_AWL12_AERIAL)         += awl12/
29 SUBDIR_$(CONFIG_VENDOR_AWL13_AWL13)          += awl13/
30
31 all:
32      for i in $(SUBDIR_y) ; do $(MAKE) -C $$i || exit $? ; done
```

図 13.11 オリジナルプロダクト(my-product)に hello を登録

先ほど作成した Makefile では、Atmark Dist のインストール(romfs)ターゲットに対応していないため、ビルドされた実行ファイルは作成されますが、ユーザーランドイメージに実行ファイルがインストールされることはありません。ユーザーランドイメージに自動的にインストールされるように、romfs ターゲットを追加しましょう。ここでは、Armadillo 上の/usr/bin/ディレクトリ以下に hello がインストールされるように記述してみます。(18-19 行目を追加)

```
12 %.o: %.c
13      $(CC) $(CFLAGS) -c -o $@ $<
14
15 clean:
16      $(RM) *~ *.o hello
17
18 romfs: hello
19      $(ROMFSINST) /usr/bin/hello
```

図 13.12 romfs ターゲットの追加

これで、my-product に hello が追加されました。my-product をビルドして、イメージファイルを書き換えてみましょう。「11.1. Linux カーネル/ユーザーランドをビルドする」の手順の中で、AtmarkTechno Products に"Armadillo-810"を選択している箇所では"my-product"を選択します。ビ

ルドして出来上がったユーザーランド(romfs.img.gz)をフラッシュメモリに書き込むには、「12. フラッシュメモリの書き換え方法」を参照してください。

フラッシュメモリを書き換えた後 Armadillo を再起動すると、/usr/bin/hello が組み込まれたユーザーランドとなっています。

```
[armadillo ~]# ls /usr/bin/hello
/usr/bin/hello
[armadillo ~]# hello
Hello World!
```

図 13.13 hello が組み込まれたユーザーランドイメージ

13.3. システムの最適化を行う

ここでは、システム開発の最終段階の最適化について説明します。

ベースとした Armaidllo-810 では、システムに不要なアプリケーションなどが含まれていると思います。不要なアプリケーションを省くことでイメージファイルがスリムになり起動速度が向上したり、空きメモリ容量が増えるなどのシステムの負荷が軽減します。

また、セキュリティーについても考慮すべきでしょう。Armadillo のデフォルトの root パスワードは、「root」となっています。デフォルトのままにしてしまうと簡単にハッキングされてしまう恐れがあります。

必要のないアプリケーションを削除したり、パスワードの変更を行うには、make menuconfig などを行いシステムを変更します。

手順 13.1 必要のないアプリケーションを削除する

1. make menuconfig を行い「Kernel/Library/Defaults Selection --->」を選択します。

```
[ATDE ~]$ cd atmark-dist
[ATDE ~/atmark-dist]$ make menuconfig
```

```
 atmark-dist v1.31.0 Configuration
--------------------------------------------------------------------------------
                              Main Menu
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ------------------------------------------------------------------------
               Vendor/Product Selection  --->
               Kernel/Library/Defaults Selection  --->
               ---
               Load an Alternate Configuration File
               Save Configuration to an Alternate File


  ------------------------------------------------------------------------

                   <Select>    < Exit >    < Help >
```

2. 「Customize Vendor/User Settings」を選択して"Exit"を 2 回して「Do you wish to save your new kernel configuration?」で"Yes"とします。

```
 atmark-dist v1.31.0 Configuration
--------------------------------------------------------------------------
                    Kernel/Library/Defaults Selection
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ------------------------------------------------------------------
            --- Kernel is linux-3.x
            (default) Cross-dev
            (None) Libc Version
            [ ] Default all settings (lose changes) (NEW)
            [ ] Customize Kernel Settings (NEW)
            [*] Customize Vendor/User Settings (NEW)
            [ ] Update Default Vendor Settings (NEW)

  ------------------------------------------------------------------

                    <Select>    < Exit >    < Help >
```

3. Userland Configuration メニューが表示されます。

```
 atmark-dist v1.31.0 Configuration
--------------------------------------------------------------------------
                        Userland Confgiguration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ------------------------------------------------------------------
              Vendor specific  --->
              Core Applications  --->
              Library Configuration  --->
              Flash Tools  --->
              Filesystem Applications  --->
              Network Applications  --->
              Miscellaneous Applications  --->
              BusyBox  --->
              Tinylogin  --->
              X Window System  --->
  ------------------------------------------------------------------

                    <Select>    < Exit >    < Help >
```

4. ここでは、例として「gstreamer」を削除してみます。「Miscellaneous Applications --->」を選択しメニューをスクロールすると「Multimedia tools」に gstreamer の項目があります。

```
 atmark-dist v1.31.0 Configuration
--------------------------------------------------------------------------
                      Miscellaneous Applications
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ------------------------------------------------------------------
               --- Multimedia tools
               [*] gstreamer
               [*]   gst-feedback
               [*]   gst-inspect
               [*]   gst-launch
               [*]   gst-typefind
               [ ]   gst-xmlinspect
               [ ]   gst-xmllaunch
                      plugins  --->
               --- Audio tools
  ------------------------------------------------------------------

                   <Select>    < Exit >    < Help >
```

5. gstreamer にカーソルを合わせて選択を解除して、"Exit"を 2 回して「Do you wish to save your new kernel configuration?」で"Yes"とすることで選択を解除することができます。

```
  ------------------------------------------------------------------
               --- Multimedia tools
               [ ] gstreamer
               --- Audio tools
```

6. ビルドを行う前に、atmark-dist/romfs/ディレクトリを一旦削除しておきます。この操作は、選択を解除したアプリケーションが romfs/ディレクトリに残ってしまうことを防止するために必要となります。

```
[ATDE ~/atmark-dist]$ rm -rf romfs/
```

手順 13.2 root パスワードを変更する

1. 「手順 13.1. 必要のないアプリケーションを削除する」と同様に、make menuconfig を使い「Userland Configuration」メニューを開きます。

2. 「Vendor specific --->」を選択します。

```
 atmark-dist v1.31.0 Configuration
--------------------------------------------------------------------------------
                             Vendor specific
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ------------------------------------------------------------------
               [ ] change root password
               (Auto) generate file-system option
               --- Kernel modules
               [ ] Armadillo-WLAN

  ------------------------------------------------------------------

                   <Select>    < Exit >    < Help >
```

3. 「change root passwd」を選択すると、root パスワードを変更することができます。

```
  ------------------------------------------------------------------
               [*] change root password
                    root password: "root"
               (Auto) generate file-system option
               --- Kernel modules
               [ ] Armadillo-WLAN

  ------------------------------------------------------------------
```

13.4. オリジナルプロダクトのコンフィグレーションを更新する

make menuconfig で修正を加えたコンフィグレーションは、一時ファイルとして保存されています。一時ファイルは make clean や make distclean などで Atmark Dist をクリーンアップした場合に削除されてしまいます。再度コンフィグレーションを復元するためには、一からコンフィグレーション手順を再現しなくてはなりません。

Atmark Dist をクリーンアップした場合でも、設定したコンフィグレーションを恒久的に復元させることができるように、プロダクトのデフォルトコンフィグレーションを上書き更新する手順を説明します。

手順 13.3 プロダクトのデフォルトコンフィグレーションを上書き更新する

1. 「手順 13.1. 必要のないアプリケーションを削除する」と同様に、make menuconfig を使い「Kernel/Library/Defaults Selection」メニューを開きます。

2. 「Update Default Vendor Settings」を選択しておきます。「Customize Vendor/User Settings」でコンフィグレーションを変更した場合などに、自動的にプロダクトのデフォルトコンフィグレーションが上書き更新されるようになります。

```
 atmark-dist v1.31.0 Configuration
--------------------------------------------------------------------------------
                    Kernel/Library/Defaults Selection
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
  ------------------------------------------------------------------------
             --- Kernel is linux-3.x
             (default) Cross-dev
             (None) Libc Version
             [ ] Default all settings (lose changes) (NEW)
             [ ] Customize Kernel Settings (NEW)
             [ ] Customize Vendor/User Settings (NEW)
             [*] Update Default Vendor Settings (NEW)

  ------------------------------------------------------------------------

                    <Select>    < Exit >    < Help >
```

「Update Default Vendor Settings」を選択した場合に更新されるデフォルトコンフィグファイルを「表 13.1. デフォルトコンフィグファイル」に示します。

表 13.1 デフォルトコンフィグファイル

対象	デフォルトコンフィグレーションファイル
Linux カーネル	[プロダクトディレクトリ]/config.linux-3.x[a]
Userland	[プロダクトディレクトリ]/config.vendor
Busybos-1.20.2	[プロダクトディレクトリ]/config.busybox-1.20.2

[a] ファイルが存在しない場合は、Linux カーネルのデフォルトコンフィグが使用されます

14. プログラミングガイド

14.1. アプリケーションでカメラデバイスを扱う方法

ここではカメラデバイスを制御するアプリケーションを実装する方法について、C 言語で実装されている既存のソースコードをサンプルに、カメラデバイスの初期化方法やデータの取得方法などについて説明します。

Linux カーネルのカメラドライバは、Video for Linux Two(V4L2)というビデオキャプチャ API として実装するのが一般的です。Armadillo-810 カメラモジュール 01 (B コネクタ用)のデバイスドライバも V4L2 デバイスとして実装されているため、ユーザーアプリケーションからは、この V4L2 API で扱うことができます。

Video for Linux Two(V4L2)についての詳しい情報は、「LINUX MEDIA INFRASTRUCTURE API」の「Video for Linux Two API Specification」や Linux の解説書籍などをご覧ください。

LINUX MEDIA INFRASTRUCTURE API

http://linuxtv.org/downloads/v4l-dvb-apis/index.html

14.1.1. カメラデバイスを制御するシステムコール

カメラデバイスを制御するには、通常のファイルを操作する場合と同様にシステムコールを介して行われます。カメラデバイスを制御する場合に利用するシステムコールについて説明します。

システムコールのマニュアルを参照するには、次のようにします。

```
[ATDE ~]$ man [システムコール]
```

14.1.1.1. open システムコール

ユーザーアプリケーションから V4L2 API でカメラデバイスを制御するには、ビデオデバイスファイル(/dev/videoN)を通して行われます。デバイスファイルに対してアクションを行う場合は、通常のファイルと同様に open()システムコールを使いファイルディスクリプタを取得します。

```
int open(const char *pathname, int flags);
```

図 14.1 open システムコールの書式

"flags"には、必ず O_RDWR (読み込み/書き込み許可)が含まれるように指定します。このフラグを指定せずに open()した場合、ioctl()が失敗するなどの意図しない挙動となってしまう場合があります。

open()の戻り値がファイルディスクリプタとなります。open()でエラーが発生した場合は、戻り値が"-1"となります。

アプリケーションがカメラデバイスを開放する場合には、open()で取得したファイルディスクリプタを開放します。ファイルディスクリプタの開放には、close()システムコールを使います。

```
int close(int fd);
```

図 14.2 close システムコールの書式

```
int fd;
fd = open("/dev/video0", O_RDWR);
if (fd == -1) {
  perror("open");
  return;
}
        :
ファイルディスクリプタに対しての処理
        :
close(fd);
```

図 14.3 open()と close()のサンプル

14.1.1.2. ioctl システムコール

V4L2 API を使用してカメラドライバにアクションを行うには、ioctl()システムコールを利用します。

```
int ioctl(int fd, int request, ...);
```

図 14.4 ioctl システムコールの書式

"request"には、V4L2 のリクエストコードを指定します。ほとんどの場合、リクエストコードに渡すデータを第 3 引数に指定します。 代表的な V4L2 リクエストコードを「表 14.1. 画像キャプチャーで利用する代表的な V4L2 リクエストコード」に示します。

表 14.1 画像キャプチャーで利用する代表的な V4L2 リクエストコード

リクエストコード	説明
VIDIOC_QUERYCAP	カメラデバイスに保有する機能を問い合わせます。 アプリケーションが利用したい機能を保有しているかどうかを確認するために行います。
VIDIOC_S_FMT VIDIOC_G_FMT	カメラデバイスに画像データのフォーマットを設定・取得します。 PixelFormat を指定する場合は、「表 14.2. Armadillo-810 カメラモジュール 01 (B コネクタ用)で対応可能な PixelFormat」を参照してください。
VIDIOC_REQBUFS	画像データ用のバッファを要求します。
VIDIOC_QUERYBUF	VIDIOC_REQBUFS で要求したバッファの情報を受けとります。
VIDIOC_QBUF VIDIOC_DQBUF	バッファをストリーミングキューに繋いだり、外したりします。
VIDIOC_STREAMON VIDIOC_STREAMOFF	ストリーミングを開始・停止します。開始されるとストリーミングキューに繋がれている空のバッファに画像データが書き込まれます。
VIDIOC_G_CTRL VIDIOC_S_CTRL	コントロールの設定・取得を行います。主にホワイトバランスや色合いなどの調整に用います。[a]

[a]Armadillo-810 カメラモジュール 01 (B コネクタ用)のカメラドライバでは未対応となっています。

表 14.2 Armadillo-810 カメラモジュール 01 (B コネクタ用)で対応可能な PixelFormat

PixelFromat	説明
V4L2_PIX_FMT_YUYV V4L2_PIX_FMT_UYVY	1 ピクセルが 16bit 長の YUV データ
V4L2_PIX_FMT_NV12 V4L2_PIX_FMT_NV21	1 ピクセルが 12bit 長で輝度データと色差データが分離された YUV データ
V4L2_PIX_FMT_NV16 V4L2_PIX_FMT_NV61	1 ピクセルが 16bit 長で輝度データと色差データが分離された YUV データ
V4L2_PIX_FMT_RGB565	1 ピクセルが 16bit 長の RGB データ

14.1.1.3. mmap システムコール

カメラドライバが確保した画像データ用のバッファを利用する場合、アプリケーションからアクセスできるメモリ空間にマッピングしなおす必要があります。この操作には、mmap()システムコールを利用します。

```
void *mmap(void *addr, size_t length, int prot, int flags, int fd, off_t offset);
```

図 14.5 mmap システムコールの書式

mmap()したメモリ空間が不要となった場合は、munmap()システムコールを使用してメモリ空間をアンマッピングします。

```
int munmap(void *addr, size_t length);
```

図 14.6 munmap システムコールの書式

14.1.1.4. select システムコール

カメラドライバが画像データの準備を完了するまでアプリケーションをウェイトさせておくには、select()システムコールを利用します。

```
int select(int nfds, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout);
```

図 14.7 select システムコールの書式

14.1.2. V4L2 実装例の解説

ここでは、カメラモデル開発セット付属の DVD に収録されている V4L2 サンプルコードを例にとって説明します。ソースコード(v4l2-sample-[version].tar.gz)は、DVD の/sample/ディレクトリに収録されています。また、アットマークテクノ ダウンロードサイトからも取得することができます。

アットマークテクノ ダウンロードサイト - Armadillo-810 - サンプル

http://download.atmark-techno.com/armadillo-810/sample/

14.1.2.1. 初期化処理

まずはじめに、カメラデバイスをオープンしています。カメラデバイスを扱う場合は、このオープン処理が第一歩となります。Armadillo-810 カメラモジュール 01 (B コネクタ用)のデバイスファイルは、"/dev/video1"[1](CAMERA_DEV)となります。

```
84   fd = open(CAMERA_DEV, O_RDWR, 0);
85   if (fd < 0) {
86     perror("open");
87     return -1;
88   }
```

図 14.8 【camera2ppm.c】カメラデバイスをオープン

次にキャプチャーする画像データのフォーマットを指定しています。640x480 の VGA サイズ(CAMERA_WIDTH, CAMERA_HEIGHT)で、PixelFormat には YUYV(V4L2_PIX_FMT_YUYV)を指定しています。ioctl(VIDIOC_S_FMT)が成功(ret == 0)した場合でも、指定したフォーマットがカメラドライバで未対応であった場合には、fmt 内のデータが対応可能なフォーマットに書き換わっている場合があります。意図していないフォーマットで動作するのを防止するために、ioctl(VIDIOC_S_FMT)の後に fmt 内のデータを確認しています。

```
 90   memset(&fmt, 0, sizeof(fmt));
 91   fmt.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
 92   fmt.fmt.pix.width = CAMERA_WIDTH;
 93   fmt.fmt.pix.height = CAMERA_HEIGHT;
 94   fmt.fmt.pix.pixelformat = V4L2_PIX_FMT_YUYV;
 95   fmt.fmt.pix.field = V4L2_FIELD_INTERLACED;
 96   ret = xioctl(fd, VIDIOC_S_FMT, &fmt);
 97   if (ret < 0 || fmt.fmt.pix.pixelformat != V4L2_PIX_FMT_YUYV ||
 98       fmt.fmt.pix.width <= 0 || fmt.fmt.pix.height <= 0) {
 99     perror("ioctl(VIDIOC_S_FMT)");
100     return -1;
101   }
```

図 14.9 【camera2ppm.c】画像データフォーマットを設定

このサンプルでは ioctl()のシグナルに対する使い勝手の悪さを克服するために、xioctl()という関数を実装しています。ioctl()の処理の最中にシグナルを受けた場合はエラー終了してしまいますが、エラー要因がシグナル受信(EINTR)だった場合に自動的に ioctl()を再試行してくれるラッパー関数となっています。

[1]デフォルト状態の Linux カーネルの場合。コンフィグレーションを変更した場合には他のデバイス番号となる場合があります。

```
56 static int xioctl(int fd, int request, void *arg)
57 {
58   for (; ; ) {
59     int ret = ioctl(fd, request, arg);
60     if (ret < 0) {
61       if (errno == EINTR)
62         continue;
63       return -errno;
64     }
65     break;
66   }
67
68   return 0;
69 }
```

続いて、カメラドライバに対して画像データ用のバッファを要求しています。メモリマップ用(V4L2_MEMORY_MMAP)のバッファを 2 面(MMAP_COUNT)として ioctl(VIDIOC_REQBUFS)を呼び出しています。

カメラドライバが ioctl(VIDIOC_REQBUFS)を受けとると、ioctl(VIDIOC_S_FMT)で指定された画像サイズのバッファをメモリ空間から確保します。この場合に、画像サイズ × 面数の合計があまりにも大きな場合は ioctl(VIDIOC_REQBUFS)はエラーとなってしまいます。アプリケーションで最低限必要な面数を指定するのがベストでしょう。

```
112   memset(&req, 0, sizeof(req));
113   req.count = MMAP_COUNT;
114   req.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
115   req.memory = V4L2_MEMORY_MMAP;
116   ret = xioctl(fd, VIDIOC_REQBUFS, &req);
117   if (ret < 0) {
118     perror("ioctl(VIDIOC_REQBUFS)");
119     return -1;
120   }
```

図 14.10 【camera2ppm.c】画像データバッファを要求

ioctl(VIDIOC_REQBUFS)で要求したバッファを面数分取得し、アプリケーションがアクセスできるようにバッファ用のメモリをユーザー空間にマッピング(mmap)しています。

```
123   for (i = 0; i < count; i++) {
124     memset(&buf, 0, sizeof(buf));
125     buf.type   = V4L2_BUF_TYPE_VIDEO_CAPTURE;
126     buf.memory = V4L2_MEMORY_MMAP;
127     buf.index  = i;
128     ret = xioctl(fd, VIDIOC_QUERYBUF, &buf);
129     if (ret < 0) {
130       perror("ioctl(VIDIOC_QUERYBUF)");
131       return -1;
132     }
133
134     mmap_p[i] = mmap(NULL, buf.length, PROT_READ, MAP_SHARED, fd, buf.m.offset);
135     if (mmap_p[i] == MAP_FAILED) {
136       perror("mmap");
137       return -1;
138     }
139     mmap_l[i] = buf.length;
140   }
```

図 14.11 【camera2ppm.c】画像データバッファの取得とユーザー空間へのマッピング

14.1.2.2. 画像データを取得するには

V4L2 のキャプチャー処理の基本的な流れは、下記の工程を繰り返します。

I. ビデオストリームに空の画像バッファをエンキュー(enqueue)

II. ビデオドライバが空の画像バッファをフィル(fill)

III. ビデオストリームからフィルされたバッファをデキュー(dequeue)

(I.) と (III.)はアプリケーションが ioctl()を用いて行う工程です。(II.) はビデオドライバが担う工程となっています。バッファをエンキュー後、どのタイミングでデキューすればいいのか？と疑問を持たれるかもしれないのですが、これは select()を利用することでデキューのタイミングを測ることができます。サンプルコードでも select()を利用してデキューのタイミングを測っています。

初めてビデオストリームにエンキューする箇所です。ここではキャプチャーが開始されていないため(後述)エンキューされたとしてもフィルされることはありません。キャプチャー開始に備えてあらかじめエンキューしています。

```
142   for (i = 0; i < count; i++) {
143     memset(&buf, 0, sizeof(buf));
144     buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
145     buf.memory = V4L2_MEMORY_MMAP;
146     buf.index = i;
147     ret = xioctl(fd, VIDIOC_QBUF, &buf);
148     if (ret < 0) {
149       perror("ioctl(VIDIOC_QBUF)");
150       return -1;
151     }
152   }
```

図 14.12 【camera2ppm.c】ビデオストリームにバッファをエンキュー

デキューの処理部分は次のコードです。

```
178       memset(&buf, 0, sizeof(buf));
179       buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
180       buf.memory = V4L2_MEMORY_MMAP;
181       ret = xioctl(fd, VIDIOC_DQBUF, &buf);
182       if (ret < 0 || buf.bytesused < (__u32)(2 * length)) {
183         perror("ioctl(VIDOC_DQBUF)");
184         return -1;
185       }
```

図 14.13 【camera2ppm.c】ビデオストリームからバッファをデキュー

前述した(I.)～(III.)の工程を処理しているのが次のコードです。

```
161   for (i = 0, pyuyv = yuyvbuf; i < PICTURE_NUM; i++) {
162     fd_set fds;
163
164     FD_ZERO(&fds); ❶
165     FD_SET(fd, &fds); ❷
166     for (; ; ) {
167       ret = select(fd + 1, &fds, NULL, NULL, NULL); ❸
168       if (ret < 0) {
169         if (errno == EINTR) ❹
170           continue;
171         perror("select");
172         return -1;
173       }
174       break;
175     }
176
177     if (FD_ISSET(fd, &fds)) { ❺
178       memset(&buf, 0, sizeof(buf));
179       buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
180       buf.memory = V4L2_MEMORY_MMAP;
181       ret = xioctl(fd, VIDIOC_DQBUF, &buf);
182       if (ret < 0 || buf.bytesused < (__u32)(2 * length)) {
183         perror("ioctl(VIDOC_DQBUF)");
184         return -1;
185       }
186
187       memcpy(pyuyv, mmap_p[buf.index], 2 * length); ❻
188       pyuyv += 2 * length;
189
190       ret = xioctl(fd, VIDIOC_QBUF, &buf); ❼
191       if (ret < 0) {
192         perror("ioctl(VIDIOC_QBUF)");
193         return -1;
194       }
195     }
```

❶　ファイルディスクリプタ集合(fds)を空にしています。

❷ ファイルディスクリプタ集合(fds)にカメラデバイスのファイルディスクリプタ(fd)をセットしています

❸ ファイルディスクリプタ集合(fds)を readfds に指定して select()を実行しています。こうすることで、カメラドライバがバッファをフィルするまでアプリケーションをウェイトさせることができます。

❹ select()も ioctl()と同様に処理中にシグナルを受信した場合にエラーとなってしまうため、エラー要因がシグナル受信(EINTR)であれば再試行される仕組みとなっています。

❺ FD_ISSET()はファイルディスクリプタ集合(fds)の内、指定のファイルディスクリプタ(fd)が読み込み準備完了となっているかを検査することができます。

❻ 画像データをアプリケーションが持つ YUYV データ保管用のバッファにコピーしています。画像バッファを再利用するために画像データを退避させる目的でしょう。

❼ 画像バッファを再度エンキューしています。

図 14.14 【camera2ppm.c】画像データの取得

14.1.2.3. キャプチャー開始と停止

ビデオストリームにエンキューした画像バッファをフィルするには、キャプチャーを開始しなくてはなりません。キャプチャーを開始すると空の画像バッファがなくなるまで順次フィルを行います。

```
154   type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
155   ret = xioctl(fd, VIDIOC_STREAMON, &type);
156   if (ret < 0) {
157     perror("ioctl(VIDIOC_STREAMON)");
158     return -1;
159   }
```

図 14.15 【camera2ppm.c】キャプチャー開始

キャプチャーは停止することができます。

```
198   type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
199   xioctl(fd, VIDIOC_STREAMOFF, &type);
```

図 14.16 【camera2ppm.c】キャプチャー停止

14.1.2.4. 取得した画像データのその後

ioctl(VIDIOC_S_FMT)で PixelFormat に V4L2_PIX_FMT_YUYV を設定しているため、取得できる画像データは YUYV フォーマットとなっています。

サンプルコードでは、取得した画像データから PPM(Portable Pix Map)ファイル(.ppm)を作成しています。PPM ファイルは、RGB24 ビットカラーを表現するフォーマットとなります。取得した画像データは YUYV フォーマットのため PPM ファイルを作成するためには、YUYV to RGB24 フォーマット変換をする必要があります。サンプルコードでは、yuyv_to_rgb()関数で行っています。

```
214   for (i = 0, pyuyv = yuyvbuf, prgb = rgbbuf; i < PICTURE_NUM;
215        i++, pyuyv += 2 * length, prgb += 3 * length)
216     yuyv_to_rgb(pyuyv, prgb, length);
```

図 14.17 【camera2ppm.c】画像データを YUYV フォーマットから RGB24 フォーマットに変換

RGB24 フォーマットに変換された画像データから PPM ファイルを作成しているのは、ppm_writefile() 関数です。

```
225   ppm_writefile(rgbbuf, width, height, PICTURE_NUM);
```

図 14.18 【camera2ppm.c】画像データ(RGB24)を PPM ファイルとして保存

14.1.3. サンプルコードを Armadillo で動かしてみる

ここでは、サンプルコードを実際に Armadillo-810 で動作させる手順を紹介します。サンプルコードの動作を確認したり、改変して挙動を確認したりする場合などに利用してください。

14.1.3.1. サンプルコードをビルドする

サンプルコードには、Armadillo-810 に最適化してビルドすることができる Makefile が付属しています。make コマンドを用いて簡単にビルドすることができます。

```
[ATDE ~/v4l2-sample]$ make
arm-linux-gnueabihf-gcc -Wall -Wextra -O3  -mcpu=cortex-a9 -march=armv7-a -mfpu=neon -c -o
camera2ppm.o camera2ppm
arm-linux-gnueabihf-gcc  camera2ppm.o -lnetpbm -o v4l2-sample
[ATDE ~/v4l2-sample]$ ls v4l2-sample
v4l2-sample
```

図 14.19 サンプルコードをビルド

14.1.3.2. v4l2-sample を Armadillo-810 上で実行させる

ビルドしてできあがった v4l2-sample を Armadillo-810 上で実行させてみましょう。Armadillo-810 にファイルを転送する方法はいくつかありますが、ここでは FTP を利用します。また、v4l2-sample は、libnetpbm という共有ライブラリを必要とします。一緒に転送しておきます。

```
[ATDE ~/v4l2-sample]$ ftp armadillo810-0.local
Connected to armadillo810-0.local.
220 localhost FTP server (GNU inetutils 1.4.1) ready.
Name (armadillo810-0.local:atmark): ftp
331 Guest login ok, type your name as password.
Password:
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd pub
250 CWD command successful.
ftp> put v4l2-sample
local: v4l2-sample remote: v4l2-sample
200 PORT command sucessful.
150 Opening BINARY mode data connection for 'v4l2-sample'.
226 Transfer complete.
9636 bytes sent in 0.00 secs (4252.2 kB/s)
ftp> put /usr/arm-linux-gnueabihf/lib/libnetpbm.so.10 libnetpbm.so.10
local: /usr/arm-linux-gnueabihf/lib/libnetpbm.so.10 remote: libnetpbm.so.10
200 PORT command sucessful.
150 Opening BINARY mode data connection for 'libnetpbm.so.10'.
226 Transfer complete.
88944 bytes sent in 0.01 secs (7883.4 kB/s)
ftp>
```

図 14.20 FTP で v4l2-sample を Armadillo-810 に転送

FTP で Armadillo-810 にファイルを転送すると/home/ftp/pub にファイルが作成されています。

実行する前に、デフォルトイメージで動作している uvc-gadget を停止させます。uvc-gadget は、カメラデバイスを利用しているため v4l2-sample がカメラデバイスを利用することができない状態となっています。

```
[armadillo ~/home/ftp/pub]# killall uvc-gadget
```

図 14.21 uvc-gadget を停止

続いて v4l2-sample 実行してみます。転送したライブラリを利用できるように、ライブラリのサーチパスにカレントディレクトリを指定しています。

```
[armadillo ~/home/ftp/pub]# LD_LIBRARY_PATH=. ./v4l2-sample
convert time: 4.948 msec/flame
[armadillo ~/home/ftp/pub]# ls
camera00.ppm     camera08.ppm     camera16.ppm     camera24.ppm
camera01.ppm     camera09.ppm     camera17.ppm     camera25.ppm
camera02.ppm     camera10.ppm     camera18.ppm     camera26.ppm
camera03.ppm     camera11.ppm     camera19.ppm     camera27.ppm
camera04.ppm     camera12.ppm     camera20.ppm     camera28.ppm
camera05.ppm     camera13.ppm     camera21.ppm     camera29.ppm
camera06.ppm     camera14.ppm     camera22.ppm     libnetpbm.so.10
camera07.ppm     camera15.ppm     camera23.ppm     v4l2-sample
```

図 14.22 v4l2-sample を実行

作成された PPM ファイルを参照するには ATDE にインストールされている"eog"というイメージビューワーを利用します。PPM ファイルは FTP を利用して転送します。

```
ftp> get camera15.ppm
local: camera15.ppm remote: camera15.ppm
200 PORT command sucessful.
150 Opening BINARY mode data connection for 'camera15.ppm' (921615 bytes).
226 Transfer complete.
921615 bytes received in 0.26 secs (3432.5 kB/s)
ftp> quit
221 Goodbye.
[ATDE ~/v4l2-sample]$ eog camera15.ppm
```

図 14.23 ATDE で PPM ファイルを表示

15. AV コーデックミドルウェア

15.1. AV コーデックミドルウェアとは

AV コーデックミドルウェアは、Armadillo-800 シリーズでマルチメディア処理 (H.264/AVC 動画・AAC 音声・JPEG 画像の変換処理) をスムーズかつ効率的に行うためのミドルウェアです。Armadillo-800 シリーズに搭載されているアプリケーションプロセッサー「R-Mobile A1」には、メインの ARM コア以外にリアルタイム制御用の SH-4A コアとマルチメディア処理専用プロセッサが搭載されています。マルチメディア処理は、多くの場合システムに負荷をかけることが多く、メイン CPU で処理をするとシステム全体のパフォーマンスが低下します。AV コーデックミドルウェアを利用することで、メイン CPU のパフォーマンスを落すことなく、マルチメディアの処理を行うことができるようになります。

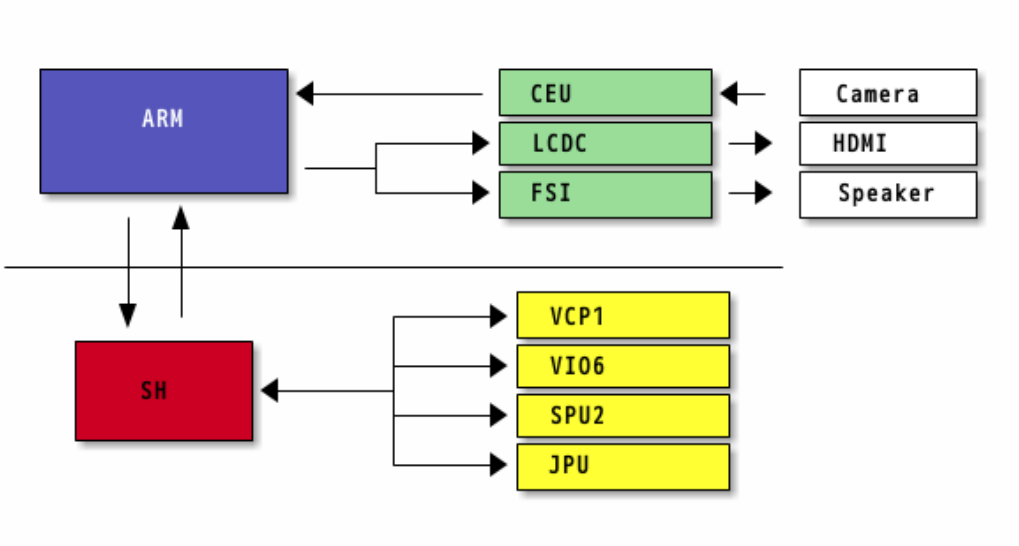


図 15.1 AV コーデックミドルウェア使用時の内蔵コアの対応

上図に示すように、AV コーデックミドルウェアを使う場合には R-Mobile A1 に搭載されている以下の専用プロセッサは SH 側で管理されます。

- Video Multi Codec (VCP1)
- Video I/O 6 (VIO6)
- Sound Processing Unit 2 (SPU2)
- JPEG Processing Unit (JPU)

AV コーデックミドルウェアが対応しているフォーマットは以下の通りです。

- デコーダーが対応しているフォーマット
  - H.264/AVC
  - AAC
- エンコーダーが対応しているフォーマット
  - H.264/AVC
  - AAC
  - JPEG

マルチメディアは、比較的大きなデータを扱います。そのためマルチメディア処理を行う専用プロセッサや管理する SH もメインメモリを利用する必要があります。AV コーデックミドルウェアを利用する時には、Armadillo-800 シリーズに搭載されている DDR3 SDRAM 512MB のうち 128MB を AV コーデックミドルウェアに割り当てる必要があります。AV コーデックミドルウェアドライバが追加された linux-3.4-at6 以降では、デフォルトで Linux カーネルが管理するメモリを 384MB に制限しています。

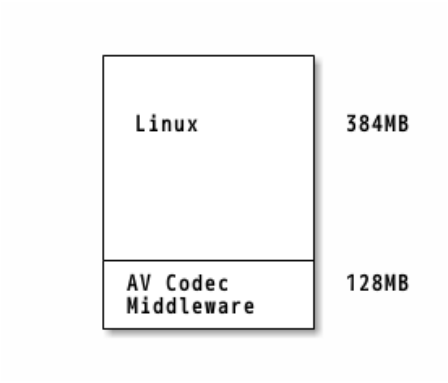


図 15.2 AV コーデックミドルウェア使用時のメモリマップ

このように、多くのプロセッサが協調動作をすることで、Armadillo-800 シリーズでは H.264/AVC 動画や AAC 音声などのマルチメディアデータをスムーズかつ効率的に扱うことができます。

さらに AV コーデックミドルウェアは、アプリケーションプログラムからマルチメディアデータを容易に扱えるよう、Linux で標準のマルチメディアフレームワーク GStreamer [http://gstreamer.freedesktop.org/]に対応しています。

図 15.3 GStreamer ロゴ

GStreamer は、エレメントと呼ばれる処理単位を繋ぎ合わせパイプラインを作成することで、複雑な要求に対応できるマルチメディアフレームワークです。MP4 ファイルを扱うエレメントや、様々な画像フォーマットを変換するエレメントなどが標準で用意されているため、このフレームワークを活用することで、様々なフォーマットのマルチメディアデータを統一的な操作で、簡単に扱うことができます。

15.2. AV コーデックミドルウェアの仕様

15.2.1. AAC デコーダー

表 15.1 AAC デコーダー仕様

符号化方式	・準拠規格 ISO/IEC 13818-7:2006、ISO/IEC 14496-3:2009 ・対応フォーマット AAC-LC、HE-AAC v1、HE-AAC v2
ビットレート	・AAC-LC 8k～576k bit/sec、VBR ・HE-AAC v1/v2 8k～144k bit/sec、VBR
入力フォーマット	・RAW 形式 / ADTS 形式

入力チャネル	・AAC-LC [a] ・1 チャネル(モノラル) ・2 チャネル (ステレオ、デュアルモノラル、パラメトリックステレオ) ・4 チャネル (3/1) [b] ・5 チャネル (3/2)[b] ・5.1 チャネル (3/2+LFE)[b] ・HE-AAC v1/v2 ・1 チャネル (モノラル) ・2 チャネル (ステレオ、デュアルモノラル、パラメトリックステレオ)
出力チャネル	1 チャネルか 2 チャネル (ダウンミックス対応)
サンプリング周波数	・AAC-LC: 8k/11.025k/12k/16k/22.05k/24k/32k/44.1k/48k/64k/88.2k/96k Hz ・HE-AAC v1/v2: 8k/11.025k/12k/16k/22.05k/24k Hz

[a] 3 チャネル(3/0、2/1)、4 チャネル(2/2)は非サポート

[b] (/)は、(前方/後方スピーカのチャネル数)を示す

15.2.2. H.264/AVC デコーダー

表 15.2 H.264/AVC デコーダー仕様

最大ビットレート	40M bit/sec [a]
入力フォーマット	Video Elementary Stream
出力フォーマット	・YUV420 ・RGB32 ・RGB24 ・RGB16
サポートプロファイル/レベル	・Baseline Profile Level 4.1 ・Constrained Baseline Profile Level 4.1 ・Main Profile Level 4.1 ・High Profile Level 4.1
プロファイル共通非サポートツール	・ASO (Arbitrary Slice Ordering) ・FMO (Flexible Macroblock Ordering) ・RS (Redundant Slices)
ピクチャ構造	・フレーム構造 (プログレッシブシーケンス / インタレースシーケンス [b]) ・フィールド構造 (インタレース) ・フレーム / フィールド混在構造 (インタレースシーケンス)
エントロピー符号化	CAVLC / CABAC
ピクチャタイプ	I / P / B ピクチャ
マルチスライス	サポート。1 フレームあたりの最大スライス数は 68 スライスまで(画像サイズ 1920x1080 の場合、1 スライス/1 マクロブロックライン相当)
マルチリファレンス	サポート
マルチシーケンス	サポート。ただし、シーケンスを通じて以下の条件満すこと ・画像サイズ、フレームレートが変化しないこと ・エンコード時のビットレート設定が同じであること
マルチストリーム	非サポート
データ・パーティショニング	非サポート
画像サイズ	・プログレッシブシーケンスの場合: 128x96〜1920x1080 [c] ・インターレースシーケンスの場合: 352x480〜1920x1080 [d]
画像の拡大・縮小	幅、高さともに拡大: 最大 16 倍まで、縮小: 最少 1/16 まで

[a] 最大ピーク時のビットレートが 40M bit/sec 以下であること

[b] インタレースシーケンスの場合、30fps の場合画像サイズ 1440x1080 まで、画像サイズ 1920x1080 であれば 20fps まで

[c] 水平 2 画素、垂直 2 ライン単位で設定可能

[d] 水平 2 画素、垂直 4 ライン単位で設定可能

15.2.3. AAC エンコーダー

表 15.3 AAC エンコーダー仕様

準拠規格	・ISO/IEC 13818-7:2006

ビットレート	16k ～ 288k bps、VBR [a]
対応チャネル	・1 チャネル(モノラル) ・2 チャネル (ステレオ、デュアルモノラル)
入力フォーマット	16bit PCM
入力サンプリング周波数	8k/11.025k/12k/16k/22.05k/24k/32k/44.1k/48k Hz
出力フォーマット	AAC-LC(RAW 形式、ADTS 形式)

[a] 1 チャネルあたり

15.2.4. H.264/AVC エンコーダー

表 15.4 H.264/AVC エンコーダー仕様

最大ビットレート	40M bit/sec [a]
最大フレームレート	水平画像サイズ 1280 以下かつ垂直画像サイズ 720 以下の場合 ・60 frame/sec 上記以外 ・30 frame/sec
最大参照フレーム	2 フレーム
入力画像サイズ	画像幅 ・80～1920 [b] 画像高さ ・80～1088 [c] [d]
入力フォーマット	YUV420
出力フォーマット	Video Elementary Stream
プロファイル/レベル	条件により自動で選択 ・Baseline Profile Level 4.1 以下条件を全て満たす場合に選択されます ・フレーム構造(プログレッシブシーケンス) ・B ピクチャなし ・画像サイズが 1920×1080 未満 ・Main Profile Level 4.1 以下条件を満たすときに選択されます ・B ピクチャあり ・High Profile Level 4.1 以下条件を満たすときに選択されます ・画像サイズが 1920×1080
プロファイル共通非サポートツール	・ASO (Arbitrary Slice Order) ・FMO (Flexible Macroblock Ordering) ・RS (Redundant Slice) ・MBAFF (Macroblock-Adaptive Frame-Field) coding ・Weighted Prediction
ピクチャ構造	フレーム構造 (プログレッシブシーケンス)
エントロピー符号化	Baseline Profile の場合 ・CAVLC Main Profile または High Profile の場合 ・CABAC

動き探索範囲	水平 ・ -64～63.75 画素 ・ -32～31.75 画素(画像幅が 144 以下のとき) 垂直 ・ -32～31.75 サブサンプリング ・ 1/4 サブサンプリング
ピクチャタイプ	I / P / B ピクチャ
イントラ MB リフレッシュ	サポート [e]
イントラ予測方式	4×4、8×8、16×16 画素単位イントラ予測
変換方式	4×4、8×8 整数変換
マルチスライス	非サポート
マルチリファレンス	非サポート
階層エンコード	非サポート
マルチシーケンス	非サポート
マルチストリーム	非サポート
リエントラント対応	なし

[a] 最大ビットレートの設定値であり、レートを保証するものではありません
[b] 2 画素単位で指定可能
[c] 2 ライン単位で指定可能
[d] 画像幅×画像高さが 32 の倍数であること
[e] 先頭のみ I ピクチャのとき適用

15.3. GStreamer - マルチメディアフレームワーク

GStreamer は、オープンソースのマルチメディアフレームワークです。小さなコアライブラリに様々な機能をプラグインとして追加できるようになっており、多彩な形式のデータを扱うことができます。GStreamer で扱うことができるデータフォーマットの一例を下記に示します。

- コンテナフォーマット: mp4, avi, mpeg-ps/ts, mkv/webm, ogg
- 動画コーデック: H.264/AVC, Vorbis
- 音声コーデック: AAC, Theora, wav
- 画像フォーマット: JPEG, PNG, BMP
- ストリーミング: http, rtp

GStreamer では、マルチメディアデータをストリームとして扱います。ストリームを流すパイプラインの中に、エレメントと呼ばれる処理単位を格納し、それらをグラフ構造で繋ぎ合わせることで、デコードやエンコードなどの処理を行います。例えば、「図 15.4. GStreamer の実行例」に示すコマンドを実行した場合のパイプラインは「図 15.5. GStreamer のパイプライン例」となります。

```
[armadillo ~]# gst-launch-1.0 filesrc location=/mnt/big-buck-bunny-30sec-fullhd.mp4 \
 ! qtdemux name=demux0 \
 demux0.audio_0 ! queue ! acmaacdec ! audioresample ! audio/x-raw,rate=48000,channels=2 ! alsasink \
 demux0.video_0 ! queue ! acmh264dec ! acmfbdevsink device=/dev/fb0
```

図 15.4 GStreamer の実行例

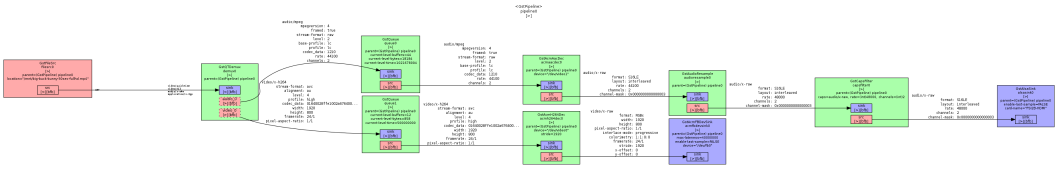


図 15.5 GStreamer のパイプライン例

拡大してご覧ください。

GStreamer のパイプラインは、シェルスクリプトのパイプ構文の構造に似ています。GStreamer の各エレメントとシェルスクリプト内のコマンドを対比することができます。構文的な違いとして、GStreamer のパイプラインは「! 」を使って各エレメントを繋ぎますが、シェルスクリプトは「| 」を使います。

シェルスクリプトで使うコマンドが引数を取るように、GStreamer のエレメントも引数を取ることができます。この引数は、「プロパティ」と呼ばれます。

各エレメントは、データの入出力の口となる「パッド(pad)」を持っています。実際にエレメント同士を繋いでいるのはパッドです。パッドにはデータを次のエレメントに渡す「ソースパッド(source pad)」とデータを受け取る「シンクパッド(sink pad)」が存在します。図中の「src 」と書かれた小さな箱と「sink 」と書かれた小さな箱がそれぞれに該当します。

パッドは自分が出力可能な、または入力可能なフォーマットを知っています。これを「ケイパビリティー (Capability)」と言います。パッドは、パイプラインが作られる時に繋がる相手パッドが持っているケイパビリティーを確認します。お互いのケイパビリティーが一致しない場合はデータの受け渡しができませんので、エラーとなり、最終的にパイプライン生成自体がエラーとなります。

ソースパッドしか持たないエレメントを「ソースエレメント」、シンクパッドしか持たないエレメントを「シンクエレメント」と呼びます。図中左端にある「filesrc0」がソースエレメントで、右端の「alsasink0」「acmfbdevsink」の 2 つがシンクエレメントになります。マルチメディアデータは、一番左側のソースエレメントから右端のシンクエレメントに流れることで形を変えていき、最終的に動画や音声として再生されることになります。

上記例は、GStreamer のデバッグ/プロトタイピング用のコマンドラインツールである gst-launch-1.0 を使って説明しましたが、GStreamer はライブラリとして提供されているため、GStreamer を使ったマルチメディア機能を自作のアプリケーションプログラムに組み込むことができます。API やアプリケーション開発マニュアルは、gstreamer.freedesktop.org の Documentation ページ [http://gstreamer.freedesktop.org/documentation/]から参照することができます。

AV コーデックミドルウェア用のエレメントには、下記のものがあります[1]。以降の章では、これらのエレメントの使い方を中心に説明します。

- H.264 デコーダー: acmh264dec
- AAC デコーダー: acmaacdec
- H.264 エンコーダー: acmh264enc
- AAC エンコーダー: acmaacenc

[1]Armadillo-800 シリーズ用の環境では、NEON 対応した libjpeg turbo が導入されています。libjpeg turbo を使うことで十分に高速な JPEG デコードを行うことができるため、AV コーデックミドルウェアには JPEG デコーダーは含まれていません。

・ JPEG エンコーダー: acmjpegenc
・ フレームバッファ用シンクエレメント: acmfbdevsink

環境にインストールされているエレメント一覧を取得したり、各エレメントの取れるケイパビリティや指定可能なプロパティは gst-inspect-1.0 コマンドを使うことで調べることができます。

```
[armadillo ~]# gst-inspect-1.0
acmaacdec:  acmaacdec: ACM AAC audio decoder
acmh264enc:  acmh264enc: ACM H264 video encoder
acmfbdevsink:  acmfbdevsink: ACM fbdev video sink
acmh264dec:  acmh264dec: ACM H264 video decoder
acmaacenc:  acmaacenc: ACM AAC audio encoder
acmjpegenc:  acmjpegenc: ACM Jpeg encoder
video4linux2:  v4l2radio: Radio (video4linux2) Tuner
video4linux2:  v4l2sink: Video (video4linux2) Sink
video4linux2:  v4l2src: Video (video4linux2) Source
fbdevsink:  fbdevsink: fbdev video sink
udp:  udpsink: UDP packet sender
udp:  multiudpsink: UDP packet sender
udp:  dynudpsink: UDP packet sender
udp:  udpsrc: UDP packet receiver
(省略)
```

図 15.6 エレメント一覧の取得

```
[armadillo ~]# gst-inspect-1.0 acmh264dec
Factory Details:
  Rank:         primary (256)
  Long-name:            ACM H264 video decoder
  Klass:                Codec/Decoder/Video
  Description:          ACM H.264/AVC decoder
  Author:               Atmark Techno, Inc.

Plugin Details:
  Name:                 acmh264dec
  Description:          ACM H264 Decoder
  Filename:             /usr/lib/gstreamer-1.0/libgstacmh264dec.so
  Version:              1.0.0
  License:              LGPL
  Source module:        gst-plugins-acm
  Binary package:       GStreamer ACM Plugins
  Origin URL:           http://armadillo.atmark-techno.com/

GObject
 +----GInitiallyUnowned
       +----GstObject
             +----GstElement
                   +----GstVideoDecoder
                         +----GstAcmH264Dec

Pad Templates:
  SINK template: 'sink'
    Availability: Always
    Capabilities:
      video/x-h264
```

```
          stream-format: avc
              alignment: au
                  width: [ 80, 1920 ]
                 height: [ 80, 1080 ]
              framerate: [ 0/1, 2147483647/1 ]

  SRC template: 'src'
    Availability: Always
    Capabilities:
      video/x-raw
                  format: RGB16
                   width: [ 80, 1920 ]
                  height: [ 80, 1080 ]
              framerate: [ 0/1, 2147483647/1 ]
      video/x-raw
                  format: RGB
                   width: [ 80, 1920 ]
                  height: [ 80, 1080 ]
              framerate: [ 0/1, 2147483647/1 ]
      video/x-raw
                  format: RGBx
                   width: [ 80, 1920 ]
                  height: [ 80, 1080 ]
              framerate: [ 0/1, 2147483647/1 ]
      video/x-raw
                  format: NV12
                   width: [ 80, 1920 ]
                  height: [ 80, 1080 ]
              framerate: [ 0/1, 2147483647/1 ]


Element Flags:
  no flags set

Element Implementation:
  Has change_state() function: gst_video_decoder_change_state

Element has no clocking capabilities.
Element has no indexing capabilities.
Element has no URI handling capabilities.

Pads:
  SINK: 'sink'
    Implementation:
      Has chainfunc(): gst_video_decoder_chain
      Has custom eventfunc(): gst_video_decoder_sink_event
      Has custom queryfunc(): gst_video_decoder_sink_query
      Has custom iterintlinkfunc(): gst_pad_iterate_internal_links_default
    Pad Template: 'sink'
  SRC: 'src'
    Implementation:
      Has custom eventfunc(): gst_video_decoder_src_event
      Has custom queryfunc(): gst_video_decoder_src_query
      Has custom iterintlinkfunc(): gst_pad_iterate_internal_links_default
    Pad Template: 'src'

Element Properties:
  name                : The name of the object
```

```
                        flags: readable, writable
                        String. Default: "acmh264dec0"
  parent              : The parent of the object
                        flags: readable, writable
                        Object of type "GstObject"
  device              : The video device eg: /dev/video0
                        flags: readable, writable
                        String. Default: null
  stride              : Stride of output video. (0 is unspecified)
                        flags: readable, writable
                        Unsigned Integer. Range: 0 - 65535 Default: 0
  x-offset            : X Offset of output video. (0 is unspecified)
                        flags: readable, writable
                        Unsigned Integer. Range: 0 - 65535 Default: 0
  y-offset            : Y Offset of output video. (0 is unspecified)
                        flags: readable, writable
                        Unsigned Integer. Range: 0 - 65535 Default: 0
  buf-pic-cnt         : Number of buffering picture
                        flags: readable, writable
                        Unsigned Integer. Range: 2 - 145 Default: 17
  enable-vio6         : FALSE: disable, TRUE: enable
                        flags: readable, writable
                        Boolean. Default: true
```

図 15.7 エレメント情報の取得

15.4. 有効化/無効化

AV コーデックミドルウェアは、SH で動作するファームウェアと、Linux 上で動作するデバイスドライバが協調して機能します。そのため、AV コーデックミドルウェアを有効化するには、SH にファームウェアをロードさせた後、ドライバをロードする必要があります。

SH ファームウェアのロードとドライバのロードを行うには、/sys/devices/platform/acm.0/codec に"encoder"または"decoder"という文字列を書き込みます。AV コーデックミドルウェアを無効化する場合は、"none"という文字列を書き込みます。

```
[armadillo ~]# echo encoder > /sys/devices/platform/acm.0/codec
acm_h264enc: H.264 Encoder of AV Codec Middleware
acm_aacenc: AAC Encoder of AV Coenc Middleware
acm_jpegenc: JPEG Encoder of AV Codec Middleware
```

図 15.8 AV コーデックミドルウェアの有効化(エンコーダー)

```
[armadillo ~]# echo decoder > /sys/devices/platform/acm.0/codec
acm_h264dec: H.264 Decoder of AV Codec Middleware
acm_aacdec: AAC Decoder of AV Codec Middleware
```

図 15.9 AV コーデックミドルウェアの有効化(デコーダー)

```
[armadillo ~]# echo none > /sys/devices/platform/acm.0/codec
```

図 15.10 AV コーデックミドルウェアの無効化

標準状態では、起動時に/etc/config/rc.local で AV コーデックミドルウェアを有効化しています。Armadillo-810 の場合エンコーダーが、Armadillo-840 の場合はデコーダーが有効になります。詳細は「9.1.4. /etc/config/rc.local」を参照してください。

AV コーデックミドルウェアの現在の状態は、/sys/devices/platform/acm.0/codec を読み出すことで確認できます。

```
[armadillo ~]# cat /sys/devices/platform/acm.0/codec
decoder [encoder] none
```

図 15.11 AV コーデックミドルウェアの状態確認(エンコーダーが有効化されている場合)

```
[armadillo ~]# cat /sys/devices/platform/acm.0/codec
[decoder] encoder none
```

図 15.12 AV コーデックミドルウェアの状態確認(デコーダーが有効化されている場合)

```
[armadillo ~]# cat /sys/devices/platform/acm.0/codec
decoder encoder [none]
```

図 15.13 AV コーデックミドルウェアの状態確認(無効化されている場合)

15.5. エンコード

15.5.1. コンテナの扱い

ビデオ(映像)とオーディオ(音声)データを一つのファイルにまとめる(多重化する)ためのフォーマットをコンテナフォーマットと言います。H.264/AVC と AAC を格納可能なコンテナフォーマットには、MP4(MPEG-4 Part 14)、AVI(Audio Video Interface)、Matroska などがあります。また、MPEG2 TS(MPEG-2 Transport Stream)や MPEG2 PS(MPEG-2 Program Stream)は、拡張規格で H.264/AVC と AAC に対応しています。

GStreamer では、ビデオやオーディオのデータをコンテナに格納する場合、マルチプレクサエレメントを使用します。

ビデオのみを MP4 コンテナに格納するパイプラインの例を「図 15.14. ビデオをエンコードしてコンテナに格納する」に示します。MP4 コンテナに格納する場合、qtmux エレメントを使用します。

```
[armadillo ~]# gst-launch-1.0 -e videotestsrc \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmh264enc ! queue \
 ! qtmux ! filesink location=/mnt/output.mp4
```

図 15.14 ビデオをエンコードしてコンテナに格納する

画像の入力ソースとして、様々なパターンの画像を様々なフォーマットで生成できる videotestsrc を使用しています。acmh264enc エレメントは、入力フォーマットとして NV12 形式のみを受付ますので、acmh264enc の入力側のケイパビリティを指定しています[2]。videotestsrc は指定されたフォーマットで画像を出力します。

また、パイプライン停止時に EOS イベントを発行するように、gst-launch-1.0 コマンドに-e オプションを付けています。エンコードを終了するには、Ctrl-C で gst-launch-1.0 コマンドを停止してください。

オーディオのみを MP4 コンテナに格納するパイプラインの例を「図 15.15. オーディオをエンコードしてコンテナに格納する」に示します。ここでも、様々なパターンのオーディオデータを生成できる audiotestsrc を使用しています。audiotestsrc が出力するフォーマットと、acmaacenc が受け付けられるフォーマットを合わせるため、ケイパビリティを指定しているのはビデオの場合と同じです。

```
[armadillo ~]# gst-launch-1.0 -e audiotestsrc wave=sine freq=1000 \
 ! audio/x-raw,format=S16LE,layout=interleaved,rate=48000,channels=2 \
 ! acmaacenc ! queue \
 ! qtmux ! filesink location=/mnt/output.mp4
```

図 15.15 オーディオをエンコードしてコンテナに格納する

ビデオとオーディオを同時に MP4 コンテナに格納するパイプラインの例を「図 15.16. ビデオとオーディオをエンコードしてコンテナに格納する」に示します。

```
[armadillo ~]# gst-launch-1.0 -e videotestsrc \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmh264enc ! queue ! muxer.video_0 \
 audiotestsrc wave=sine freq=1000 \
 ! audio/x-raw,format=S16LE,layout=interleaved,rate=48000,channels=2 \
 ! acmaacenc ! queue ! muxer.audio_0 \
 qtmux name="muxer" ! filesink location=/mnt/output.mp4
```

図 15.16 ビデオとオーディオをエンコードしてコンテナに格納する

qtmux エレメントは、入力されるビデオとオーディオそれぞれに対して、動的にシンクパッドを作成します。ビデオ用に動的に生成されたパッドには video_*N* という名前が、オーディオ用に生成されたパッドには audio_*N* という名前がつきます。

パッド名は省略して、下記のように記述することもできます。GStreamer では、エレメント同士を接続(リンク)する際に、お互いのソースパッド(出力)とシンクパッド(入力)が受け渡しできるフォーマット(ケイパビリティ)が一致するかネゴシエーションを行い、ケイパビリティが一致した場合だけパッドがリンクされます。queue エレメントはバッファリングを行うためのエレメントで、どのようなフォーマットのデータも受け渡しでき、シンクパッドとソースパッドのケイパビリティは同じものになります。そのため、qtmux が動的に生成したオーディオ用のパッドにはオーディオを扱う acmaacenc にリンクされている queue エレメントのソースパッドが、ビデオ用のパッドにはビデオを扱う acmh264enc エレメントがリンクされている queue エレメントのソースシンクパッドがリンクされます。

```
[armadillo ~]# gst-launch-1.0 -e videotestsrc \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmh264enc ! queue ! muxer. \
```

[2]エレメントが受け付けられるフォーマットは、gst-inspect-1.0 を実行したときの"SINK template"の"Capabilities"で確認できます。

```
audiotestsrc wave=sine freq=1000 \
! audio/x-raw,format=S16LE,layout=interleaved,rate=48000,channels=2 \
! acmaacenc ! queue ! muxer. \
qtmux name="muxer" ! filesink location=/mnt/output.mp4
```

図 15.17 ビデオとオーディオをエンコードしてコンテナに格納する(パッド名の省略)

15.5.2. ビデオのエンコード

15.5.2.1. 入力ソースを指定する

v4l2src エレメントを使うことで、V4L2(Video for Linux 2)デバイスとして実装されているカメラデバイスから画像を取得できます。どのデバイスから画像を取得するかは、v4l2src エレメントの device プロパティにデバイスファイル名を指定することで変更できます。Armadillo-810 カメラモデルの場合、Armadillo-810 カメラモジュール 01 (B コネクタ用)のデバイスファイルは/dev/video1 となります。

```
[armadillo ~]# gst-launch-1.0 -e v4l2src device="/dev/video1" \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmh264enc ! queue \
 ! qtmux ! filesink location=/mnt/output.mp4
```

図 15.18 カメラモジュールからの入力画像をエンコードする

UVC 対応 USB カメラなども同様に v4l2src で扱うことができます。どのデバイスファイルが、どのカメラに対応しているかは/sys/class/video4linux/video*N*/name で確認できます。Armadillo-810 カメラモジュールなど、R-Mobile A1 の CEU に接続されたカメラの場合、"sh_mobile_ceu.*N*"と表示されます。UVC 対応 USB カメラの場合、"USB Camera"などと表示されます。

```
[armadillo ~]# cat /sys/class/video4linux/video0/name
sh_mobile_ceu.0
[armadillo ~]# cat /sys/class/video4linux/video4/name
USB Camera
```

図 15.19 どのデバイスファイルがどのカメラに対応しているか確認する

USB カメラによっては NV12 フォーマットで出力できない場合もあります。そのような場合は、videoconvert エレメントを使うことで画像フォーマットを変更できます。

```
[armadillo ~]# gst-launch-1.0 -e v4l2src device="/dev/video4" \
 ! videoconvert ! video/x-raw,format=NV12,framerate=30/1 \
 ! acmh264enc ! queue \
 ! qtmux ! filesink location=/mnt/output.mp4
```

図 15.20 USB カメラからの入力画像をエンコードする

15.5.2.2. フレームレートを指定する

Armadillo-810 カメラモジュール 01 (B コネクタ用)からの入力画像は、30fps 固定となっています。しかし、そこまでフレームレートが必要ない場合、もっと低いフレームレートでエンコードすることもできます。フレームレートの変更には、videorate エレメントを使います。「図 15.21. フレームレートを指定する」のように指定すると、15fps でエンコードを行います。

```
[armadillo ~]# gst-launch-1.0 -e v4l2src device="/dev/video1" \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! videorate ! video/x-raw,framerate=15/1 \
 ! acmh264enc ! queue \
 ! qtmux ! filesink location=/mnt/output.mp4
```

図 15.21 フレームレートを指定する

15.5.2.3. エンコード品質を指定する

acmh264enc エレメントのプロパティを指定することで、エンコード品質を調整できます。エンコード品質に影響する acmh264enc エレメントのプロパティを下記に示します。

表 15.5 エンコード品質に影響する acmh264enc エレメントのプロパティ

プロパティ	意味	最小値	最大値	デフォルト値
bitrate	目標平均ビットレート[bit/sec]	16,000	40,000,000	8,000,000
max-frame-size	最大フレームサイズ[a][byte]	0	5,000,000	0(推奨値[b]を使用)
rate-control-mode	・ 0: 固定ビットレート(ピクチャスキップあり) ・ 1: 固定ビットレート(ピクチャスキップなし) ・ 2: 可変ビットレート(ピクチャスキップなし)	0	2	2
max-gop-length	最大 GOP 長 ・ 0: ストリームの先頭のみ I ピクチャ ・ 1: 全て I ピクチャ ・ 2 以上: 指定された GOP 長で GOP を生成	0	120	30
b-pic-mode	B ピクチャモード ・ 0: B ピクチャを使用しない ・ 1〜3: B ピクチャを N フレーム挿入[c]	0	3	3

[a]0 以外を指定する場合は、目標平均ビットレート/8[byte]以上を指定すること
[b]固定ビットレート制御時: 目標平均ビットレート/3*4/8[byte]、可変ビットレート制御時: 目標平均ビットレート*2/8[byte]
[c]B ピクチャを挿入する場合、rate-control-mode=0(ピクチャスキップあり)は指定不可

15.5.2.4. 入力画像サイズの制限とオフセットの指定

acmh264enc エレメントへの入力画像サイズには、下記の制限があります。

・ 入力画像幅: 80〜1920 画素(2 の倍数であること)
・ 入力画像高さ: 80〜1088 ライン(2 の倍数であること)
・ 入力画像幅 × 入力画像高さは 32 の倍数であること

また、acmh264enc エレメントの x-offset プロパティと y-offset プロパティを使うことで、入力画像の一部だけをエンコードできます。x-offset と y-offset プロパティには下記の制限があります。

・ [入力画像幅]×[y-offset]+[x-offset]は 32 の倍数であること
・ [入力画像幅]×[y-offset]/2+[x-offset]は 32 の倍数であること

「図 15.22. オフセットを指定する」のように指定すると、640×480 サイズの入力画像のうち、中央の 320×240 サイズだけエンコードを行います。acmh264enc エレメントの出力サイズを設定するために、出力側のケイパビリティも指定している点に注意してください。

```
[armadillo ~]# gst-launch-1.0 -e videotestsrc \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
```

```
 ! acmh264enc x-offset=160 y-offset=120 ! video/x-h264,width=320,height=240 ! queue \
 ! qtmux name="muxer" ! filesink location=output.mp4
```

図 15.22 オフセットを指定する

15.5.3. オーディオのエンコード

15.5.3.1. 入力ソースを指定する

alsasrc エレメントを使うことで、ALSA(Advanced Linux Sound Architecture)デバイスとして実装されているオーディオデバイスから録音できます。どのデバイスから音声を取得するかは、alsasrc エレメントの device プロパティに ALSA デバイス名を指定することで変更できます。Armadillo-840 液晶モデルの場合、Armadillo-840 拡張ボード 01 (C コネクタ用)のマイク入力インターフェース(CON5)に対応する ALSA デバイスは hw:1 となります。

```
[armadillo ~]# gst-launch-1.0 -e alsasrc device="hw:1" \
 ! audio/x-raw,format=S16LE,layout=interleaved,rate=48000,channels=2 \
 ! acmaacenc ! queue \
 ! qtmux ! filesink location=/mnt/output.mp4
```

図 15.23 マイク入力インターフェースからの入力音声をエンコードする

音声入力となることができる ALSA デバイスの一覧は、 arecord -l コマンドで調べられます。card *N* の番号を、alsasrc の device プロパティに指定してください。

```
[armadillo ~]# arecord -l
**** List of CAPTURE Hardware Devices ****
card 1: FSI2AWM8978 [FSI2A-WM8978], device 0: wm8978 wm8978-hifi-0 []
  Subdevices: 1/1
  Subdevice #0: subdevice #0
card 2: Camera [USB Camera], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
```

図 15.24 ALSA 入力デバイスの一覧表示

15.5.3.2. エンコード品質を指定する

acmaacenc エレメントのプロパティを指定することで、エンコード品質を調整できます。エンコード品質に影響する acmaacenc エレメントのプロパティを下記に示します。

表 15.6 エンコード品質に影響する acmaacenc エレメントのプロパティ

プロパティ	意味	最小値	最大値	デフォルト値
bitrate	目標平均ビットレート[bit/sec]	16,000	288,000	64,000
enable-cbr	固定ビットレート有効 ・ 0: 可変ビットレート ・ 1: 固定ビットレート	0	1	0

15.5.4. JPEG のエンコード

15.5.4.1. JPEG をファイルに保存する

Armadillo-810 カメラモジュール 01 (B コネクタ用)のカメラデバイスから取得した画像を JPEG として保存する場合、下記コマンドを実行します。10 フレームの画像を取得し、output0.jpeg 〜 output9.jpeg の 10 枚の JPEG 画像が生成されます。画像取得開始からカメラの AGC(Auto Gain Control)、AWB(Auto White Balance)の効果が十分に反映されるまでに 6 フレーム程度かかりますので、output0.jpeg 〜 output5.jpeg は色合い、明るさが激しく変化します。

最後の 1 フレームだけ取得したい場合は、 multifilesink エレメントに max-files=1 プロパティを指定してください。

```
[armadillo ~]# gst-launch-1.0 -e v4l2src device=/dev/video1 num-buffers=10 \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmjpegenc ! multifilesink location=/mnt/output%d.jpeg
```

図 15.25 カメラモジュールからの入力画像をエンコードする

下記コマンドを実行すると、JPEG エンコードした画像を Motion JPEG として mov ファイルに格納することができます。

```
[armadillo ~]# gst-launch-1.0 -e v4l2src device=/dev/video1 \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmjpegenc ! qtmux ! filesink location=/mnt/output.mov
```

図 15.26 Motion JPEG としてファイルに保存する

15.5.4.2. エンコード品質を指定する

acmjpegenc エレメントの quality プロパティを指定することで、エンコード品質を調整できます。0〜100 でエンコード品質を指定します。値が小さいほど画像が荒くなりますが、画像サイズは小さくなります。デフォルト値は 75 に設定されています。

15.5.4.3. 入出力画像サイズの制限とオフセットの指定

acmjpegenc エレメントへの入出力画像サイズには、下記の制限があります。

- 入力画像幅: 16〜1920 画素(8 の倍数であること)
- 入力画像高さ: 16〜1072 ライン(16 の倍数であること)
- 出力画像幅: 16〜1920 画素(4 の倍数であること)
- 出力画像高さ: 16〜1072 ライン(4 の倍数であること)

また、acmjpegenc エレメントの x-offset プロパティと y-offset プロパティを使うことで、入力画像の一部だけをエンコードできます。x-offset と y-offset プロパティには、下記の制限があります。

- [入力画像高さ]-[y-offset]が 16 の倍数であること

「図 15.27. オフセットを指定する」のように指定すると、640×480 サイズの入力画像のうち、中央の 320×240 サイズだけエンコードを行います。ajpegenc エレメントの出力サイズを設定するために、出力側のケイパビリティも指定している点に注意してください。

```
[armadillo ~]# gst-launch-1.0 -e videotestsrc num-buffers=10 \
 ! video/x-raw,format=NV12,width=640,height=480,framerate=30/1 \
 ! acmjpegenc x-offset=160 y-offset=112 ! image/jpeg,width=320,height=240 \
 ! multifilesink location=/mnt/output%d.jpeg
```

図 15.27 オフセットを指定する

16. SD ブートの活用

本章では、SD ブートをおこなうためのブートディスクの作成方法や、ブートディスクにルートファイルシステムを構築する方法など、SD ブートを活用するために必要な情報について説明します。SD ブートとは、SD カードに保存されたブートローダーイメージを起動させることを示します。

開発時に SD ブートを利用すると、以下のようなメリットがあります。

- フラッシュメモリのブートローダーを復旧することができる
- フラッシュメモリに収まらないサイズのソフトウェアを動作させることができる
- SD カードを取り替えるだけでシステムイメージを変更することができる

SD ブートをおこなった場合でも、ブートローダーの設定(保守モードの setenv/setboodevice コマンドで設定する項目)についてはフラッシュメモリに保存されます。

SD カードに対する作業は、ATDE で行います。そのため、ATDE に SD カードを接続する必要があります。詳しくは「4.2.2. 取り外し可能デバイスの使用」を参照してください。

ATDE に SD カードを接続すると、自動的に/media/ディレクトリにマウントされます。本章に記載されている手順を実行するためには、次のように SD カードをアンマウントしておく必要があります。

```
[ATDE ~]$ mount
(省略)
/dev/sdb1 on /media/52E6-5897 type vfat
(rw,nosuid,nodev,relatime,uid=1000,gid=1000,fmask=0022,dmask=0077,codepage=cp437,iocharset=utf8,sh
ortname=mixed,showexec,utf8,flush,errors=remount-ro,uhelper=udisks)
[ATDE ~]$ sudo umount /dev/sdb1
[ATDE ~]$
```

図 16.1 自動マウントされた SD カードのアンマウント

本章で使用するブートローダーイメージファイルなどは、開発セット付属の DVD に収録されています。最新版のファイルは、"Armadillo サイト"でダウンロードすることができます。新機能の追加や不具合の修正などが行われているため、DVD に収録されているものよりも新しいバージョンがリリースされているかを確認して、最新バージョンのソースコードを利用することを推奨します。

Armadillo サイト - Armadillo-810 ドキュメント・ダウンロード

http://armadillo.atmark-techno.com/armadillo-810/downloads

16.1. ブートディスクの作成

ATDE でブートディスクを作成します。ブートディスクの作成に使用するファイルを次に示します。

表 16.1 ブートディスクの作成に使用するファイル

ファイル	ファイル名
SD ブート用ブートローダーイメージ	loader-armadillo810-mmcsd-*[version]*.bin

SD カードにブートローダーイメージを配置する際、「表 16.2. ブートディスクの制約」に示す制約があります。本章に示す手順を実行した場合は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。

表 16.2 ブートディスクの制約

項目	制約
パーティション番号	1
パーティションのシステムタイプ	0xb(Win95 FAT32)
ファイルシステム	FAT32
ブートローダーイメージファイル名	sdboot.bin
ブートローダーイメージファイルの配置場所	ルートディレクトリ直下

「表 16.3. ブートディスクの構成例」に示すブートディスクを作成する手順を、「手順 16.1. ブートディスクの作成例」に示します。

表 16.3 ブートディスクの構成例

パーティション番号	パーティションサイズ	ファイルシステム	説明
1	128MByte	FAT32	SD ブート用のブートローダーイメージを配置します。
2	残り全て	ext3	ルートファイルシステムを構築するために ext3 ファイルシステムを構築しておきます。

手順 16.1 ブートディスクの作成例

1. SD ブート用のブートローダーイメージファイルを取得します。

```
[ATDE ~]$ ls
loader-armadillo810-mmcsd-v3.2.0.bin
```

フラッシュメモリ用のブートローダーイメージを SD カードに配置しても起動することができません。事前にファイル名を確認してください。ブートローダーイメージファイルには以下 2 種類があります。

格納場所	イメージファイル
SD カード	loader-armadillo810-mmcsd-[version].bin
フラッシュメモリ	loader-armadillo810-nor-[version].bin

2. 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。

```
[ATDE ~]$ sudo fdisk /dev/sdb ❶

Command (m for help): o ❷
Building a new DOS disklabel with disk identifier 0x8cb9edcc.
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)

Command (m for help): n ❸
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): ❹
Using default response p
Partition number (1-4, default 1): ❺
Using default value 1
First sector (2048-3862527, default 2048): ❻
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-3862527, default 3862527): +128M ❼

Command (m for help): n ❽
Partition type:
   p   primary (1 primary, 0 extended, 3 free)
   e   extended
Select (default p): ❾
Using default response p
Partition number (1-4, default 2): ❿
Using default value 2
First sector (264192-3862527, default 264192): ⓫
Using default value 264192
Last sector, +sectors or +size{K,M,G} (264192-3862527, default 3862527): ⓬
Using default value 3862527

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

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

Calling ioctl() to re-read partition table.

WARNING: If you have created or modified any DOS 6.x
partitions, please see the fdisk manual page for additional
information.
Syncing disks.
[ATDE ~]$
```

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。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。

❷　新しく空の DOS パーティションテーブルを作成します。

❸　新しくパーティションを追加します。

❹ パーティション種別にはデフォルト値(p: プライマリ)を指定するので、そのまま改行を入力してください。

❺ パーティション番号にはデフォルト値(1)を指定するので、そのまま改行を入力してください。

❻ 開始セクタにはデフォルト値(使用可能なセクタの先頭)を使用するので、そのまま改行を入力してください。

❼ 最終シリンダは、128MByte 分を指定します。

❽ 新しくパーティションを追加します。

❾ パーティション種別にはデフォルト値(p: プライマリ)を指定するので、そのまま改行を入力してください。

❿ パーティション番号にはデフォルト値(2)を指定するので、そのまま改行を入力してください。

⓫ 開始セクタにはデフォルト値(第 1 パーティションの最終セクタの次のセクタ)を使用するので、そのまま改行を入力してください。

⓬ 最終セクタにはデフォルト値(末尾セクタ)を使用するので、そのまま改行を入力してください。

⓭ パーティションのシステムタイプを変更します。

⓮ 第 1 パーティションを指定します。

⓯ パーティションのシステムタイプに 0xb(Win95 FAT32)を指定します。

⓰ 変更を SD カードに書き込みます。

3. パーティションリストを表示し、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。

```
[ATDE ~]$ sudo fdisk -l /dev/sdb

Disk /dev/sdb: 1977 MB, 1977614336 bytes
61 heads, 62 sectors/track, 1021 cylinders, total 3862528 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x8cb9edcc

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            2048      264191      131072    b  W95 FAT32
/dev/sdb2          264192     3862527     1799168   83  Linux
```

4. 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。

```
[ATDE ~]$ sudo mkfs.vfat -F 32 /dev/sdb1  ❶
mkfs.vfat 3.0.13 (30 Jun 2012)
[ATDE ~]$ sudo mkfs.ext3 /dev/sdb2  ❷
mke2fs 1.42.5 (29-Jul-2012)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
```

```
Stride=0 blocks, Stripe width=0 blocks
112448 inodes, 449792 blocks
22489 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=461373440
14 block groups
32768 blocks per group, 32768 fragments per group
8032 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912

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

[ATDE ~]$
```

❶ 第 1 パーティションに FAT32 ファイルシステムを構築します。

❷ 第 2 パーティションに ext3 ファイルシステムを構築します。

5. SD ブート用のブートローダーイメージファイルを第 1 パーティションに配置します。

```
[ATDE ~]$ ls
loader-armadillo810-mmcsd-v3.2.0.bin
[ATDE ~]$ mkdir sd ❶
[ATDE ~]$ sudo mount -t vfat /dev/sdb1 sd ❷
[ATDE ~]$ sudo cp loader-armadillo810-mmcsd-v3.2.0.bin sd/sdboot.bin ❸
[ATDE ~]$ sudo umount sd ❹
[ATDE ~]$ rm -r sd ❺
```

❶ SD カードをマウントするための sd/ディレクトリを作成します。

❷ 第 1 パーティションを sd/ディレクトリにマウントします。

❸ sd/ディレクトリにブートローダーイメージをコピーします。ファイル名"sdboot.bin" にリネームする必要があります。

❹ sd/ディレクトリにマウントした第 1 パーティションをアンマウントします。

❺ sd/ディレクトリを削除します。

アンマウントが完了する前に SD カードを作業用 PC から取り外すと、SD カードのデータが破損する場合があります。

16.2. ルートファイルシステムの構築

「16.1. ブートディスクの作成」で作成したブートディスクにルートファイルシステムを構築します。

Atmark Dist または Debian GNU/Linux のルートファイルシステムを構築することができます。ルートファイルシステムの構築に使用するファイルを次に示します。

表 16.4 ルートファイルシステムの構築に使用するファイル

Linux ディストリビューション	ファイル名	ファイルの説明
Atmark Dist	romfs-a810-*[version]*.img.gz	Atmark Dist で作成したユーザーランドイメージ
Debian GNU/Linux	debian-wheezy-armhf_a810_*[version]*.tar.gz	ARM(armhf)アーキテクチャ用 Debian GNU/Linux 7.0 (コードネーム 「wheezy」)のルートファイルシステムアーカイブ

ブートディスクに構築した Atmark Dist ルートファイルシステムからでも、netflash を使用してフラッシュメモリを書き替えることができます。開発時にはフラッシュメモリの復旧用として準備しておくことを推奨します。

16.2.1. Atmark Dist を構築する

Atmark Dist で作成したユーザーランドイメージから、ルートファイルシステムを構築する手順を次に示します。

手順 16.2 Atmark Dist イメージからルートファイルシステムを構築する

1. Atmark Dist で作成したユーザーランドイメージファイル(romfs-a810-v1.00.img.gz)を準備しておきます。

```
[ATDE ~]$ ls
romfs-a810-v1.00.img.gz
```

2. ユーザーランドイメージファイルをマウントします。

```
[ATDE ~]$ mkdir romfs ❶
[ATDE ~]$ gzip -c -d romfs-a810-v1.00.img.gz > romfs-a810-v1.00.img ❷
[ATDE ~]$ ls
romfs  romfs-a810-v1.00.img  romfs-a810-v1.00.img.gz
[ATDE ~]$ sudo mount -o loop romfs-a810-v1.00.img romfs ❸
[ATDE ~]$ ls romfs ❹
bin   dev  home  linuxrc     media  opt   root  sys  usr
boot  etc  lib   lost+found  mnt    proc  sbin  tmp  var
```

❶ ユーザーランドイメージファイルをマウントするための romfs/ディレクトリを作成します。

❷ gzip 形式で圧縮されているユーザーランドイメージファイルを伸長します。

❸ ユーザーランドイメージファイルを romfs/ディレクトリにマウントします。"-o"オプションで"loop"を指定する必要があります。

❹ マウントに成功し、ルートファイルシステムが見えるようになったことを確認します。

イメージファイルをマウントするには

Atmark Dist で作成したユーザーランドイメージファイルのマウントには「loop デバイス」を使用します。loop デバイスを使用すると、イメージファイルをブロック型デバイスとして扱うことができます。loop デバイスを使用したマウントを行うためには、mount コマンドの"-o"オプションで"loop"を指定する必要があります。

3. ルートファイルシステムをブートディスクの第 2 パーティションに構築します。

```
[ATDE ~]$ mkdir sd ❶
[ATDE ~]$ sudo mount -t ext3 /dev/sdb2 sd ❷
[ATDE ~]$ sudo cp -a romfs/* sd ❸
[ATDE ~]$ sudo umount romfs ❹
[ATDE ~]$ rm -r romfs ❺
```

❶ SD カードをマウントするための sd/ディレクトリを作成します。

❷ 第 2 パーティションを sd/ディレクトリにマウントします。

❸ romfs/ディレクトリから sd/ディレクトリにルートファイルシステムをコピーします。

❹ romfs/ディレクトリにマウントしたユーザーランドイメージファイルをアンマウントします。

❺ romfs/ディレクトリを削除します。

4. ユーザーランドイメージファイルの/etc/fstab はフラッシュメモリ用の設定になっているため、SD カード用の設定に変更します。

```
[ATDE ~]$ sudo vi sd/etc/fstab
/dev/mmcblk0p2          /              ext3    defaults              0 1  ❶
proc                    /proc          proc    defaults              0 0
usbfs                   /proc/bus/usb  usbfs   defaults              0 0
sysfs                   /sys           sysfs   defaults              0 0
udev                    /dev           tmpfs   mode=0755             0 0
/dev/flashblk/firmware  /opt/firmware  squashfs        defaults       0 0
/dev/flashblk/license   /opt/license   squashfs        defaults       0 0
[ATDE ~]$ sudo umount sd ❷
[ATDE ~]$ rm -r sd ❸
```

❶ "/dev/ram0"を"/dev/mmcblk0p2"に、"ext2"を"ext3"に変更します。

❷ sd/ディレクトリにマウントしたブートディスクの第 2 パーティションをアンマウントします。

❸ sd/ディレクトリを削除します。

アンマウントが完了する前に SD カードを作業用 PC から取り外すと、SD カードのデータが破損する場合があります。

16.2.2. Debian GNU/Linux を構築する

Debian GNU/Linux ルートファイルシステムアーカイブから、ルートファイルシステムを構築する手順を次に示します。

手順 16.3 Debian GNU/Linux ルートファイルシステムアーカイブからルートファイルシステムを構築する

1. Debian GNU/Linux ルートファイルシステムアーカイブを準備しておきます。

```
[ATDE ~]$ ls
debian-wheezy-armhf_a810_20130116.tar.gz
```

2. ルートファイルシステムをブートディスクの第 2 パーティションに構築します。

```
[ATDE ~]$ mkdir sd ❶
[ATDE ~]$ sudo mount -t ext3 /dev/sdb2 sd ❷
[ATDE ~]$ sudo tar zxf debian-wheezy-armhf_a810_20130116.tar.gz -C sd ❸
[ATDE ~]$ sudo umount sd ❹
[ATDE ~]$ rm -r sd ❺
```

❶ SD カードをマウントするための sd/ディレクトリを作成します。

❷ 第 2 パーティションを sd/ディレクトリにマウントします。

❸ ルートファイルシステムアーカイブを sd/ディレクトリに展開します。

❹ sd/ディレクトリにマウントしたブートディスクの第 2 パーティションをアンマウントします。

❺ sd/ディレクトリを削除します。

アンマウントが完了する前に SD カードを作業用 PC から取り外すと、SD カードのデータが破損する場合があります。

16.3. Linux カーネルイメージの配置

「16.2.1. Atmark Dist を構築する」または、「16.2.2. Debian GNU/Linux を構築する」で作成したルートファイルシステムに Linux カーネルイメージを配置します。Linux カーネルイメージの配置に使用するファイルを次に示します。

表 16.5 ブートディスクの作成に使用するファイル

ファイル	ファイル名
Linux カーネルイメージ	linux-a810-*[version]*.bin.gz

SD カードに Linux カーネルイメージを配置する際は、次の条件を満たすようにしてください。この条件から外れた場合、ブートローダーが Linux カーネルイメージを検出することが出来なくなる場合があります。

表 16.6 ブートローダーが Linux カーネルを検出可能な条件

項目	条件
ファイルシステム	ext2 または ext3
圧縮形式	gzip 形式 または 非圧縮
Linux カーネルイメージファイル名(gzip 形式)	Image.gz, linux.gz, Image.bin.gz, linux.bin.gz のいずれか
Linux カーネルイメージファイル名(非圧縮)	Image, linux, Image.bin, linux.bin のいずれか
Linux カーネルイメージファイルの配置場所	/boot/ディレクトリ直下

Linux カーネルイメージをルートファイルシステムに配置する手順を次に示します。

手順 16.4 Linux カーネルイメージの配置例

1. Linux カーネルイメージを準備しておきます。

```
[ATDE ~]$ ls
linux-a810-v1.00.bin.gz
```

2. Linux カーネルイメージをブートディスクの第 2 パーティションに配置します。

```
[ATDE ~]$ mkdir sd ❶
[ATDE ~]$ sudo mount -t ext3 /dev/sdb2 sd ❷
[ATDE ~]$ sudo mkdir -p sd/boot ❸
[ATDE ~]$ sudo cp linux-a810-v1.00.bin.gz sd/boot/Image.bin.gz ❹
[ATDE ~]$ sudo umount sd ❺
[ATDE ~]$ rm -r sd ❻
```

❶ SD カードをマウントするための sd/ディレクトリを作成します。

❷ 第 2 パーティションを sd/ディレクトリにマウントします。

❸ Linux カーネルイメージを配置するための boot/ディレクトリを作成します。

❹ Linux カーネルイメージを sd/boot/ディレクトリにコピーします。

❺ sd/ディレクトリにマウントしたブートディスクの第 2 パーティションをアンマウントします。

❻ sd/ディレクトリを削除します。

アンマウントが完了する前に SD カードを作業用 PC から取り外すと、SD カードのデータが破損する場合があります。

16.4. SD ブートの実行

「16.1. 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する方法を説明します。

Armadillo に電源を投入する前に次の準備を行います。

1. SD スロット(CON5)にブートディスクを接続します。
2. ブートディスクのブートローダーイメージを起動させ(SD ブート)、ブートローダーの起動後に保守モードとなるように、Armadillo-810 拡張ボード 01 (A コネクタ用)の JP1 をショート、JP2 をオープン、JP3 をショートに設定してます。

準備の完了後、電源を投入すると SD ブートさせることができます。SD ブートに成功した場合は、「図 16.2. SD ブート時の起動メッセージ」のように起動メッセージが表示されます。起動デバイス(Armadillo-810/の後に表示される文字列)が"mmcsd"になっていることを確認してください。

```
Hermit-At v3.2.0 (Armadillo-810/mmcsd) compiled at 15:48:02, Feb 06 2013
hermit>
```

図 16.2 SD ブート時の起動メッセージ

「16.2. ルートファイルシステムの構築」で構築したルートファイルシステムで起動する場合は、「図 16.3. ルートファイルシステムの起動設定」のように setenv コマンドで Linux カーネル起動オプションを設定します。setenv コマンドの詳細については「10.3. ブートローダーの機能」を参照してください。

```
hermit> setenv console=ttySC2,115200 noinitrd rootwait root=/dev/mmcblk0p2 mem=384M
hermit> setenv
1: console=ttySC2,115200
2: noinitrd
3: rootwait
4: root=/dev/mmcblk0p2
5: mem=384M
```

図 16.3 ルートファイルシステムの起動設定

Linux カーネル起動オプションを出荷状態(Linux カーネル起動オプションが設定されていない状態)に戻すには、以下のようにコマンドを実行します。

```
hermit> clearenv
```

「16.3. Linux カーネルイメージの配置」で配置した Linux カーネルイメージで起動する場合は、保守モードで「図 16.4. Linux カーネルの起動設定」のように setbootdevice コマンドで Linux カーネルイ

メージを指定します。setbootdevice コマンドの詳細については「10.3.2. Linux カーネルイメージの指定方法」を参照してください。

```
hermit> setbootdevice mmcblk0p2
hermit> setbootdevice
bootdevice: mmcblk0p2
```

図 16.4 Linux カーネルの起動設定

起動デバイス設定を出荷状態(フラッシュメモリから起動)に戻すには、以下のようにコマンドを実行します。

```
hermit> setbootdevice flash
```

17. 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。JTAG ICE を使用するためには、Armadillo-810 拡張ボード 01 (A コネクタ用)が必要です。

17.1. 準備

17.1.1. JTAG ケーブルの接続

JTAG ICE のケーブルを、Armadillo-810 拡張ボード 01 (A コネクタ用)の JTAG インターフェースに接続します。信号配列などの JTAG インターフェースについての詳細は、「21.1.3.7. CON7 JTAG インターフェース」を参照してください。

17.1.2. ジャンパの設定

Armadillo-810 拡張ボード 01 (A コネクタ用)の JP2 をショートします。

17.2. 接続確認

「17.1. 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

17.3. 各種デバッガへの対応について

お使いのデバッガが Armadillo-810 に対応しているか等の情報につきましては、各メーカにお問い合わせください。

18. 顔認識ミドルウェア「FSE」

「FSE (Face Sensing Engine)」は、顔検出や特徴点抽出などの機能を持つ OKI (沖電気工業株式会社) 製の顔認識エンジンです。

アットマークテクノ ユーザーズサイトで購入製品登録を行うことで、FSE の主な機能を利用できる評価用デモアプリをダウンロードすることができます。

アットマークテクノ ユーザーズサイト

https://users.atmark-techno.com/

顔認識エンジン「FSE」の主な機能は次のとおりです。

1. 顔認識
2. 特徴点抽出
3. 類似度判定

顔認識エンジン「FSE」の詳しい情報は、Armadillo サイトをご覧ください。

Armadillo サイト - 顔認識エンジン「FSE」

http://armadillo.atmark-techno.com/software/fse

19. ハードウェア仕様

19.1. インターフェースレイアウト

Armadillo-810 のインターフェースレイアウトは次の通りです。

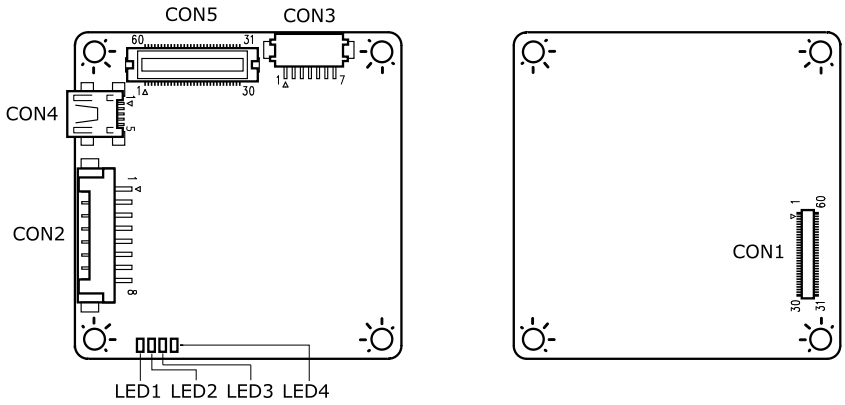


図 19.1 Armadillo-810 のインターフェースレイアウト図

表 19.1 Armadillo-810 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	拡張インターフェース 2 (B コネクタ)	BtoB コネクタ 60P(0.4mm ピッチ) DF40C-60DP-0.4V(51)/HIROSE ELECTRIC	対向コネクタ例: DF40HC(4.0)-60DS-0.4V(51)/HIROSE ELECTRIC 挿抜寿命:30 回
CON2	シリアルインターフェース 1	ピンヘッダ 8P(2mm ピッチ) DF3DZ-8P-2H(51)/HIROSE ELECTRIC	信号レベル: RS232C 対向コネクタ例: DF3-8S-2C/HIROSE ELECTRIC 挿抜寿命:50 回
CON3	シリアルインターフェース 2	ピンヘッダ 7P(1.25mm ピッチ) DF13A-7P-1.25H(51)/HIROSE ELECTRIC	信号レベル: 3.3V CMOS 対向コネクタ例: DF13-7S-1.25C/HIROSE ELECTRIC 挿抜寿命:50 回
CON4	USB インターフェース	USB mini B コネクタ 54819-0572/Molex[a]	USB2.0 Device(High Speed 対応)
CON5	拡張インターフェース 1 (A コネクタ)	BtoB コネクタ 60P(0.5mm ピッチ) DF17(4.0)-60DS-0.5V(57)/HIROSE ELECTRIC	対向コネクタ例: DF17(4.0)-60DP-0.5V(57)/HIROSE ELECTRIC 挿抜寿命:50 回
LED1～LED4	ユーザー LED	LED(黄色、面実装)	

[a]製品リビジョン Rev.3.2 以前は、UB-M5BR-G14-4S(LF)(SN)/J.S.T. Mfg.が搭載されています。

19.2. インターフェース仕様

19.2.1. CON1 拡張インターフェース 2 (B コネクタ)

CON1 は入出力拡張用のインターフェースです。用途によって機能を選択できるように複数の機能が割り当てられたピンが多数接続されています。主にカメラ用の信号が接続されています。

搭載コネクタ　　DF40C-60DP-0.4V(51)/HIROSE ELECTRIC

許容電流: 0.3A 以下(端子 1 本あたり)

対向コネクタ例　　DF40HC(4.0)-60DS-0.4V(51)/HIROSE ELECTRIC

表 19.2 CON1 信号配列

ピン番号	信号名	I/O	機能
1	GND	Power	電源(GND)
2	GND	Power	電源(GND)
3	GND	Power	電源(GND)
4	EXT_IO36	In/Out	拡張入出力、R-Mobile A1 の VIO_D0_0 ピンに接続
5	EXT_IO37	In/Out	拡張入出力、R-Mobile A1 の VIO_D1_0 ピンに接続
6	EXT_IO38	In/Out	拡張入出力、R-Mobile A1 の VIO_D2_0 ピンに接続
7	EXT_IO39	In/Out	拡張入出力、R-Mobile A1 の VIO_D3_0 ピンに接続
8	EXT_IO40	In/Out	拡張入出力、R-Mobile A1 の VIO_D4_0 ピンに接続
9	EXT_IO41	In/Out	拡張入出力、R-Mobile A1 の VIO_D5_0 ピンに接続
10	EXT_IO42	In/Out	拡張入出力、R-Mobile A1 の VIO_D6_0 ピンに接続
11	EXT_IO43	In/Out	拡張入出力、R-Mobile A1 の VIO_D7_0 ピンに接続
12	GND	Power	電源(GND)
13	EXT_IO44	In/Out	拡張入出力、R-Mobile A1 の VIO_CLK_0 ピンに接続
14	GND	Power	電源(GND)
15	EXT_IO45	In/Out	拡張入出力、R-Mobile A1 の VIO_FIELD_0 ピンに接続
16	EXT_IO46	In/Out	拡張入出力、R-Mobile A1 の VIO_HD_0 ピンに接続
17	EXT_IO47	In/Out	拡張入出力、R-Mobile A1 の VIO_VD_0 ピンに接続
18	GND	Power	電源(GND)
19	EXT_IO48	In/Out	拡張入出力、R-Mobile A1 の VIO_CKO_0 ピンに接続
20	GND	Power	電源(GND)
21	NC	-	未接続
22	NC	-	未接続
23	GND	Power	電源(GND)
24	GND	Power	電源(GND)
25	GND	Power	電源(GND)
26	GND	Power	電源(GND)
27	NC	-	未接続
28	NC	-	未接続
29	NC	-	未接続
30	EXT_IO49	In/Out	拡張入出力、R-Mobile A1 の SCIFA_SCK_2 ピンに接続
31	VCC_5V	Power	電源(VCC_5V)
32	VCC_5V	Power	電源(VCC_5V)
33	VCC_5V	Power	電源(VCC_5V)
34	VCC_5V	Power	電源(VCC_5V)
35	NC	-	未接続
36	GND	Power	電源(GND)
37	I2C_SCL_1	In/Out	I2C クロック、R-Mobile A1 の I2C_SCL_1 ピンに接続
38	I2C_SDA_1	In/Out	I2C データ、R-Mobile A1 の I2C_SDA_1 ピンに接続
39	GND	Power	電源(GND)
40	EXT_IO50	In/Out	拡張入出力、R-Mobile A1 の SCIFA_RXD_1 ピンに接続
41	EXT_IO51	In/Out	拡張入出力、R-Mobile A1 の SCIFA_TXD_1 ピンに接続
42	NC	-	未接続
43	GND	Power	電源(GND)
44	NC	-	未接続
45	NC	-	未接続
46	NC	-	未接続

ピン番号	信号名	I/O	機能
47	NC	-	未接続
48	NC	-	未接続
49	NC	-	未接続
50	EXT_IO52	In/Out	拡張入出力、R-Mobile A1 の VIO_D8_0 ピンに接続
51	EXT_IO53	In/Out	拡張入出力、R-Mobile A1 の VIO_D9_0 ピンに接続
52	EXT_IO54	In/Out	拡張入出力、R-Mobile A1 の VIO_D10_0 ピンに接続
53	EXT_IO55	In/Out	拡張入出力、R-Mobile A1 の VIO_D11_0 ピンに接続
54	EXT_IO56	In/Out	拡張入出力、R-Mobile A1 の VIO_D12_0 ピンに接続
55	EXT_IO57	In/Out	拡張入出力、R-Mobile A1 の VIO_D13_0 ピンに接続
56	EXT_IO58	In/Out	拡張入出力、R-Mobile A1 の VIO_D14_0 ピンに接続
57	EXT_IO59	In/Out	拡張入出力、R-Mobile A1 の VIO_D15_0 ピンに接続
58	VCC_3.3V	Power	電源(VCC_3.3V)
59	VCC_3.3V	Power	電源(VCC_3.3V)
60	VCC_3.3V	Power	電源(VCC_3.3V)

表 19.3 CON1 拡張入出力ピンのマルチプレクス

ピン番号	機能			
	CAMERA	UART	GPIO	PWM
1				
2				
3				
4	VIO_D0_0		PORT34	
5	VIO_D1_0		PORT33	
6	VIO_D2_0		PORT32	
7	VIO_D3_0		PORT31	
8	VIO_D4_0		PORT30	
9	VIO_D5_0		PORT29	
10	VIO_D6_0		PORT28	
11	VIO_D7_0		PORT27	
12				
13	VIO_CLK_0		PORT35	
14				
15	VIO_FIELD_0		PORT38 (IRQ25)	
16	VIO_HD_0		PORT37	
17	VIO_VD_0		PORT39	
18				
19	VIO_CKO_0		PORT36	
20				
21				
22				
23				
24				
25				
26				
27				
28				
29				
30			PORT22	
31				
32				
33				
34				
35				

ピン番号	機能			
	CAMERA	UART	GPIO	PWM
36				
37				
38				
39				
40		SCIFA_RXD_1	PORT195	
41		SCIFA_TXD_1	PORT196	
42				
43				
44				
45				
46				
47				
48				
49				
50	VIO_D8_0		PORT182	
51	VIO_D9_0		PORT181	
52	VIO_D10_0		PORT180 (IRQ24)	TPU0TO3
53	VIO_D11_0		PORT179	
54	VIO_D12_0		PORT178	
55	VIO_D13_0	SCIFA_TXD_6	PORT26	
56	VIO_D14_0	SCIFA_RXD_6	PORT25	
57	VIO_D15_0	SCIFA_SCK_6	PORT24	
58				
59				
60				

表 19.4 CON1 拡張入出力ピンの信号状態 [a]

ピン番号	信号名	R-Mobile A1 の信号名	リセット中の状態	リセット後の状態
1				
2				
3				
4	EXT_IO36	VIO_D0_0	PD	ID
5	EXT_IO37	VIO_D1_0	PD	ID
6	EXT_IO38	VIO_D2_0	PD	ID
7	EXT_IO39	VIO_D3_0	PD	ID
8	EXT_IO40	VIO_D4_0	PU	IU
9	EXT_IO41	VIO_D5_0	PU	IU
10	EXT_IO42	VIO_D6_0	PU	IU
11	EXT_IO43	VIO_D7_0	PU	IU
12				
13	EXT_IO44	VIO_CLK_0	PU	IU
14				
15	EXT_IO45	VIO_FIELD_0	PU	IU
16	EXT_IO46	VIO_HD_0	PD	ID
17	EXT_IO47	VIO_VD_0	PD	ID
18				
19	EXT_IO48	VIO_CKO_0	PU	IU
20				
21				
22				
23				
24				

ピン番号	信号名	R-Mobile A1 の信号名	リセット中の状態	リセット後の状態
25				
26				
27				
28				
29				
30	EXT_IO49	SCIFA_SCK_2	PU	IU
31				
32				
33				
34				
35				
36				
37				
38				
39				
40	EXT_IO50	SCIFA_RXD_1	PD	ID
41	EXT_IO51	SCIFA_TXD_1	PD	ID
42				
43				
44				
45				
46				
47				
48				
49				
50	EXT_IO52	VIO_D8_0	PU	IU
51	EXT_IO53	VIO_D9_0	PU	IU
52	EXT_IO54	VIO_D10_0	PD	ID
53	EXT_IO55	VIO_D11_0	PD	ID
54	EXT_IO56	VIO_D12_0	PU	IU
55	EXT_IO57	VIO_D13_0	ID	ID
56	EXT_IO58	VIO_D14_0	ID	ID
57	EXT_IO59	VIO_D15_0	PU	IU
58				
59				
60				

[a]記号一覧

IU: 入力バッファ有効、プルアップ有効

ID: 入力バッファ有効、プルダウン有効

PU: 入出力バッファ無効、プルアップ有効

PD: 入出力バッファ無効、プルダウン有効

19.2.1.1. Camera

R-Mobile A1 の CEU コントローラ(CEU0)に接続されています。

CEU0　　信号レベル: 3.3V CMOS

画像フォーマット: YUV422(8bit/16bit)

最大ピクセル数: 64M ピクセル(8188 x 8188 ピクセル)

19.2.1.2. UART

R-Mobile A1 のシリアル(UART)コントローラ(SCIFA1、SCIFA6)に接続されています。最大 2 ポート UART として使用できます。

SCIFA1	信号レベル: 3.3V CMOS
	最大 Baudrate: 1Mbps
	フロー制御: なし
SCIFA6	信号レベル: 3.3V CMOS
	最大 Baudrate: 1Mbps
	フロー制御: なし

19.2.1.3. PWM

R-Mobile A1 の PWM コントローラ(TPU0)に接続されています。

- 信号レベル: 3.3V CMOS

19.2.1.4. GPIO

R-Mobile A1 の GPIO コントローラに接続されています。最大 24bitGPIO として使用可能です。

- 信号レベル: 3.3V CMOS

19.2.2. CON2 シリアルインターフェース 1

CON2 は非同期(調歩同期)シリアルインターフェースで、レベル変換 IC を介して R-Mobile A1 の UART コントローラに接続されています。

SCIFB	信号レベル: RS232C
	最大 Baudrate: 230kbps
	フロー制御: RTS、CTS
搭載コネクタ	DF3DZ-8P-2H(51)/HIROSE ELECTRIC
	許容電流: 3A 以下(端子 1 本あたり)
対向コネクタ例	DF3-8S-2C/HIROSE ELECTRIC

CON2 にはピンヘッダ 8P(2mm ピッチ)が実装されています。Armadillo-810 カメラモデル開発セット付属品の D-Sub9/8 ピン シリアル変換ケーブルにより、D-Sub9 ピンに変換可能です。

表 19.5 CON2 信号配列

ピン番号	信号名	I/O	機能
1	VIN	Power	電源(VIN)[a]
2	GND	Power	電源(GND)
3	SCIFB_RXD	In	受信データ、レベル変換 IC を介して R-Mobile A1 の SCIFB_RXD ピンに接続
4	GND	Power	電源(GND)
5	SCIFB_TXD	Out	送信データ、レベル変換 IC を介して R-Mobile A1 の SCIFB_TXD ピンに接続
6	GND	Power	電源(GND)
7	SCIFB_CTS	In	送信可能、レベル変換 IC を介して R-Mobile A1 の SCIFB_CTS ピンに接続
8	SCIFB_RTS	Out	送信要求、レベル変換 IC を介して R-Mobile A1 の SCIFB_RTS ピンに接続

[a]CON2 の 1 ピンと CON5 の 57、58、59、60 ピンは接続されています。電源回路の構成につきましては、「19.4. 電源回路の構成」をご参照ください。

19.2.3. CON3 シリアルインターフェース 2

CON3 は非同期(調歩同期)シリアルインターフェースです。

SCIFA2　信号レベル: 3.3V CMOS

最大 Baudrate: 1Mbps

フロー制御: RTS、CTS

搭載コネクタ　DF13A-7P-1.25H(51)/HIROSE ELECTRIC

許容電流: 1A 以下(端子 1 本あたり)

対向コネクタ例　DF13-7S-1.25C/HIROSE ELECTRIC

Armadillo-810 カメラモデル開発セット付属品の開発用 USB シリアル変換アダプタを接続して、PC と通信可能です。開発用 USB シリアル変換アダプタにはスイッチが実装されており、起動モードを切替することが可能です。

表 19.6 CON3 信号配列

ピン番号	信号名	I/O	機能
1	SCIFA_RXD_2	In	受信データ、R-Mobile A1 の PORT200 ピンに接続
2	GND	Power	電源(GND)
3	SCIFA_TXD_2	Out	送信データ、R-Mobile A1 の PORT201 ピンに接続
4	VCC_3.3V	Power	電源出力(VCC_3.3V)
5	SCIFA_CTS_2	In	送信可能、R-Mobile A1 の PORT95 ピンに接続
6	HERMIT_EN_N	In	起動モード設定[a]、R-Mobile A1 の FCE0_N ピンに接続、VCC_3.3V で 10kΩ プルアップ (Low: 保守モード、High: OS 自動起動モード)
7	SCIFA_RTS_2	Out	送信要求、R-Mobile A1 の PORT96 ピンに接続

[a]CON3 の 6 ピンと CON5 の 34 ピンは接続されています。

19.2.4. CON4 USB インターフェース

CON4 は USB デバイスインターフェースです。

USB0　データ転送モード: USB2.0 High Speed/Full Speed

搭載コネクタ　　54819-0572/Molex[1]

表 19.7 CON4 信号配列

ピン番号	信号名	I/O	機能
1	VBUS	Power	電源(VBUS)
2	USB_DM_0	In/Out	USB のマイナス側信号、R-Mobile A1 の DM_0 ピンに接続[a]
3	UBS_DP_0	In/Out	USB のプラス側信号、R-Mobile A1 の DP_0 ピンに接続[a]
4	NC	-	未接続
5	GND	Power	電源(GND)

[a]CON5 の USB と同様の信号が配線されており、どちらか一方のみ使用可能です。

19.2.5. CON5 拡張インターフェース 1 (A コネクタ)

CON5 は入出力拡張用のインターフェースです。用途によって機能を選択できるように複数の機能が割り当てられたピンが接続されています。UART、GPIO、USB、SD、I2S、SPI などに使用可能な信号や外部リセット信号、起動モード設定信号などが接続されています。

搭載コネクタ　　DF17(4.0)-60DS-0.5V(57)/HIROSE ELECTRIC

許容電流: 0.3A 以下(端子 1 本あたり)

対向コネクタ例　　DF17(4.0)-60DP-0.5V(57)/HIROSE ELECTRIC

表 19.8 CON5 信号配列

ピン番号	信号名	I/O	機能
1	GND	Power	電源(GND)
2	USB_DM_0	In/Out	USB マイナス側信号、R-Mobile A1 の DM_0 ピンに接続[a]
3	USB_DP_0	In/Out	USB プラス側信号、R-Mobile A1 の DP_0 ピンに接続[a]
4	GND	Power	電源(GND)
5	USB_DM_1	In/Out	USB マイナス側信号、R-Mobile A1 の DM_1 ピンに接続
6	USB_DP_1	In/Out	USB プラス側信号、R-Mobile A1 の DP_1 ピンに接続
7	GND	Power	電源(GND)
8	EXT_IO0	In/Out	拡張入出力、R-Mobile A1 の SDHICLK_0 ピンに接続
9	EXT_IO1	In/Out	拡張入出力、R-Mobile A1 の SDHICMD_0 ピンに接続
10	EXT_IO2	In/Out	拡張入出力、R-Mobile A1 の SDHID0_0 ピンに接続
11	EXT_IO3	In/Out	拡張入出力、R-Mobile A1 の SDHID1_0 ピンに接続
12	EXT_IO4	In/Out	拡張入出力、R-Mobile A1 の SDHID2_0 ピンに接続
13	EXT_IO5	In/Out	拡張入出力、R-Mobile A1 の SDHID3_0 ピンに接続
14	EXT_IO6	In/Out	拡張入出力、R-Mobile A1 の SDHICD_0 ピンに接続
15	EXT_IO7	In/Out	拡張入出力、R-Mobile A1 の SDHIWP_0 ピンに接続
16	EXT_IO8	In/Out	拡張入出力、R-Mobile A1 の PORT97 ピンに接続
17	GND	Power	電源(GND)
18	EXT_IO9	In/Out	拡張入出力、R-Mobile A1 の PORT66 ピンに接続
19	EXT_IO10	In/Out	拡張入出力、R-Mobile A1 の PORT67 ピンに接続
20	EXT_IO11	In/Out	拡張入出力、R-Mobile A1 の PORT68 ピンに接続
21	EXT_IO12	In/Out	拡張入出力、R-Mobile A1 の PORT69 ピンに接続
22	EXT_IO13	In/Out	拡張入出力、R-Mobile A1 の PORT70 ピンに接続
23	EXT_IO14	In/Out	拡張入出力、R-Mobile A1 の PORT71 ピンに接続
24	EXT_IO15	In/Out	拡張入出力、R-Mobile A1 の PORT72 ピンに接続
25	EXT_IO16	In/Out	拡張入出力、R-Mobile A1 の PORT73 ピンに接続
26	EXT_IO17	In/Out	拡張入出力、R-Mobile A1 の PORT74 ピンに接続

[1]製品リビジョン Rev.3.2 以前は、UB-M5BR-G14-4S(LF)(SN)/J.S.T. Mfg.が搭載されています。

ピン番号	信号名	I/O	機能
27	EXT_IO18	In/Out	拡張入出力、R-Mobile A1 の PORT75 ピンに接続
28	VCC_3.3V	Power	電源(VCC_3.3V)
29	VCC_3.3V	Power	電源(VCC_3.3V)
30	VCC_3.3V	Power	電源(VCC_3.3V)
31	EXT_RESET_N	In	外部リセット、リセット IC を介して R-Mobile A1 の RESETP_N ピンに接続、VCC_3.3V で 10kΩ プルアップ (Low: リセット状態、High: リセット解除)
32	JTAG_EN	In	SD/JTAG 設定、R-Mobile A1 の DBGMD10 ピンに接続 (Low: SD 有効、High: JTAG 有効)
33	SDBOOT_EN	In	起動デバイス設定、R-Mobile A1 の MD3 ピンに接続、GND に 10kΩ プルダウン (Low: オンボードフラッシュメモリから起動、High: SD から起動)
34	HERMIT_EN_N	In	起動モード設定[b]、R-Mobile A1 の FCE0_N ピンに接続、VCC_3.3V で 10kΩ プルアップ (Low: 保守モード、High: OS 自動起動モード)
35	EXT_VIN_EN_N	In	電源系統マニュアル設定[c]、VBUS または VIN で 10kΩ プルアップ
36	GND	Power	電源(GND)
37	EXT_IO19	In/Out	拡張入出力、R-Mobile A1 の D27 ピンに接続
38	EXT_IO20	In/Out	拡張入出力、R-Mobile A1 の D28 ピンに接続
39	EXT_IO21	In/Out	拡張入出力、R-Mobile A1 の D29 ピンに接続
40	EXT_IO22	In/Out	拡張入出力、R-Mobile A1 の D30 ピンに接続
41	EXT_IO23	In/Out	拡張入出力、R-Mobile A1 の D31 ピンに接続
42	EXT_IO24	In/Out	拡張入出力、R-Mobile A1 の PORT98 ピンに接続
43	EXT_IO25	In/Out	拡張入出力、R-Mobile A1 の PORT99 ピンに接続
44	EXT_IO26	In/Out	拡張入出力、R-Mobile A1 の PORT100 ピンに接続
45	GND	Power	電源(GND)
46	EXT_IO27	In/Out	拡張入出力、R-Mobile A1 の FSIACK ピンに接続
47	EXT_IO28	In/Out	拡張入出力、R-Mobile A1 の FSIAIBT ピンに接続
48	EXT_IO29	In/Out	拡張入出力、R-Mobile A1 の FSIAILR ピンに接続
49	EXT_IO30	In/Out	拡張入出力、R-Mobile A1 の FSIAOSLD ピンに接続
50	EXT_IO31	In/Out	拡張入出力、R-Mobile A1 の DBGMD11 ピンに接続
51	EXT_IO32	In/Out	拡張入出力、R-Mobile A1 の FMSOCK ピンに接続
52	EXT_IO33	In/Out	拡張入出力、R-Mobile A1 の FSIAOMC ピンに接続
53	EXT_IO34	In/Out	拡張入出力、R-Mobile A1 の FSIAOBT ピンに接続
54	EXT_IO35	In/Out	拡張入出力、R-Mobile A1 の FSIAOLR ピンに接続
55	GND	Power	電源(GND)
56	NC	-	未接続
57	VIN	Power	電源(VIN)[d]
58	VIN	Power	電源(VIN)[d]
59	VIN	Power	電源(VIN)[d]
60	VIN	Power	電源(VIN)[d]

[a]CON4 の USB と同様の信号が配線されており、どちらか一方のみ使用可能です。使用する場合は GPIO(PORT19)を Low 出力に設定してください。

[b]CON3 の 6 ピンと CON5 の 34 ピンは接続されています。

[c]詳細につきましては、「19.4. 電源回路の構成」を参照してください。

[d]CON5 の 57、58、59、60 ピンと CON2 の 1 ピンは接続されています。電源回路の構成につきましては、「19.4. 電源回路の構成」をご参照ください。

表 19.9 CON5 拡張入出力ピンのマルチプレクス

ピン番号	機能							
	SD	UART	I2S	SSI(SPI)	MMC	GPIO	PWM	IrDA
1								
2								
3								
4								
5								
6								
7								
8	SDHICLK_0					PORT82		
9	SDHICMD_0					PORT76		
10	SDHID0_0					PORT77		
11	SDHID1_0					PORT78		
12	SDHID2_0					PORT79		
13	SDHID3_0					PORT80		
14	SDHICD_0					PORT81(IRQ26)		
15	SDHIWP_0					PORT83		
16						PORT97(IRQ12)		
17								
18	SDHICLK_1				MMCCLK_0	PORT66	TPU0TO2	
19	SDHICMD_1			MSIOF1_SS1	MMCCMD_0	PORT67(IRQ20)		
20	SDHID0_1			MSIOF1_RSCK	MMCD0_0	PORT68(IRQ16)		
21	SDHID1_1			MSIOF1_RSYNC	MMCD1_0	PORT69(IRQ17)		
22	SDHID2_1			MSIOF1_MCK0	MMCD2_0	PORT70(IRQ18)		
23	SDHID3_1			MSIOF1_MCK1	MMCD3_0	PORT71(IRQ19)		
24	SDHICD_1			MSIOF1_TSCK	MMCD4_0	PORT72		
25	SDHIWP_1			MSIOF1_TSYNC	MMCD5_0	PORT73		
26				MSIOF1_TXD	MMCD6_0	PORT74		
27				MSIOF1_RXD	MMCD7_0	PORT75		
28								
29								
30								
31								
32								
33								
34								

ピン番号	機能							
	SD	UART	I2S	SSI(SPI)	MMC	GPIO	PWM	IrDA
35								
36								
37		SCIFA_CTS_3_N				PORT162		IRDA_OUT
38		SCIFA_RTS_3_N				PORT161		IRDA_IN
39		SCIFA_TXD_3				PORT160		
40		SCIFA_RXD_3				PORT159		
41		SCIFA_SCK_3				PORT158		IRDA_FIRSEL
42						PORT98(IRQ13)		
43						PORT99(IRQ14)		
44						PORT100(IRQ15)		
45								
46			FSIACK			PORT11(IRQ2)		
47		SCIFA_TXD_4	FSIAIBT			PORT13(IRQ0)		
48		SCIFA_RXD_4	FSIAILR			PORT12(IRQ2)		
49			FSIAOSLD			PORT9		
50			FSIAISLD			PORT5		
51		SCIFA_TXD_5				PORT20(IRQ1)		
52		SCIFA_RXD_5	FSIAOMC			PORT10(IRQ3)		
53			FSIAOBT			PORT8		
54			FSIAOLR			PORT7		
55								
56								
57								
58								
59								
60								

表 19.10 CON5 拡張入出力ピンの信号状態 [a]

ピン番号	信号名	R-Mobile A1 の信号名	リセット中の状態	リセット後の状態
1				
2				
3				
4				
5				
6				
7				
8	EXT_IO0	SDHICLK_0	PD	O
9	EXT_IO1	SDHICMD_0	PD	ID
10	EXT_IO2	SDHID0_0	PD	ID
11	EXT_IO3	SDHID1_0	PD	ID
12	EXT_IO4	SDHID2_0	PD	ID
13	EXT_IO5	SDHID3_0	PD	ID
14	EXT_IO6	SDHICD_0	PU	IU
15	EXT_IO7	SDHIWP_0	PU	IU
16	EXT_IO8	PORT97	PD	ID
17				
18	EXT_IO9	PORT66	PD	ID
19	EXT_IO10	PORT67	PU	IU
20	EXT_IO11	PORT68	PU	IU
21	EXT_IO12	PORT69	PU	IU
22	EXT_IO13	PORT70	PU	IU
23	EXT_IO14	PORT71	PU	IU
24	EXT_IO15	PORT72	PU	PU
25	EXT_IO16	PORT73	PU	PU
26	EXT_IO17	PORT74	PU	PU
27	EXT_IO18	PORT75	PU	PU
28				
29				
30				
31				
32				
33				
34				
35				
36				
37	EXT_IO19	D27	PD	ID
38	EXT_IO20	D28	PD	ID
39	EXT_IO21	D29	PD	ID
40	EXT_IO22	D30	PD	ID
41	EXT_IO23	D31	PD	ID
42	EXT_IO24	PORT98	PD	ID
43	EXT_IO25	PORT99	PD	ID
44	EXT_IO26	PORT100	PD	ID
45				
46	EXT_IO27	FSIACK	PD	ID
47	EXT_IO28	FSIAIBT	PD	ID
48	EXT_IO29	FSIAILR	PD	ID
49	EXT_IO30	FSIAOSLD	L	O
50	EXT_IO31	DBGMD11	ID	ID
51	EXT_IO32	FMSOCK	PD	ID
52	EXT_IO33	FSIAOMC	PD	ID

ピン番号	信号名	R-Mobile A1 の信号名	リセット中の状態	リセット後の状態
53	EXT_IO34	FSIAOBT	L	O
54	EXT_IO35	FSIAOLR	L	O
55				
56				
57				
58				
59				
60				

[a]記号一覧

IU: 入力バッファ有効、プルアップ有効

ID: 入力バッファ有効、プルダウン有効

L : 出力バッファ有効、Low レベル出力

O : 出力バッファ有効

PU: 入出力バッファ無効、プルアップ有効

PD: 入出力バッファ無効、プルダウン有効

19.2.5.1. USB

R-Mobile A1 の USB コントローラに接続されています。USB は 2 ポート接続されています。

USB0 データ転送モード: USB2.0 High Speed/Full Speed

USB1 データ転送モード: USB2.0 High Speed/Full Speed

19.2.5.2. SD

R-Mobile A1 の SD コントローラ(SDHI0)に接続されています。

SDHI0 信号入出力レベル: 3.3V CMOS

19.2.5.3. UART

R-Mobile A1 のシリアル(UART)コントローラ(SCIFA3、SCIFA4、SCIFA5)に接続されています。最大 3 ポート UART として使用できます。

SCIFA3 信号レベル: 3.3V CMOS

最大 Baudrate: 1Mbps

フロー制御: RTS、CTS

SCIFA4 信号レベル: 3.3V CMOS

最大 Baudrate: 1Mbps

フロー制御: なし

SCIFA5 信号レベル: 3.3V CMOS

最大 Baudrate: 1Mbps

フロー制御: なし

19.2.5.4. I2S

R-Mobile A1 の I2S コントローラ(FSIA)に接続されています。

FSIA 信号レベル: 3.3V CMOS

19.2.5.5. SSI(SPI)

R-Mobile A1 の SSI(SPI)コントローラ(MSIOF1)に接続されています。

MSIOF1 信号レベル: 3.3V CMOS

19.2.5.6. MMC

R-Mobile A1 の MMC コントローラ(MMC0)に接続されています。

MMC0 信号レベル: 3.3V CMOS

19.2.5.7. GPIO

R-Mobile A1 の GPIO コントローラに接続されています。最大 36bitGPIO として使用可能です。

GPIO 信号レベル: 3.3V CMOS

19.2.5.8. PWM

R-Mobile A1 の PWM コントローラ(TPU0)に接続されています。

TPU2 信号レベル: 3.3V CMOS

19.2.5.9. IrDA

R-Mobile A1 の IrDA コントローラに接続されています。

IrDA 信号レベル: 3.3V CMOS

19.2.5.10. 外部リセット

確実にリセットさせるため、外部リセットには 100μs 以上の Low 期間を設定してください。

19.2.6. LED1～LED4 ユーザー LED

LED1～LED4 はユーザー側で自由に利用できる面実装の黄色 LED です。LED に接続された R-Mobile A1 の信号が GPIO の出力モードに設定されている場合に制御できます。

表 19.11 ユーザー LED の機能

LED	機能
LED1	R-Mobile A1 の WE3_N ピンに接続 (Low: 消灯、High: 点灯)

LED	機能
LED2	R-Mobile A1 の WE2_N ピンに接続 (Low: 消灯、High: 点灯)
LED3	R-Mobile A1 の RDWR ピンに接続 (Low: 消灯、High: 点灯)
LED4	R-Mobile A1 の IOIS16_N ピンに接続 (Low: 消灯、High: 点灯)

19.3. 電気的仕様

19.3.1. 絶対最大定格

表 19.12 絶対最大定格

項目	記号	Min	Max	単位	備考
電源電圧	VIN, VBUS	-0.3	5.3	V	
入力電圧	VIO	-0.5	VCC_3.3V+0.5	V	
動作温度範囲	Topr	-20	70	℃	ただし結露なきこと

絶対最大定格はあらゆる使用条件、又は試験条件であっても瞬時たりとも越えてはならない値です。上記の値に対して余裕をもってご使用ください。

19.3.2. 推奨動作条件

表 19.13 推奨動作条件

項目	記号	Min	Typ	Max	単位	備考
電源電圧	VIN, VBUS	4.35	5	5.25	V	
使用周囲温度	Ta	-20	25	70	℃	ただし結露なきこと

19.3.3. 入出力インターフェースの電気的仕様

表 19.14 入出力インターフェースの電気的仕様

項目	記号	Min	Typ	Max	単位	備考
入出力インターフェース電源電圧	VCC_3.3V	3.135	3.3	3.465	V	
入力電圧	VIH	VCC_3.3Vx0.8	-	VCC_3.3V +0.3	V	
	VIL	-0.3	-	VCC_3.3V x 0.2	V	
出力電圧	VOH	VCC_3.3V-0.5	-	-	V	IOH=-2mA の時
	VOL	-	-	0.5	V	IOL=2mA の時
出力電流(per pin)	IOL	-	-	2.0	mA	
	IOH	-	-	-2.0	mA	
出力電流(total)	ΣIOL	-	-	120	mA	
	ΣIOH	-	-	-40	mA	
出力電流(I2C)	I2C_IOL	-	-	5	mA	
プルアップ/ダウン抵抗値	PULL	25	50	100	kΩ	

19.4. 電源回路の構成

電源回路の構成は次の通りです。

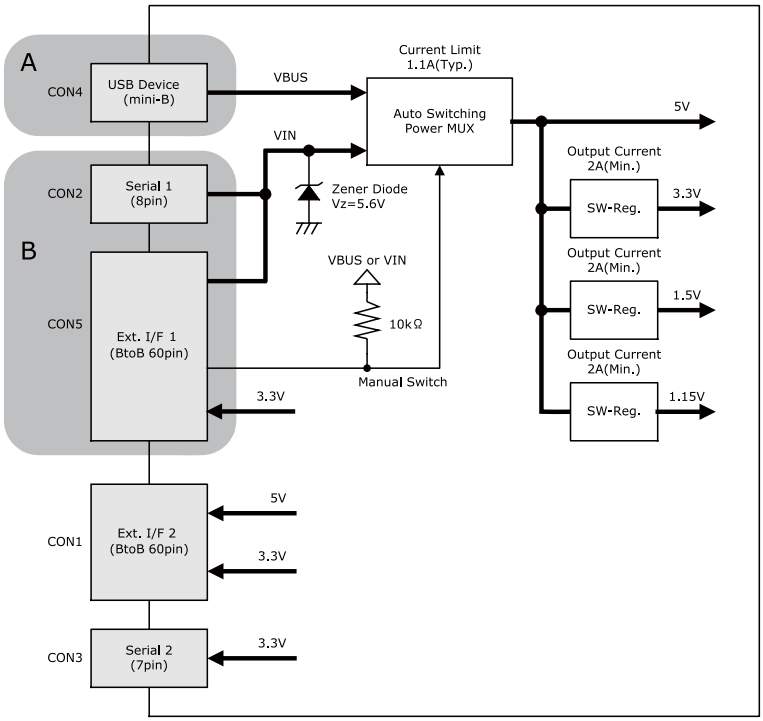


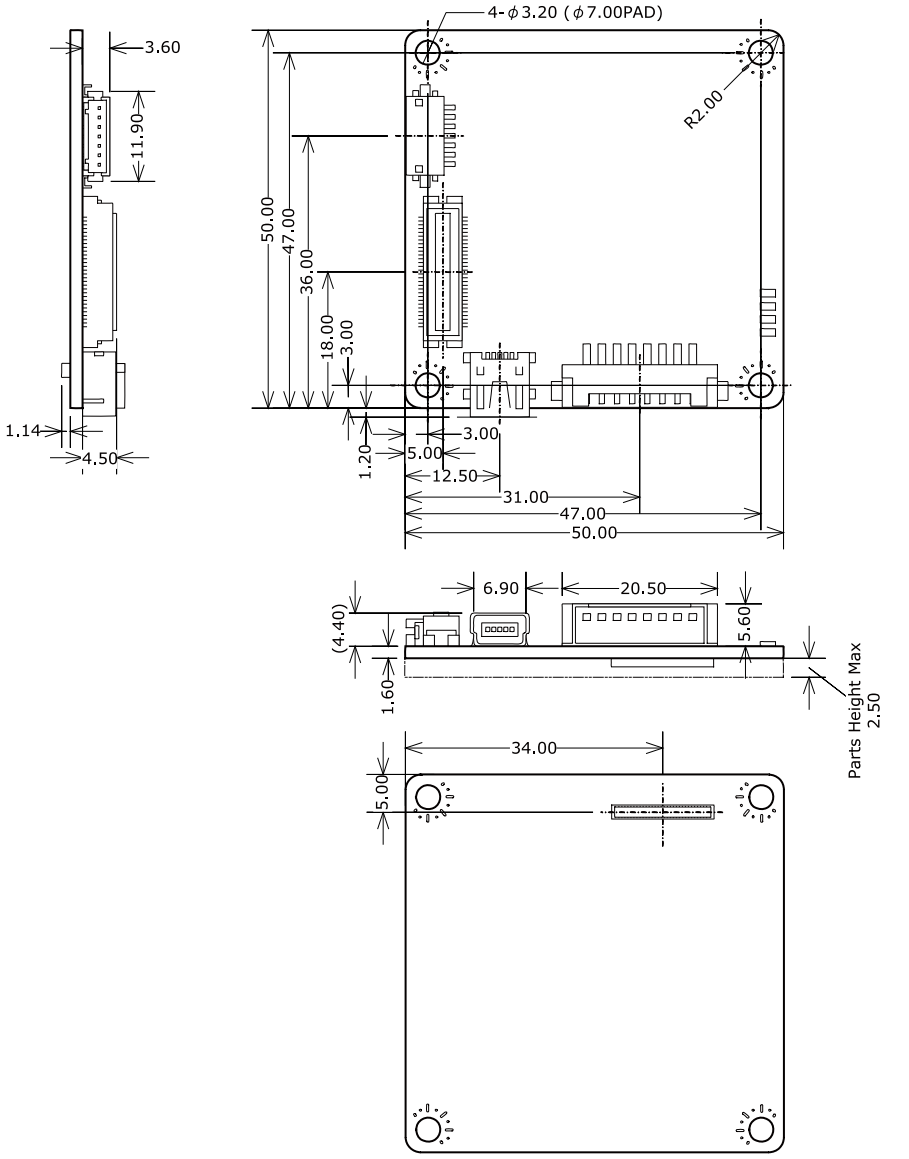
図 19.2 Armadillo-810 の電源回路の構成

Armadillo-810 では CON2、CON4、CON5 の 3 箇所から電源を供給することができます。電源系統の切替スイッチが実装されており、A 系統(CON4)と B 系統(CON2、CON5 のどちらか一方)の両方に電源を接続した場合は自動的に B 系統側から電源が供給されます。電源系統のマニュアル固定ピンが CON5 の 35 ピンに配置されており、このピンを GND に接続することにより、電源の供給を B 系統側だけに固定することができます。

過電流保護の IC を搭載しており、Armadillo-810 の供給可能電流は内部回路と合計で 1.1A(Typ.)となります。

CON2 と CON5 から同時に電源を供給しないでください。故障の原因となります。

20. 基板形状図



[Unit : mm]

図 20.1 基板形状および固定穴寸法

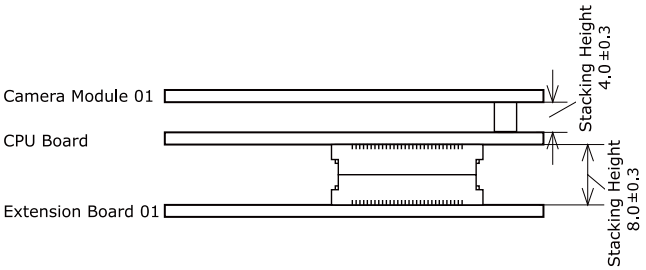


図 20.2 スタッキング高さ例(Armadllo-810 カメラモデル)

21. 拡張ボード/オプションモジュール

本章では、Armadillo-810 に接続可能な拡張ボード、オプションモジュールおよびケーブルについて説明します。

21.1. Armadillo-810 拡張ボード 01 (A コネクタ用)

21.1.1. 概要

Armadillo-810 拡張ボード 01 (A コネクタ用)は Armadillo-810 の拡張インターフェース 1 (A コネクタ)に接続可能な拡張ボードです。リアルタイムクロック、SD スロット等を搭載しています。

Armadillo-810 拡張ボード 01 (A コネクタ用)の主な仕様は次の通りです。

表 21.1 Armadillo-810 拡張ボード 01 (A コネクタ用)の仕様

シリアル(UART)	最大 3 ポート増設可能
GPIO	最大 23bit
USB	USB2.0 Host(High Speed 対応)
SD/MMC	SD スロット x 1、追加 1 ポート増設可能
オーディオ	増設可能(I2S ポート使用)
拡張インターフェース	UART、GPIO、SPI、I2S、SD、MMC、PWM 等
カレンダー時計	リアルタイムクロック搭載(バックアップ機能付き)
スイッチ	リセットスイッチ
JTAG	10 ピン(2.54mm ピッチ)[a]
電源電圧	DC 5V±5%
使用周囲温度	-20～70℃(ただし結露なきこと)
基板サイズ	50 x 50mm(突起部を除く)

[a]オプション品の「8 ピン JTAG 変換ケーブル(Armadillo-400/800 シリーズ対応)」(OP-JC8P25-00)を使用して ARM 標準 20 ピンに変換することが可能です。

Armadillo-810 拡張ボード 01 (A コネクタ用)のブロック図は次の通りです。

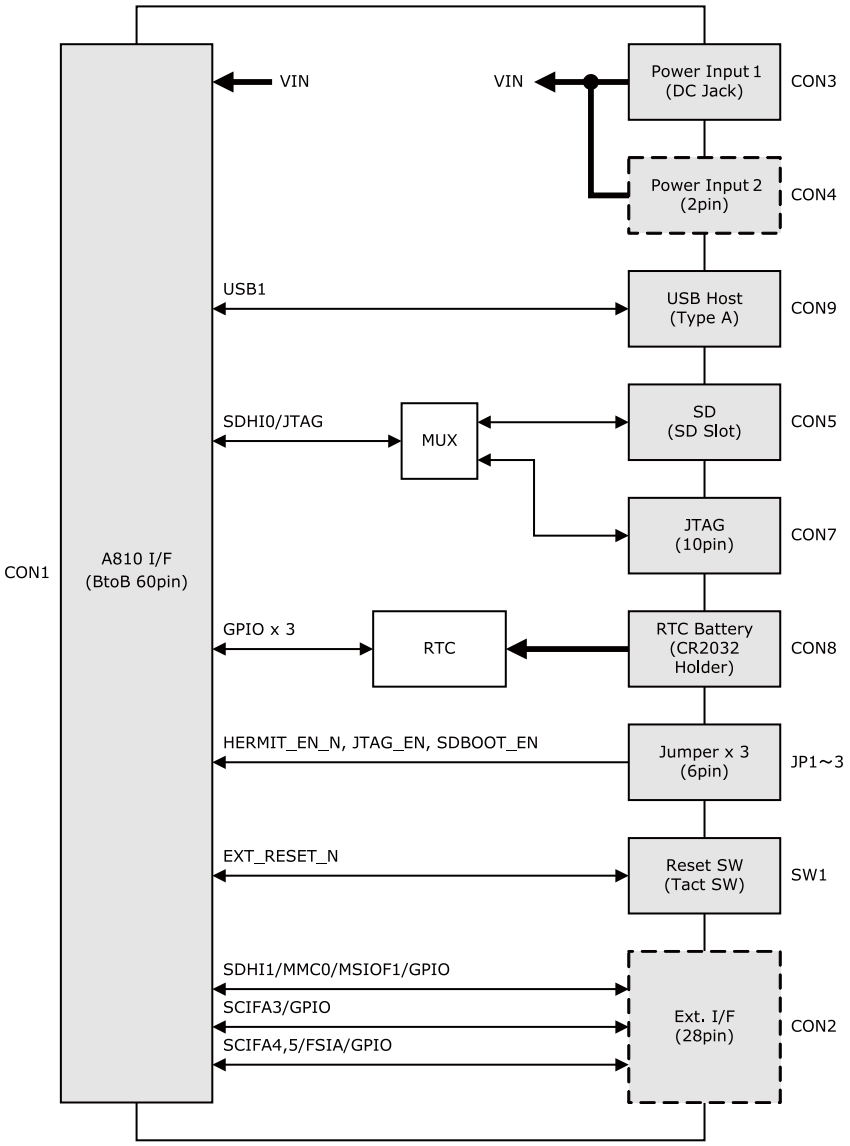


図 21.1 Armadillo-810 拡張ボード 01 (A コネクタ用)のブロック図

21.1.2. インターフェースレイアウト

Armadillo-810 拡張ボード 01 (A コネクタ用)のインターフェースレイアウトは次の通りです。

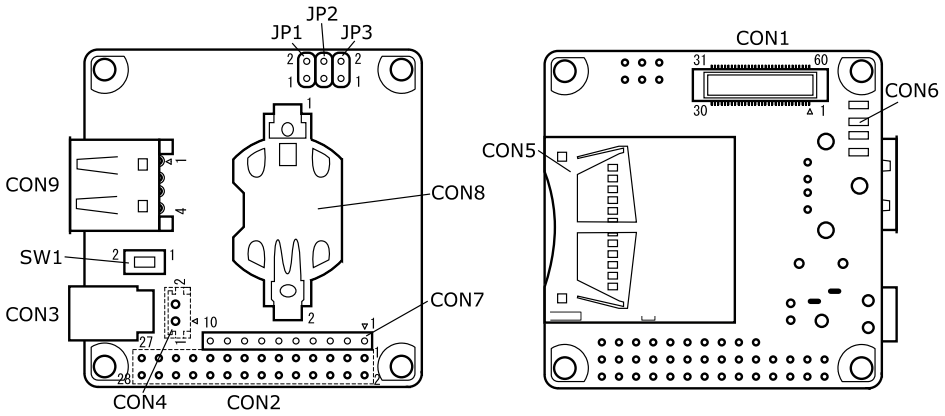


図 21.2 Armadillo-810 拡張ボード 01 (A コネクタ用)のインターフェースレイアウト図

表 21.2 Armadillo-810 拡張ボード 01 (A コネクタ用)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	Armadillo-810 接続インターフェース	BtoB コネクタ 60P(0.5mm ピッチ) DF17(4.0)-60DP-0.5V(57)/HIROSE ELECTRIC	挿抜寿命:50 回 Armadillo-810 の拡張インターフェース 1 (A コネクタ)と接続
CON2	拡張インターフェース	ピンヘッダ 28P(2.54mm ピッチ)	コネクタ非搭載(搭載コネクタ例：A1-28PA-2.54DSA(71)/HIROSE ELECTRIC)
CON3	電源入力 1	DC ジャック HEC3600-016110/HOSIDEN	対応プラグ： EIAJ#2
CON4	電源入力 2	ピンヘッダ 2P(2.5mm ピッチ)	コネクタ非搭載(搭載コネクタ例：B2B-EH/J.S.T. Mfg.)
CON5	SD インターフェース	SD スロット SCDA9A0400/ALPS ELECTRIC	信号線は CON7 と共通
CON6	Reserved	Pad	このインターフェースを使用する場合の動作は保証されていません
CON7	JTAG インターフェース	ピンヘッダ 10P(2.54mm ピッチ) A2-10PA-2.54DSA(71)/HIROSE ELECTRIC	信号線は CON5 と共通
CON8	RTC 外部バックアップインターフェース	電池ボックス SMTU2032-LF.TR/Renata SA	対応電池: CR2032
CON9	USB インターフェース	USB Type A コネクタ UBA-4R-D14T-4D(LF)(SN)/J.S.T. Mfg.	USB2.0 Host(High Speed 対応)
JP1	起動モード設定ジャンパ	ピンヘッダ 6P(2.54mm ピッチ) A1-6PA-2.54DSA(71)/HIROSE ELECTRIC	オープン: OS 自動起動モード ショート: 保守モード
JP2	SD/JTAG 設定ジャンパ		オープン: SD(CON5)有効/JTAG(CON7)無効 ショート: SD(CON5)無効/JTAG(CON7)有効
JP3	起動デバイス設定ジャンパ	オープン: オンボードフラッシュメモリブート ショート: SD(CON5)ブート	
SW1	リセットスイッチ	タクトスイッチ SKHLACA010/ALPS ELECTRIC	

21.1.3. インターフェース仕様

21.1.3.1. CON1 Armadillo-810 接続インターフェース

CON1 は Armadillo-810 の拡張インターフェース 1 (A コネクタ)との接続インターフェースです。

搭載コネクタ　　DF17(4.0)-60DP-0.5V(57)/HIROSE ELECTRIC

許容電流: 0.3A 以下(端子 1 本あたり)

表 21.3 CON1 信号配列

ピン番号	信号名	I/O	機能
1	GND	Power	電源(GND)
2	USB_DM_0	In/Out	USB マイナス側信号、CON6 の 2 ピンと接続
3	USB_DP_0	In/Out	USB プラス側信号、CON6 の 3 ピンと接続
4	GND	Power	電源(GND)
5	USB_DM_1	In/Out	USB マイナス側信号、CON9 の 2 ピンと接続
6	USB_DP_1	In/Out	USB プラス側信号、CON9 の 3 ピンと接続
7	GND	Power	電源(GND)
8	A1_SDHICLK_0	In	SD クロック、CON5 の 5 ピンと接続
9	A1_SDHICMD_0	In/Out	SD コマンド/レスポンス、CON5 の 2 ピンと接続
10	A1_SDHID0_0	In/Out	データバス(bit0)、CON5 の 7 ピンと接続
11	A1_SDHID1_0	In/Out	データバス(bit1)、CON5 の 8 ピンと接続
12	A1_SDHID2_0	In/Out	データバス(bit2)、CON5 の 9 ピンと接続
13	A1_SDHID3_0	In/Out	データバス(bit3)、CON5 の 1 ピンと接続
14	A1_SDHIDCD_0	Out	カード検出、CON5 の 10 ピンと接続 (Low: カード挿入、High: カード未挿入)
15	A1_SDHIWP_0	Out	ライトプロテクト検出、CON5 の 12 ピンと接続 (Low: 書き込み可能、High; 書き込み不可能)
16	SD0_PWR_EN	In	SD 電源操作、CON5 のパワースイッチのイネーブルピンと接続 (Low: 電源切断、High: 電源供給)
17	GND	Power	電源(GND)
18	A1_PORT66	In/Out	拡張入出力、CON2 の 1 ピンと接続
19	A1_PORT67	In/Out	拡張入出力、CON2 の 2 ピンと接続
20	A1_PORT68	In/Out	拡張入出力、CON2 の 3 ピンと接続
21	A1_PORT69	In/Out	拡張入出力、CON2 の 4 ピンと接続
22	A1_PORT70	In/Out	拡張入出力、CON2 の 5 ピンと接続
23	A1_PORT71	In/Out	拡張入出力、CON2 の 6 ピンと接続
24	A1_PORT72	In/Out	拡張入出力、CON2 の 7 ピンと接続
25	A1_PORT73	In/Out	拡張入出力、CON2 の 8 ピンと接続
26	A1_PORT74	In/Out	拡張入出力、CON2 の 9 ピンと接続
27	A1_PORT75	In/Out	拡張入出力、CON2 の 10 ピンと接続
28	VCC_3.3V	Power	電源(VCC_3.3V)
29	VCC_3.3V	Power	電源(VCC_3.3V)
30	VCC_3.3V	Power	電源(VCC_3.3V)
31	EXT_RESET_N	Out	リセット、CON7 の 2 ピン、SW1 と接続
32	JTAG_EN	Out	SD/JTAG 設定、JP2 と接続
33	SDBOOT_EN	Out	起動デバイス設定、JP3 と接続
34	HERMIT_EN_N	Out	起動モード設定、JP1 と接続
35	EXT_VIN_EN_N	Out	電源系統マニュアル設定[a]、0Ω 抵抗で GND に接続
36	GND	Power	電源(GND)
37	A1_PORT162	In/Out	拡張入出力、CON2 の 16 ピンと接続
38	A1_PORT161	In/Out	拡張入出力、CON2 の 14 ピンと接続
39	A1_PORT160	In/Out	拡張入出力、CON2 の 13 ピンと接続

ピン番号	信号名	I/O	機能
40	A1_PORT159	In/Out	拡張入出力、CON2 の 15 ピンと接続
41	USB1_PWR_EN	In	USB 電源操作(Low: 電源切断、High: 電源供給)
42	RTC_SCL	In	I2C クロック、リアルタイムクロックに接続、VCC_3.3V で 1kΩ プルアップ
43	RTC_SDA	In/Out	I2C データ、リアルタイムクロックに接続、VCC_3.3V で 1kΩ プルアップ
44	RTC_INT1_N	Out	リアルタイムクロック割り込み、リアルタイムクロックに接続、VCC_3.3V で 10kΩ プルアップ
45	GND	Power	電源(GND)
46	A1_PORT11	In/Out	拡張入出力、CON2 の 21 ピンと接続
47	A1_PORT13	In/Out	拡張入出力、CON2 の 25 ピンと接続
48	A1_PORT12	In/Out	拡張入出力、CON2 の 23 ピンと接続
49	A1_PORT9	In/Out	拡張入出力、CON2 の 26 ピンと接続
50	A1_PORT5	In/Out	拡張入出力、CON2 の 24 ピンと接続
51	A1_PORT20	In/Out	拡張入出力、CON2 の 22 ピンと接続
52	A1_PORT10	In/Out	拡張入出力、CON2 の 20 ピンと接続
53	A1_PORT8	In/Out	拡張入出力、CON2 の 19 ピンと接続
54	A1_PORT7	In/Out	拡張入出力、CON2 の 18 ピンと接続
55	GND	Power	電源(GND)
56	NC	-	未接続
57	VCC_5V	Power	電源(VCC_5V)
58	VCC_5V	Power	電源(VCC_5V)
59	VCC_5V	Power	電源(VCC_5V)
60	VCC_5V	Power	電源(VCC_5V)

[a]詳細につきましては、「21.1.3.12. 電源回路の構成」を参照してください。

21.1.3.2. CON2 拡張インターフェース

CON2 は入出力拡張用のインターフェースです。

搭載コネクタ例　　A1-28PA-2.54DSA(71)/HIROSE ELECTRIC

表 21.4 CON2 信号配列

ピン番号	信号名	I/O	機能
1	A1_PORT66	In/Out	拡張入出力、CON1 の 18 ピンと接続
2	A1_PORT67	In/Out	拡張入出力、CON1 の 19 ピンと接続
3	A1_PORT68	In/Out	拡張入出力、CON1 の 20 ピンと接続
4	A1_PORT69	In/Out	拡張入出力、CON1 の 21 ピンと接続
5	A1_PORT70	In/Out	拡張入出力、CON1 の 22 ピンと接続
6	A1_PORT71	In/Out	拡張入出力、CON1 の 23 ピンと接続
7	A1_PORT72	In/Out	拡張入出力、CON1 の 24 ピンと接続
8	A1_PORT73	In/Out	拡張入出力、CON1 の 25 ピンと接続
9	A1_PORT74	In/Out	拡張入出力、CON1 の 26 ピンと接続
10	A1_PORT75	In/Out	拡張入出力、CON1 の 27 ピンと接続
11	VCC_3.3V	Power	電源(VCC_3.3V)
12	GND	Power	電源(GND)
13	A1_PORT160	In/Out	拡張入出力、CON1 の 39 ピンと接続
14	A1_PORT161	In/Out	拡張入出力、CON1 の 38 ピンと接続
15	A1_PORT159	In/Out	拡張入出力、CON1 の 40 ピンと接続
16	A1_PORT162	In/Out	拡張入出力、CON1 の 37 ピンと接続
17	GND	Power	電源(GND)
18	A1_PORT7	In/Out	拡張入出力、CON1 の 54 ピンと接続
19	A1_PORT8	In/Out	拡張入出力、CON1 の 53 ピンと接続
20	A1_PORT10	In/Out	拡張入出力、CON1 の 52 ピンと接続
21	A1_PORT11	In/Out	拡張入出力、CON1 の 46 ピンと接続
22	A1_PORT20	In/Out	拡張入出力、CON1 の 51 ピンと接続

ピン番号	信号名	I/O	機能
23	A1_PORT12	In/Out	拡張入出力、CON1 の 48 ピンと接続
24	A1_PORT5	In/Out	拡張入出力、CON1 の 50 ピンと接続
25	A1_PORT13	In/Out	拡張入出力、CON1 の 47 ピンと接続
26	A1_PORT9	In/Out	拡張入出力、CON1 の 49 ピンと接続
27	VCC_3.3V	Power	電源(VCC_3.3V)
28	GND	Power	電源(GND)

表 21.5 CON2 拡張入出力ピンのマルチプレクス

ピン番号	機能						
	SD	UART	I2S	SSI(SPI)	MMC	GPIO	その他
1	SDHICLK_1				MMCCLK_0	PORT66	TPU0TO2
2	SDHICMD_1			MSIOF1_SS1	MMCCMD_0	PORT67(IRQ20)	
3	SDHID0_1			MSIOF1_RSCK	MMCD0_0	PORT68(IRQ16)	
4	SDHID1_1			MSIOF1_RSYNC	MMCD1_0	PORT69(IRQ17)	
5	SDHID2_1			MSIOF1_MCK0	MMCD2_0	PORT70(IRQ18)	
6	SDHID3_1			MSIOF1_MCK1	MMCD3_0	PORT71(IRQ19)	
7	SDHICD_1			MSIOF1_TSCK	MMCD4_0	PORT72	
8	SDHIWP_1			MSIOF1_TSYNC	MMCD5_0	PORT73	
9				MSIOF1_TXD	MMCD6_0	PORT74	
10				MSIOF1_RXD	MMCD7_0	PORT75	
11							
12							
13		SCIFA_TXD_3				PORT160	
14		SCIFA_RTS_3_N				PORT161	IRDA_IN
15		SCIFA_RXD_3				PORT159	
16		SCIFA_CTS_3_N				PORT162	IRDA_OUT
17							
18			FSIAOLR			PORT7	
19			FSIAOBT			PORT8	
20		SCIFA_RXD_5	FSIAOMC			PORT10(IRQ3)	
21			FSIACK			PORT11(IRQ2)	
22		SCIFA_TXD_5				PORT20(IRQ1)	
23		SCIFA_RXD_4	FSIAILR			PORT12(IRQ2)	
24			FSIAISLD			PORT5	
25		SCIFA_TXD_4	FSIAIBT			PORT13(IRQ0)	
26			FSIAOSLD			PORT9	
27							
28							

21.1.3.3. CON3 電源入力 1

CON3 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。下図の極性マークのある AC アダプターが使用できます。

図 21.3 AC アダプターの極性マーク

搭載コネクタ　　HEC3600-016110/HOSIDEN

21.1.3.4. CON4 電源入力 2

CON4 は電源入力インターフェースです。コネクタは実装されていません。

搭載コネクタ例　　B2B-EH/J.S.T. Mfg.

表 21.6 CON4 信号配列

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

21.1.3.5. CON5 SD インターフェース

CON5 は SD インターフェースです。CON7 の JTAG 機能と排他になっており、JP2 の操作によりどちらを使用するかを設定します。SD インターフェースに供給する電源は CON1 の 16 ピンより、ON/OFF の制御が可能です。Low 出力で電源切断、High 出力で電源供給されます。

搭載コネクタ　　SCDA9A0400/ALPS ELECTRIC

表 21.7 CON5 信号配列

ピン番号	信号名	I/O	機能
1	SD0_DAT3	In/Out	データバス(bit3)、CON1 の 13 ピンと接続、VCC_3.3V で 47kΩ プルアップ
2	SD0_CMD	In/Out	SD コマンド/レスポンス、CON1 の 9 ピンと接続、VCC_3.3V で 47kΩ プルアップ
3	GND	Power	電源(GND)
4	VCC_3.3V	Power	電源(VCC_3.3V)
5	SD0_CLK	In/Out	SD クロック、CON1 の 8 ピンと接続、47kΩ プルダウン
6	GND	Power	電源(GND)
7	SD0_DAT0	In/Out	データバス(bit0)、CON1 の 10 ピンと接続、VCC_3.3V で 47kΩ プルアップ
8	SD0_DAT1	In/Out	データバス(bit1)、CON1 の 11 ピンと接続、VCC_3.3V で 47kΩ プルアップ
9	SD0_DAT2	In/Out	データバス(bit2)、CON1 の 12 ピンと接続、VCC_3.3V で 47kΩ プルアップ
10	SD0_CD	In	カード検出(Low: カード挿入、High: カード未挿入)、CON1 の 14 ピンと接続、VCC_3.3V で 47kΩ プルアップ
11	GND	Power	電源(GND)
12	SD0_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)、CON1 の 15 ピンと接続、VCC_3.3V で 47kΩ プルアップ
13	GND	Power	電源(GND)
14	GND	Power	電源(GND)

21.1.3.6. CON6 USB インターフェース(Reserved)

CON6 には USB の信号線が接続されています。このインターフェースを使用する場合の動作は保証しておりません。

表 21.8 CON6 信号配列

ピン番号	信号名	I/O	機能
1	VBUS	Power	電源(VBUS)
2	USB_DM_0	In/Out	USB のマイナス側信号、CON1 の 1 ピンと接続
3	USB_DP_0	In/Out	USB のプラス側信号、CON1 の 2 ピンと接続
4	GND	Power	電源(GND)

21.1.3.7. CON7 JTAG インターフェース

CON7 は JTAG インターフェースです。CON5 の SD 機能と排他になっており、JP2 の操作によりどちらを使用するかを設定します。

搭載コネクタ　　A2-10PA-2.54DSA(71)/HIROSE ELECTRIC

表 21.9 CON7 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	JTAG_TRST_N	In	テストリセット、CON1 の 9 ピンと接続、VCC_3.3V で 10kΩ プルアップ
3	JTAG_TDI	In	テストデータ入力、CON1 の 12 ピンと接続、VCC_3.3V で 10kΩ プルアップ
4	JTAG_TMS	In	テストモード選択、CON1 の 10 ピンと接続、VCC_3.3V で 10kΩ プルアップ
5	JTAG_TCK	In	テストクロック、CON1 の 11 ピンと接続、VCC_3.3V で 10kΩ プルアップ
6	JTAG_TDO	Out	テストデータ出力、CON1 の 8 ピンと接続、VCC_3.3V で 10kΩ プルアップ
7	JTAG_SRST_N	In	リセット、CON1 の 31 ピン、リセットスイッチ(SW1)と接続
8	GND	Power	電源(GND)
9	JTAG_RTCK	In	テストクロック、CON1 の 13 ピンと接続、VCC_3.3V で 10kΩ プルアップ
10	JTAG_EDBGREQ	In	ブレーク要求、CON1 の 15 ピンと接続

オプション品の「8 ピン JTAG 変換ケーブル(Armadillo-400/800 シリーズ対応)」(OP-JC8P25-00)を使用して ARM 標準 20 ピンに変換することが可能です。

21.1.3.8. CON8 RTC 外部バックアップインターフェース

CON8 はリアルタイムクロックの外部バックアップインターフェースです。電池ボックス(対応バッテリ: CR2032、BR2032)が実装されています。

搭載コネクタ　　SMTU2032-LF.TR/Renata SA

Armadillo-810 拡張ボード 01 (A コネクタ用)にはリアルタイムクロックが搭載されています。リアルタイムクロックは積層セラミックコンデンサにより、電源切断後も数分間動作することが可能です。長時間電源が切断されても時刻データを保持させたい場合は、RTC 外部バックアップインターフェース(CON8)に別途バッテリを接続することができます。

リアルタイムクロックの主な仕様は次の通りです。

表 21.10 リアルタイムクロック仕様

リアルタイムクロック	セイコーインスツル リアルタイムクロック(S-35390A)
バックアップ	300 秒(Typ.)、60 秒(Min.) RTC 外部バックアップインターフェース(CON8)経由で外部バッテリを接続可能
電源電圧	DC2.0～3.5V

リアルタイムクロックの平均月差は、周囲温度 25℃で±30 秒程度(参考値)です。時間精度は、周囲温度に大きく影響を受けますので、ご使用の際は十分に特性の確認をお願いします。

21.1.3.9. CON9 USB インターフェース

CON9 は USB ホストインターフェースです。USB Type A コネクタを実装しています。USB インターフェースに供給する電源は CON1 の 41 ピンから ON/OFF の制御ができます。

・データ転送モード: USB2.0 High Speed/Full Speed/Low Speed

搭載コネクタ　　UBA-4R-D14T-4D(LF)(SN)/J.S.T. Mfg.

表 21.11 CON9 信号配列

ピン番号	信号名	I/O	機能
1	VBUS	Power	電源(VBUS)
2	USB_DM_1	In/Out	USB のマイナス側信号、CON1 の 5 ピンと接続
3	USB_DP_1	In/Out	USB のプラス側信号、CON1 の 6 ピンと接続
4	GND	Power	電源(GND)

21.1.3.10. JP1～JP3 設定ジャンパ

JP1～JP3 は設定ジャンパです。JP1 は起動モードの設定、JP2 は SD/JTAG 機能の設定、JP3 は起動デバイスの設定に使用します。

搭載コネクタ　　A1-6PA-2.54DSA(71)/HIROSE ELECTRIC

表 21.12 JP1 信号配列

ピン番号	信号名	I/O	機能
1	JP1	In	CON1 の 34 ピンと接続
2	GND	Power	電源(GND)

表 21.13 JP2 信号配列

ピン番号	信号名	I/O	機能
1	JP2	In	CON1 の 32 ピンと接続
2	JP2PU	Out	VCC_3.3V で 100Ω プルアップ

表 21.14 JP3 信号配列

ピン番号	信号名	I/O	機能
1	JP3	In	CON1 の 33 ピンと接続
2	JP3PU	Out	VCC_3.3V で 100Ω プルアップ

表 21.15 ジャンパの機能

ジャンパ	設定	説明
JP1	オープン	オートブートモード： OS を自動起動するモード
	ショート	保守モード: 対話形式でブートローダの機能を使用する動作モード
JP2	オープン	SD インターフェース(CON5)有効/JTAG インターフェース(CON7)無効
	ショート	SD インターフェース(CON5)無効/JTAG インターフェース(CON7)有効
JP3	オープン	NOR フラッシュブート: Armadillo-810 上の NOR フラッシュメモリのブートローダイメージを起動
	ショート	SD ブート: SD カード(CON5)のブートローダイメージを起動

21.1.3.11. SW1 リセットスイッチ

SW1 はリセットスイッチです。Armadillo-810 の外部リセットピンに接続されており、押下でリセットされます。

搭載スイッチ　　SKHLACA010/ALPS ELECTRIC

表 21.16 SW1 信号配列

ピン番号	信号名	I/O	機能
1	SW1	In	リセット、CON1 の 31 ピン、CON7 の 2 ピンと接続 押されていない状態: オープン、押された状態: GND とショート
2	GND	Power	電源(GND)

21.1.3.12. 電源回路の構成

電源回路の構成は次の通りです。

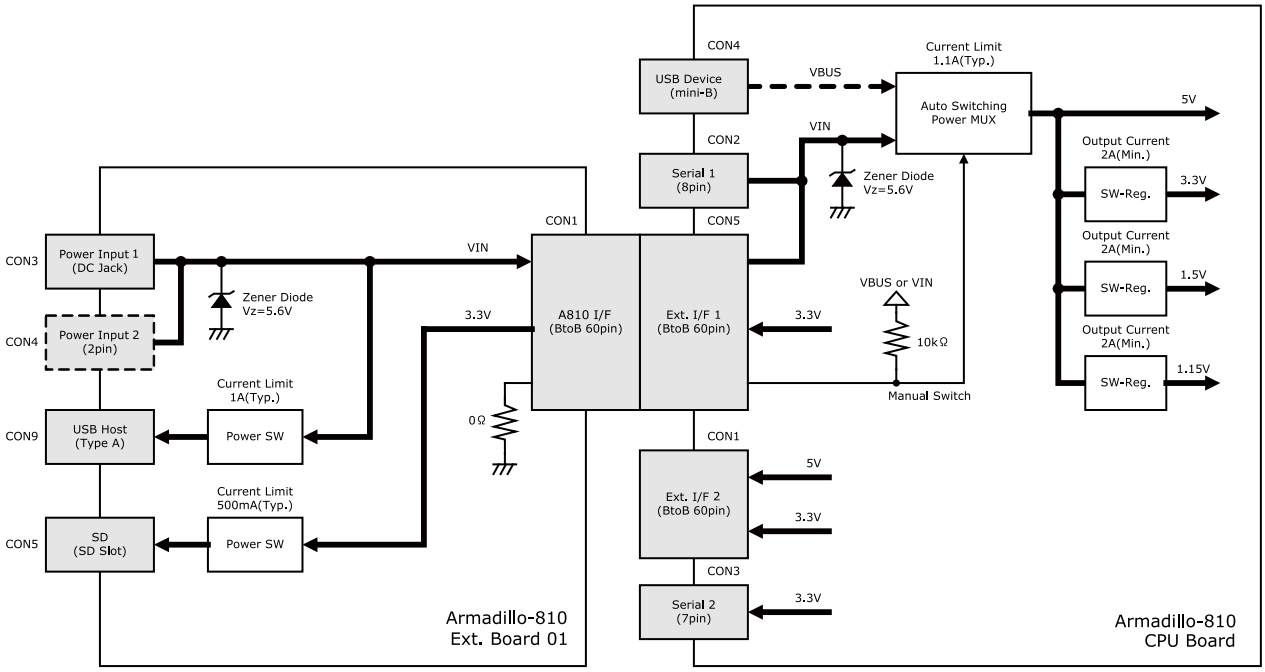


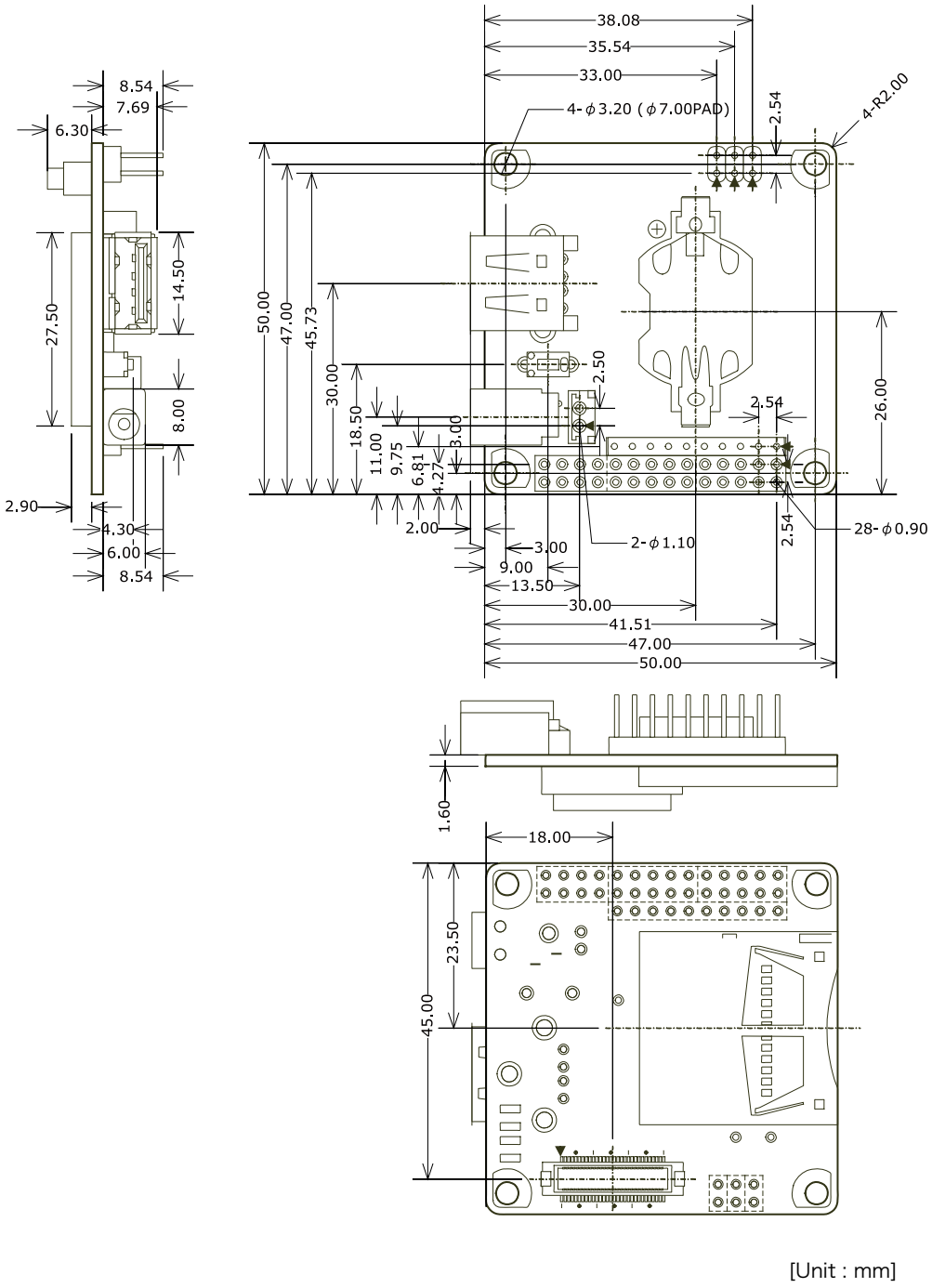
図 21.4 Armadillo-810 と Armadillo-810 拡張ボード 01 (A コネクタ用)接続時の電源回路の構成

Armadillo-810 と Armadillo-810 拡張ボード 01 (A コネクタ用)を接続した場合、Armadillo-810 拡張ボード 01 (A コネクタ用)の CON1 の 35 ピンが GND に接続されており、Armadillo-810 の CON4 からは電源供給がされなくなります。

過電流保護の IC を搭載しており、CON5 への供給可能電流は 500mA(Typ.)、CON9 への供給可能電流は 1A(Typ.)となります。

Armadillo-810 拡張ボード 01 (A コネクタ用)の CON3、CON4 および Armadillo-810 の CON2 から同時に電源を供給しないでください。故障の原因となります。

21.1.4. 基板形状図



[Unit : mm]

図 21.5 基板形状および固定穴寸法

21.2. Armadillo-810 カメラモジュールセット 01 (B コネクタ用)

21.2.1. 概要

Armadillo-810 カメラモジュール 01 (B コネクタ用)は、Armadillo-810 の拡張インターフェース 2 (B コネクタ)に接続可能なカメラモジュールです。

Armadillo-810 カメラモジュールセット 01 (B コネクタ用)の詳細につきましては、アットマークテクノ ユーザーズサイトよりダウンロードできるデータシートをご参照ください。

Armadillo-810 カメラモジュール 01 (B コネクタ用)の主な仕様は次の通りです。

表 21.17 Armadillo-810 カメラモジュール 01 (B コネクタ用)の仕様

撮像素子	1/4 インチ CMOS カラーイメージセンサ Omnivision Technologies 社製 OV7725
出力最大画素数	640 x 480 ピクセル VGA)
出力信号形式	YUV422(8Bit)
フレームレート	30fps
画像調整機能	AWB(Auto White Balance) 自動/手動 AGC(Auto Gain Control) 自動/手動 AEC(Auto Exprosure Control) 自動/手動
外部 IF 形式	SCCB(Standard Serial Camera Control Bus)
電源電圧	DC 3.3V
動作温度範囲	-20～+70℃(ただし結露なきこと)
外形サイズ	50 x 50 mm(突起部を除く)

表 21.18 レンズの仕様

	カメラレンズ(水平画角 79°)[a]	カメラレンズ(水平画角 120°)[b]
メーカー型番	HPB1007-A1	HPB1027-A1
焦点距離	2.9mm	1.95mm
F 値	2.0±5%	2.2±5%
画角	水平 79° 垂直 58°	水平 120° 垂直 97°
構成	4G IR カットフィルタ付き	6G IR カットフィルタ付き
ひずみ率	-4.9%	-13.1%
寸法	φ14.0 x 16.75mm	φ14.0 x 14.9mm

[a]標準搭載

[b]カメラモデル開発セットに同梱

Armadillo-810 カメラモジュール 01 (B コネクタ用)のブロック図は次の通りです。

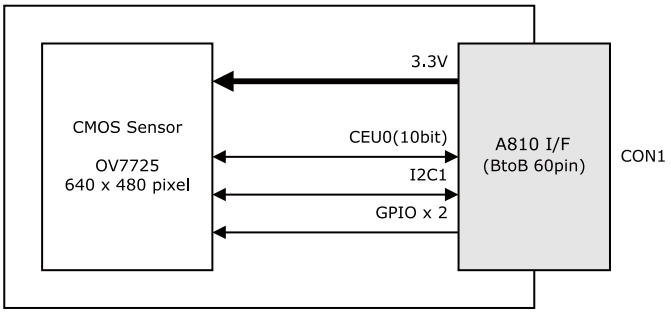
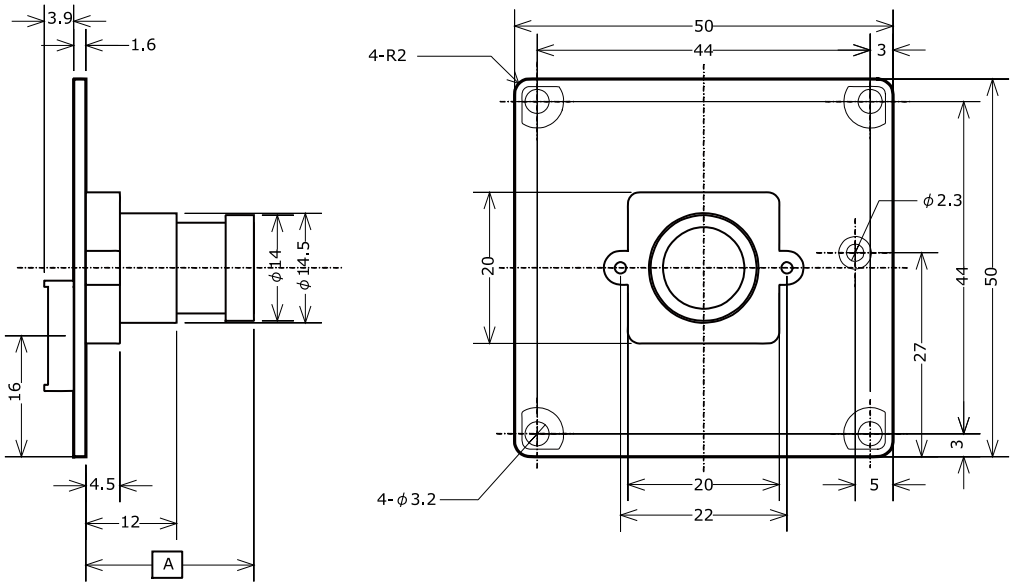


図 21.6 Armadillo-810 カメラモジュール 01 (B コネクタ用)のブロック図

Armadillo-810 カメラモジュールセット 01 (B コネクタ用)の詳細につきましては、アットマークテクノ ユーザーズサイトよりダウンロードできるデータシートをご参照ください。

21.2.2. 基板形状図



Lens	A
HPB1007-A1	22.22±1.0mm
HPB1027-A1	21.07±1.0mm

[Unit : mm]

図 21.7 基板形状および固定穴寸法

21.3. 開発用 USB シリアル変換アダプタ

開発用 USB シリアル変換アダプタは、FT232RL を搭載した USB-シリアル変換アダプタです。シリアルの信号レベルは 3.3V CMOS です。Armadillo-810 のシリアルインターフェース 2(CON3)に接続して使用することが可能です。

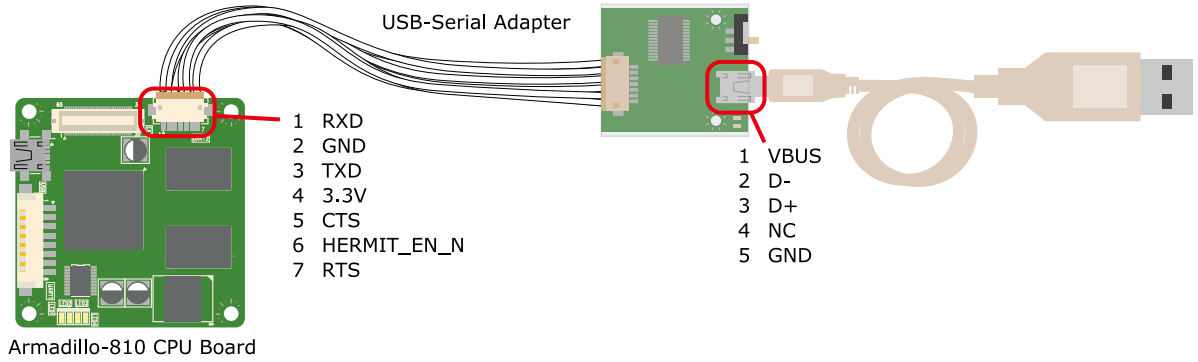


図 21.8 開発用 USB シリアル変換アダプタの配線

スライドスイッチにより、Armadillo-810 の起動モードを設定することができます。

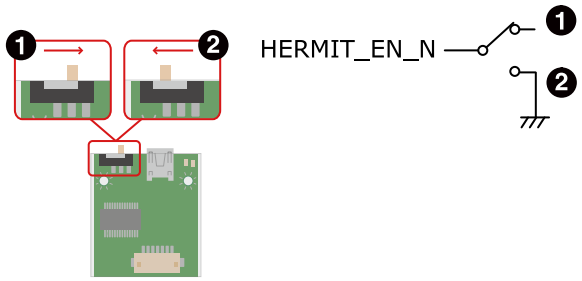


図 21.9 スライドスイッチについて

❶ OS 自動起動モード

❷ 保守モード

開発用 USB シリアル変換アダプタ(Armadillo-800 シリーズ対応)の取扱い上の注意

USB シリアル変換アダプタには電源投入順序があります。Armadillo-810 に接続する際は、以下の手順に従ってご使用ください。接続手順に従わない場合は、USB シリアル変換アダプタが故障する可能性がありますのでご注意ください。

1. 起動中の作業用 PC と USB シリアル変換アダプタを USB2.0 ケーブルで接続します。
2. Armadillo-810 のシリアルインターフェース 2 (CON3)に USB シリアル変換アダプタを接続します。
3. 上記接続を確認後、Armadillo-810 に電源を投入します。

また、Armadillo-810 に USB シリアル変換アダプタを接続した状態のまま、作業用 PC または USB シリアル変換アダプタから USB2.0 ケーブル

を抜く場合や作業用 PC をシャットダウンする場合は、Armadillo-810 の電源が切断されていることを確認してから行ってください。

21.4. D-Sub9/8 ピン シリアル変換ケーブル

D-Sub9/8 ピン シリアル変換ケーブルは、Armadillo-810 のシリアルインターフェース 1(CON2)を D-Sub9 ピンに変換するためのケーブルです。

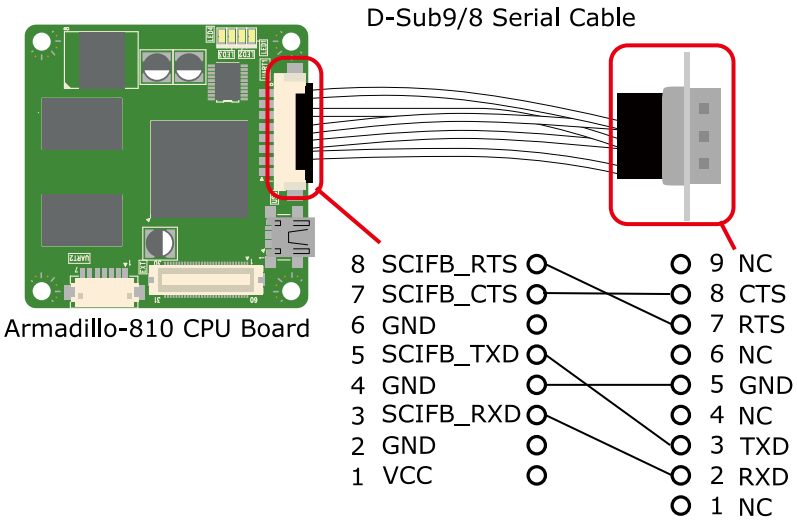


図 21.10 D-Sub9/8 ピン シリアル変換ケーブルの配線

22. ユーザー登録

アットマークテクノ製品をご利用のユーザーに対して、購入者向けの限定公開データの提供や大切なお知らせをお届けするサービスなど、ユーザー登録すると様々なサービスを受けることができます。サービスを受けるためには、「アットマークテクノ ユーザーズサイト」にユーザー登録をする必要があります。

ユーザー登録すると次のようなサービスを受けることができます。

- 製品仕様や部品などの変更通知の閲覧・配信
- 購入者向けの限定公開データのダウンロード
- 該当製品のバージョンアップに伴う優待販売のお知らせ配信
- 該当製品に関する開発セミナーやイベント等のお知らせ配信

詳しくは、「アットマークテクノ ユーザーズサイト」をご覧ください。

アットマークテクノ ユーザーズサイト

https://users.atmark-techno.com/

22.1. 購入製品登録

ユーザー登録完了後に、購入製品登録することで、「購入者向けの限定公開データ[1]」をダウンロードすることができるようになります。

Armadillo-810 購入製品登録

https://users.atmark-techno.com/armadillo-810/register

Armadillo-810 の購入製品登録を行うには、Armadillo-810 から取り出した「正規認証ファイル」をアットマークテクノ ユーザーズサイトからアップロードする必要があります。Armadillo-810 から正規認証ファイル(board-info.txt)を取り出す手順は以下の通りです。

作業用 PC の OS によって手順が異なりますので、ご注意ください。

22.1.1. 正規認証ファイルを取り出す手順(作業用 PC の OS が Linux)

作業用 PC の OS が Linux の場合、minicom を使用して正規認証ファイルを取得します。

1. 作業用 PC から minicom を立ち上げて、Armadillo-810 に root ユーザーでログインします。デバイスファイル名(/dev/ttyUSB0)は、ご使用の作業用 PC により異なる場合があります。開発用 USB シリアル変換アダプタが接続されている USB ポートのデバイスファイル名に適宜置き換えて下さい。

```
[PC ~]$ LANG=C minicom --noinit --wrap --device /dev/ttyUSB0
```

[1]拡張ボードの回路図データや、ミドルウェアパッケージなど

```
armadillo810-0 login: root
Password:
[root@armadillo810-0 (ttySC2) ~]#
```

2. "get-board-info-a810"コマンドを実行して正規認証ファイル(board-info.txt)を作成します。

```
[root@armadillo810-0 (ttySC2) ~]# get-board-info-a810
[root@armadillo810-0 (ttySC2) ~]# ls
board-info.txt
[root@armadillo810-0 (ttySC2) ~]#
```

3. "lsz"コマンドで正規認証ファイルを作業用 PC に転送します。

```
[root@armadillo810-0 (ttySC2) ~]# lsz board-info.txt
```

以下の画面が表示されたら任意のキーを入力します。

```
[root@armadillo810-0 (ttySC2) ~]# lsz board-info.txt

                +----------[zmodem download - Press CTRL-C to quit]-----------+
                |                                                             |
                |Receiving: board-info.txt                                    |
                |Bytes received:     775/    775   BPS:10318                  |
                |                                                             |
                |Transfer complete                                            |
                |                                                             |
                | READY: press any key to continue...                         |
                +-------------------------------------------------------------+

[root@armadillo810-0 (ttySC2) ~]#
```

4. minicom を終了させると作業用 PC に正規認証ファイルが転送されています。転送されたファイルのサイズが 775 バイトである事を確認してください。

```
[PC ~]$ ls -l board-info.txt
-rw-r--r--    1 atmark   atmark         775 Jan  1 09:08 board-info.txt
```

取り出した正規認証ファイルを「Armadillo-810 購入製品登録」ページの「正規認証ファイル」欄に指定し、アップロードしてください。

22.1.2. 正規認証ファイルを取り出す手順(作業用 PC の OS が Windows)

作業用 PC の OS が Windows の場合、TeraTerm を使用して正規認証ファイルを取得します。

1. TeraTerm を立ち上げ、正規認証ファイルの格納ディレクトリを設定します。TeraTermメニューの[ファイル] - [ディレクトリ変更]をクリックします。

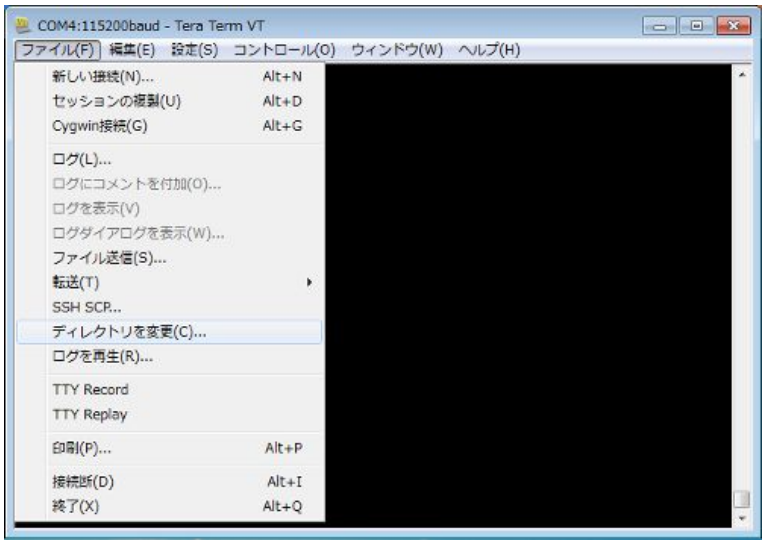


図 22.1 ディレクトリ変更の選択

ディレクトリ変更のウィンドウに、任意のファイル格納先を指定してください。

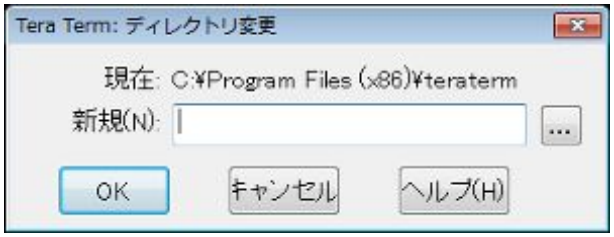


図 22.2 格納先の指定

2. TeraTerm から Armadillo-810 に root ユーザーでログインします。

```
armadillo810-0 login: root
Password:
[root@armadillo810-0 (ttySC2) ~]#
```

3. "get-board-info-a810"コマンドを実行して正規認証ファイル(board-info.txt)を作成します。

```
[root@armadillo810-0 (ttySC2) ~]# get-board-info-a810
[root@armadillo810-0 (ttySC2) ~]# ls
board-info.txt
[root@armadillo810-0 (ttySC2) ~]#
```

4. "lsz"コマンドで正規認証ファイルを作業用 PC に転送します。

```
[root@armadillo810-0 (ttySC2) ~]# lsz --disable-timeouts board-info.txt
```

コマンド実行後に、TeraTerm メニューの[ファイル] - [転送] - [ZMODEM] - [受信]を選択してください。

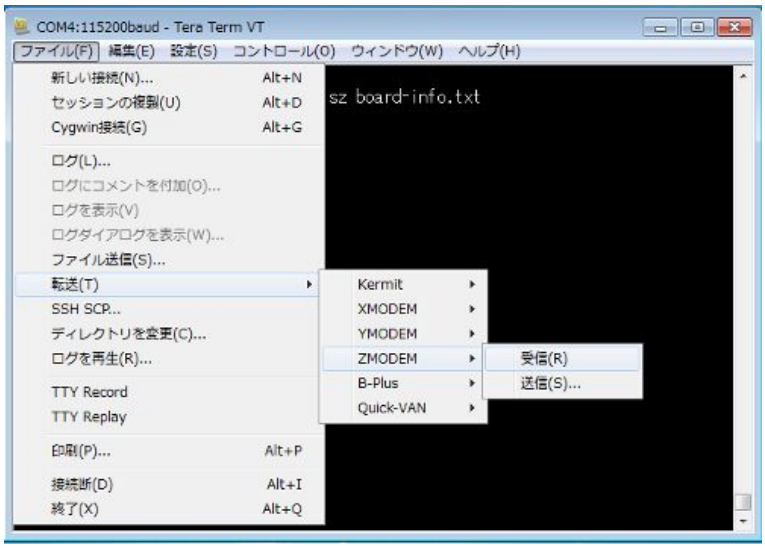


図 22.3 ZMODEM の選択

5.

```
[PC ~]$ ls board-info.txt
board-info.txt
```

先ほど設定したディレクトリに、正規認証ファイルが転送されています。転送されたファイルのサイズが 775 バイトであることを確認してください。

図 22.4 正規認証ファイルの確認

取り出した正規認証ファイルを「Armadillo-810 購入製品登録」ページの「正規認証ファイル」欄に指定し、アップロードしてください。

改訂履歴

バージョン	年月日	改訂内容
1.0.0	2013/02/14	・初版発行
1.0.1	2013/02/19	・「開発用 USB シリアル変換アダプタの取扱い上の注意」を追加
1.1.0	2013/03/28	・誤記修正 ・「21.1.3.1. CON1 Armadillo-810 接続インターフェース」を修正 ・「図 21.8. 開発用 USB シリアル変換アダプタの配線」を修正 ・「図 21.10. D-Sub9/8 ピン シリアル変換ケーブルの配線」を修正 ・「図 21.9. スライドスイッチについて」を修正 ・「22.1. 購入製品登録」の正規認証ファイルを取り出す手順を変更
1.1.1	2013/05/13	・フラッシュメモリのパーティション書き込み制限についての内容を追加 ・「表 21.7. CON5 信号配列」の誤記修正 ・表記ゆれ修正
1.2.0	2013/07/12	・誤記修正 ・ATDE5 のアーカイブ形式変更に伴う修正 ・「6.3. シリアル」から「ファイルを転送する」を削除 ・「表 21.7. CON5 信号配列」の誤記修正 ・「2.6. 電波障害について」の VCCI クラス A の情報を更新
1.3.0	2013/10/16	・「表 21.3. CON1 信号配列」の誤記修正 ・「表 21.4. CON2 信号配列」の誤記修正 ・「6.1.1.2. カメラ設定」を追記
1.4.0	2014/01/29	・AV コーデックミドルウェアに対応 ・「表 3.1. Armadillo-810 仕様」に RAM とフラッシュメモリの型番を追加 ・「表 4.5. Armadillo-810 のインターフェース内容」に USB mini B コネクタの型番を追加 ・「表 4.6. Armadillo-810 拡張ボード 01 (A コネクタ用)のインターフェース内容」に各コネクタの型番を追加 ・「表 6.14. direction の設定」の内容を修正 ・「19.1. インターフェースレイアウト」を追加 ・「19.2.1. CON1 拡張インターフェース 2 (B コネクタ)」に搭載コネクタの型番を追加 ・「表 19.4. CON1 拡張入出力ピンの信号状態 」を追加 ・「表 19.10. CON5 拡張入出力ピンの信号状態 」を追加 ・「19.3. 電気的仕様」を「19. ハードウェア仕様」に移動 ・「表 19.12. 絶対最大定格」にプルアップ/ダウン抵抗値を追加 ・「21.1.2. インターフェースレイアウト」を追加 ・「21.1.3.1. CON1 Armadillo-810 接続インターフェース」に搭載コネクタの型番を追加 ・「表 21.5. CON2 拡張入出力ピンのマルチプレクス」を追加 ・誤記、表記ゆれ修正
1.4.1	2014/07/31	・誤記修正

Armadillo-810 製品マニュアル
Version 1.4.1
2014/07/31

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