Armadillo-X1
製品マニュアル

AX1100-D00Z
AX1100-U01Z
AX1100-U00Z

Version 1.1.0
2016/12/08

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-X1 製品マニュアル

株式会社アットマークテクノ

Version 1.1.0
2016/12/08

目次

図目次

表目次

1. はじめに

このたびは Armadillo-X1 をご利用いただき、ありがとうございます。

Armadillo-X1 は、NXP Semiconductors 製アプリケーションプロセッサ「i.MX7Dual」を採用し、標準インターフェースとして、USB 2.0 ホストポートやギガビット・イーサネットポート、無線 LAN コンボモジュールを搭載したシングルボードコンピューターです。i.MX7Dual の CPU - ARM Cortex-A7 デュアルコア(1GHz) を最大限活用することができるように、標準 OS として Debian GNU/Linux を採用しています。

Armadillo-X1 では、Debian GNU/Linux がプリインストールされているため、オープンソースソフトウェアを含む多くのソフトウェア資産を活用し、自由にオリジナルのアプリケーションを開発することができます。開発言語としては、C/C++言語だけでなく、Oracle Java や Ruby なども利用することができるため、PC ライクな開発が可能です。

ハードウェアを拡張する方法は、2 つ用意されています。

1 つ目は、「アドオンモジュール」を利用します。Armadillo-X1 には、Armadillo-IoT ゲートウェイのセンサ接続用の「アドオンインターフェース」を備え、RS232C/422/485、接点入出力など一般的なセンサ接続に広く使われるインターフェースの他、EnOcean や Wi-SUN など新しい省電力無線通信規格に対応したアドオンモジュールを利用することができます。

2 つ目は、Armadillo-X1 に搭載されている 100 ピンの「拡張インターフェース」を利用します。このインターフェースには、組み込みシステムで求められる次の機能を利用できるように設計されています。これを利用した拡張基板を開発することで、様々なシステムに対応することができます。

- ギガネット・イーサネット
- USB 2.0 ホスト/デバイスインターフェース
- LCD インターフェース
- カメラインターフェース
- SD/SDIO インターフェース
- SPI
- GPIO　など

Armadillo-X1 は、ソフトウェアの柔軟な開発方法と、2 つのハードウェアの拡張機能を選択し利用することにより、お客様のビジネススタイルに合わせたシステム開発をスピーディーに、円滑に実現することが可能です。

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. 本書で扱うこと扱わないこと

1.1.1. 扱うこと

本書では、Armadillo-X1 の使い方、製品仕様(ソフトウェアおよびハードウェア)、オリジナルの製品を開発するために必要となる情報、その他注意事項について記載しています。Linux あるいは組み込み機器に不慣れな方でも読み進められるよう、コマンドの実行例なども記載しています。

また、Armadillo-X1 の機能をサポートする専用アプリケーションについても、その使い方を中心に説明しています。

Armadillo-X1 は一つの機器だけで完結するものではなく、接続するセンサや、クラウドシステムなどとの連携が不可欠です。そのため、参照すべきドキュメントも多岐に渡ります。本書では、アットマークテクノが運営する Armadillo サイトやユーザーズサイトを始め、開発に有用な情報を得る方法についても、随時説明しています。

1.1.2. 扱わないこと

本書では、一般的な Linux のプログラミング、デバッグ方法やツールの扱い方、各種モジュールの詳細仕様など、一般的な情報や、他に詳しい情報があるものは扱いません。また、(Armadillo-X1 を使用した)最終製品あるいはサービスに、固有な情報や知識も含まれていません。

1.2. 本書で必要となる知識と想定する読者

本書は、読者として Armadillo-X1 を使ってオリジナルのゲートウェイ機器を開発するエンジニアを想定して書かれています。また、「Armadillo-X1 を使うと、どのようなことが実現可能なのか 」を知りたいと考えている設計者・企画者も対象としています。Armadillo-X1 は組込みプラットフォームとして実績のある Armadillo をベースとしているため、標準で有効になっている機能以外にも様々な機能を実現することができます。

ソフトウェアエンジニア

端末からのコマンドの実行方法など、基本的な Linux の扱い方を知っているエンジニアを対象読者として想定しています。プログラミング言語として C/C++を扱えることは必ずしも必要ではありませんが、基礎的な知識がある方が理解しやすい部分もあります。

ハードウェアエンジニア

電子工学の基礎知識を有したエンジニアを対象読者として想定しています。回路図や部品表を読み、理解できる必要があります。

1.3. ユーザー限定コンテンツ

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と、製品をご購入いただいたユーザーに限定して公開している限定コンテンツにアクセスできるようになります。主な限定コンテンツには、下記のものがあります。

- 各種信頼性試験データ・納入仕様書等製造関連情報
- アドオンモジュール回路図

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。

1.4. 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-X1 ドキュメント・ダウンロード

http://armadillo.atmark-techno.com/armadillo-x1/downloads

1.5. 本書の構成

本書には、Armadillo-X1 をベースに、オリジナルの製品を開発するために必要となる情報を記載しています。また、取扱いに注意が必要な事柄についても説明しています。

◆ **はじめにお読みください。**

「1. はじめに」、「2. 注意事項」

◆ **Armadillo-X1 の仕様を紹介します。**

「3. 製品概要」

◆ **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. Armadillo の電源を入れる前に」、「5. 起動と終了」、「6. 動作確認方法」、

◆ **工場出荷状態のソフトウェア仕様について紹介します。**

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

◆ **システム開発に必要な情報を紹介します。**

「10. ビルド手順」、「11. イメージファイルの書き換え方法」、「12. 開発の基本的な流れ」

◆ **アドオンモジュールの開発や、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。**

「15. 電気的仕様」、「14. SD ブートの活用」、「16. インターフェース仕様」、「17. 基板形状図」、「18. アドオンモジュール」、「19. オプション品」、「20. 設計情報」

◆ **ソフトウェアのカスタマイズ方法を紹介します。**

「21. Howto」

◆ **ご購入ユーザーに限定して公開している情報の紹介やユーザー登録について紹介します。**

「22. ユーザー登録」

1.6. 表記について

1.6.1. 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

1.6.2. 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
=>	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

1.6.3. 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。

1.7. 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. 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。

2.2. 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	基板間コネクタ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(LAN、SD/SDIO、USB)以外へのコネクタやカードの着脱は、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おり、静電気により破壊されるおそれがあります。本製品を開封するときは、低湿度状態にならないよう注意し、静電防止用マットの使用、導電靴や人体アースなどによる作業者の帯電防止対策、備品の放電対策、静電気対策を施された環境下で行ってください。また、本製品を保管する際は、静電気を帯びやすいビニール袋やプラスチック容器などは避け、導電袋や導電性の容器・ラックなどに収納してください。
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。
使用場所の制限	テレビ・ラジオに近接する場所で使用すると、受信障害を招く恐れがあります。
電波に関する注意事項 (2.4GHz 帯無線)	2.4GHz 帯の電波を使用する機能(無線 LAN 等)は、自動ドアなどの自動制御電子機器に影響が出る場合、すぐに使用を中止してください。  この無線機(AEH-AR9462)は 2.4GHz 帯を使用します。変調方式として DS-SS 及び OFDM 方式を採用しています。

[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。

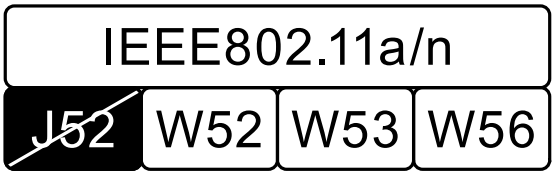
この無線機(AEH-AR9462)は 2.4GHz 帯を使用します。変調方式として FH-SS 方式を採用しています。

電波に関する注意事項(5GHz帯無線)

この無線機(AEH-AR9462)は 5GHz 帯を使用します。

W52、W53 の屋外での利用は電波法により禁じられています。

W53、W56 での AP モードは、2016 年 6 月現在工事設計認証を受けていないため使用しないでください。



5.0GHz(W52,W53): Indoor Use Only

2.3. 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。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。

本製品の標準出荷状態でプリインストールされている以下のソフトウェアは、オープンソースソフトウェアではありません。

・ ボード情報取得ツール(get_board_info)

2.4. 書込み禁止領域について

i.MX 7Dual 内蔵エレクトリカルヒューズ(e-Fuse)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。

2.5. 電波障害について

この装置は、クラス B 情報技術装置です。この装置は、家庭環境で使用することを目的としていますが、この装置がラジオやテレビジョン受信機に近接して使用されると、受信障害を引き起こすことがあります。取扱説明書に従って正しい取り扱いをして下さい。VCCI-B

この装置を、VCCI の技術基準に適合させるためには、DC ジャック(CON10)から AC アダプタで電源供給する必要があります。

アドオンモジュールは、モジュール単体で VCCI の適合確認試験を実施していません。Armadillo-X1 と接続することで、VCCI の技術基準に適合することを確認しています。アドオンモジュールは、Armadillo-X1 のアドオンインターフェースに接続し使用してください。

2.6. 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

2.7. 輸出について

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

- 当社製品および関連技術は、大量破壊兵器の開発等の軍事目的、その他国内外の法令により製造・使用・販売・調達が禁止されている機器には使用することができません。

2.8. 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。

2.9. 無線モジュールの安全規制について

本製品に搭載されている WLAN+BT コンボモジュール AEH-AR9462 は電気通信事業法に基づく設計認証を受けています。

また、本製品に搭載されている WLAN+BT コンボモジュール AEH-AR9462、ラインアップしている無線アドオンモジュールは、電波法に基づく工事設計認証を受けています。

これらの無線モジュールを国内で使用するときに無線局の免許は必要ありません。

以下の事項を行うと法律により罰せられることがあります。

- 無線モジュールやアンテナを分解/改造すること。
- 無線モジュールや筐体、基板等に直接印刷されている証明マーク・証明番号、または貼られている証明ラベルをはがす、消す、上からラベルを貼るなどし、見えない状態にすること。

認証番号は次の通りです。

表 2.1 WLAN+BT コンボモジュール: AEH-AR9462 適合証明情報

項目	内容
型式又は名称	AR5B22
電波法に基づく工事設計認証における認証番号	003WWA111393
電波法に基づく工事設計認証における認証番号	003WWA111394
電波法に基づく工事設計認証における認証番号	003GZA111395
電波法に基づく工事設計認証における認証番号	003XWA111396
電波法に基づく工事設計認証における認証番号	003YWA111397
電気通信事業法に基づく設計認証における認証番号	D111398003

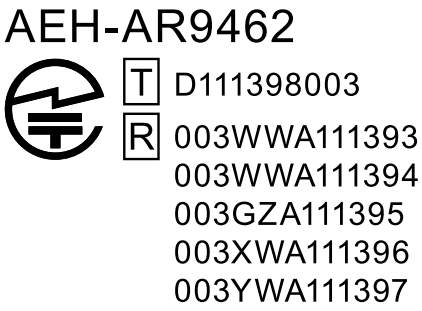


図 2.1 WLAN+BT コンボモジュール: AEH-AR9462 認証マーク

表 2.2 RN4020 アドオンモジュール: RN4020 適合証明情報

項目	内容
型式又は名称	RN4020
電波法に基づく工事設計認証における認証番号	201-140392

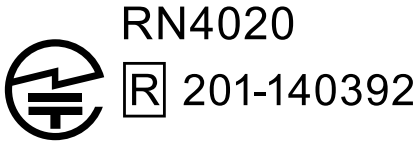


図 2.2 RN4020 アドオンモジュール: RN4020 認証マーク

表 2.3 EnOcean アドオンモジュール: BP35A3 適合証明情報

項目	内容
型式又は名称	BP35A3
電波法に基づく工事設計認証における認証番号	003-140290

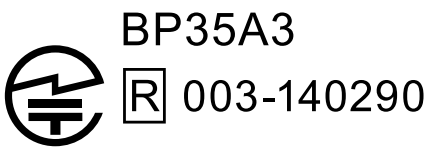
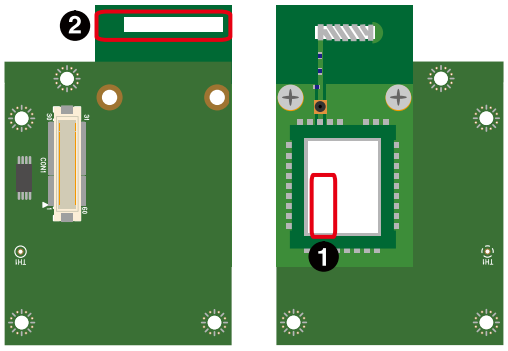


図 2.3 EnOcean アドオンモジュール: BP35A3 認証マーク

EnOcean アドオンモジュール には 2 つの認証マーク表示がありますが、BP35A3 の認証番号は 003-140290 です。



❶ シールに記載されている認証マーク及び認証番号（206-000372）は、BP35A3 が搭載しているモジュール、TCM410J としての認証番号です。

❷ BP35A3 の認証マーク及び認証番号（003-140290）はこちらに記載されています。

図 2.4 EnOcean アドオンモジュールの認証マーク

表 2.4 Wi-SUN アドオンモジュール: BP35A1 適合証明情報

項目	内容
型式又は名称	BP35A1
電波法に基づく工事設計認証における認証番号	003-140032

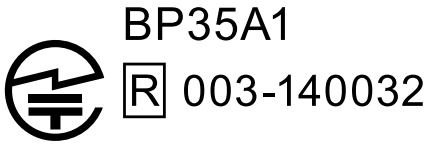


図 2.5 Wi-SUN アドオンモジュール: BP35A1 認証マーク

AEH-AR9462、RN4020 の各国電波法規制への対応情報は以下の通りです。

・ 当社製品は、原則として日本国内での使用を想定して開発・製造されています。
・ 海外の法令および規則への適合については当社はなんらの保証を行うものではありません。
・ 当社製品を輸出、または当社製品を組み込んだ最終製品を海外で販売する場合、日本国および関係する諸外国の関連法令・規制に従い、必要な手続を行っていただきますようお願いいたします。

表 2.5 AEH-AR9462 各国電波法規制への対応情報

項目	内容
FCC ID	2AE3B-AEH-AR9462
IC	20662-AEHAR9462

表 2.6 RN4020 各国電波法規制への対応情報

項目	内容
FCC ID	T9JRN4020
IC	6514A-RN4020

3. 製品概要

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-X1 とは

Armadillo-X1 は、NXP Semiconductors 製アプリケーションプロセッサ「i.MX7Dual」を採用し、標準インターフェースとして、USB 2.0 ホストポートやギガビット・イーサネットポート、無線 LAN コンボモジュールを搭載したシングルボードコンピューターです。i.MX7Dual の CPU - ARM Cortex-A7 デュアルコア(1GHz) を最大限活用することができるように、標準 OS として Debian GNU/Linux を採用しています。

Linux をベースとしたソフトウェアスタック

Debian GNU/Linux がプリインストールされているため、オープンソースソフトウェアを含む多くのソフトウェア資産を活用し、自由にオリジナルのアプリケーションを開発することができます。開発言語としては、C/C++言語だけでなく、Oracle Java や Ruby なども利用することができるため、PC ライクな開発が可能です。

２つの拡張用のインターフェースで機能拡張

１つ目は、「アドオンモジュール」を利用します。Armadillo-X1 には、Armadillo-IoT ゲートウェイのセンサ接続用の「アドオンインターフェース」を備え、RS232C/422/485、接点入出力な

ど一般的なセンサ接続に広く使われるインターフェースの他、EnOcean や Wi-SUN など新しい省電力無線通信規格に対応したアドオンモジュールを利用することができます。

2 つ目は、100 ピンの「拡張インターフェース」を利用します。このインターフェースには、組み込みシステムで求められる次の機能を利用できるように設計されています。これを利用した拡張基板を開発することで、様々なシステムに対応することができます。

- ギガネット・イーサネット
- USB 2.0 ホスト/デバイスインターフェース
- LCD インターフェース
- カメラインターフェース
- SD/SDIO インターフェース
- SPI
- GPIO　など

また、拡張用のインターフェース規格は公開されているため、オリジナルのモジュールを開発できます。拡張用モジュールのみを開発するだけで様々な要求に対応することができるため、CPU ボードから全て開発する場合に比べて、開発期間とコストを低減できます。

3.2. 製品ラインアップ

Armadillo-X1 の製品ラインアップは次の通りです。

表 3.1 Armadillo-X1 ラインアップ

名称	型番
Armadillo-X1 開発セット	AX1100-D00Z
Armadillo-X1 量産ボード (WLAN コンボ搭載)	AX1100-U01Z
Armadillo-X1 量産ボード (WLAN コンボ非搭載)	AX1100-U00Z

アドオンモジュールのラインアップは次の通りです。

表 3.2 アドオンモジュールラインアップ

名称	型番
Armadillo-IoT RS232C アドオンモジュール RS00	OP-AGA-RS00-00
Armadillo-IoT 絶縁 RS232C/422/485 アドオンモジュール RS01	OP-AGA-RS01-00
Armadillo-IoT 絶縁 RS485 アドオンモジュール RS02	OP-AGA-RS02-00
Armadillo-IoT RN4020 アドオンモジュール BT00	OP-AGA-BT00-00
Armadillo-IoT EnOcean アドオンモジュール EN00[a]	OP-AGA-EN00-00
Armadillo-IoT Wi-SUN アドオンモジュール WS00	OP-AGA-WS00-00
Armadillo-IoT 絶縁デジタル入出力/アナログ入力アドオンモジュール DA00	OP-AGA-DA00-00

[a]発売予定

3.2.1. Armadillo-X1 開発セット

Armadillo-X1 開発セット(型番: AX1100-D00Z)は、Armadillo-X1 を使った開発がすぐに開始できるように、開発に必要なものを一式含んだセットです。

表 3.3 Armadillo-X1 開発セットのセット内容

Armadillo-X1
WLAN+BT コンボモジュール
WLAN+BT コンボモジュール用外付けアンテナ x2
WLAN+BT コンボモジュール用アンテナケーブル x2

アンテナ固定金具
SD スロット拡張ボード
100 ピンコネクタ ピッチ変換基板
100 ピンコネクタ 延長ケーブル
USB シリアル変換アダプタ
USB2.0 ケーブル(A-miniB タイプ)
AC アダプタ(5V/2.0A EIAJ#2)
開発用 DVD-ROM
ネジ・スペーサー類

3.2.2. Armadillo-X1 量産用

Armadillo-X1 を使った製品の量産用モデルとして、Armadillo-X1 量産ボード (WLAN コンボ搭載)(型番: AX1100-U01Z)、Armadillo-X1 量産ボード (WLAN コンボ非搭載)(型番: AX1100-U00Z)をラインアップしています。

アドオンモジュールや無線 LAN モジュール、その他付属品など、量産時に必要なものを同時に発注することができます。

3.3. 仕様

Armadillo-X1 の主な仕様は次のとおりです。

表 3.4 仕様

型番	AX1100-D00Z	AX1100-U01Z	AX1100-U00Z
プロセッサ	NXP Semiconductors i.MX 7Dual		
	ARM Cortex-A7 x 2 - 命令/データキャッシュ 32KByte/32KByte - L2 キャッシュ 512KByte - 内部 SRAM 256KByte - メディアプロセッシングエンジン(NEON)搭載 - Thumb code(16bit 命令セット)サポート ARM Cortex-M4 - 命令/データキャッシュ 16KByte/16KByte		
システムクロック	CPU コアクロック(ARM Cortex-A7): 996MHz CPU コアクロック(ARM Cortex-M4): 240MHz DDR クロック: 533MHz 源発振クロック: 32.768kHz, 24MHz		
RAM	DDR3L: 512MByte バス幅 32bit		
ROM	QSPI NOR 型フラッシュメモリ: 8MByte eMMC(SLC Mode): 約 3.8GiB		
LAN(Ethernet)	RJ-45 x 1 1000BASE-T/100BASE-TX/10BASE-T, AUTO-MDIX 対応		
無線 LAN	WLAN+BT コンボモジュール: AEH-AR9462 搭載 IEEE 802.11a/b/g/n		非搭載
シリアル(UART)	3.3V CMOS x 1		
USB	USB 2.0 Host x 1 (High Speed)		
カレンダー時計	リアルタイムクロック 外部バックアップ用電源入力コネクタ搭載[a]		
アドオンモジュール[b]	CON7 アドオンインターフェースには SD スロット拡張ボードが搭載[c]	非搭載[d]	
スイッチ	ユーザースイッチ x 1		
LED	ユーザー LED x 1		
電源電圧	DC 5V±5%		

型番	AX1100-D00Z	AX1100-U01Z	AX1100-U00Z
消費電力(参考値)[e]	約 2.6W(待機時), 約 3.6W(通信時)[f]		約 1.6W(待機時、通信時共に)
使用温度範囲[g]	-20～70℃		
外形サイズ	100.0 x 64.8mm(突起部を除く)		

[a]電池は付属していません。
[b]アドオンモジュールは 1 個搭載可能です。
[c]アドオンインターフェースに接続可能な他のアドオンモジュールは別売です。
[d]アドオンモジュール(セミオーダーで選択可能)の接続が可能です。
[e]LAN、USB、シリアルコネクタにケーブル、デバイスを接続した状態での消費電力です。外部接続機器の消費分は含みません。
[f]通信時の電波環境により消費電力は変化します。
[g]結露無きこと。

3.4. Armadillo-X1 の外観

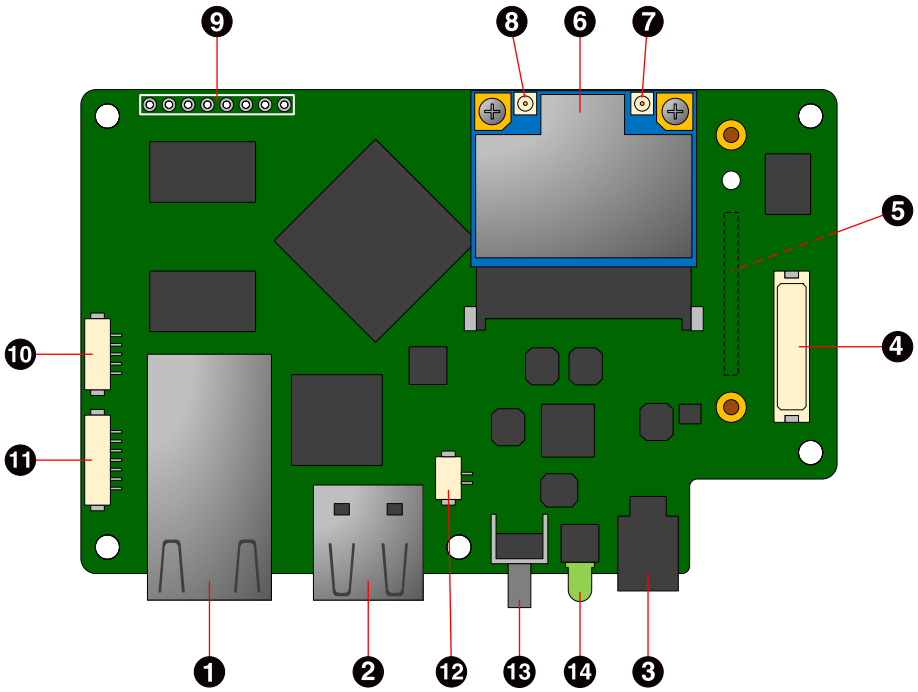


図 3.1 Armadillo-X1 の外観

表 3.5 各部名称と機能

番号	名称	説明
1	LAN コネクタ	LAN ケーブルを接続します。
2	USB コネクタ	USB メモリ等を接続します。
3	電源コネクタ	付属の AC アダプタを接続します。
4	アドオンコネクタ	機能拡張用のコネクタです。各種アドオンモジュールが接続できます。
5	拡張コネクタ	機能拡張用のコネクタです。
6	WLAN+BT コンボモジュール	AEH-AR9462/VoxMicro が搭載されています。
7	アンテナコネクタ(CH1)	付属のアンテナケーブルを接続します。
8	アンテナコネクタ(CH0)	付属のアンテナケーブルを接続します。

番号	名称	説明
9	JTAG インターフェース	ARM JTAG デバッガを接続します。
10	-	機能拡張用のコネクタです。
11	シリアルコネクタ	付属の USB シリアル変換アダプタを接続します。
12	RTC バックアップ電源コネクタ	外付けバッテリー等を接続します。
13	ユーザースイッチ	ユーザーで自由に機能を設定できるタクトスイッチです。
14	ユーザー LED	ユーザーで自由に機能を設定できる緑色 LED です。

3.5. ブロック図

Armadillo-X1 のブロック図は次のとおりです。

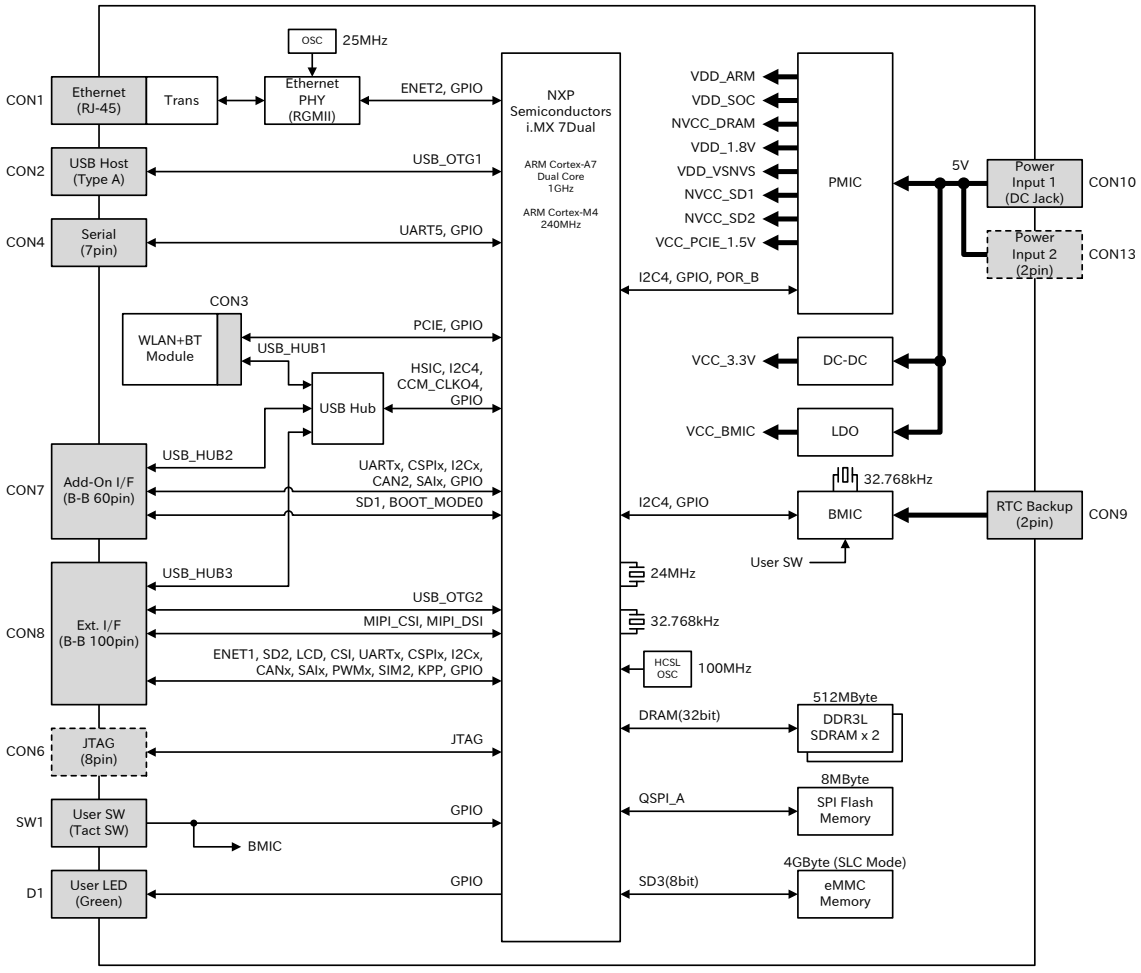


図 3.2 Armadillo-X1 ブロック図

3.6. ソフトウェア構成

Armadillo-X1 で動作するソフトウェアの構成について説明します。

Armadillo-X1 で利用可能なソフトウェアを「表 3.6. Armadillo-X1 で利用可能なソフトウェア」に示します。

表 3.6 Armadillo-X1 で利用可能なソフトウェア

ソフトウェア	説明
U-Boot	ブートローダーです。工場出荷状態ではブートローダーイメージは QSPI フラッシュメモリに配置されていますが、SD カードに配置することもできます。
Linux カーネル	uImage 形式の Linux カーネルイメージが利用可能です。工場出荷状態では Linux カーネルイメージは eMMC に配置されていますが、ブートローダーの機能により SD カード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。工場出荷状態では Debian GNU/Linux のルートファイルシステムは eMMC に配置されていますが、Linux カーネルがサポートしている SD カードなどのストレージデバイスに配置することもできます。

Armadillo-X1 の QSPI フラッシュメモリのメモリマップを「表 3.7. QSPI フラッシュメモリ メモリマップ」に示します。

表 3.7 QSPI フラッシュメモリ メモリマップ

物理アドレス	サイズ	説明
0x00000000 ｜ 0x001003FF	約 1 MByte	U-Boot ブートローダーイメージ
0x00100400 ｜ 0x001403FF	256 KBytes	ライセンス情報
0x00140400 ｜ 0x007FFFFF	約 6.7 MBytes	予約領域

Armadillo-X1 の eMMC のメモリマップを「表 3.8. eMMC メモリマップ」に示します。

表 3.8 eMMC メモリマップ

パーティション	サイズ	説明
1	32 MBytes	Linux カーネルイメージ/Device Tree Blob
2	約 3.4 GBytes	Debian GNU/Linux
3	128 MBytes	予約領域

4. Armadillo の電源を入れる前に

4.1. 準備するもの

Armadillo を使用する前に、次のものを必要に応じて準備してください。

作業用 PC	Linux または Windows が動作し、ネットワークインターフェースと 1 つ以上の USB ポートを持つ PC です。「4.2. 開発/動作確認環境の構築」を参照して、作業用 PC 上に開発/動作確認環境を構築してください。
ネットワーク環境	Armadillo と作業用 PC をネットワーク通信ができるようにしてください。
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 の動作を確認する場合などに利用します。
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[1]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。

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[2]
- 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。

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。本製品に対応している ATDE は、ATDE6 の v20160916 以降です。

ATDE6 は Debian GNU/Linux 8(コードネーム jessie)をベースに、Armadillo-X1 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-X1 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。

[1]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。
[2]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。

4.2.1. ATDE6 セットアップ

4.2.1.1. VMware のインストール

ATDE6 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、ATDE6 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 ATDE6 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

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. ATDE6 アーカイブの取得

ATDE6 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。

本製品に対応している ATDE6 のバージョンは v20160916 以降です。

作業用 PC の動作環境(ハードウェア、VMware、ATDE6 の対応アーキテクチャなど)により、ATDE6 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. ATDE6 アーカイブの展開

ATDE6 のアーカイブを展開します。ATDE6 のアーカイブは、tar.xz 形式の圧縮ファイルです。

Windows での展開方法を「手順 4.1. Windows で ATDE6 のアーカイブ展開する」に、Linux での展開方法を「手順 4.2. Linux で tar.xz 形式のファイルを展開する」に示します。

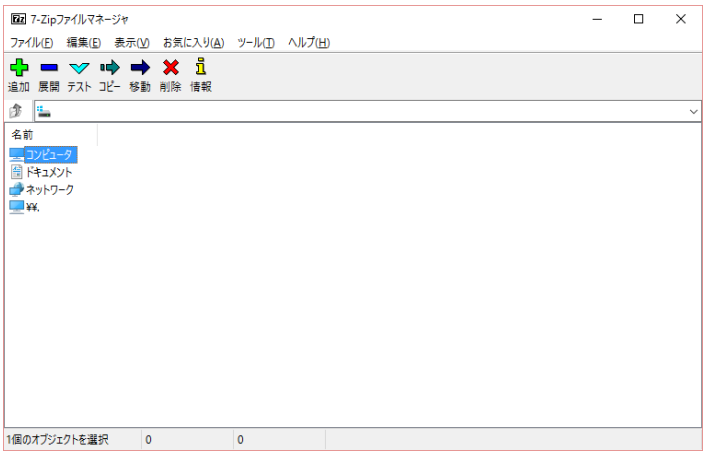
手順 4.1 Windows で ATDE6 のアーカイブ展開する

1. **7-Zip のインストール**

   7-Zip をインストールします。7-Zip は、圧縮解凍ソフト 7-Zip(http://sevenzip.sourceforge.jp)または、開発セット付属の DVD から取得可能です。

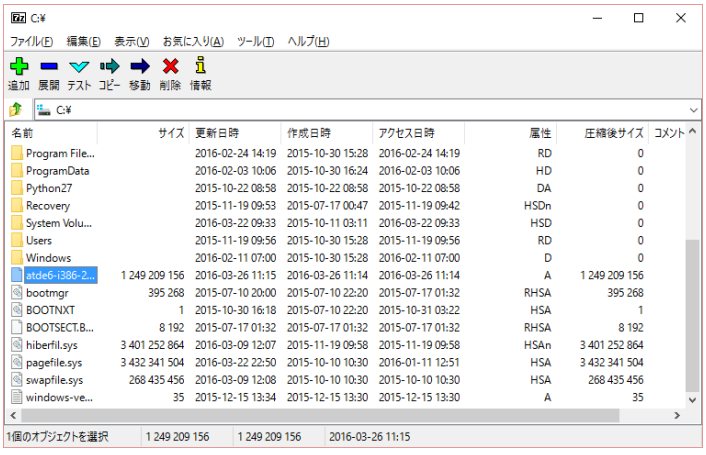
2. **7-Zip の起動**

   7-Zip を起動します。



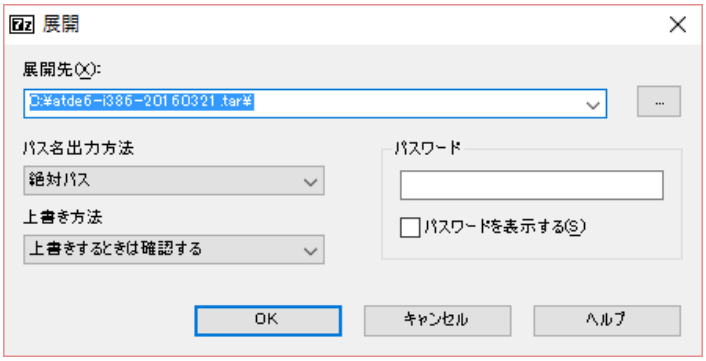
3. **xz 圧縮ファイルの選択**

   xz 圧縮ファイルを展開して、tar 形式のファイルを出力します。tar.xz 形式のファイルを選択して、「展開」をクリックします。



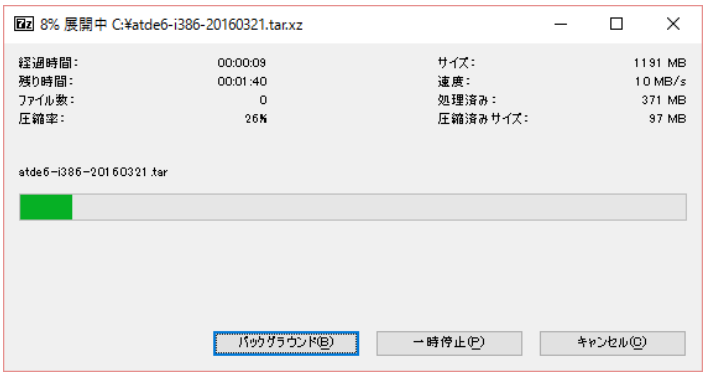
4. **xz 圧縮ファイルの展開先の指定**

   「展開先」を指定して、「OK」をクリックします。



5. **xz 圧縮ファイルの展開**

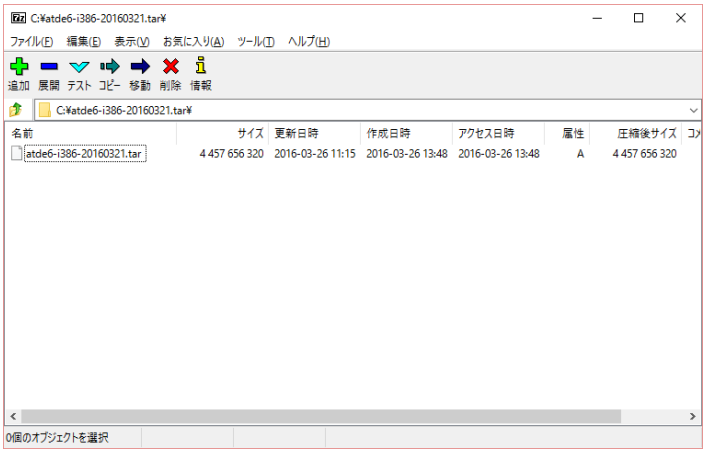
展開が始まります。



6. **tar アーカイブファイルの選択**

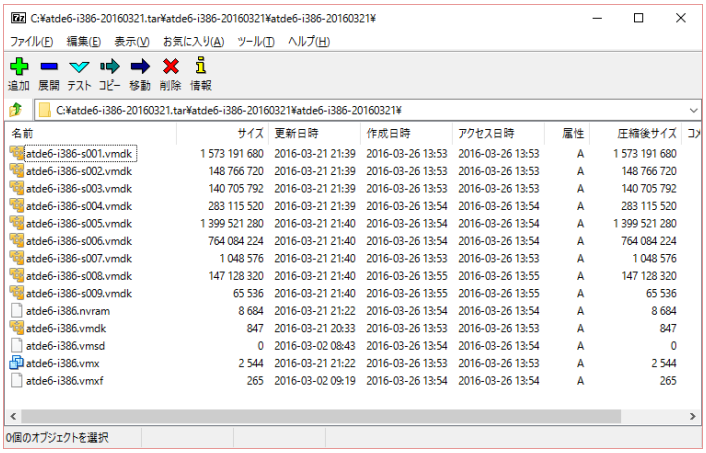
xz 圧縮ファイルの展開が終了すると、tar 形式のファイルが出力されます。

tar アーカイブファイルを出力したのと同様の手順で、tar アーカイブファイルから ATDE6 のデータイメージを出力します。tar 形式のファイルを選択して「展開」をクリックし、「展開先」を指定して、「OK」をクリックします。



7. **展開の完了確認**

tar アーカイブファイルの展開が終了すると、ATDE6 アーカイブの展開は完了です。「展開先」に指定したフォルダに ATDE6 のデータイメージが出力されています。



手順 4.2 Linux で tar.xz 形式のファイルを展開する

1. **tar.xz 圧縮ファイルの展開**

   tar の Jxf オプション使用して tar.xz 圧縮ファイルを展開します。

```
[PC ~]$ tar Jxf atde6-i386-[version].tar.xz
```

2. **展開の完了確認**

   tar.xz 圧縮ファイルの展開が終了すると、ATDE6 アーカイブの展開は完了です。atde6-i386-*[version]*ディレクトリに ATDE6 のデータイメージが出力されています。

```
[PC ~]$ ls atde6-i386-[version]/
atde6-i386.nvram      atde6-i386-s005.vmdk  atde6-i386.vmdk
atde6-i386-s001.vmdk  atde6-i386-s006.vmdk  atde6-i386.vmsd
atde6-i386-s002.vmdk  atde6-i386-s007.vmdk  atde6-i386.vmx
atde6-i386-s003.vmdk  atde6-i386-s008.vmdk  atde6-i386.vmxf
atde6-i386-s004.vmdk  atde6-i386-s009.vmdk
```

4.2.1.4. ATDE6 の起動

ATDE6 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、ATDE6 を起動することができます。ATDE6 にログイン可能なユーザーを、「表 4.1. ユーザー名とパスワード」に示します[3]。

表 4.1 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://

[3]特権ユーザーで GUI ログインを行うことはできません。

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。

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-X1 の動作確認を行うためには、「表 4.2. 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 4.2 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
USB シリアル変換アダプタ	Future Devices FT232R USB UART
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

ATDE6 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。ATDE6 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。

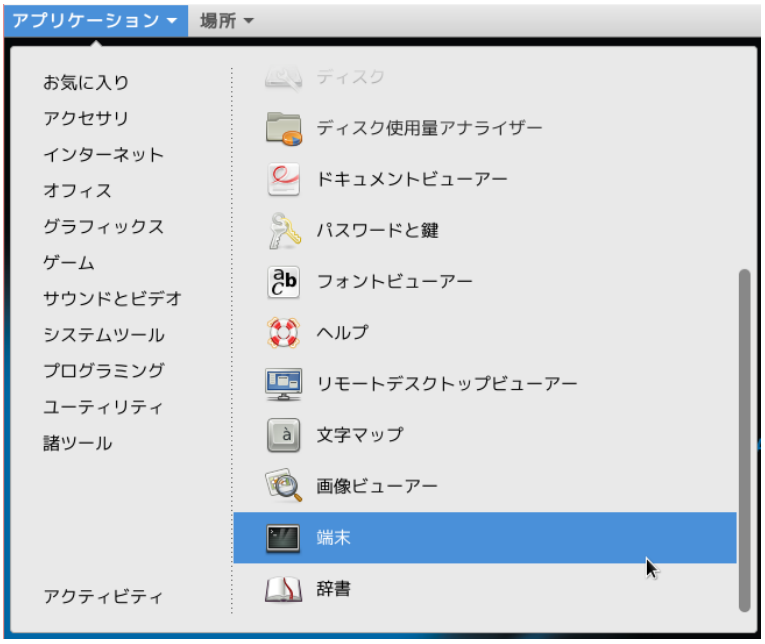


図 4.1 GNOME 端末の起動

「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 4.3. 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 4.3 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。

```
[PC ~]$ 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。

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

図 4.4 minicom 起動方法

デバイスファイル名は、環境によって/dev/ttyS0 や/dev/ttyUSB1 など、本書の実行例とは異なる場合があります。

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

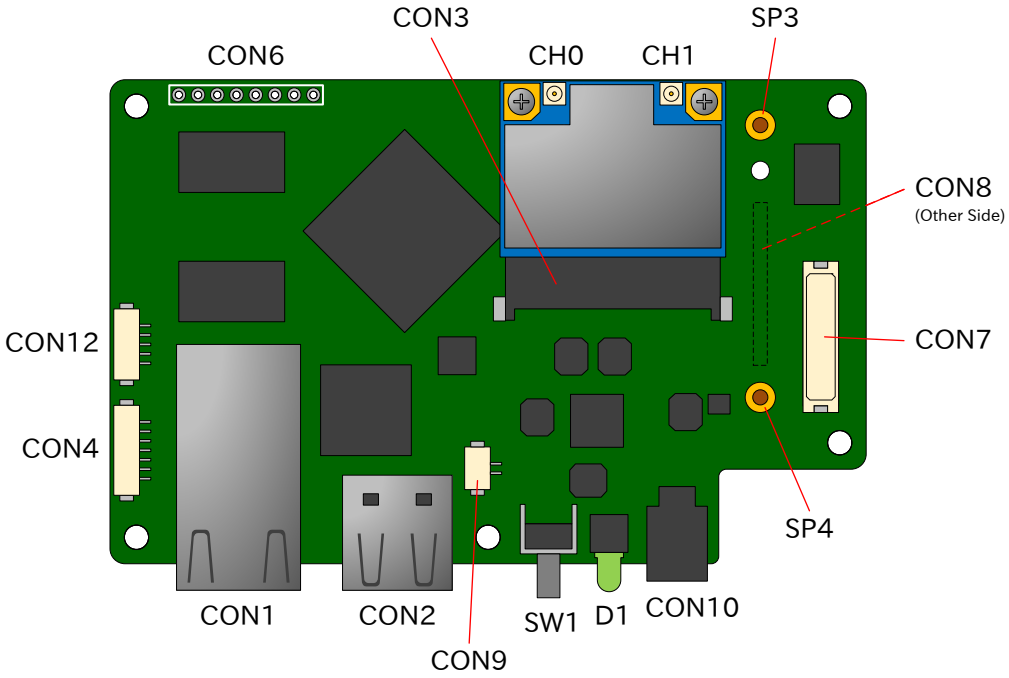


図 4.6 インターフェースレイアウト図

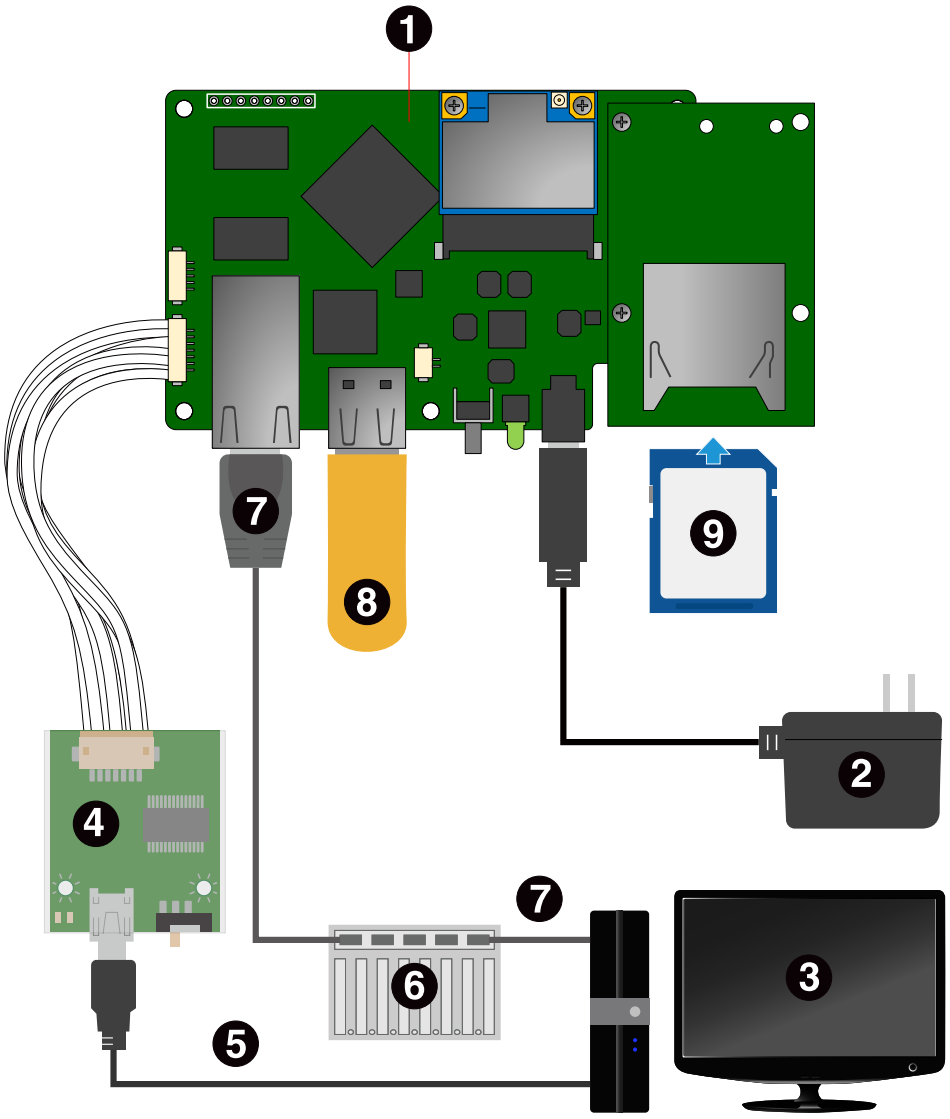
表 4.4 インターフェース内容

部品番号	インターフェース名	形状	備考
CON1	LAN インターフェース	RJ-45 コネクタ	
CON2	USB ホストインターフェース	Type A コネクタ	
CON3	WLAN インターフェース	PCI Express Mini Card コネクタ	挿抜寿命： 40 回[a]
CON4	シリアルインターフェース	ピンヘッダ 7 ピン(1.25mm ピッチ)	挿抜寿命： 40 回[a]
CON6	JTAG インターフェース	ピンヘッダ 8 ピン(2.54mm ピッチ)	コネクタ非搭載
CON7	アドオンインターフェース	基板間コネクタ 60 ピン(0.5mm ピッチ)	挿抜寿命： 40 回[a]
CON8	拡張インターフェース	基板間コネクタ 100 ピン(0.4mm ピッチ)	挿抜寿命： 20 回[a]
CON9	RTC バックアップインターフェース	ピンヘッダ 2 ピン(1.25mm ピッチ)	挿抜寿命： 20 回[a]
CON10	電源入力インターフェース 1	DC ジャック	対応プラグ: EIAJ#2
CON12	-	ピンヘッダ 5 ピン(1.25mm ピッチ)	
CH0	WLAN+BT アンテナインターフェース	U.FL コネクタ	挿抜寿命： 20 回[a]
CH1	WLAN アンテナインターフェース	U.FL コネクタ	挿抜寿命： 20 回[a]
SW1	ユーザースイッチ	タクトスイッチ	
D1	ユーザー LED	LED(緑色)	
SP3	アドオンモジュール用スタッド	スペーサー(M2, L=8mm)	
SP4			

[a]挿抜寿命は製品出荷時における目安であり、実際の挿抜可能な回数を保証するものではありません。

4.4. 接続方法

Armadillo-X1 と周辺装置の接続例を次に示します。



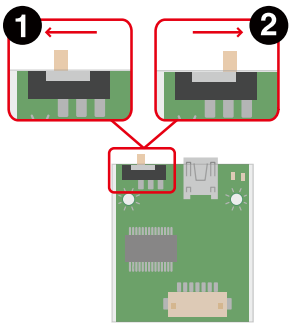
❶ Armadillo-X1 + SD スロット拡張ボード[4]
❷ AC アダプタ(5V/2.0A EIAJ#2)[4]
❸ 作業用 PC
❹ USB シリアル変換アダプタ[4]
❺ USB2.0 ケーブル(A-miniB タイプ)[4]
❻ LAN HUB
❼ LAN ケーブル
❽ USB メモリ
❾ SD カード

図 4.7 Armadillo-X1 の接続例

[4]Armadillo-X1 開発セット付属品

4.5. スライドスイッチの設定について

USB シリアル変換アダプタのスライドスイッチを操作することで、ブートローダーの起動モードを変更することができます。



❶ ブートローダーは保守モード[5]になります。

❷ ブートローダーはオートブートモード[6]になります。

図 4.8 スライドスイッチの設定

4.6. 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。

4.6.1. 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。

```
[PC ~]# vi [file]
```

図 4.9 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)を行います。vi はコマンドモードの状態で起動します。

[5]ブートローダーのコマンドプロンプトが起動します。
[6]OS を自動起動します。

4.6.2. 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 4.5. 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 4.5 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 4.10. 入力モードに移行するコマンドの説明」に示します。

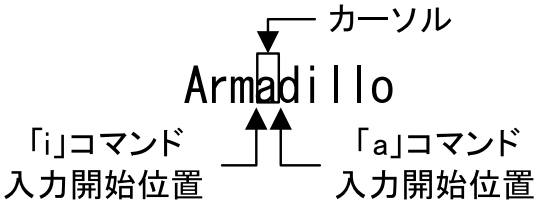


図 4.10 入力モードに移行するコマンドの説明

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、「4.6.4. 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。

4.6.3. 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 4.6. 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 4.6 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

4.6.4. 文字の削除

文字を削除する場合は、コマンドモードで「表 4.7. 文字の削除コマンド」に示すコマンドを入力します。

表 4.7 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 4.11. 文字を削除するコマンドの説明」に示します。

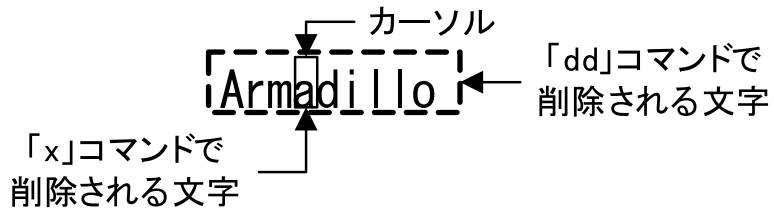


図 4.11 文字を削除するコマンドの説明

4.6.5. 保存と終了

ファイルの保存、終了を行うコマンドを「表 4.8. 保存・終了コマンド」に示します。

表 4.8 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-X1 に電源を接続するとき USB シリアル変換アダプタのスライドスイッチによって起動モードが変わります。 詳しくは「4.5. スライドスイッチの設定について」を参照してください。 本節では、保守モードに設定しているときの例を示します。 オートブートモードを選択した場合は、途中でコマンドを入力することなく起動が完了します。

また、CON7 アドオンインターフェースに、SD スロット拡張ボードを接続している場合は、「19.1.3. インターフェース仕様」を参照して SD スロット拡張ボードの SW1 を「NOMAL」に設定します。

```
U-Boot 2016.07-at3 (Sep 16 2016 - 15:29:22 +0900)

CPU:   Freescale i.MX7D rev1.1 996 MHz (running at 792 MHz)
CPU:   Extended Commercial temperature grade (-20C to 105C) at 45C
Reset cause: POR
       Watchdog enabled
I2C:   ready
DRAM:  512 MiB
Board Type: Armadillo-X1(0a100000)
Revison: 0001
S/N: 12
DRAM: 00001d05
XTAL: 00
X1 Addon EEPROM Detect
Atmark Techno Ext SD Slot Detect
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
SF: Detected N25Q512 with page size 256 Bytes, erase size 64 KiB, total 64 MiB
In:    serial
Out:   serial
Err:   serial
Found PFUZE300! deviceid 0x30, revid 0x11
Net:   FEC0
=>
```

図 5.1 電源投入直後のログ

Linux システムを起動するには、次のように "boot"コマンドを実行してください。コマンドを実行するとブートローダーが Linux システムを起動させます。シリアル通信ソフトウェアには Linux の起動ログが表示されます。

```
=> boot
switch to partitions #0, OK
mmc1(part 0) is current device
switch to partitions #0, OK
mmc1(part 0) is current device
reading boot.scr
```

```
** Unable to read file boot.scr **
reading boot.scr
** Unable to read file boot.scr **
reading uImage
9784016 bytes read in 241 ms (38.7 MiB/s)
Booting from mmc ...
reading armadillo_x1.dtb
50454 bytes read in 17 ms (2.8 MiB/s)
## Booting kernel from Legacy Image at 82000000 ...
   Image Name:   Linux-3.14.76-at3
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    9783952 Bytes = 9.3 MiB
   Load Address: 80008000
   Entry Point:  80008000
   Verifying Checksum ... OK
## Flattened Device Tree blob at 84800000
   Booting using the fdt blob at 0x84800000
   Loading Kernel Image ... OK
   Using Device Tree in place at 84800000, end 8480f515

Starting kernel ...

Booting Linux on physical CPU 0x0
Linux version 3.14.76-at3 (atmark@atde6) (gcc version 4.9.2 ( 4.9.2-10) ) #1080 SMP PREEMPT Fri Sep
16 17:15:59 JST 2016
CPU: ARMv7 Processor [410fc075] revision 5 (ARMv7), cr=10c53c7d
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine model: Atmark-Techno Armadillo-X1 Board
cma: CMA: reserved 64 MiB at 9c000000
Memory policy: Data cache writealloc
PERCPU: Embedded 8 pages/cpu @9bbb5000 s8256 r8192 d16320 u32768
Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 130048
Kernel command line: console=ttymxc4,115200 root=/dev/mmcblk2p2 rootwait rw
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)
Memory: 433948K/524288K available (8734K kernel code, 511K rwdata, 7712K rodata, 2548K init, 441K
bss, 90340K reserved, 0K highmem)
Virtual kernel memory layout:
    vector  : 0xffff0000 - 0xffff1000   (   4 kB)
    fixmap  : 0xfff00000 - 0xfffe0000   ( 896 kB)
    vmalloc : 0xa0800000 - 0xff000000   (1512 MB)
    lowmem  : 0x80000000 - 0xa0000000   ( 512 MB)
    pkmap   : 0x7fe00000 - 0x80000000   (   2 MB)
    modules : 0x7f000000 - 0x7fe00000   (  14 MB)
      .text : 0x80008000 - 0x81017d58   (16448 kB)
      .init : 0x81018000 - 0x81295040   (2549 kB)
      .data : 0x81296000 - 0x81315dc0   ( 512 kB)
       .bss : 0x81315dcc - 0x8138422c   ( 442 kB)
SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=2, Nodes=1
Preemptible hierarchical RCU implementation.
    RCU restricting CPUs from NR_CPUS=4 to nr_cpu_ids=2.
RCU: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=2
NR_IRQS:16 nr_irqs:16 16
Switching to timer-based delay loop
sched_clock: 32 bits at 3000kHz, resolution 333ns, wraps every 1431655765682ns
Architected cp15 timer(s) running at 8.00MHz (phys).
sched_clock: 56 bits at 8MHz, resolution 125ns, wraps every 2147483648000ns
```

```
Ignoring duplicate/late registration of read_current_timer delay
Console: colour dummy device 80x30
Calibrating delay loop (skipped), value calculated using timer frequency.. 6.00 BogoMIPS (lpj=30000)
pid_max: default: 32768 minimum: 301
Mount-cache hash table entries: 1024 (order: 0, 4096 bytes)
Mountpoint-cache hash table entries: 1024 (order: 0, 4096 bytes)
CPU: Testing write buffer coherency: ok
/cpus/cpu@0 missing clock-frequency property
/cpus/cpu@1 missing clock-frequency property
CPU0: thread -1, cpu 0, socket 0, mpidr 80000000
Setting up static identity map for 0x8084fd48 - 0x8084fda0
CPU1: Booted secondary processor
CPU1: thread -1, cpu 1, socket 0, mpidr 80000001
Brought up 2 CPUs
SMP: Total of 2 processors activated (12.00 BogoMIPS).
CPU: All CPU(s) started in SVC mode.
devtmpfs: initialized
VFP support v0.3: implementor 41 architecture 2 part 30 variant 7 rev 5
pinctrl core: initialized pinctrl subsystem
regulator-dummy: no parameters
NET: Registered protocol family 16
DMA: preallocated 256 KiB pool for atomic coherent allocations
cpuidle: using governor ladder
cpuidle: using governor menu
Use WDOG1 as reset source
GPIO line 13 (MCU_INTB) hogged as input
GPIO line 60 (LAN1_INT_N) hogged as input
GPIO line 62 (LAN1_COMA_MODE) hogged as output/low
syscon 30340000.iomuxc-gpr: regmap [mem 0x30340000-0x3034ffff] registered
syscon 30350000.ocotp-ctrl: regmap [mem 0x30350000-0x3035ffff] registered
syscon 30360000.anatop: regmap [mem 0x30360000-0x3036ffff] registered
vdd1p0d: 800 <--> 1200 mV at 1000 mV
vdd1p2: 1100 <--> 1300 mV
syscon 30390000.src: regmap [mem 0x30390000-0x3039ffff] registered
DDR type is DDR3!
prom_parse: Bad cell count for /regulators0/regulator@0
hw-breakpoint: found 5 (+1 reserved) breakpoint and 4 watchpoint registers.
hw-breakpoint: maximum watchpoint size is 8 bytes.
imx7d-pinctrl 302c0000.iomuxc-lpsr: initialized IMX pinctrl driver
imx7d-pinctrl 30330000.iomuxc: Invalid fsl,pins property in node /soc/aips-bus@30000000/
iomuxc@30330000/imx7d-sdb/uart7grp
imx7d-pinctrl 30330000.iomuxc: Invalid fsl,pins property in node /soc/aips-bus@30000000/
iomuxc@30330000/imx7d-sdb/ecspi1grp
imx7d-pinctrl 30330000.iomuxc: Invalid fsl,pins property in node /soc/aips-bus@30000000/
iomuxc@30330000/imx7d-sdb/usdhc1grp
imx7d-pinctrl 30330000.iomuxc: Invalid fsl,pins property in node /soc/aips-bus@30000000/
iomuxc@30330000/imx7d-sdb/usdhc1grp_100mhz
imx7d-pinctrl 30330000.iomuxc: Invalid fsl,pins property in node /soc/aips-bus@30000000/
iomuxc@30330000/imx7d-sdb/usdhc1grp_200mhz
imx7d-pinctrl 30330000.iomuxc: Invalid fsl,pins property in node /soc/aips-bus@30000000/
iomuxc@30330000/imx7d-sdb/usdhc1grp_power_off
imx7d-pinctrl 30330000.iomuxc: initialized IMX pinctrl driver
MU is ready for cross core communication!
bio: create slab <bio-0> at 0
mxs-dma 33000000.dma-apbh: initialized
USB_OTG1_VBUS: 5000 mV
vgaarb: loaded
i2c-core: driver [max17135] using legacy suspend method
```

```
i2c-core: driver [max17135] using legacy resume method
SCSI subsystem initialized
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
30800000.aips-bus:usbphy_nop1 supply vcc not found, using dummy regulator
30800000.aips-bus:usbphy_nop2 supply vcc not found, using dummy regulator
i2c i2c-2: IMX I2C adapter registered
gpio_bmic 3-0014: version: 2.0
bmic_regulator 3-0016: version: 1.0
BMIC_VREF: 1710 <--> 3600 mV at 2948 mV
i2c i2c-3: IMX I2C adapter registered
Linux video capture interface: v2.00
pps_core: LinuxPPS API ver. 1 registered
pps_core: Software ver. 5.3.6 - Copyright 2005-2007 Rodolfo Giometti <giometti@linux.it>
PTP clock support registered
MIPI CSI2 driver module loaded
Advanced Linux Sound Architecture Driver Initialized.
Bluetooth: Core ver 2.18
NET: Registered protocol family 31
Bluetooth: HCI device and connection manager initialized
Bluetooth: HCI socket layer initialized
Bluetooth: L2CAP socket layer initialized
Bluetooth: SCO socket layer initialized
cfg80211: Calling CRDA to update world regulatory domain
armadillo_iotg_addon addon: Atmark Techno SD board detected at Add-On Module I/F 1(Rev 0,
SerialNumber=6).
VDD_SD1: 3300 mV
USB_AOM_USB_VBUS: 5000 mV
armadillo_iotg_addon addon: No add-on expansion board detected at Add-On Module I/F 2.
Switched to clocksource arch_sys_counter
NET: Registered protocol family 2
TCP established hash table entries: 4096 (order: 2, 16384 bytes)
TCP bind hash table entries: 4096 (order: 3, 32768 bytes)
TCP: Hash tables configured (established 4096 bind 4096)
TCP: reno registered
UDP hash table entries: 256 (order: 1, 8192 bytes)
UDP-Lite hash table entries: 256 (order: 1, 8192 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.
imx rpmsg driver is registered.
Bus freq driver module loaded
futex hash table entries: 512 (order: 3, 32768 bytes)
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
Key type id_resolver registered
Key type id_legacy registered
jffs2: version 2.2. (NAND) c 2001-2006 Red Hat, Inc.
fuse init (API version 7.22)
msgmni has been set to 975
io scheduler noop registered
io scheduler deadline registered
io scheduler cfq registered (default)
```

```
imx-sdma 30bd0000.sdma: no event needs to be remapped
imx-sdma 30bd0000.sdma: loaded firmware 4.1
imx-sdma 30bd0000.sdma: initialized
pfuze100-regulator 3-0009: Full layer: 1, Metal layer: 1
pfuze100-regulator 3-0009: FAB: 0, FIN: 0
pfuze100-regulator 3-0009: pfuze3000 found.
SW1A: 700 <--> 1475 mV at 1100 mV
SW1B: 700 <--> 1475 mV at 1000 mV
SW2: 1500 <--> 1850 mV at 1800 mV
SW3: 900 <--> 1650 mV at 1350 mV
SWBST: 5000 <--> 5150 mV at 5000 mV
VSNVS: 1000 <--> 3000 mV at 3000 mV
VREFDDR: 750 mV
VLDO1: 1800 <--> 3300 mV at 1800 mV
VLDO2: 800 <--> 1550 mV at 1500 mV
VCCSD: 2850 <--> 3300 mV at 3300 mV
V33: 2850 <--> 3300 mV at 3300 mV
VLDO3: 1800 <--> 3300 mV at 3300 mV
VLDO4: 1800 <--> 3300 mV at 3300 mV
30a70000.serial: ttymxc4 at MMIO 0x30a70000 (irq = 62, base_baud = 1500000) is a IMX
console [ttymxc4] enabled
serial: Freescale lpuart driver
imx sema4 driver is registered.
[drm] Initialized drm 1.1.0 20060810
[drm] Initialized vivante 1.0.0 20120216 on minor 0
brd: module loaded
loop: module loaded
(hci_tty): inside hci_tty_init
(hci_tty): allocated 248, 0
fsl-quadspi 30bb0000.qspi: Unsupported cmd 0x65
fsl-quadspi 30bb0000.qspi: Unsupported cmd 0x61
fsl-quadspi 30bb0000.qspi: Unsupported cmd 0x65
fsl-quadspi 30bb0000.qspi: n25q512ax3 (65536 Kbytes)
3 ofpart partitions found on MTD device 30bb0000.qspi
Creating 3 MTD partitions on "30bb0000.qspi":
0x000000000000-0x000000100000 : "bootloader"
0x000000100000-0x000000140000 : "license"
0x000000140000-0x000004000000 : "reserved"
fsl-quadspi 30bb0000.qspi: QuadSPI SPI NOR flash driver
CAN device driver interface
30bf0000.ethernet supply phy not found, using dummy regulator
pps pps0: new PPS source ptp0
libphy: fec_enet_mii_bus: probed
fec 30bf0000.ethernet eth0: registered PHC device 0
PPP generic driver version 2.4.2
usbcore: registered new interface driver cdc_ether
usbcore: registered new interface driver cdc_eem
usbcore: registered new interface driver sierra_net
usbcore: registered new interface driver cdc_ncm
usbcore: registered new interface driver qmi_wwan
ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
ehci-pci: EHCI PCI platform driver
ehci-mxc: Freescale On-Chip EHCI Host driver
usbcore: registered new interface driver cdc_acm
cdc_acm: USB Abstract Control Model driver for USB modems and ISDN adapters
usbcore: registered new interface driver cdc_wdm
usbcore: registered new interface driver usb-storage
usbcore: registered new interface driver usbserial
```

```
usbcore: registered new interface driver usbserial_generic
usbserial: USB Serial support registered for generic
usbcore: registered new interface driver ftdi_sio
usbserial: USB Serial support registered for FTDI USB Serial Device
usbcore: registered new interface driver option
usbserial: USB Serial support registered for GSM modem (1-port)
usbcore: registered new interface driver sierra
usbserial: USB Serial support registered for Sierra USB modem
usbcore: registered new interface driver usb_serial_simple
usbserial: USB Serial support registered for zio
usbserial: USB Serial support registered for funsoft
usbserial: USB Serial support registered for flashloader
usbserial: USB Serial support registered for google
usbserial: USB Serial support registered for vivopay
usbserial: USB Serial support registered for moto_modem
usbserial: USB Serial support registered for hp4x
usbserial: USB Serial support registered for suunto
usbserial: USB Serial support registered for siemens_mpi
usb3503 3-0008: switched to HUB mode
usb3503 3-0008: usb3503_probe: probed in hub mode
30b10200.usbmisc supply vbus-wakeup not found, using dummy regulator
30b20200.usbmisc supply vbus-wakeup not found, using dummy regulator
30b30200.usbmisc supply vbus-wakeup not found, using dummy regulator
ci_hdrc ci_hdrc.0: EHCI Host Controller
ci_hdrc ci_hdrc.0: new USB bus registered, assigned bus number 1
ci_hdrc ci_hdrc.0: USB 2.0 started, EHCI 1.00
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 1 port detected
30b30000.usb supply vbus not found, using dummy regulator
ci_hdrc ci_hdrc.1: EHCI Host Controller
ci_hdrc ci_hdrc.1: new USB bus registered, assigned bus number 2
ci_hdrc ci_hdrc.1: USB 2.0 started, EHCI 1.00
hub 2-0:1.0: USB hub found
hub 2-0:1.0: 1 port detected
mousedev: PS/2 mouse device common for all mice
input: 30370000.snvs-pwrkey as /devices/soc/30000000.aips-bus/30370000.snvs-pwrkey/input/input0
snvs_pwrkey 30370000.snvs-pwrkey: i.MX snvs powerkey probed
i2c-core: driver [isl29023] using legacy suspend method
i2c-core: driver [isl29023] using legacy resume method
bmic_rtc 3-0011: version: 1.0
bmic_rtc 3-0011: rtc core: registered bmic_rtc as rtc0
snvs_rtc 30370034.snvs-rtc-lp: rtc core: registered 30370034.snvs-rtc-l as rtc1
i2c /dev entries driver
IR NEC protocol handler initialized
IR RC5(x) protocol handler initialized
IR RC6 protocol handler initialized
IR JVC protocol handler initialized
IR Sony protocol handler initialized
IR RC5 (streamzap) protocol handler initialized
IR SANYO protocol handler initialized
IR MCE Keyboard/mouse protocol handler initialized
usbcore: registered new interface driver uvcvideo
USB Video Class driver (1.1.1)
i2c-core: driver [mag3110] using legacy suspend method
i2c-core: driver [mag3110] using legacy resume method
bmic_thermal 3-0013: version: 1.0
imx2-wdt 30280000.wdog: timeout 10 sec (nowayout=0)
Bluetooth: HCI UART driver ver 2.2
```

```
Bluetooth: HCI H4 protocol initialized
Bluetooth: HCI BCSP protocol initialized
Bluetooth: HCILL protocol initialized
Bluetooth: HCIATH3K protocol initialized
usbcore: registered new interface driver bcm203x
usbcore: registered new interface driver btusb
usbcore: registered new interface driver ath3k
sdhci: Secure Digital Host Controller Interface driver
sdhci: Copyright(c) Pierre Ossman
sdhci-pltfm: SDHCI platform and OF driver helper
sdhci-esdhc-imx 30b60000.usdhc: allocated mmc-pwrseq
sdhci-esdhc-imx 30b60000.usdhc: could not get ultra high speed state, work on normal mode
mmc2: no vqmmc regulator found
mmc2: no vmmc regulator found
mmc2: SDHCI controller on 30b60000.usdhc [30b60000.usdhc] using ADMA
sdhci-esdhc-imx 30b40000.usdhc: Got CD GPIO #128.
sdhci-esdhc-imx 30b40000.usdhc: Got WP GPIO #129.
mmc0: no vqmmc regulator found
mmc0: SDHCI controller on 30b40000.usdhc [30b40000.usdhc] using ADMA
usb 2-1: new high-speed USB device number 2 using ci_hdrc
caam 30900000.caam: Instantiated RNG4 SH0
mmc2: BKOPS_EN bit is not set
mmc2: new high speed DDR MMC card at address 0001
mmcblk2: mmc2:0001 Q1J55L 3.56 GiB
mmcblk2boot0: mmc2:0001 Q1J55L partition 1 2.00 MiB
mmcblk2boot1: mmc2:0001 Q1J55L partition 2 2.00 MiB
mmcblk2rpmb: mmc2:0001 Q1J55L partition 3 4.00 MiB
caam 30900000.caam: Instantiated RNG4 SH1
 mmcblk2: p1 p2 p3
caam 30900000.caam: device ID = 0x0a160300 (Era 8)
caam 30900000.caam: job rings = 3, qi = 0
 mmcblk2boot1: unknown partition table
 mmcblk2boot0: unknown partition table
hub 2-1:1.0: USB hub found
hub 2-1:1.0: 3 ports detected
caam algorithms registered in /proc/crypto
hmac-sha1-caam alg registration failed
sha1-caam alg registration failed
hmac-sha224-caam alg registration failed
sha224-caam alg registration failed
hmac-sha256-caam alg registration failed
sha256-caam alg registration failed
hmac-md5-caam alg registration failed
md5-caam alg registration failed
xcbc-aes-caam alg registration failed
xcbc-aes-caam alg registration failed
caam_jr 30901000.jr0: registering rng-caam
snvs-secvio 30370000.caam-snvs: violation handlers armed - non-secure state
usbcore: registered new interface driver usbhid
usbhid: USB HID core driver
usbcore: registered new interface driver r8712u
bmic_adc 3-0012: version: 1.0
coresight-tmc 30086000.etr: TMC initialized
coresight-tmc 30084000.etf: TMC initialized
coresight-tpiu 30087000.tpiu: TPIU initialized
coresight-funnel 30083000.funnel: FUNNEL initialized
coresight-funnel 30041000.funnel: FUNNEL initialized
coresight-replicator replicator: REPLICATOR initialized
```

```
coresight-etm3x 3007c000.etm: ETM initialized
coresight-etm3x 3007d000.etm: ETM initialized
usbcore: registered new interface driver snd-usb-audio
NET: Registered protocol family 26
nf_conntrack version 0.5.0 (7804 buckets, 31216 max)
ipip: IPv4 over IPv4 tunneling driver
gre: GRE over IPv4 demultiplexor driver
ip_gre: GRE over IPv4 tunneling driver
ip_tables: (C) 2000-2006 Netfilter Core Team
TCP: cubic registered
Initializing XFRM netlink socket
NET: Registered protocol family 10
mip6: Mobile IPv6
ip6_tables: (C) 2000-2006 Netfilter Core Team
sit: IPv6 over IPv4 tunneling driver
ip6_gre: GRE over IPv6 tunneling driver
mmc0: new high speed SDHC card at address b368
mmcblk0: mmc0:b368 NCard 3.73 GiB
NET: Registered protocol family 17
 mmcblk0: p1 p2
NET: Registered protocol family 15
can: controller area network core (rev 20120528 abi 9)
NET: Registered protocol family 29
can: raw protocol (rev 20120528)
can: broadcast manager protocol (rev 20120528 t)
can: netlink gateway (rev 20130117) max_hops=1
Bluetooth: RFCOMM TTY layer initialized
Bluetooth: RFCOMM socket layer initialized
Bluetooth: RFCOMM ver 1.11
Bluetooth: BNEP (Ethernet Emulation) ver 1.3
Bluetooth: BNEP filters: protocol multicast
Bluetooth: BNEP socket layer initialized
Bluetooth: HIDP (Human Interface Emulation) ver 1.2
Bluetooth: HIDP socket layer initialized
8021q: 802.1Q VLAN Support v1.8
Key type dns_resolver registered
cpu cpu0: dev_pm_opp_get_opp_count: device OPP not found (-19)
ThumbEE CPU extension supported.
imx6q-pcie 33800000.pcie: PCI host bridge to bus 0000:00
pci_bus 0000:00: root bus resource [io  0x1000-0x10000]
pci_bus 0000:00: root bus resource [mem 0x40000000-0x4fefffff]
pci_bus 0000:00: root bus resource [bus 00-ff]
PCI: bus0: Fast back to back transfers disabled
PCI: bus1: Fast back to back transfers disabled
pci 0000:00:00.0: BAR 0: assigned [mem 0x40000000-0x400fffff]
pci 0000:00:00.0: BAR 14: assigned [mem 0x40100000-0x401fffff]
pci 0000:00:00.0: BAR 15: assigned [mem 0x40200000-0x402fffff pref]
pci 0000:00:00.0: BAR 6: assigned [mem 0x40300000-0x4030ffff pref]
pci 0000:01:00.0: BAR 0: assigned [mem 0x40100000-0x4017ffff 64bit]
pci 0000:01:00.0: BAR 6: assigned [mem 0x40200000-0x4020ffff pref]
pci 0000:00:00.0: PCI bridge to [bus 01]
pci 0000:00:00.0:   bridge window [mem 0x40100000-0x401fffff]
pci 0000:00:00.0:   bridge window [mem 0x40200000-0x402fffff pref]
pcieport 0000:00:00.0: Signaling PME through PCIe PME interrupt
pci 0000:01:00.0: Signaling PME through PCIe PME interrupt
PCI: enabling device 0000:01:00.0 (0140 -> 0142)
ieee80211 phy0: Atheros AR9462 Rev:2 mem=0xa0b00000, irq=157
regulator-dummy: disabling
```

```
imx mcc test is registered.
input: gpio-keys as /devices/gpio-keys/input/input1
bmic_rtc 3-0011: setting system clock to 1970-01-01 00:00:11 UTC (11)
ALSA device list:
  No soundcards found.
usb 2-1.1: new full-speed USB device number 3 using ci_hdrc
Warning: unable to open an initial console.
Freeing unused kernel memory: 2548K (81018000 - 81295000)
systemd-udevd[151]: starting version 215
random: systemd-udevd urandom read with 43 bits of entropy available
usb 2-1.1: string descriptor 0 read error: -22
usb 2-1.1: USB disconnect, device number 3
EXT4-fs (mmcblk2p2): mounted filesystem with ordered data mode. Opts: (null)
usb 2-1.1: new full-speed USB device number 4 using ci_hdrc
systemd[1]: systemd 215 running in system mode. (+PAM +AUDIT +SELINUX +IMA +SYSVINIT +LIBCRYPTSETUP ⏎
+GCRYPT +ACL +XZ -SECCOMP -APPARMOR)
systemd[1]: Detected architecture 'arm'.
usb 2-1.1: string descriptor 0 read error: -22

Welcome to Debian GNU/Linux 8 (jessie)!

systemd[1]: Set hostname to <armadillo>.
systemd[1]: Cannot add dependency job for unit display-manager.service, ignoring: Unit display- ⏎
manager.service failed to load: No such file or directory.
systemd[1]: Expecting device dev-ttymxc4.device...
         Expecting device dev-ttymxc4.device...
systemd[1]: Starting Forward Password Requests to Wall Directory Watch.
systemd[1]: Started Forward Password Requests to Wall Directory Watch.
systemd[1]: Starting Remote File Systems (Pre).
[  OK  ] Reached target Remote File Systems (Pre).
systemd[1]: Reached target Remote File Systems (Pre).
systemd[1]: Starting Encrypted Volumes.
[  OK  ] Reached target Encrypted Volumes.
systemd[1]: Reached target Encrypted Volumes.
systemd[1]: Starting Dispatch Password Requests to Console Directory Watch.
systemd[1]: Started Dispatch Password Requests to Console Directory Watch.
systemd[1]: Starting Paths.
[  OK  ] Reached target Paths.
systemd[1]: Reached target Paths.
systemd[1]: Starting Arbitrary Executable File Formats File System Automount Point.
[  OK  ] Set up automount Arbitrary Executable File Formats F...utomount Point.
systemd[1]: Set up automount Arbitrary Executable File Formats File System Automount Point.
systemd[1]: Starting Swap.
[  OK  ] Reached target Swap.
systemd[1]: Reached target Swap.
systemd[1]: Starting Root Slice.
[  OK  ] Created slice Root Slice.
systemd[1]: Created slice Root Slice.
systemd[1]: Starting User and Session Slice.
[  OK  ] Created slice User and Session Slice.
systemd[1]: Created slice User and Session Slice.
systemd[1]: Starting /dev/initctl Compatibility Named Pipe.
[  OK  ] Listening on /dev/initctl Compatibility Named Pipe.
systemd[1]: Listening on /dev/initctl Compatibility Named Pipe.
systemd[1]: Starting Delayed Shutdown Socket.
[  OK  ] Listening on Delayed Shutdown Socket.
systemd[1]: Listening on Delayed Shutdown Socket.
systemd[1]: Starting Journal Socket (/dev/log).
```

```
[  OK  ] Listening on Journal Socket (/dev/log).
systemd[1]: Listening on Journal Socket (/dev/log).
systemd[1]: Starting udev Control Socket.
[  OK  ] Listening on udev Control Socket.
systemd[1]: Listening on udev Control Socket.
systemd[1]: Starting udev Kernel Socket.
[  OK  ] Listening on udev Kernel Socket.
systemd[1]: Listening on udev Kernel Socket.
systemd[1]: Starting Journal Socket.
[  OK  ] Listening on Journal Socket.
systemd[1]: Listening on Journal Socket.
systemd[1]: Starting System Slice.
[  OK  ] Created slice System Slice.
systemd[1]: Created slice System Slice.
systemd[1]: Starting system-getty.slice.
[  OK  ] Created slice system-getty.slice.
systemd[1]: Created slice system-getty.slice.
systemd[1]: Starting system-serial\x2dgetty.slice.
[  OK  ] Created slice system-serial\x2dgetty.slice.
systemd[1]: Created slice system-serial\x2dgetty.slice.
systemd[1]: Starting Increase datagram queue length...
         Starting Increase datagram queue length...
systemd[1]: Mounted Huge Pages File System.
systemd[1]: Mounting Debug File System...
         Mounting Debug File System...
systemd[1]: Started Create list of required static device nodes for the current kernel.
systemd[1]: Starting Create Static Device Nodes in /dev...
         Starting Create Static Device Nodes in /dev...
systemd[1]: Starting Load Kernel Modules...
         Starting Load Kernel Modules...
systemd[1]: Starting udev Coldplug all Devices...
         Starting udev Coldplug all Devices...
systemd[1]: Started Set Up Additional Binary Formats.
systemd[1]: Mounted POSIX Message Queue File System.
systemd[1]: Starting Slices.
[  OK  ] Reached target Slices.
systemd[1]: Reached target Slices.
systemd[1]: Starting Remount Root and Kernel File Systems...
         Starting Remount Root and Kernel File Systems...
systemd[1]: Expecting device dev-mtdblock1.device...
         Expecting device dev-mtdblock1.device...
[  OK  ] Mounted Debug File System.
systemd[1]: Mounted Debug File System.
[  OK  ] Started Increase datagram queue length.
systemd[1]: Started Increase datagram queue length.
random: nonblocking pool is initialized
[  OK  ] Started Create Static Device Nodes in /dev.
systemd[1]: Started Create Static Device Nodes in /dev.
[  OK  ] Started Load Kernel Modules.
systemd[1]: Started Load Kernel Modules.
[  OK  ] Started Remount Root and Kernel File Systems.
systemd[1]: Started Remount Root and Kernel File Systems.
[  OK  ] Started udev Coldplug all Devices.
systemd[1]: Started udev Coldplug all Devices.
systemd[1]: Started Various fixups to make systemd work better on Debian.
systemd[1]: Starting Load/Save Random Seed...
         Starting Load/Save Random Seed...
systemd[1]: Mounting FUSE Control File System...
```

```
         Mounting FUSE Control File System...
systemd[1]: Starting Apply Kernel Variables...
         Starting Apply Kernel Variables...
systemd[1]: Mounting Configuration File System...
         Mounting Configuration File System...
systemd[1]: Starting udev Kernel Device Manager...
         Starting udev Kernel Device Manager...
systemd[1]: Starting Local File Systems (Pre).
[  OK  ] Reached target Local File Systems (Pre).
systemd[1]: Reached target Local File Systems (Pre).
systemd-udevd[249]: starting version 215
systemd[1]: Starting Local File Systems.
[  OK  ] Reached target Local File Systems.
systemd[1]: Reached target Local File Systems.
systemd[1]: Starting Create Volatile Files and Directories...
         Starting Create Volatile Files and Directories...
systemd[1]: Starting Remote File Systems.
[  OK  ] Reached target Remote File Systems.
systemd[1]: Reached target Remote File Systems.
systemd[1]: Starting Syslog Socket.
[  OK  ] Listening on Syslog Socket.
systemd[1]: Listening on Syslog Socket.
systemd[1]: Starting Journal Service...
         Starting Journal Service...
[  OK  ] Started Journal Service.
systemd[1]: Started Journal Service.
         Starting Trigger Flushing of Journal to Persistent Storage...
[  OK  ] Mounted Configuration File System.
[  OK  ] Mounted FUSE Control File System.
[  OK  ] Started udev Kernel Device Manager.
[  OK  ] Started Load/Save Random Seed.
[  OK  ] Started Apply Kernel Variables.
[  OK  ] Started Create Volatile Files and Directories.
[  OK  ] Started Trigger Flushing of Journal to Persistent Storage.
systemd-journald[260]: Received request to flush runtime journal from PID 1
[  OK  ] Found device /dev/ttymxc4.
[  OK  ] Found device /dev/mtdblock1.
         Starting Trigger Flushing of Journal to Persistent Storage...
[  OK  ] Created slice system-systemd\x2drfkill.slice.
         Starting Load/Save RF Kill Switch Status of rfkill1...
         Starting Load/Save RF Kill Switch Status of rfkill0...
         Mounting /opt/license...
         Starting Update UTMP about System Boot/Shutdown...
         Starting LSB: Raise network interfaces....
         Starting Copy rules generated while the root was ro...
[  OK  ] Mounted /opt/license.
[  OK  ] Started Load/Save RF Kill Switch Status of rfkill1.
[  OK  ] Started Load/Save RF Kill Switch Status of rfkill0.
[  OK  ] Started Copy rules generated while the root was ro.
systemd-journald[260]: Received request to flush runtime journal from PID 1
[  OK  ] Started Trigger Flushing of Journal to Persistent Storage.
[  OK  ] Started Update UTMP about System Boot/Shutdown.
[  OK  ] Started LSB: Raise network interfaces..
[  OK  ] Reached target Network.
[  OK  ] Reached target Network is Online.
[  OK  ] Reached target System Initialization.
[  OK  ] Listening on Avahi mDNS/DNS-SD Stack Activation Socket.
[  OK  ] Listening on D-Bus System Message Bus Socket.
```

```
[  OK  ] Reached target Sockets.
[  OK  ] Reached target Timers.
[  OK  ] Reached target Basic System.
         Starting Bluetooth service...
         Starting Restore /etc/resolv.conf if the system cras...s shut down....
         Starting Network Manager...
         Starting Regular background program processing daemon...
[  OK  ] Started Regular background program processing daemon.
         Starting Lighttpd Daemon...
         Starting Modem Manager...
         Starting /etc/rc.local Compatibility...
         Starting Login Service...
         Starting LSB: exim Mail Transport Agent...
         Starting Avahi mDNS/DNS-SD Stack...
         Starting D-Bus System Message Bus...
[  OK  ] Started D-Bus System Message Bus.
[  OK  ] Started Avahi mDNS/DNS-SD Stack.
[  OK  ] Started Bluetooth service.
[  OK  ] Reached target Bluetooth.
         Starting System Logging Service...
         Starting Permit User Sessions...
[  OK  ] Started Restore /etc/resolv.conf if the system crash...was shut down..
[  OK  ] Started Permit User Sessions.
[  OK  ] Started System Logging Service.
[  OK  ] Started Login Service.
         Starting Authenticate and Authorize Users to Run Privileged Tasks...
         Starting Hostname Service...
[  OK  ] Started Hostname Service.
[  OK  ] Started Authenticate and Authorize Users to Run Privileged Tasks.
[  OK  ] Started Lighttpd Daemon.
[  OK  ] Started /etc/rc.local Compatibility.
[  OK  ] Started Network Manager.
[  OK  ] Started Modem Manager.
fec 30bf0000.ethernet eth0: Freescale FEC PHY driver [Vitesse VSC8501]
(mii_bus:phy_addr=30bf0000.etherne:00, irq=-1)
         Starting change status LED...
         IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
Starting Getty on tty1...
[  OK  ] Started Getty on tty1.
         Starting Serial Getty on ttymxc4...
[  OK  ] Started Serial Getty on ttymxc4.
[  OK  ] Reached target Login Prompts.
[  OK  ] Started change status LED.
IPv6: ADDRCONF(NETDEV_UP): wlan0: link is not ready
         Starting WPA supplicant...
[  OK  ] Started WPA supplicant.
[  OK  ] Started LSB: exim Mail Transport Agent.
[  OK  ] Reached target Multi-User System.
[  OK  ] Reached target Graphical Interface.
         Starting Update UTMP about System Runlevel Changes...
[  OK  ] Started Update UTMP about System Runlevel Changes.

Debian GNU/Linux 8 armadillo ttymxc4

armadillo login:
```

図 5.2 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
atmark	atmark	一般ユーザ

初めて機器を接続したときは、必ず以下の手順に従って、初期パスワードを変更してください。

1. root でログイン

   初期パスワードを変更します。

```
[armadillo ~]# passwd
Enter new UNIX password: # 新しいパスワードを入力
Retype new UNIX password: # 再入力
```

2. atmark でログイン

   初期パスワードを変更します。

```
[armadillo ~]$ passwd
Enter new UNIX password: # 新しいパスワードを入力
Retype new UNIX password: # 再入力
[armadillo ~]$
```

Armadillo-X1 はネットワークに接続されることを前提としている機器ですので、 初期パスワードのままご利用になるとセキュリティリスクが非常に高まります。 セキュリティ強度の高いパスワードに変更され、その後も適切にパスワードを運用されることを強くお勧めします。

5.3. debian のユーザを管理する

1. ユーザを作成

   例として guest というユーザを作成します。

```
[armadillo ~]# adduser guest
Adding user `[user_name]' ...
Adding new group `guest' (1001) ...
Adding new user `guest' (1001) with group `guest' ...
Creating home directory `/home/guest' ...
Copying files from `/etc/skel' ...
Enter new UNIX password: # パスワードを入力
Retype new UNIX password: # 再入力
```

```
passwd: password updated successfully
Changing the user information for guest
Enter the new value, or press ENTER for the default
        Full Name []: # Enter を入力
        Room Number []: # Enter を入力
        Work Phone []: # Enter を入力
        Home Phone []: # Enter を入力
        Other []: # Enter を入力
Is the information correct? [Y/n] # Enter を入力
```

2. パスワードの変更

   例として guest ユーザのパスワードを変更します。

```
[armadillo ~]# passwd guest
Enter new UNIX password: # 新しいパスワードを入力
Retype new UNIX password: # 再入力
```

3. sudo を許可する

   例として guest ユーザに sudo を許可します。vi の使い方については、「4.6. vi エディタの使用方法」を参照にしてください。

```
[armadillo ~]# visudo
...
# User privilege specification
root    ALL=(ALL:ALL) ALL
guest   ALL=(ALL:ALL) ALL # この行を追加します
...
```

4. ユーザを削除

   例として guest ユーザを削除します。

```
[armadillo ~]# userdel guest
```

ホームディレクトリも合わせて消したいときは、"r"オプションをつけます。

```
[armadillo ~]# userdel -r guest
```

5.4. 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
         Starting Synchronise Hardware Clock to System Clock...
         Unmounting /opt/license
[  OK  ] Stopped target Bluetooth.
         Stopping User Manager for UID 0...
         Stopping WPA supplicant...
         Stopping Hostname Service...
         Stopping Authenticate and Authorize Users to Run Privileged Tasks...
         Stopping Bluetooth service...
[  OK  ] Stopped target Graphical Interface.
[  OK  ] Stopped target Multi-User System.
         Stopping Network Manager...
         Stopping Regular background program processing daemon...
IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
         Stopping Lighttpd Daemon...
IPv6: ADDRCONF(NETDEV_UP): wlan0: link is not ready
         Stopping Modem Manager...
[  OK  ] Stopped target Login Prompts.
         Stopping Getty on tty1...
         Stopping Serial Getty on ttymxc4...
         Stopping LSB: exim Mail Transport Agent...
         Stopping Avahi mDNS/DNS-SD Stack...
         Stopping D-Bus System Message Bus...
         Stopping System Logging Service...
[  OK  ] Stopped Bluetooth service.
[  OK  ] Stopped Network Manager.
[  OK  ] Stopped Regular background program processing daemon.
[  OK  ] Stopped Modem Manager.
[  OK  ] Stopped Avahi mDNS/DNS-SD Stack.
[  OK  ] Stopped D-Bus System Message Bus.
[  OK  ] Stopped System Logging Service.
[  OK  ] Stopped Hostname Service.
[  OK  ] Stopped Authenticate and Authorize Users to Run Privileged Tasks.
[  OK  ] Stopped Getty on tty1.
[  OK  ] Stopped Serial Getty on ttymxc4.
[  OK  ] Stopped Lighttpd Daemon.
[  OK  ] Stopped WPA supplicant.
[  OK  ] Stopped User Manager for UID 0.
[  OK  ] Unmounted /opt/license.
[  OK  ] Stopped LSB: exim Mail Transport Agent.
[  OK  ] Stopped target Network is Online.
         Stopping Login Service...
[  OK  ] Removed slice user-0.slice.
[  OK  ] Removed slice system-serial\x2dgetty.slice.
[  OK  ] Removed slice system-getty.slice.
         Stopping /etc/rc.local Compatibility...
[  OK  ] Stopped /etc/rc.local Compatibility.
[  OK  ] Stopped target Network.
         Stopping Permit User Sessions...
[  OK  ] Stopped Login Service.
[  OK  ] Stopped Permit User Sessions.
[  OK  ] Started Synchronise Hardware Clock to System Clock.
[  OK  ] Stopped target Remote File Systems.
[  OK  ] Stopped target Remote File Systems (Pre).
[  OK  ] Stopped target Basic System.
[  OK  ] Stopped target Slices.
[  OK  ] Stopped target Paths.
```

```
[  OK  ] Stopped target Timers.
[  OK  ] Stopped target Sockets.
[  OK  ] Closed Avahi mDNS/DNS-SD Stack Activation Socket.
[  OK  ] Closed Syslog Socket.
[  OK  ] Closed D-Bus System Message Bus Socket.
[  OK  ] Stopped target System Initialization.
         Stopping Load/Save RF Kill Switch Status of rfkill0...
         Stopping Load/Save RF Kill Switch Status of rfkill1...
         Stopping Update UTMP about System Boot/Shutdown...
         Stopping Apply Kernel Variables...
[  OK  ] Stopped Apply Kernel Variables.
         Stopping Load Kernel Modules...
[  OK  ] Stopped Load Kernel Modules.
[  OK  ] Stopped target Encrypted Volumes.
         Stopping LSB: Raise network interfaces....
[  OK  ] Stopped target Swap.
[  OK  ] Removed slice User and Session Slice.
[  OK  ] Stopped Load/Save RF Kill Switch Status of rfkill0.
[  OK  ] Stopped Load/Save RF Kill Switch Status of rfkill1.
[  OK  ] Stopped Update UTMP about System Boot/Shutdown.
[  OK  ] Stopped LSB: Raise network interfaces..
         Stopping Load/Save Random Seed...
         Stopping Create Volatile Files and Directories...
[  OK  ] Stopped Create Volatile Files and Directories.
[  OK  ] Stopped target Local File Systems.
         Unmounting /run/user/0...
[  OK  ] Removed slice system-systemd\x2drfkill.slice.
[  OK  ] Stopped Load/Save Random Seed.
[  OK  ] Unmounted /run/user/0.
[  OK  ] Reached target Unmount All Filesystems.
[  OK  ] Stopped target Local File Systems (Pre).
         Stopping Create Static Device Nodes in /dev...
[  OK  ] Stopped Create Static Device Nodes in /dev.
         Stopping Remount Root and Kernel File Systems...
[  OK  ] Stopped Remount Root and Kernel File Systems.
[  OK  ] Reached target Shutdown.
systemd-shutdown[1]: Sending SIGTERM to remaining processes...
systemd-journald[261]: Received SIGTERM from PID 1 (systemd-shutdow).
systemd-shutdown[1]: Sending SIGKILL to remaining processes...
systemd-shutdown[1]: Unmounting file systems.
systemd-shutdown[1]: Unmounting /sys/kernel/config.
systemd-shutdown[1]: Unmounting /sys/fs/fuse/connections.
systemd-shutdown[1]: Unmounting /sys/kernel/debug.
EXT4-fs (mmcblk2p2): re-mounted. Opts: (null)
EXT4-fs (mmcblk2p2): re-mounted. Opts: (null)
EXT4-fs (mmcblk2p2): re-mounted. Opts: (null)
systemd-shutdown[1]: All filesystems unmounted.
systemd-shutdown[1]: Deactivating swaps.
systemd-shutdown[1]: All swaps deactivated.
systemd-shutdown[1]: Detaching loop devices.
systemd-shutdown[1]: All loop devices detached.
systemd-shutdown[1]: Detaching DM devices.
systemd-shutdown[1]: All DM devices detached.
systemd-shutdown[1]: Halting system.
```

```
imx2-wdt 30280000.wdog: Device shutdown: Expect reboot!
reboot: System halted
```

図 5.3 終了方法

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. 動作確認を行う前に

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。最新版のイメージファイルは、Armadillo 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。

イメージファイルの書き換えについては、「11. イメージファイルの書き換え方法」を参照してください。

6.2. ネットワーク

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. 接続可能なネットワーク

Armadillo-X1 は、複数の種類のネットワークに接続することができます。接続可能なネットワークと Linux から使用するネットワークデバイスの対応を次に示します。

表 6.1 ネットワークとネットワークデバイス

ネットワーク	ネットワークデバイス	備考
有線 LAN	eth0	
無線 LAN	wlan0	AEH-AR9462-LX 搭載

6.2.2. ネットワークの設定方法

Armadillo-X1 では、通常の Linux システムと同様、ネットワークインターフェースの設定は NetworkManager を使用します。NetworkManager はデフォルトで eth0(LAN のネットワークインターフェース)が自動で up し、DHCP でネットワーク設定を取得するようになっています。

NetworkManager はすべてのネットワーク設定をコネクションとして管理します。コネクションには「どのようにネットワークへ接続するか」、「どのようにネットワークを作成するか」を記述し、`/etc/NetworkManager/system-connections/`に保存します。 また、1 つのデバイスに対して複数のコネクションを保存することは可能ですが、1 つのデバイスに対して有効化にできるコネクションは 1 つだけです。

NetworkManager は、従来の`/etc/network/interfaces` を使った設定方法もサポートしていますが、本書では **`nmcli`** を用いた方法を中心に紹介します。

6.2.2.1. nmcli について

nmcli は NetworkManager を操作するためのコマンドラインツールです。

「図 6.1. nmcli のコマンド書式」 に nmcli の書式を示します。このことから、nmcli は「オブジェクト(OBJECT)というものが存在し、それぞれのオブジェクトに対してコマンド(COMMAND)を実行する。」という書式でコマンドを入力することがわかります。また、オブジェクトそれぞれに help が用意されていることもここから読み取れます。

```
nmcli [ OPTIONS ] OBJECT { COMMAND | help }
```

図 6.1 nmcli のコマンド書式

各オブジェクトについての詳しい情報は **man nmcli** を参照してください。

Armadillo-X1 には nmcli の他ユーザーフレンドリーな nmtui もインストールされていますが本書では取り扱いません。

6.2.3. nmcli の基本的な使い方

ここでは **nmcli** の、基本的な使い方を説明します。

6.2.3.1. コネクションの一覧

登録されているコネクションの一覧を確認するには、次のようにコマンドを実行します。 [1]

```
[armadillo ~]# nmcli connection
NAME                UUID                                  TYPE            DEVICE
Wired connection 1  64e2e184-ede4-4cc6-ab70-0713d7cb0f0b  802-3-ethernet  eth0
```

図 6.2 コネクションの一覧

6.2.3.2. コネクションの有効化・無効化

コネクションを有効化するには、次のようにコマンドを実行します。

```
[armadillo ~]# nmcli connection up [ID]
```

図 6.3 コネクションの有効化

コネクションを無効化するには、次のようにコマンドを実行します。

```
[armadillo ~]# nmcli connection down [ID]
```

図 6.4 コネクションの無効化

6.2.3.3. コネクションの作成

コネクションを作成するには、次のようにコマンドを実行します。

[1] **nmcli connection show *[ID]*** によって、より詳細な情報を表示することもできます。

```
[armadillo ~]# nmcli connection add con-name [ID] \
type [type] ifname [interface name]
```

図 6.5 コネクションの作成

[ID] にはコネクションの名前(任意)、*[type]* には ethernet, wifi といった接続タイプ、*[interface name]* にはインターフェース名(デバイス)を入力します。具体的なコネクションの作成方法はそれぞれのデバイスの章で説明します。

/etc/NetworkManager/system-connections/に*[ID]*の名前でコネクションファイルが作成されます。これを vi などで編集しコネクションを修正することも可能です。

6.2.3.4. コネクションの削除

コネクションを削除するには、次のようにコマンドを実行します。

```
[armadillo ~]# nmcli connection delete [ID]
```

図 6.6 コネクションの削除

/etc/NetworkManager/system-connections/のコネクションファイルも同時に削除されます。

6.2.3.5. コネクションを修正する

具体的なコネクションの修正方法を紹介します。

wifi の設定情報を nmcli connection modify コマンドで設定情報を編集すると、パスフレーズ情報がリセットされます。 編集する際は、都度パスフレーズも同時に設定してください。 パスフレーズの設定方法は、「6.2.5. 無線 LAN」を参照してください。

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。

6.2.3.5.1. 固定 IP アドレスに設定する

「表 6.2. 固定 IP アドレス設定例」の内容に設定する例を、「図 6.7. 固定 IP アドレス設定」に示します。

表 6.2 固定 IP アドレス設定例

項目	設定
IP アドレス	192.0.2.10
マスク長	24
デフォルトゲートウェイ	192.0.2.1

```
[armadillo ~]# nmcli connection modify [ID] \
ipv4.method manual ipv4.addresses "192.0.2.10/24 192.0.2.1"
```

図 6.7 固定 IP アドレス設定

6.2.3.5.2. DHCP に設定する

DHCP に設定する例を、「図 6.8. DHCP 設定」に示します。

```
[armadillo ~]# nmcli connection modify [ID] \
ipv4.method auto -ipv4.addresses "192.0.2.10/24 192.0.2.1"
```

図 6.8 DHCP 設定

-ipv4.addresses のように、プロパティ名の先頭に '-' を付けることで設定したプロパティを削除することができます。反対に '+' を付けることでプロパティを追加することができます。

6.2.3.5.3. DNS サーバーを指定する

DNS サーバーを指定する例を、「図 6.9. DNS サーバーの指定」に示します。

```
[armadillo ~]# nmcli connection modify [ID] ipv4.dns 192.0.2.10
```

図 6.9 DNS サーバーの指定

6.2.3.6. コネクションの修正を反映する

有効化されているコネクションをコネクションを修正した場合、かならず修正したコネクションを再度有効化してください。

```
[armadillo ~]# nmcli connection down [ID]
[armadillo ~]# nmcli connection up [ID]
```

図 6.10 コネクションの修正の反映

6.2.3.7. デバイスの一覧

デバイスの一覧(デバイス名、タイプ、状態、有効なコネクション)を確認するには、次のようにコマンドを実行します。[2]

```
[armadillo ~]# nmcli device
DEVICE    TYPE      STATE         CONNECTION
eth0      ethernet  connected     Wired connection 1
ttyACM3   gsm       disconnected  --
wlan0     wifi      disconnected  --
gre0      gre       unmanaged     --
gretap0   gretap    unmanaged     --
ip6gre0   ip6gre    unmanaged     --
ip6tnl0   ip6tnl    unmanaged     --
tunl0     ipip      unmanaged     --
lo        loopback  unmanaged     --
sit0      sit       unmanaged     --
ip6_vti0  vti6      unmanaged     --
```

図 6.11 デバイスの一覧

6.2.3.8. デバイスの接続

デバイスを接続するには、次のようにコマンドを実行します。

```
[armadillo ~]# nmcli device connect [ifname]
```

図 6.12 デバイスの接続

デバイスを接続するには、接続しようとしているデバイスの有効なコネクションが必要です。"Error: neither a valid connection nor device given" というメッセージが表示された場合には、**nmcli connection** などで有効なコネクションがあるかを確認してください。

6.2.3.9. デバイスの切断

デバイスを切断するには、次のようにコマンドを実行します。

```
[armadillo ~]# nmcli device disconnect [ifname]
```

図 6.13 デバイスの切断

6.2.4. 有線 LAN

ここでは有線 LAN の使用方法について説明します。

[2] **nmcli device** と **nmcli device status** は等価です。

また、**nmcli device show** から、より詳細な情報を表示することができます。

6.2.4.1. 有線 LAN インターフェース(eth0)のコネクションの作成

有線 LAN インターフェース用のコネクションを作成するには、次のようにコマンドを実行します。

```
[armadillo ~]# nmcli connection add type ethernet ifname eth0
Connection 'ethernet-eth0' (ac491d33-9647-4096-8b91-5c7abcf5850d) successfully added.
```

図 6.14 有線 LAN インターフェース(eth0)のコネクションを作成

6.2.4.2. 有線 LAN のネットワーク設定を変更する

ネットワークの設定方法は「6.2.3.5. コネクションを修正する」を参照してください。コネクションを修正を行った後にはかならず「6.2.3.6. コネクションの修正を反映する」を参考に、修正の反映を行ってください。

6.2.4.3. 有線 LAN の接続を確認する

有線 LAN で正常に通信が可能か確認します。設定を変更した場合、かならず変更したインターフェースを再度有効化してください。

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。以下の例では、通信機器が「192.0.2.20」という IP アドレスを持っていると想定しています。

```
[armadillo ~]# ping 192.0.2.20
```

図 6.15 有線 LAN の PING 確認

有線 LAN 以外のコネクションが有効化されている場合、ネットワーク通信に有線 LAN が使用されない場合があります。確実に有線 LAN の接続確認をする場合は、事前に有線 LAN 以外のコネクションを無効化してください。

6.2.5. 無線 LAN

ここでは、Armadillo-X1 に搭載されている無線 LAN モジュールの使用方法について説明します。

例として、WPA2-PSK(AES)のアクセスポイントに接続します。WPA2-PSK(AES)以外のアクセスポイントへの接続方法などについては、**man nm-settings** を参考にしてください。また、以降の説明では、アクセスポイントの ESSID を*[essid]*、パスフレーズを*[passphrase]*と表記します。

6.2.5.1. 無線 LAN アクセスポイントに接続する

無線 LAN アクセスポイントに接続するためには、次のようにコマンドを実行します。

```
[armadillo ~]# nmcli device wifi connect [essid] password [passphrase]
```

図 6.16 無線 LAN アクセスポイントに接続する

6.2.5.2. 無線 LAN(wlan0)のコネクションの作成

「6.2.5.1. 無線 LAN アクセスポイントに接続する」方法は簡単ですが、この方法は DHCP にしか対応していません。そのため、無線 LAN を固定 IP にする場合や、細かなネットワーク設定を行うためにはコネクションから作成する必要があります。

無線 LAN(wlan0)のコネクションの作成するためには、次のようにコマンドを実行します。

```
[armadillo ~]# nmcli connection add type wifi ifname wlan0 ssid [essid] ❶
Connection 'wifi-wlan0' (d3cbb49d-b843-4dbf-94d5-7e7275449e8a) successfully added.
[armadillo ~]# nmcli connection modify wifi-wlan0 \ ❷
802-11-wireless-security.key-mgmt wpa-psk \ ❸
802-11-wireless-security.psk [passphrase]
```

図 6.17 無線 LAN(wlan0)のコネクションの作成

❶ type を wifi に設定し、無線 LAN のコネクションを作成します。

❷ 暗号化キー管理方式を wpa-psk に設定します。

❸ パスフレーズを設定します。

接続先のアクセスポイントによっては、以下のようなメッセージが出力され、アクセスポイントに接続できないことがあります。

```
wlan0: authenticate with 00:3a:9d:42:cc:92
wlan0: send auth to 00:3a:9d:42:cc:92 (try 1/3)
wlan0: authenticated
wlan0: associate with 00:3a:9d:42:cc:92 (try 1/3)
wlan0: RX AssocResp from 00:3a:9d:42:cc:92 (capab=0x431 status=0 aid=1)
wlan0: associated
cfg80211: Calling CRDA to update world regulatory domain
cfg80211: World regulatory domain updated:
cfg80211:  DFS Master region: unset
cfg80211:   (start_freq - end_freq @ bandwidth), (max_antenna_gain,
max_eirp)
cfg80211:   (2402000 KHz - 2472000 KHz @ 40000 KHz), (N/A, 2000 mBm)
cfg80211:   (2457000 KHz - 2482000 KHz @ 40000 KHz), (N/A, 2000 mBm)
cfg80211:   (2474000 KHz - 2494000 KHz @ 20000 KHz), (N/A, 2000 mBm)
cfg80211:   (5170000 KHz - 5250000 KHz @ 80000 KHz), (N/A, 2000 mBm)
cfg80211:   (5250000 KHz - 5330000 KHz @ 80000 KHz), (N/A, 2000 mBm)
cfg80211:   (5490000 KHz - 5730000 KHz @ 160000 KHz), (N/A, 2000 mBm)
cfg80211:   (5735000 KHz - 5835000 KHz @ 80000 KHz), (N/A, 2000 mBm)
cfg80211:   (57240000 KHz - 63720000 KHz @ 2160000 KHz), (N/A, 0 mBm)
cfg80211: Calling CRDA for country: JP
cfg80211: Regulatory domain changed to country: JP
cfg80211:  DFS Master region: JP
cfg80211:   (start_freq - end_freq @ bandwidth), (max_antenna_gain,
max_eirp)
cfg80211:   (2402000 KHz - 2482000 KHz @ 40000 KHz), (N/A, 2000 mBm)
cfg80211:   (2474000 KHz - 2494000 KHz @ 20000 KHz), (N/A, 2000 mBm)
cfg80211:   (4910000 KHz - 4990000 KHz @ 40000 KHz), (N/A, 2300 mBm)
cfg80211:   (5030000 KHz - 5090000 KHz @ 40000 KHz), (N/A, 2300 mBm)
```

```
cfg80211:   (5170000 KHz - 5250000 KHz @ 80000 KHz), (N/A, 2000 mBm)
cfg80211:   (5250000 KHz - 5330000 KHz @ 80000 KHz), (N/A, 2000 mBm)
cfg80211:   (5490000 KHz - 5710000 KHz @ 160000 KHz), (N/A, 2300 mBm)
```

6.2.5.3. 無線 LAN のネットワーク設定を変更する

ネットワークの設定方法は「6.2.3.5. コネクションを修正する」を参照してください。コネクションの修正を行う際は「図 6.17. 無線 LAN(wlan0)のコネクションの作成」を参考にパスフレーズも合わせて設定してください。 また、コネクションを修正を行った後にはかならず「6.2.3.6. コネクションの修正を反映する」を参考に、修正の反映を行ってください。

6.2.5.4. 無線 LAN の接続を確認する

無線 LAN で正常に通信が可能か確認します。

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。以下の例では、通信機器が「192.0.2.20」という IP アドレスを持っていると想定しています。

```
[armadillo ~]# ping 192.0.2.20
```

図 6.18 無線 LAN の PING 確認

無線 LAN 以外のコネクションが有効化されている場合、ネットワーク通信に無線 LAN が使用されない場合があります。確実に無線 LAN の接続確認をする場合は、事前に無線 LAN 以外のコネクションを無効化してください。

6.2.6. 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 6.19. iptables」のようにコマンドを実行してください。

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

図 6.19 iptables

6.2.7. 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。

6.2.7.1. 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://[Armadillo の IP アドレス]/ にアクセスすると、lighttpd のトップページ(index.html)が表示されます。

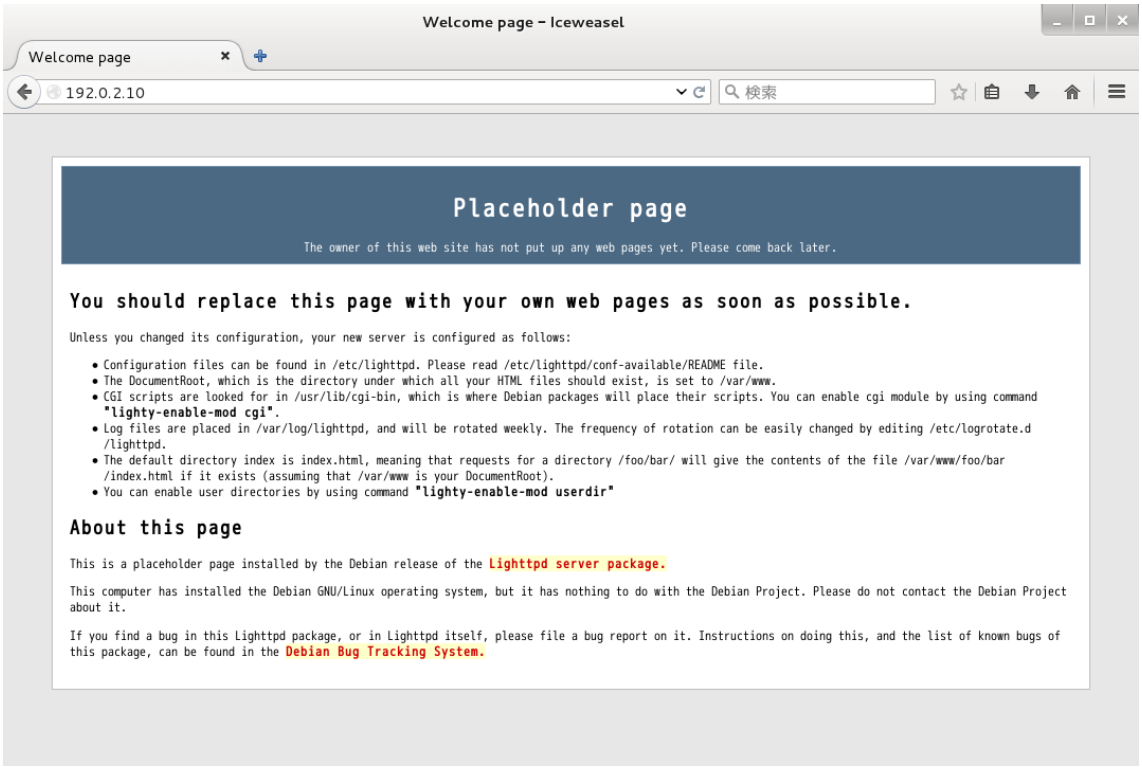


図 6.20 Armadillo トップページ

6.3. ストレージ

Armadillo-X1 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 6.3 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	インターフェース
SD/SDHC/SDXC カード	/dev/mmcblk*[a]	/dev/mmcblk*p1	SD インターフェース(CON4)
USB フラッシュメモリ	/dev/sd*[b]	/dev/sd*1	USB ホストインターフェース(CON7)

[a]microSD/microSDHC/microSDXC カードを接続した場合は、認識された順に mmcblk0 mmcblk1 となります。
[b]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。

6.3.1. ストレージの使用方法

ここでは、SDHC カードを接続した場合を例にストレージの使用方法を説明します。以降の説明では、共通の操作が可能な場合に、SD/SDHC/SDXC カードを SD カードと表記します。

SDXC/microSDXC カードを使用する場合は、事前に「6.3.2. ストレージのパーティション変更とフォーマット」を参照してフォーマットを行う必要があります。これは、Linux カーネルが exFAT ファイルシステムを扱

うことができないためです。通常、購入したばかりの SDXC/microSDXC カードは exFAT ファイルシステムでフォーマットされています。

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 fstype には、ファイルシステムタイプを指定します[3]。FAT32 ファイルシステムの場合は vfat[4]、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 SDHC 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 SDHC 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

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。

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

図 6.23 ストレージのアンマウント

6.3.2. 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 SDHC 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

[3]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。
[4]通常、購入したばかりの SDHC 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。

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 ページ等を参照してください。

```
[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

6.4. LED

Armadillo-X1 の LED は、GPIO が接続されているためソフトウェアで制御することができます。

利用しているデバイスドライバは LED クラスとして実装されているため、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各 LED の対応を次に示します。

表 6.4 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	インターフェース	デフォルトトリガ
/sys/class/leds/led1/	ユーザー LED1	default-on

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。

6.4.1. 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 6.26 LED を点灯させる

Armadillo-X1 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 6.27 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 6.28 LED の状態を表示する

6.4.2. 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 6.5 trigger の種類

設定	説明
none	トリガを設定しません。
mmc0	SD インターフェース(CON4)のアクセスランプにします。
mmc2	eMMC 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。
default-on	主に Linux カーネルから使用します。LED が点灯します。

以下のコマンドを実行すると、LED が 2 秒点灯、1 秒消灯を繰り返します。

```
[armadillo ~]# echo timer > /sys/class/leds/[LED]/trigger
[armadillo ~]# echo 2000 > /sys/class/leds/[LED]/delay_on
[armadillo ~]# echo 1000 > /sys/class/leds/[LED]/delay_off
```

図 6.29 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] rc-feedback nand-disk mmc0 mmc2 timer oneshot heartbeat backlight gpio de
fault-on rfkill0 phy0rx phy0tx phy0assoc phy0radio phy0tpt rfkill1
```

図 6.30 LED のトリガを表示する

6.5. RTC

Armadillo-X1 は、Board Management IC の RTC 機能を利用しています。

電源が切断されても時刻を保持させたい場合は、RTC バックアップインターフェース(CON13)に外付けバッテリー(対応バッテリー例: CR1220)を接続することができます。

6.5.1. 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 6.6 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)

2015 年 6 月 2 日 12 時 34 分 56 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 060212342015.56 ❷
Tue Jun  2 12:34:56 JST 2015
[armadillo ~]# date ❸
Tue Jun  2 12:34:57 JST 2015
```

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 6.31 システムクロックを設定

Armadillo-X1 が接続しているネットワーク内にタイムサーバーがある場合は、NTP(Network Time Protocol)クライアントを利用してシステムクロックを設定することができます。

```
[armadillo ~]# ntpdate [NTP SERVER]
2 Jun 12:34:56 ntpdate[742]: adjust time server x.x.x.x offset 0.004883
 sec
[armadillo ~]# date
Tue Jun  2 12:34:57 JST 2015
```

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
Tue Jun  2 12:35:08 2015  -0.897934 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 6.32 ハードウェアクロックを設定

6.6. ユーザースイッチ

Armadillo-X1 のユーザースイッチのデバイスドライバは、インプットデバイスとして実装されています。インプットデバイスのデバイスファイルからボタンプッシュ/リリースイベントを取得することができます。

ユーザースイッチのインプットデバイスファイルと、各スイッチに対応したイベントコードを次に示します。

表 6.7 インプットデバイスファイルとイベントコード

ユーザースイッチ	インプットデバイスファイル	イベントコード
SW1	/dev/input/event1	2 (KEY_1)

インプットデバイスは検出された順番にインデックスが割り振られます。USB デバイスなどを接続してインプットデバイスを追加している場合は、デバイスファイルのインデックスが異なる可能性があります。

6.6.1. イベントを確認する

ユーザースイッチのボタンプッシュ/リリースイベントを確認するために、ここでは evtest コマンドを利用します。evtest を停止するには、Ctrl+c を入力してください。

```
[armadillo ~]# evtest /dev/input/event1
Input driver version is 1.0.1
Input device ID: bus 0x19 vendor 0x1 product 0x1 version 0x100
Input device name: "gpio-keys"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 2 (KEY_1)
    Event code 3 (KEY_2)
    Event code 4 (KEY_3)
Properties:
Testing ... (interrupt to exit)
Event: time 1458887649.091957, type 1 (EV_KEY), code 3 (KEY_2), value 1 ❶
Event: time 1458887649.091957, ---------- EV_SYN ------------
Event: time 1458887650.311954, type 1 (EV_KEY), code 3 (KEY_2), value 0 ❷
Event: time 1458887650.311954, ---------- EV_SYN ------------
                :
[armadillo ~]#
```

❶ SW2 のボタンプッシュイベントを検出したときの表示。

❷ SW2 のボタンリリースイベントを検出したときの表示。

図 6.33 ユーザースイッチ: イベントの確認

6.7. 温度センサ

Armadillo-X1 の温度センサーは、i.MX 7Dual の TEMPMON(Temperature Monitor)を利用しています。

6.7.1. 温度を取得する

/sys/class/thermal/thermal_zone1/temp ファイルの値を読み出すことによって、i.MX 7Dual の測定温度を取得することができます。

```
[armadillo ~]# cat /sys/class/thermal/thermal_zone1/temp
50000 ❶
```

❶ 温度はミリ°C の単位で表示されます。この例では 50.000°C を示しています。

図 6.34 i.MX 7Dual の測定温度を取得する

6.8. AD コンバーター

Armadillo-X1 は、BMIC(Board Management IC)の AD コンバーター機能により、電源電圧および RTC バックアップインターフェース(CON13)に接続された外付けバッテリーの電圧を取得することができます。

6.8.1. 電圧を取得する

電源電圧は、分圧されて AD コンバーターへ入力されています。電源電圧を取得するためには、まず AD コンバーターへの入力電圧を取得する必要があります。 外部バッテリーの電圧は分圧されていないため、AD コンバーターの入力電圧がそのまま外部バッテリーの電圧となります。

AD コンバーターは IIO(Industrial I/O) デバイスとして実装しています。 /sys/bus/iio/devices/iio:device0/ディレクトリ以下のファイルから入力電圧を算出することができます。

IIO デバイスは、デバイスを認識した順番で iio:deviceN (N は'0'からの連番)となります。IIO デバイスは、IIO デバイス名から特定することができます。BMIC の AD コンバーターの IIO デバイス名は "3-0012"です。

```
[armadillo ~]# cat /sys/bus/iio/devices/iio:device0/name
3-0012
```

AD コンバータへの入力電圧は、AD 変換値と最小入力電圧変動から算出する事ができます。

```
[AD コンバータへの入力電圧 (mV)] = [in_voltage_raw] × [in_voltage_scale]
```

図 6.35 AD コンバータへの入力電圧の計算式

/sys/bus/iio/devices/iio:device0/ディレクトリ以下にある、入力電圧の算出に必要なファイルを次に示します。

表 6.8 入力電圧の算出に必要なファイル

ファイル	説明
in_voltage0_raw	シングルエンド入力 CH0(電源電圧)の AD 変換値
in_voltage1_raw	シングルエンド入力 CH1(外部バッテリー電圧)の AD 変換値
in_voltage_scale	シングルエンド入力の最小入力電圧変動

例として、電源電圧の取得方法について記載します。

```
[armadillo ~]# cat /sys/bus/iio/devices/iio:device0/in_voltage0_raw
4095
[armadillo ~]# cat /sys/bus/iio/devices/iio:device0/in_voltage_scale
0.721191406
```

図 6.36 AD コンバーターへの入力電圧を取得する

「図 6.36. AD コンバーターへの入力電圧を取得する」の例では、AD コンバーターの入力電圧は、約 2.957V (4095 × 0.721191406 [mV])である事がわかります。

AD コンバーターへの入力電圧から、電源電圧を求める計算式を次に示します。

```
[電源電圧 (mV)] = [AD コンバーターへの入力電圧] × (200 + 200) ÷ 200
```

図 6.37 電源電圧の計算式

「図 6.36. AD コンバーターへの入力電圧を取得する」を例にとると、AD コンバーターの入力電圧 1.261V から、電源電圧は約 11.770V であることを求めることができます。

awk コマンドを利用して、次のように電源電圧を表示することができます。

```
[armadillo ~]# adin_raw=`cat /sys/bus/iio/devices/iio:device0/
in_voltage0_raw`
[armadillo ~]# adin_scale=`cat /sys/bus/iio/devices/iio:device0/
in_voltage_scale`
[armadillo ~]# echo $adin_raw $adin_scale | awk '{printf ("%d\n",$1*
$2*(200+200)/200)}'
4435
```

6.8.2. 電源電圧を監視する

vintrigger コマンドを利用して、電源電圧が指定した電圧になった場合に任意のコマンドを実行させることができます。

vintrigger を複数起動することはできません。

vintrigger コマンドのヘルプは次の通りです。

```
[armadillo ~]# vintrigger
Usage: vintrigger -o|-u VOLTAGE [-i INTERVAL] [COMMAND ARGS]
Options:
  -o, --over=VOLTAGE
      Execute the program COMMAND when the detected voltage is equal
      to or over the VOLTAGE[mV].
  -u, --under=VOLTAGE
      Execute the program COMMAND when the detected voltage is equal
      to or under the VOLTAGE[mV].
VOLTAGE: Range: 0 - 28980

  -i, --interval=INTERVAL
      Compare with Vin to the VOLTAGE at INTERVAL second intervals.
INTERVAL: Range: 0 - 4294967295 (Default: 60)

  -h, --help
      Print usage(this message) and exit.
  -v, --version
      Print version information and exit.
```

図 6.38 vintrigger コマンドのヘルプ

30 秒間隔で電源電圧を監視し、4000mV(4V)以下になった場合に、LED2 を点灯させる例を次に示します。

```
[armadillo ~]# vintrigger -u 4000 -i 30 echo 1 > /sys/class/leds/led2/brightness
```

図 6.39 vintrigger コマンド例

vintrigger コマンドのログは/var/log/messages ファイルに出力されます。

```
[armadillo ~]# cat /var/log/messages
                :
Jul  1 09:38:52 armadillo vintrigger[812]: waiting for an under range
alert (4000 mV).❶
Jul  1 09:38:52 armadillo vintrigger[812]: exceeded the limit. executing
command.❷
```

❶ 指定した電圧(4000mV)以下になることを待機します。
❷ 指定した電圧に達したのでコマンドを実行します。

6.9. Armadillo-IoT RS232C アドオンモジュール RS00

Armadillo-IoT RS232C アドオンモジュール RS00(以降、RS232C アドオンモジュールと記載します)は RS232C レベルのシリアルポートが 1 ポート搭載されています。RS232C アドオンモジュールのシリアルポートのデバイスドライバは、TTY デバイスとして実装されているため TTY デバイスファイルから制御を行うことができます。

対応する TTY デバイスファイルを次に示します。

表 6.9 TTY デバイスファイル

TTY デバイスファイル
/dev/ttymxc6

RS232C アドオンモジュールが接続されているとき、Linux カーネルの起動ログに次のように出力されます。

```
armadillo_iotg_addon addon: Atmark Techno RS232C board detected at Add-
On Module I/F 1(Rev 2, SerialNumber=xxxx)
```

6.10. Armadillo-IoT 絶縁 RS232C/422/485 アドオンモジュール RS01

Armadillo-IoT 絶縁 RS232C/422/485 アドオンモジュール RS01(以降、絶縁シリアルアドオンモジュールと記載します)は、電気的に絶縁された RS232C/RS422/RS485 のシリアルポートが 1 ポート登載されています。絶縁シリアルアドオンモジュールのシリアルポートのデバイスドライバは、TTY デバイスとして実装されているため TTY デバイスファイルから制御を行うことができます。

対応する TTY デバイスファイルを次に示します。

表 6.10 TTY デバイスファイル

TTY デバイスファイル
/dev/ttymxc6

絶縁シリアルアドオンモジュールが接続されているとき、Linux カーネルの起動ログに次のように出力されます。

```
armadillo_iotg_addon addon: Atmark Techno RS485/RS422/RS232C board
detected at Add-On Module I/F 1(Rev 2, SerialNumber=xxxx).
```

6.10.1. RS422/RS485 の通信設定を変更する

Armadillo-X1 に電源を投入する前に 絶縁シリアルアドオンモジュール:SW1.1 を OFF に設定すると、TTY デバイスの RS485 設定が自動的に有効化されます。

Armadillo-X1 の電源投入後に 絶縁シリアルアドオンモジュール:SW1.1 の設定を変更しないでください。故障の原因となる可能性があります。

変更が可能な RS485 設定と、自動的に有効化された場合の初期値を「表 6.11. RS485 設定と初期値」に示します。flags は各ビットごとの論理和を示します。

表 6.11 RS485 設定と初期値

設定		説明	初期値
flags	ENABLED(bit0)	0: RS485 無効 1: RS485 有効	1
	RTS_ON_SEND(bit1)	0: データ送信時の RTS(Driver Enable)が Low 1: データ送信時の RTS(Driver Enable)が High	1
	RTS_AFTER_SEND(bit2)	0: データ非送信時の RTS(Driver Enable)が Low 1: データ非送信時の RTS(Driver Enable)が High	0
	RX_DURING_TX(bit4)	0: 半二重通信 1: 全二重通信	0
delay_rts_before_send		送信前遅延時間(ミリ秒)	0
delay_rts_after_send		送信後遅延時間(ミリ秒)	0

flags の RTS_ON_SEND と RTS_AFTER_SEND は初期値を変更しないでください。変更した場合はデータ送信を行うことができなくなります。

RS485 が有効化された TTY デバイスをコンソールとして利用することはできません。

RS485 設定は、アプリケーションプログラムまたは、Linux カーネル起動オプションで変更することができます。

アプリケーションプログラムの作成方法については、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/serial/serial-rs485.txt)を参照してください。

Linux カーネル起動オプションでは、次のオプション指定子で RS485 設定を行います。

表 6.12 Linux カーネル起動オプションからの RS485 設定

オプション指定子	説明
imx.rs485_uart7=	UART7(ttymxc6)の RS485 設定を指定します。

RS485 設定のフォーマットは次の通りです。

```
<flags>,<delay_rts_before_send>,<delay_rts_after_send>
```

例として、絶縁シリアルアドオンモジュールの RS485 設定を全二重通信にする場合は、保守モードで起動してから次のようにコマンドを実行してください。

```
=> setenv optargs imx.rs485_uart7=0x13,0,0
=> saveenv
```

6.11. Armadillo-IoT RN4020 アドオンモジュール BT00

Armadillo-IoT RN4020 アドオンモジュール BT00(以降、RN4020 アドオンモジュールと記載します)は Microchip 製 RN4020 が搭載されています。RN4020 は、Bluetooth(R) version 4.1 に対応しており、Bluetooth Low Energy 4.1 プロトコルスタックが内蔵されています。

RN4020 アドオンモジュールは、TTY デバイスファイルから ASCII コマンドを使用した制御を行うことができます。対応する TTY デバイスファイルを次に示します。

表 6.13 TTY デバイスファイル

TTY デバイスファイル
/dev/ttymxc6

RN4020 アドオンモジュールが接続されているとき、Linux カーネルの起動ログに次のように出力されます。

```
armadillo_iotg_addon addon: Atmark Techno RN4020 board detected at Add-
On Module I/F 1(Rev 2, SerialNumber=xxxx).
```

6.11.1. 設定情報を取得する

RN4020 アドオンモジュールを制御する例として、RN4020 の設定情報の取得を行います。

RN4020 アドオンモジュールに搭載されている RN4020 の設定情報を取得する手順を次に示します。

手順 6.1 設定情報の取得

1. cu コマンドを実行して/dev/ttymxc6 に接続します。ボーレートは 115200bps です。

```
[armadillo ~]$ cu -l /dev/ttymxc6 -s 115200
Connected.
```

2. D (Dump configuration)コマンドを実行すると、RN4020 の設定情報が表示されます。まず、入力したコマンドを表示するために、Ctrl+a に続けて e を入力して下さい。 D コマンドを実行すると、以下のように設定情報が取得できます。

```
D
BTA=001EC01CF9A4
```

```
Name=RNF9A4
Connected=no
Bonded=no
Server Service=80000000
Features=00000000
TxPower=4
```

3. cu を終了するには、"~."(チルダ「~」に続いてドット「.」)を入力します。

```
Disconnected.
[armadillo ~]$
```

その他の ASCII コマンドや、RN4020 の詳細な情報については Microchip 製ドキュメントを参照してください。

RN4020

http://www.microchip.com/wwwproducts/en/RN4020

6.12. Armadillo-IoT EnOcean アドオンモジュール EN00

Armadillo-IoT EnOcean アドオンモジュール EN00(以降、EnOcean アドオンモジュールと記載します)は ROHM 製 BP35A3 が搭載されています。BP35A3 には EnOcean 無線トランシーバー TCM410J が搭載されています。

EnOcean アドオンモジュールは、TTY デバイスファイルから EnOcean Serial Protocol 3(ESP3)で通信することができます。対応する TTY デバイスファイルを次に示します。

表 6.14 TTY デバイスファイル

TTY デバイスファイル
/dev/ttymxc6

EnOcean アドオンモジュールが接続されているとき、Linux カーネルの起動ログに次のように出力されます。

```
armadillo_iotg_addon addon: Atmark Techno EnOcean board detected at Add-
On Module I/F 1(Rev 2, SerialNumber=xxxx).
```

6.12.1. EnOcean 無線データを受信する

EnOcean 無線データを受信する例として、ROHM 製スイッチモジュール PTM 210J を使用します

EnOcean アドオンモジュールで受信する手順を次に示します。

手順 6.2 EnOcean 無線データの受信

1. stty コマンドを実行して TTY デバイスの通信設定を行います。ボーレートは 57600bps です。

```
[armadillo ~]$ stty -F /dev/ttymxc6 57600 raw
```

2. hexdump コマンドを実行して受信データを 16 進数でダンプします。

```
[armadillo ~]$ hexdump -v /dev/ttymxc6
0000000 0055 0207 0a0a 0020 e928 8447 0114 bd38
0000010 0055 0207 0a0a 0020 e928 0047 0181 ba39
```

3. hexdump を終了するには、"Ctrl+c"を入力します。

PTM 210J など、EnOcean 製品の情報については ROHM 社 Web ページを参照してください。

EnOcean 製品のご紹介 | ローム 半導体 ROHM

http://www.rohm.co.jp/web/japan/enocean

EnOcean Serial Protocol の詳細については EnOcean GmbH 製ドキュメントを参照してください。

EnOcean Serial Protocol 3 (ESP3)

http://www.enocean.com/esp

6.13. Armadillo-IoT Wi-SUN アドオンモジュール WS00

Armadillo-IoT Wi-SUN アドオンモジュール WS00(以降、Wi-SUN アドオンモジュールと記載します)は ROHM 製 BP35A1 が搭載されています。

Wi-SUN アドオンモジュールは、TTY デバイスファイルから ASCII コマンドを使用した制御を行うことができます。対応する TTY デバイスファイルを次に示します。

表 6.15 TTY デバイスファイル

TTY デバイスファイル
/dev/ttymxc6

Wi-SUN アドオンモジュールが接続されているとき、Linux カーネルの起動ログに次のように出力されます。

```
armadillo_iotg_addon addon: Atmark Techno Wi-SUN board detected at Add-
On Module I/F 1(Rev 2, SerialNumber=xxxx).
```

6.13.1. 設定情報を取得する

Wi-SUN アドオンモジュールを制御する例として、BP35A1 の設定情報の取得を行います。

Wi-SUN アドオンモジュールに搭載されている BP35A1 の設定情報を取得する手順を次に示します。

手順 6.3 設定情報の取得

1. cu コマンドを実行して/dev/ttymxc6 に接続します。ボーレートは 115200bps です。

```
[armadillo ~]$ cu -l /dev/ttymxc6 -s 115200
Connected.
```

2. SKINFO コマンドを実行すると、BP35A1 の設定情報が表示されます。

```
SKINFO
EINFO FE80:0000:0000:0000:021D:1290:0004:0FBE 001D129000040FBE 21 FFFF FFFE
OK
```

3. cu を終了するには、"~."(チルダ「~」に続いてドット「.」)を入力します。

```
Disconnected.
[armadillo ~]$
```

その他の ASCII コマンドや、BP35A1 の詳細な情報については ROHM 製ドキュメントを参照してください。

「ROHM Sub-GHz シリーズ」サポートページ ドキュメントダウンロード | 半導体のローム ROHM

http://micro.rohm.com/jp/download_support/wi-sun

6.14. Armadillo-IoT 絶縁 RS485 アドオンモジュール RS02

Armadillo-IoT 絶縁 RS485 アドオンモジュール RS02(以降、絶縁 RS485 アドオンモジュールと記載します)は、電気的に絶縁された RS422/RS485 のシリアルポートが 1 ポート登載されています。絶縁 RS485 アドオンモジュールのシリアルポートのデバイスドライバは、TTY デバイスとして実装されているため TTY デバイスファイルから制御を行うことができます。

対応する TTY デバイスファイルを次に示します。

表 6.16 TTY デバイスファイル

TTY デバイスファイル
/dev/ttymxc6

絶縁 RS485 アドオンモジュールが接続されているとき、Linux カーネルの起動ログに次のように出力されます。

```
armadillo_iotg_addon addon: Atmark Techno RS485 board detected at Add-On
Module I/F 1(Rev 1, SerialNumber=xxxx).
```

6.14.1. RS422/RS485 の通信設定を変更する

TTY デバイスの RS485 設定は自動的に有効化されます。

変更が可能な RS485 設定と、自動的に有効化された場合の初期値を「表 6.11. RS485 設定と初期値」に示します。flags は各ビットごとの論理和を示します。

表 6.17 RS485 設定と初期値

設定		説明	初期値
flags	ENABLED(bit0)	0: RS485 無効 1: RS485 有効	1
	RTS_ON_SEND(bit1)	0: データ送信時の RTS(Driver Enable)が Low 1: データ送信時の RTS(Driver Enable)が High	1
	RTS_AFTER_SEND(bit2)	0: データ非送信時の RTS(Driver Enable)が Low 1: データ非送信時の RTS(Driver Enable)が High	0
	RX_DURING_TX(bit4)	0: 半二重通信 1: 全二重通信	0
delay_rts_before_send		送信前遅延時間(ミリ秒)	0
delay_rts_after_send		送信後遅延時間(ミリ秒)	0

flags の RTS_ON_SEND と RTS_AFTER_SEND は初期値を変更しないでください。変更した場合はデータ送信を行うことができなくなります。

RS485 が有効化された TTY デバイスをコンソールとして利用することはできません。

RS485 設定は、アプリケーションプログラムまたは、Linux カーネル起動オプションで変更することができます。

アプリケーションプログラムの作成方法については、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/serial/serial-rs485.txt)を参照してください。

Linux カーネル起動オプションでは、次のオプション指定子で RS485 設定を行います。

表 6.18 Linux カーネル起動オプションからの RS485 設定

オプション指定子	説明
imx.rs485_uart7=	UART7(ttymxc6)の RS485 設定を指定します。

RS485 設定のフォーマットは次の通りです。

```
<flags>,<delay_rts_before_send>,<delay_rts_after_send>
```

例として、RS485 設定を全二重通信にする場合は、保守モードで起動してから次のようにコマンドを実行してください。

```
=> setenv optargs imx.rs485_uart7=0x13,0,0
=> saveenv
```

6.15. Armadillo-IoT 絶縁デジタル入出力/アナログ入力アドオンモジュール DA00

Armadillo-IoT 絶縁デジタル入出力/アナログ入力アドオンモジュール DA00(以降、絶縁 IO アドオンモジュールと記載します)は、電気的に絶縁されたデジタル入力 2 ポート、デジタル出力 2 ポートと 0〜5V のアナログ入力 2 ポートを追加することができます。

絶縁 IO アドオンモジュールのデジタル入出力のデバイスドライバは GPIO、アナログ入力のデバイスドライバは IIO(Industrial I/O) デバイスとして実装しています。

入出力ポートと、GPIO クラスディレクトリの対応を「表 6.19. 入出力ポートと GPIO クラスディレクトリ」に示します。IIO デバイスは、デバイスを認識した順番で iio:deviceN (N は'0'からの連番)となります。

表 6.19 入出力ポートと GPIO クラスディレクトリ

ポート	GPIO クラスディレクトリ
デジタル出力 1	/sys/devices/addon/DO1_INTF1
デジタル出力 2	/sys/devices/addon/DO2_INTF1
デジタル入力 1	/sys/devices/addon/DI1_INTF1
デジタル入力 2	/sys/devices/addon/DI2_INTF1

絶縁 IO アドオンモジュールが接続されているとき、Linux カーネルの起動ログに次のように出力されます。

```
armadillo_iotg_addon addon: Atmark Techno DI/DO/AD board detected at Add-
On Module I/F 1(Rev 1, SerialNumber=xxxx).
```

6.15.1. 出力状態を設定する

GPIO クラスディレクトリ以下の value ファイルに値を書き込むことによって、出力状態を設定することができます。"0"は開放、"1"は短絡を表わします。

デジタル出力 1 を開放に設定する例を次に示します。

```
[armadillo ~]# echo 0 > /sys/devices/addon/DO1_INTF1/value
```

図 6.40 デジタル出力状態を変更する

6.15.2. 入力状態を取得する

GPIO クラス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、入力状態を取得することができます。"0"は GND_ISO との短絡。"1"は開放または 3.15V 以上印加を表わします。

デジタル入力 1 の状態を取得する例を次に示します。

```
[armadillo ~]# cat /sys/devices/addon/DI1_INTF1/value
1
```

図 6.41 デジタル入力状態を取得する

6.15.3. 電圧を取得する

/sys/bus/iio/devices/iio:device1/ディレクトリ以下のファイルから入力電圧を算出することができます。

IIO デバイスは、デバイスを認識した順番で iio:deviceN (N は'0'からの連番)となります。IIO デバイスは、IIO デバイス名から特定することができます。絶縁 IO アドオンモジュールに搭載している AD コンバーターの IIO デバイス名は "mcp3202"です。

```
[armadillo ~]# cat /sys/bus/iio/devices/iio:device1/name
mcp3202
```

AD コンバータへの入力電圧は、AD 変換値と最小入力電圧変動から算出する事ができます。

```
[AD コンバータへの入力電圧（mV)] = [AD 変換値] × [最小入力電圧変動]
```

図 6.42 AD コンバータへの入力電圧の計算式

/sys/bus/iio/devices/iio:device1/ディレクトリ以下にある、入力電圧の算出に必要なファイルを次に示します。

表 6.20 入力電圧の算出に必要なファイル

ファイル	説明
in_voltage0_raw	シングルエンド入力 CH0 の AD 変換値
in_voltage1_raw	シングルエンド入力 CH1 の AD 変換値
in_voltage_scale	シングルエンド入力の最小入力電圧変動
in_voltage0-voltage1_raw	疑似差動入力の AD 変換値
in_voltage-voltage_scale	疑似差動入力の最小入力電圧変動

シングルエンド入力 CH0 への入力電圧を算出する例を次に示します。

```
[armadillo ~]# cat /sys/bus/iio/devices/iio:device1/in_voltage0_raw
2048
[armadillo ~]# cat /sys/bus/iio/devices/iio:device1/in_voltage_scale
1.220703125
```

図 6.43 AD コンバーターへの入力電圧を取得する

「図 6.43. AD コンバーターへの入力電圧を取得する」の例では、シングルエンド入力 CH0 への入力電圧は、2.5V (2048 × 1.220703125 [mV])である事がわかります。

awk コマンドを利用して、次のように電源電圧を表示することができます。

```
[armadillo ~]# adin_raw=`cat /sys/bus/iio/devices/iio:device1/
in_voltage0_raw`
[armadillo ~]# adin_scale=`cat /sys/bus/iio/devices/iio:device1/
in_voltage_scale`
[armadillo ~]# echo $adin_raw $adin_scale | awk '{printf ("%d",$1*$2)}'
2500
```

7. Linux カーネル仕様

本章では、工場出荷状態の Armadillo-X1 の Linux カーネル仕様について説明します。

7.1. デフォルトコンフィギュレーション

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-X1 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-3.14-x1-at[VERSION].tar.gz)に含まれる `arch/arm/configs/x1_defconfig` です。

`x1_defconfig` で有効になっている主要な設定を「表 7.1. Linux カーネル主要設定」に示します。

表 7.1 Linux カーネル主要設定

コンフィグ	説明
SMP	Symmetric Multi-Processing
SMP_ON_UP	Allow booting SMP kernel on uniprocessor systems
ARM_CPU_TOPOLOGY	Support cpu topology definition
HAVE_ARM_ARCH_TIMER	Architected timer support
VMSPLIT_2G	Memory split (2G/2G user/kernel split)
NO_HZ	Tickless System (Dynamic Ticks)
HIGH_RES_TIMERS	High Resolution Timer Support
PREEMPT	Preemptible Kernel
AEABI	Use the ARM EABI to compile the kernel
COMPACTION	Allow for memory compaction
BINFMT_ELF	Kernel support for ELF binaries

7.2. デフォルト起動オプション

工場出荷状態の Armadillo-X1 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 7.2 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=ttymxc4,115200	起動ログなどが出力されるイニシャルコンソールに ttymxc4(CON9)を、ボーレートに 115200bps を指定します。
root=/dev/mmcblk2p2	ルートファイルシステムに eMMC を指定します。
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。
rw	ルートファイルシステムを読み書き可能としてマウントします。

7.3. Linux ドライバ一覧

Armadillo-X1 で利用することができるデバイスドライバについて説明します。 各ドライバで利用しているソースコードの内主要なファイルのパスや、コンフィギュレーションに必要な情報、及びデバイスファイルなどについて記載します。

7.3.1. Armadillo-X1

Armadillo-X1 のハードウェアの構成情報やピンマルチプレクスの情報、i.MX 7Dual の初期化手順などが定義されています。

関連するソースコード

arch/arm/mach-imx/
arch/arm/mach-imx/armadillo_iotg_addon/
arch/arm/boot/dts/armadillo_x1.dts
arch/arm/boot/dts/imx7d.dtsi
arch/arm/boot/dts/armadillo_x1_addon.dtsi

Device Tree ドキュメント

Documentation/devicetree/bindings/arm/arch_timer.txt
Documentation/devicetree/bindings/arm/cpus.txt
Documentation/devicetree/bindings/arm/gic.txt
Documentation/devicetree/bindings/arm/imx/busfreq-imx.txt
Documentation/devicetree/bindings/clock/fixed-clock.txt
Documentation/devicetree/bindings/cpufreq/cpufreq-cpu0.txt
Documentation/devicetree/bindings/misc/sram.txt
Documentation/devicetree/bindings/pinctrl/fsl,imx-pinctrl.txt

カーネルコンフィギュレーション

```
System Type  --->
    [*] Freescale i.MX family                                    <CONFIG_ARCH_MXC>
          Freescale i.MX support  --->
              [*] i.MX7 Dual support                            <CONFIG_SOC_IMX7D>
               [*]   Add-On Module Auto Detect                <AIOTG_ADDON_AUTO_DETECT>
```

7.3.2. SPI フラッシュメモリ

Armadillo-X1 では、フラッシュメモリを制御するソフトウェアとして MTD(Memory Technology Device) を利用しています。MTD のキャラクタデバイスまたはブロックデバイスを経由して、ユーザーランドからアクセスすることができます。

関連するソースコード

drivers/mtd/mtdcore.c
drivers/mtd/mtdsuper.c
drivers/mtd/mtdconcat.c
drivers/mtd/mtdpart.c
drivers/mtd/mtdchar.c
drivers/mtd/cmdlinepart.c
drivers/mtd/ofpart.c
drivers/mtd/mtdblock.c
drivers/mtd/spi-nor/spi-nor.c
drivers/mtd/spi-nor/fsl-quadspi.c

デバイスファイル

デバイスファイル	デバイスタイプ	対応するパーティション名
/dev/mtd0	キャラクタ	bootloader
/dev/mtd0ro		
/dev/mtdblock0	ブロック	
/dev/mtd1	キャラクタ	license
/dev/mtd1ro		
/dev/mtdblock1	ブロック	
/dev/mtd2	キャラクタ	reserved
/dev/mtd2ro		
/dev/mtdblock2	ブロック	

カーネルコンフィギュレーション

```
Device Drivers  --->
    <*> Memory Technology Device (MTD) support  --->                    <CONFIG_MTD>
        <*>   Command line partition table parsing            <CONFIG_MTD_CMDLINE_PARTS>
        <*>   OpenFirmware partitioning information support       <CONFIG_MTD_OF_PARTS>
        <*>   Caching block device access to MTD devices             <CONFIG_MTD_BLOCK>
        <*>   SPI-NOR device support  --->                          <CONFIG_MTD_SPI_NOR>
            <*>   Freescale Quad SPI controller                  <CONFIG_SPI_FSL_QUADSPI>
```

7.3.3. UART

Armadillo-X1 のシリアルは、i.MX 7Dual の UART(Universal Asynchronous Receiver/Transmitter) を利用しています。

Armadillo-X1 の標準状態では、UART5(CON5)をコンソールとして利用しています。

フォーマット

データビット長: 7 or 8 ビット
ストップビット長: 1 or 2 ビット
パリティ: 偶数 or 奇数 or なし
フロー制御: CTS/RTS or XON/XOFF or なし
最大ボーレート: 1.5Mbps

関連するソースコード

drivers/tty/n_tty.c
drivers/tty/tty_buffer.c
drivers/tty/tty_io.c
drivers/tty/tty_ioctl.c
drivers/tty/tty_ldisc.c
drivers/tty/tty_ldsem.c
drivers/tty/tty_mutex.c
drivers/tty/tty_port.c
drivers/tty/serial/serial_core.c
drivers/tty/serial/imx.c

Device Tree ドキュメント

Documentation/devicetree/bindings/serial/fsl-imx-uart.txt

デバイスファイル

シリアルインターフェース	デバイスファイル
UART5	/dev/ttymxc4
UART7	/dev/ttymxc6

カーネルコンフィギュレーション

```
Device Drivers  --->
    Character devices  --->
        [*] Enable TTY                                                    <TTY>
            Serial drivers  --->
                <*> IMX serial port support                            <SERIAL_IMX>
                [*]   Console on IMX serial port                   <SERIAL_IMX_CONSOL>
                <*> Freescale lpuart serial port support           <SERIAL_FSL_LPUART>
                [*]   Console on Freescale lpuart serial port <SERIAL_FSL_LPUART_CONSOLE>
```

7.3.4. Ethernet

Armadillo-X1 の Ethernet(LAN)は、i.MX 7Dual の ENET(Ethernet MAC)を利用しています。

機能

通信速度: 1000Mbps(1000BASE-T), 100Mbps(100BASE-TX), 10Mbps(10BASE-T)
通信モード: Full-Duplex(全二重), Half-Duplex(半二重)[1]
Auto Negotiation サポート
キャリア検知サポート
リンク検出サポート

関連するソースコード

drivers/net/Space.c
drivers/net/loopback.c
drivers/net/mii.c
drivers/net/ethernet/freescale/fec_main.c
drivers/net/ethernet/freescale/fec_ptp.c
drivers/net/phy/mdio_bus.c
drivers/net/phy/phy.c
drivers/net/phy/phy_device.c
drivers/net/phy/vitesse.c

Device Tree ドキュメント

Documentation/devicetree/bindings/net/fsl-fec.txt

ネットワークデバイス

eth0

[1]1000BASE-T は半二重通信非サポート

カーネルコンフィギュレーション

```
Device Drivers  --->
    [*] Network device support  --->                                    <NETDEVICES>
        [*]   Ethernet driver support  --->                               <ETHERNET>
            [*]   Freescale devices                                <NET_VENDOR_FREESCALE>
            <*>     FEC ethernet controller (of ColdFire and some i.MX CPUs)       <FEC>
        -*-   PHY Device support and infrastructure  --->                    <PHYLIB>
            <*>   Drivers for the Vitesse PHYs                           <VITESSE_PHY>
```

7.3.5. WLAN

Armadillo-X1 には、Qualcomm Atheros 製 AR9462 が搭載されています。AR9462 の WLAN 機能は、「7.3.10. PCI Express」に示す PCI Express に接続されています。

機能

IEEE 802.11 a/b/g/n 準拠
最大通信速度: 300Mbps(論理値)
動作モード: インフラストラクチャモード(STA/AP), アドホックモード
チャンネル(2.4GHz): 1-14
チャンネル(5GHz): 36-48, 52-64, 100-140

ネットワークデバイス

wlan0

関連するソースコード

drivers/net/wireless/ath/ath9k/

カーネルコンフィギュレーション

```
Device Drivers  --->
    [*] Network device support  --->                                    <NETDEVICES>
        [*]   Wireless LAN  --->                                            <WLAN>
        <*>   Atheros Wireless Cards  --->                               <ATH_CARDS>
           <*>   Atheros 802.11n wireless cards support                       <ATH9>
           [*]     Atheros ath9k PCI/PCIe bus support                    <ATH9K_PCI>
           [*]   Atheros ath9k rfkill support                           <ATH9K_RFKILL>
```

7.3.6. BT

Armadillo-X1 には、Qualcomm Atheros 製 AR9462 が搭載されています。AR9462 の BT 機能は、「7.3.9. USB ハブ」に示す USB3503 に接続されています。

AR9462 は、Bluetooth(R) version 4.0 に対応しており、BLE(Bluetooth Low Energy)、HS(High Speed)および EDR(Enhanced Data Rate)が利用できます。

デバイス

hci0

関連するソースコード

drivers/bluetooth/ath3k.c
drivers/bluetooth/btusb.c

カーネルコンフィギュレーション

```
[*] Networking support  --->                                                <NET>
    <*>   Bluetooth subsystem support  --->                                 <BT>
        <*>   RFCOMM protocol support                                   <BT_RFCOMM>
        [*]     RFCOMM TTY support                                  <BT_RFCOMM_TTY>
        <*>   BNEP protocol support                                       <BT_BNEP>
        [*]     Multicast filter support                          <BT_BNEP_MC_FILTER>
        [*]     Protocol filter support                         <BT_BNEP_PROTO_FILTER>
        <*>   HIDP protocol support                                       <BT_HIDP>
              Bluetooth device drivers  --->
                  <*> HCI USB driver                                  <BT_HCIUART_H4>
                  [*]   Atheros AR300x serial support                 <BT_HCIUART_LL>
    Device Drivers  --->
    [*] Network device support  --->                                  <NETDEVICES>
        [*]   Wireless LAN  --->                                            <WLAN>
            <*>   Atheros Wireless Cards  --->                          <ATH_CARDS>
                [*]   Atheros bluetooth coexistence support       <ATH9K_BTCOEX_SUPPORT>
```

AR9462 のファームウェアは、ATDE にインストールされている firmware-atheros パッケージに含まれています。ファームウェアは Linux カーネルイメージ内に改変無く配置されます。

firmware-atheros の著作権およびライセンス情報については、ATDE 上で/usr/share/doc/firmware-atheros/copyright を参照してください。

7.3.7. SD ホスト

Armadillo-X1 の SD ホストは、i.MX 7Dual の uSDHC(Ultra Secured Digital Host Controller)を利用しています。

Armadillo-X1 では、SD スロット拡張ボードの SD インターフェース(CON2)が uSDHC1 を利用しています。

機能

カードタイプ: SD/SDHC/SDXC/SDIO
バス幅: 1bit or 4bit
スピードモード: Default Speed(24MHz), High Speed(48MHz), UHS-I(196.36MHz)
カードディテクトサポート
ライトプロテクトサポート

デバイスファイル

メモリカードの場合は、カードを認識した順番で/dev/mmcblkN (N は'0'または'1')となります。
I/O カードの場合は、ファンクションに応じたデバイスファイルとなります。

関連するソースコード

drivers/mmc/card/block.c
drivers/mmc/card/queue.c
drivers/mmc/core/
drivers/mmc/host/sdhci-esdhc-imx.c
drivers/mmc/host/sdhci-pltfm.c
drivers/mmc/host/sdhci.c

Device Tree ドキュメント

Documentation/devicetree/bindings/mmc/fsl-imx-esdhc.txt
Documentation/devicetree/bindings/mmc/mmc-pwrseq-emmc.txt

カーネルコンフィギュレーション

```
Device Drivers  --->
    <*> MMC/SD/SDIO card support  --->                                  <MMC>
        [*]   Additional delay after SDIO reset          <MMC_DELAY_AFTER_SDIO_RESET>
              *** MMC/SD/SDIO Card Drivers ***
        <*>   MMC block device driver                                  <MMC_BLOCK>
        (8)     Number of minors per block device                <MMC_BLOCK_MINORS>
        [*]     Use bounce buffer for simple hosts               <MMC_BLOCK_BOUNCE>
              *** MMC/SD/SDIO Host Controller Drivers ***
        <*>   Secure Digital Host Controller Interface support         <MMC_SDHCI>
        <*>   SDHCI platform and OF driver helper                <MMC_SDHCI_PLTFM>
        <*>     SDHCI support for the Freescale eSDHC/uSDHC i.MX controller
                                                                <MMC_SDHCI_OF_ESDHC>
```

SDIO カードを利用する場合は、arch/arm/boot/dts/armadillo_x1_addon.dtsi の"usdhc1"ノードに"use-sdio"プロパティを追加してください。

```
&usdhc1 {
    pinctrl-names = "default", "state_100mhz", "state_200mhz",
            "state_power_off";
    pinctrl-0 = <&pinctrl_usdhc1>;
    pinctrl-1 = <&pinctrl_usdhc1_100mhz>;
    pinctrl-2 = <&pinctrl_usdhc1_200mhz>;
    pinctrl-3 = <&pinctrl_usdhc1_power_off>;
    use-sdio;
};
```

"use-sdio"プロパティを追加しない場合、Advanced DMA エラーが発生する場合があります。

7.3.8. USB ホスト

Armadillo-X1 の USB ホストは、i.MX 7Dual の USB-PHY(Universal Serial Bus 2.0 Integrated PHY) および USB(Universal Serial Bus Controller) を利用しています。

Armadillo-X1 では、USB ホストインターフェース(CON2)が OTG1 を利用しています。HSIC HOST には「7.3.9. USB ハブ」に示す USB3503 が接続されています。

機能

Universal Serial Bus Specification Revision 2.0 準拠
Enhanced Host Controller Interface (EHCI)準拠
転送レート: USB2.0 High-Speed (480Mbps), Full-Speed (12Mbps), Low-Speed (1.5Mbps)

デバイスファイル

メモリデバイスの場合は、デバイスを認識した順番で/dev/sdN (N は'a'からの連番)となります。
I/O デバイスの場合は、ファンクションに応じたデバイスファイルとなります。

関連するソースコード

drivers/usb/chipidea/ci_hdrc_imx.c
drivers/usb/chipidea/ci_hdrc_msm.c
drivers/usb/chipidea/ci_hdrc_zevio.c
drivers/usb/chipidea/core.c
drivers/usb/chipidea/host.c
drivers/usb/chipidea/otg.c
drivers/usb/chipidea/usbmisc_imx.c
drivers/usb/host/ehci-hcd.c
drivers/usb/host/ehci-hub.c
drivers/usb/phy/phy-generic.c

Device Tree ドキュメント

Documentation/devicetree/bindings/usb/ci-hdrc-imx.txt
Documentation/devicetree/bindings/usb/usbmisc-imx.txt
Documentation/devicetree/bindings/usb/usb-nop-xceiv.txt
Documentation/devicetree/bindings/regulator/fixed-regulator.txt

カーネルコンフィギュレーション

```
Device Drivers  --->
    [*] USB support  --->                                          <USB_SUPPORT>
        <*>   Support for Host-side USB                                   <USB>
                *** USB Host Controller Drivers ***
        <*>     EHCI HCD (USB 2.0) support                       <USB_EHCI_HCD>
        <*>   ChipIdea Highspeed Dual Role Controller            <USB_CHIPIDEA>
        [*]     ChipIdea host controller                       <USB_CHIPIDEA_HOST>
              USB Physical Layer drivers  --->
            <*>     NOP USB Transceiver Driver                   <NOP_USB_XCEIV>
```

7.3.9. USB ハブ

Armadillo-X1 には、Microchip 製 USB3503 が搭載されています。USB3503 には、AR9462 が接続されています。

関連するソースコード

drivers/usb/misc/usb3503.c

Device Tree ドキュメント

Documentation/devicetree/bindings/usb/usb3503.txt

カーネルコンフィギュレーション

```
Device Drivers  --->
    [*] USB support  --->                                                <USB_SUPPORT>
        <*>   USB3503 HSIC to USB20 Driver                             <USB_HSIC_USB3503>
```

7.3.10. PCI Express

Armadillo-X1 の PCI Express、i.MX 7Dual の PCIe_PHY(PCI Express PHY) を利用しています。

Armadillo-X1 では、AR9462 が接続されています。

機能

PCI Express Base Specification 2.1 準拠
PIPE Specification 2.0 準拠
リンク幅: x1
転送レート: 5.0GT/s
割り込み通知方式: INTx, MSI-X

関連するソースコード

drivers/pci/host/pci-imx6.c
drivers/pci/access.c
drivers/pci/bus.c
drivers/pci/probe.c
drivers/pci/host-bridge.c
drivers/pci/remove.c
drivers/pci/pci.c
drivers/pci/pci-driver.c
drivers/pci/search.c
drivers/pci/pci-sysfs.c
drivers/pci/rom.c
drivers/pci/setup-res.c
drivers/pci/irq.c
drivers/pci/vpd.c
drivers/pci/setup-bus.c
drivers/pci/vc.c
drivers/pci/iov.c

カーネルコンフィギュレーション

```
Bus support  --->
    [*] PCI support                                                        <PCI>
    [ ] Message Signaled Interrupts (MSI and MSI-X)                    <PCI_MSI>
    [*] PCI IOV support                                                <PCI_IOV>
    [*] PCI PRI support                                                <PCI_PRI>
    [*] PCI PASID support                                            <PCI_PASID>
        PCI host controller drivers  --->
            [*] Freescale i.MX6 PCIe controller                       <PCI_IMX6>
```

PCI_MSI を有効化すると、AR9462 が利用できなくなります。

7.3.11. リアルタイムクロック

Armadillo-X1 のリアルタイムクロックは、Board Management IC の RTC 機能を利用しています。Board Management IC の RTC 機能は、I2C4 (I2C ノード: 3-0011) に接続されています。

機能

アラーム割り込みサポート

デバイスファイル

/dev/rtc
/dev/rtc0

関連するソースコード

drivers/rtc/class.c
drivers/rtc/hctosys.c
drivers/rtc/interface.c
drivers/rtc/rtc-dev.c
drivers/rtc/rtc-lib.c
drivers/rtc/rtc-proc.c
drivers/rtc/rtc-sysfs.c
drivers/rtc/systohc.c
drivers/rtc/rtc-bmic.c

カーネルコンフィギュレーション

```
Device Drivers  --->
    <*> Real Time Clock  --->
        [*]   Set system time from RTC on startup and resume               <RTC_HCTOSYS>
        [*]   Set the RTC time based on NTP synchronization                <RTC_SYSTOHC>
        (rtc0)  RTC used to set the system time                          <RTC_HCTOSYS_DEVICE>
              *** RTC interfaces ***
        [*]   /sys/class/rtc/rtcN (sysfs)                                    <RTC_INTF_SYSFS>
        [*]   /proc/driver/rtc (procfs for rtcN)                             <RTC_INTF_PROC>
        [*]   /dev/rtcN (character devices)                                   <RTC_INTF_DEV>
        [*]     RTC UIE emulation on dev interface                       <RTC_INTF_DEV_UIE_EMUL>
              *** I2C RTC drivers ***
        <*>   Atmark Techno BMIC RTC                                          <RTC_DRV_BMIC>
```

アラーム割り込みは、sysfs RTC クラスディレクトリ以下のファイルから利用できます。

wakealarm ファイルに UNIX エポックからの経過秒数、または先頭に+を付けて現在時刻からの経過秒数を書き込むと、アラーム割り込み発生時刻を指定できます。アラーム割り込み発生時刻を変更するには wakealarm ファイルに"+0"を書き込み、アラーム割り込みのキャンセル後に再設定する必要があります。アラーム割り込みの利用例を次に示します。

```
[armadillo ~]# cat /proc/interrupts | grep bmic_rtc_irq ❶
173:          0          0  gpio-mxc  13  bmic_rtc_irq
[armadillo ~]# echo +10 > /sys/class/rtc/rtc0/wakealarm ❷
[armadillo ~]# cat /sys/class/rtc/rtc0/wakealarm ❸
1458781144
[armadillo ~]# cat /sys/class/rtc/rtc0/since_epoch ❹
1458781145
[armadillo ~]# cat /proc/interrupts | grep bmic_rtc_irq ❺
173:          1          0  gpio-mxc  13  bmic_rtc_irq
```

❶ アラーム割り込みの発生回数を確認します。この例では 0 回です。

❷ アラーム割り込みの発生時刻を 10 秒後に設定します。

❸ アラーム割り込みの発生時刻(UNIX エポックからの経過秒数)を確認します。この例では 1458781144 秒です。

❹ 現在時刻(UNIX エポックからの経過秒数)を確認します。アラーム割り込みの発生時刻を超えるまで待ちます。

❺ 再度アラーム割り込みの発生回数を確認します。1 増えているのでアラーム割り込みが発生したことを確認できます。

デバイスファイル(/dev/rtc0)経由でもアラーム割り込みを利用することができます。サンプルプログラムなどのより詳細な情報については、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/rtc.txt)を参照してください。

date コマンドを利用して、UNIX エポックからの経過秒数を日時に変換することができます。

```
[armadillo ~]# date --date=@`cat /sys/class/rtc/rtc0/since_epoch`
Thu Mar 24 10:02:56 JST 2016
```

7.3.12. 温度センサ

Armadillo-X1 の温度センサーは、i.MX 7Dual の TEMPMON(Temperature Monitor)を利用しています。

起動直後の設定では、i.MX 7Dual の測定温度が 105℃以上になった場合、Linux カーネルが/sbin/poweroff コマンドを実行し、システムを停止します。

sysfs ディレクトリ

/sys/class/thermal/thermal_zone1/

関連するソースコード

drivers/thermal/imx_thermal.c
drivers/thermal/thermal_core.c
drivers/thermal/cpu_cooling.c
drivers/thermal/device_cooling.c
drivers/thermal/of-thermal.c
drivers/thermal/step_wise.c
drivers/thermal/thermal_hwmon.c

カーネルコンフィギュレーション

```
Device Drivers  --->
    <*> Generic Thermal sysfs driver  --->                                  <THERMAL>
        [*]   Expose thermal sensors as hwmon device                     <THERMAL_HWMON>
        [*]   APIs to parse thermal data out of device tree                 <THERMAL_OF>
        -*-   Step_wise thermal governor                             <THERMAL_GOV_STEP_WISE>
        [*]   generic cpu cooling support                                 <CPU_THERMAL>
        <*>   Temperature sensor driver for Freescale i.MX SoCs           <IMX_THERMAL>
```

7.3.13. AD コンバーター

Armadillo-X1 に搭載された Board Management IC の AD コンバーター機能および Armadillo-IoT 絶縁デジタル入出力/アナログ入力アドオン モジュール DA00(以降、絶縁 IO アドオンモジュールと記載します)を利用することができます。

Board Management IC の AD コンバーター機能は、I2C4(I2C ノード: 3-0012)に接続されています。Armadillo-X1 の電源電圧および Board Management IC の外部バッテリーの電圧を測定することができます。

絶縁 IO アドオンモジュールには、Microchip 製 MCP3202 が搭載されています。MCP3202 は、ECSPI1 に接続されます。

機能(Board Management IC)

分解能: 12bit
測定範囲: 0V ～ 3.3V(Board Management IC の電源電圧)

機能(MCP3202)

分解能: 12bit
測定範囲: 0V ～ 5.0V(MCP3202 の電源電圧)

sysfs ディレクトリ

デバイスを認識した順番で /sys/bus/iio/devices/iio:deviceN (N は'0'からの連番)となります。

デバイスファイル

デバイスを認識した順番で /dev/iio:deviceN (N は'0'からの連番)となります。

関連するソースコード

drivers/iio/industrialio-core.c

drivers/iio/industrialio-buffer.c
drivers/iio/industrialio-event.c
drivers/iio/industrialio-trigger.c
drivers/iio/industrialio-triggered-buffer.c
drivers/iio/inkern.c
drivers/iio/kfifo_buf.c
drivers/iio/trigger/iio-trig-sysfs.c
drivers/iio/adc/bmic_adc.c
drivers/iio/adc/mcp320x.c

カーネルコンフィギュレーション

```
Device Drivers  --->
    <*> Industrial I/O support  --->                                        <IIO>
        [*]   Enable buffer support within IIO                          <IIO_BUFFER>
        -*-     Industrial I/O buffering based on kfifo                <IIO_KFIFO_BUF>
        -*-   Enable triggered sampling support                          <IIO_TRIGGER>
        (2)     Maximum number of consumers per trigger       <IIO_CONSUMERS_PER_TRIGGER>
              Analog to digital converters  --->
                  <*> Atmark Techno BMIC ADC                                <BMIC_ADC>
                  <*> Microchip Technology MCP3x01/02/04/08                 <MCP320X>
```

7.3.14. LED

Armadillo-X1 に搭載されているソフトウェア制御可能な LED には、GPIO が接続されています。Linux では、GPIO 接続用 LED ドライバ(leds-gpio)で制御することができます。

sysfs LED クラスディレクトリ

/sys/class/leds/led1
/sys/class/leds/led2

関連するソースコード

drivers/leds/led-class.c
drivers/leds/led-core.c
drivers/leds/led-triggers.c
drivers/leds/leds-gpio-register.c
drivers/leds/leds-gpio.c
drivers/leds/trigger/ledtrig-timer.c
drivers/leds/trigger/ledtrig-oneshot.c
drivers/leds/trigger/ledtrig-heartbeat.c
drivers/leds/trigger/ledtrig-backlight.c
drivers/leds/trigger/ledtrig-gpio.c
drivers/leds/trigger/ledtrig-default-on.c

Device Tree ドキュメント

Documentation/devicetree/bindings/leds/leds-gpio.txt

カーネルコンフィギュレーション

```
Device Drivers  --->
    [*] LED Support  --->                                          <NEW_LEDS>
        <*>   LED Class Support                                  <LEDS_CLASS>
              *** LED drivers ***
        <*>   LED Support for GPIO connected LEDs                 <LEDS_GPIO>
              *** LED Triggers ***
        [*]   LED Trigger support  --->                        <LEDS_TRIGGERS>
            <*>   LED Timer Trigger                          <LEDS_TRIGGER_TIMER>
            <*>   LED One-shot Trigger                     <LEDS_TRIGGER_ONESHOT>
            <*>   LED Heartbeat Trigger                  <LEDS_TRIGGER_HEARTBEAT>
            <*>   LED backlight Trigger                  <LEDS_TRIGGER_BACKLIGHT>
            <*>   LED GPIO Trigger                            <LEDS_TRIGGER_GPIO>
            <*>   LED Default ON Trigger                <LEDS_TRIGGER_DEFAULT_ON>
```

7.3.15. ユーザースイッチ

Armadillo-X1 に搭載されているユーザースイッチには、GPIO が接続されています。GPIO が接続されユーザー空間でイベント(Press/Release)を検出することができます。Linux では、GPIO 接続用キーボードドライバ(gpio-keys)で制御することができます。

ユーザースイッチには、次に示すキーコードが割り当てられています。

表 7.3 キーコード

ユーザースイッチ	キーコード	イベントコード
SW1	KEY_1	2

デバイスファイル

/dev/input/event1[2]

関連するソースコード

drivers/input/evdev.c
drivers/input/ff-core.c
drivers/input/input-compat.c
drivers/input/input-mt.c
drivers/input/input-polldev.c
drivers/input/input.c
drivers/input/keyboard/gpio_keys.c

Device Tree ドキュメント

Documentation/devicetree/bindings/gpio/gpio_keys.txt

[2]USB デバイスなどを接続してインプットデバイスを追加している場合は、番号が異なる可能性があります

カーネルコンフィギュレーション

```
Device Drivers  --->
    Input device support  --->
        -*- Generic input layer (needed for keyboard, mouse, ...)              <INPUT>
        -*-   Polled input device skeleton                                  <INPUT_POLLDEV>
              *** Userland interfaces ***
        <*>   Event interface                                                <INPUT_EVDEV>
              *** Input Device Drivers ***
        [*]   Keyboards  --->                                               <INPUT_KEYBOARD>
            <*>   GPIO Buttons                                               <KEYBOARD_GPIO>
```

7.3.16. I2C

Armadillo-X1 の I2C インターフェースは、i.MX 7Dual の I2C(I2C Controller) を利用します。また、GPIO を利用した I2C バスドライバ(i2c-gpio)を利用することで、I2C バスを追加することができます。

Armadillo-X1 で利用している I2C バスと、接続される I2C デバイスを次に示します。

表 7.4 I2C デバイス

I2C バス	I2C デバイス	
	アドレス	デバイス名
2(I2C3)	0x50	M24C01-W EEPROM[a]
3(I2C4)	0x08	USB3503 USB ハブ
	0x09	PF3000 パワーマネジメント IC
	0x10 ～ 0x17	Board Management IC

[a]アドオンインターフェース(CON7)にアドオンモジュールを接続した場合。

Armadillo-X1 の標準状態では、CONFIG_I2C_CHARDEV が有効となっているためユーザードライバで I2C デバイスを制御することができます。ユーザードライバを利用する場合は、Linux カーネルで I2C デバイスに対応するデバイスドライバを無効にする必要があります。

機能

最大転送レート: 400kbps

デバイスファイル

/dev/i2c-2 (I2C3)
/dev/i2c-3 (I2C4)

関連するソースコード

drivers/i2c/i2c-boardinfo.c
drivers/i2c/i2c-core.c
drivers/i2c/i2c-dev.c
drivers/i2c/algos/i2c-algo-bit.c
drivers/i2c/busses/i2c-gpio.c
drivers/i2c/busses/i2c-imx.c

Device Tree ドキュメント

Documentation/devicetree/bindings/i2c/i2c-imx.txt

カーネルコンフィギュレーション

```
Device Drivers  --->
    <*> I2C support  --->                                                  <I2C>
        <*>   I2C device interface                                  <I2C_CHARDEV>
        [*]   Autoselect pertinent helper modules                <I2C_HELPER_AUTO>
              I2C Hardware Bus support  --->
            <*> GPIO-based bitbanging I2C                             <I2C_GPIO>
            <*> IMX I2C interface                                      <I2C_MXC>
```

7.3.17. SPI

Armadillo-X1 の SPI インターフェースは、i.MX 7Dual の ECSPI(Enhanced Configurable SPI)を利用します。

標準状態では無効になっている CONFIG_SPI_SPIDEV を有効化すると、ユーザードライバで SPI デバイスを制御することができます。

関連するソースコード

drivers/spi/spi-bitbang.c
drivers/spi/spi-imx.c
drivers/spi/spi.c
drivers/spi/spidev.c

Device Tree ドキュメント

Documentation/devicetree/bindings/spi/fsl-imx-cspi.txt

カーネルコンフィギュレーション

```
Device Drivers  --->
    [*] SPI support  --->                                                  <SPI>
              *** SPI Master Controller Drivers ***
        -*-   Utilities for Bitbanging SPI masters                  <SPI_BITBANG>
        <*>   Freescale i.MX SPI controllers                           <SPI_MXC>
              *** SPI Protocol Masters ***
        < >   User mode SPI device driver support                   <SPI_SPIDEV>
```

7.3.18. ウォッチドッグタイマー

Armadillo-X1 のウォッチドッグタイマーは、i.MX 7Dual の WDOG(Watchdog Timer) を利用しています。

ウォッチドッグタイマーは、U-Boot によって有効化されます。標準状態でタイムアウト時間は 10 秒に設定されます。Linux カーネルは、ウォッチドッグタイマードライバの初期化時にタイムアウト時間を 10 秒に再設定します。

何らかの要因でウォッチドッグタイマーのキックができなくなりタイムアウトすると、システムリセットが発生します。

関連するソースコード

drivers/watchdog/imx2_wdt.c

Device Tree ドキュメント

Documentation/devicetree/bindings/watchdog/fsl-imx-wdt.txt

カーネルコンフィギュレーション

```
Device Drivers  --->
    [*] Watchdog Timer Support  --->                                    <WATCHDOG>
        <*>   IMX2+ Watchdog                                            <IMX2_WDT>
```

i.MX 7Dual の WDOG は、一度有効化すると無効化することができません。そのため、halt コマンドなどを実行して Linux カーネルを停止した場合は、ウォッチドッグタイマーのキックができなくなるためシステムリセットが発生します。

WDOG ドライバーの終了処理では、タイムアウト時間を WDOG の最大値である 128 秒に設定します。

7.3.19. CPU 周波数スケーリング

Armadillo-X1 の CPU 周波数スケーリング機能は、Linux の CPUFreq を利用しています。適切に CPU 周波数を調整することにより、CPU の消費電力を抑えることができます。

sysfs ディレクトリ

/sys/devices/system/cpu/cpu0/cpufreq/
/sys/devices/system/cpu/cpu1/cpufreq/

関連するソースコード

drivers/cpufreq/cpufreq.c
drivers/cpufreq/cpufreq_conservative.c
drivers/cpufreq/cpufreq_governor.c
drivers/cpufreq/cpufreq_interactive.c
drivers/cpufreq/cpufreq_ondemand.c
drivers/cpufreq/cpufreq_performance.c
drivers/cpufreq/cpufreq_stats.c
drivers/cpufreq/cpufreq_userspace.c
drivers/cpufreq/freq_table.c
drivers/cpufreq/imx7-cpufreq.c

カーネルコンフィギュレーション

```
CPU Power Management  --->
    CPU Frequency scaling  --->
        [*] CPU Frequency scaling                                       <CPU_FREQ>
        <*>   CPU frequency translation statistics                    <CPU_FREQ_STAT>
              Default CPUFreq governor (interactive)  --->
                  (X) interactive                      <CPU_FREQ_DEFAULT_GOV_INTERACTIVE>
              <*>   'performance' governor                      <CPU_FREQ_GOV_PERFORMANCE>
              <*>   'powersave' governor                          <CPU_FREQ_GOV_POWERSAVE>
              <*>   'userspace' governor for userspace frequency scaling
                                                                  <CPU_FREQ_GOV_USERSPACE>
              <*>   'ondemand' cpufreq policy governor             <CPU_FREQ_GOV_ONDEMAND>
              -*-   'interactive' cpufreq policy governor       <CPU_FREQ_GOV_INTERACTIVE>
              <*>   'conservative' cpufreq governor           <CPU_FREQ_GOV_CONSERVATIVE>
                    ARM CPU frequency scaling drivers  --->
                        <*> Freescale i.MX7 cpufreq support           <ARM_IMX7D_CPUFREQ>
```

Armadillo-X1 が対応する CPU 周波数と、CPU(ARM Cortex-A7)の電源電圧の対応を次に示します。

表 7.5 対応する CPU 周波数と電源電圧

CPU 周波数	電源電圧
996MHz	1.075V
792MHz	0.975V

CPU 周波数を決定する機構を、"Governor"と呼びます。工場出荷状態の Armadillo-X1 の Linux カーネルで利用可能な Governor を次に示します。

表 7.6 Governor の種類

Governor	説明
performance	負荷に関わらず、常に最大クロックで動作する。
powersave	負荷に関わらず、常に最小クロックで動作する。
userspace	SYSFS ファイルからユーザーがクロックを指定する。
ondemand	負荷がかかるとクロックを上げ、負荷が下がるとクロックも下げる。負荷がかかるとクロックが急激に上がる。
conservative	負荷がかかるとクロックを上げ、負荷が下がるとクロックも下げる。ondemand よりも、負荷に対するクロックの変化が段階的に行われる。
interactive	負荷がかかるとクロックを上げ、負荷が下がるとクロックも下げる。ondemand よりも、負荷に対するクロックの変化が速やかに行われる。

Governor の変更は sysfs ディレクトリ以下の scaling_governor から行うことができます。CPU0 の Governor 変更例を次に示します。

```
[armadillo ~]# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors ❶
conservative ondemand userspace powersave interactive performance
[armadillo ~]# echo performance > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor ❷
[armadillo ~]# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor ❸
performance
```

❶ 利用可能な Governor を表示します。

❷ Governor を performance に設定します。

❸ Governor を確認します。performance に設定されていることが確認できます。

7.3.20. パワーマネジメント

Armadillo-X1 のパワーマネジメント機能は、Linux の SPM(System Power Management)および DPM(Device Power Management)を利用しています。パワーマネジメント状態を省電力モードに遷移させることにより、Armadillo-X1 の消費電力を抑えることができます。

パワーマネジメント機能を利用する場合は、Linux カーネル linux-3.14-x1-at7 以降(カーネルイメージ ulmage-x1-v7.00 以降)をご利用ください。

パワーマネジメント状態を省電力モードに遷移させると、アプリケーションの実行は一時停止し、Linux カーネルはサスペンド状態となります。起床要因が発生すると、Linux カーネルのリジューム処理が行われた後、アプリケーションの実行を再開します。

sysfs ファイル

/sys/power/state

関連するソースコード

kernel/power/

カーネルコンフィギュレーション

```
Power management options  --->
    [*] Suspend to RAM and standby                              <SUSPEND>
    [*] Run-time PM core functionality                          <PM_RUNTIME>
```

Armadillo-X1 が対応するパワーマネジメント状態と、/sys/power/state に書き込む文字列の対応を次に示します。

表 7.7 対応するパワーマネジメント状態

パワーマネジメント状態	文字列	説明
Power-On Suspend	standby	Suspend-to-RAM よりも短時間で復帰することができる。
Suspend-to-RAM	mem	Power-On Suspend よりも消費電力を抑えることができる。

起床要因として利用可能なデバイスは次の通りです。

表 7.8 起床要因として利用可能なデバイス

デバイス	起床要因の有効化	起床要因
UART5(CON5)	[armadillo ~]# **echo enabled > /sys/bus/platform/drivers/imx-uart/30a70000.serial/tty/ttymxc4/power/wakeup** ⏎ ⏎	データ受信

デバイス	起床要因の有効化	起床要因
UART7(CON7)	[armadillo ~]# **echo enabled > /sys/bus/platform/drivers/imx-uart/30a90000.serial/tty/ttymxc6/power/wakeup**	データ受信 ⏎ ⏎
Ethernet(CON1)	[armadillo ~]# **apt-get install ethtool** [armadillo ~]# **ethtool -s eth0 wol g**	Wake-on-LAN のマジックパケットを受信
USB ホスト(CON2)	[armadillo ~]# **echo enabled > /sys/bus/platform/devices/30b10000.usb/power/wakeup** [armadillo ~]# **echo enabled > /sys/bus/platform/drivers/ci_hdrc/ci_hdrc.0/power/wakeup** [armadillo ~]# **echo enabled > /sys/bus/platform/drivers/ci_hdrc/ci_hdrc.0/usb1/power/wakeup**	USB デバイスの挿抜 ⏎ ⏎ ⏎

Ethernet から起床要因である Wake-on-LAN のマジックパケットを、ATDE から送信する例を次に示します。

```
[PC ~]$ sudo apt-get install wakeonlan
[PC ~]$ wakeonlan [MAC Address] ❶
```

❶ Armadillo-X1 の有線 LAN の MAC アドレスを指定します。

8. Debian ユーザーランド仕様

本章では、工場出荷状態の Armadillo-X1 の Deabin ユーザーランドの基本的な仕様について説明します。

8.1. Debian ユーザーランド

Armadillo-X1 の標準ルートファイルシステムは、32-bit hard-float ARMv7 (「armhf」)アーキテクチャ用の Debian GNU/Linux 8 (コードネーム 「jessie」)です。出荷状態、または標準イメージを展開した直後のユーザーランド内には、Armadillo の動作に必要な最小限のパッケージや設定が含まれています。

Armadillo-X1 にインストールされた Debian GNU/Linux 8 は、eMMC または SD カード上で動作します。Linux カーネルが動作している状態で Armadillo の電源を切断する場合は、必ず 「halt」コマンドによる終了を行い、RAM 上にキャッシュされている eMMC または SD カード への書き込み処理を完了するようにしてください。再起動を行う場合も同様に、reboot コマンドによる再起動を行ってください。

8.2. パッケージ管理

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

工場出荷状態では、APT はインターネット上の Debian サイト(HTTP サーバー)から利用可能なパッケージのインデックスを取得します[1]。そのため、APT を使用するためにはネットワークを有効化し、インターネットに接続できる状態にしておく必要があります。

ネットワークを有効化する方法については、「6.2. ネットワーク」を参照してください。

システムクロックが大幅にずれた状態で、APT を利用すると警告メッセージが出力される場合があります。事前に「6.5. RTC」を参照してシステムクロックを合わせてください。

apt-get update

パッケージインデックスファイルを最新の状態にアップデートします。

引数　無し

使用例

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

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

apt-get upgrade

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

引数　無し

使用例

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

apt-get install [パッケージ名]

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

引数　パッケージ名(複数指定可能)

使用例

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

apt-get remove [パッケージ名]

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

引数　パッケージ名(複数指定可能)

使用例

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

apt-cache search [キーワード]

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

引数　キーワード(正規表現が使用可能)

使用例

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

9. ブートローダー仕様

本章では、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。

9.1. 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、USB シリアル変換アダプタのスライドスイッチの状態により、2 つのモードのどちらかに遷移します。USB シリアル変換アダプタのスライドスイッチの詳細については、「4.5. スライドスイッチの設定について」を参照してください。

表 9.1 ブートローダー起動モード

起動モードの種別	スライドスイッチ	説明
保守モード	外側	各種設定が可能な U-Boot コマンドプロンプトが起動します。
オートブートモード	内側	電源投入後、自動的に Linux カーネルを起動させます。

USB シリアル変換アダプタが未接続の場合オートブートモードとなり、Linux カーネルが起動します。

9.2. ブートローダーの機能

U-Boot の保守モードでは、Linux カーネルの起動オプションの設定などを行うことできます。

保守モードで利用できる有用なコマンドは、「表 9.2. 保守モード 有用なコマンド一覧」に示します。

表 9.2 保守モード 有用なコマンド一覧

コマンド	説明
boot	OS を起動する場合に使用します
bdinfo	ハードウェアの情報を表示します
md mm nm mw cp cmp	簡易的にメモリアクセスする場合に使用します
printenv setenv saveenv	環境変数の設定をする場合に使用します、環境変数にて OS の起動設定等をおこなうことができます
crc32	メモリ空間のチェックサムを表示する場合に使用します
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 9.1. U-Boot コマンドのヘルプを表示」のようにします。

```
=> help [コマンド]
```

図 9.1 U-Boot コマンドのヘルプを表示

9.2.1. Linux カーネルイメージと device tree blob の指定方法

ブートローダーが OS を起動させる場合、eMMC または、SD カード内に保存されている Linux カーネルイメージと device tree blob を使用することができます。

ファイルを保存しているデバイスを指定するには、環境変数 "mmcdev" を、パーティション番号を指定するには 環境変数 "mmcpart" を使用します。

Linux カーネルイメージはファイル名 "uImage" で保存されたものを使用します。device tree blob はファイル名 "armadillo_x1.dtb" で保存されたものを使用します。

"mmcdev" で設定可能な値と、起動デバイスの関係を次に示します。 CON7 アドオンインターフェースに、SD スロット拡張ボードを接続した場合と、しない場合で、起動デバイスの番号が変わります。

表 9.3 mmcdev の設定値と起動デバイス SD スロット拡張ボードを接続しない場合

設定値	起動デバイス
0	eMMC

表 9.4 mmcdev の設定値と起動デバイス SD スロット拡張ボードを接続した場合

設定値	起動デバイス
0	SD(SD スロット拡張ボードを接続した場合のみ)
1	eMMC

SD スロット拡張ボードを接続し状態で、eMMC のパーティション 1 を指定する場合、「図 9.2. eMMC のパーティション 1 に保存された Linux カーネルイメージから起動する」のようにします。

```
=> setenv mmcdev 1
=> setenv mmcpart 1
```

図 9.2 eMMC のパーティション 1 に保存された Linux カーネルイメージから起動する

ブートローダーの種類と、デフォルト値の関係を「表 9.5. ブートローダーの種類と mmcdev, mmcpart のデフォルト値」に示します。

表 9.5 ブートローダーの種類と mmcdev, mmcpart のデフォルト値

ブートローダーの種類	ブートローダーファイル名	mmcdev デフォルト値	mmcpart デフォルト値
QSPI 用	u-boot-x1-at*.bin	0(eMMC)	1
SD 用	u-boot-x1-sd-at*.bin	0(SD)	1

9.2.2. ルートファイルシステムの指定方法

ルートファイルシステムが構築されているデバイスは、環境変数 "mmcroot" で指定することができます。

eMMC のパーティション 2 を指定する場合、「図 9.3. eMMC のパーティション 2 に保存されたルートファイルシステムを指定する」のようにします。

```
=> setenv mmcroot /dev/mmcblk2p2
```

図 9.3 eMMC のパーティション 2 に保存されたルートファイルシステムを指定する

QSPI 用のブートローダーと、SD 用のブートローダーにて、"mmcroot"のデフォルト値が異なります。ブートローダーの種類と、デフォルト値の関係を「表 9.6. ブートローダーの種類と mmcroot のデフォルト値」に示します。

表 9.6 ブートローダーの種類と mmcroot のデフォルト値

ブートローダーの種類	ブートローダーファイル名	mmcroot デフォルト値
QSPI 用	u-boot-x1-at*.bin	/dev/mmcblk2p2(eMMC パーティション 2)
SD 用	u-boot-x1-sd-at*.bin	/dev/mmcblk0p2(SD パーティション 2)

9.2.3. 環境変数の保存

環境変数は"saveenv"コマンドにて保存することができます。保存を行わずに、Armadillo-X1 の電源を切ると setenv で設定した環境変数は消えてしまいます。

QSPI 用のブートローダーを使用した場合、環境変数は QSPI 内に保存されます。SD 用のブートローダーを使用した場合、環境変数は SD 内に保存されます。

全ての環境変数をデフォルト値に戻すには、「図 9.4. 全ての環境変数をデフォルト値に戻す」のようにした後、電源を切断・接続します。

```
=> env default -a
=> saveenv
```

図 9.4 全ての環境変数をデフォルト値に戻す

9.2.4. Linux カーネル起動オプション

9.2.4.1. 代表的な Linux カーネル起動オプション

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-X1 で使用することができる、代表的な起動オプションを「表 9.7. Linux カーネルの起動オプションの一例」に紹介します。

表 9.7 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 ttymxc1 を、ボーレートに 115200 を指定しています。 `console=ttymxc1,115200`

オプション指定子	説明

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。
	initrd をルートファイルシステムとする場合には、以下の例のように設定します。
	`root=/dev/ram0`
	SD カードにルートファイルシステムを配置する場合には、SD カードのデバイスファイルを指定します。次の例では、デバイスに microSD 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。
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。

9.2.4.2. Linux カーネル起動オプションの設定方法

Linux カーネル起動オプションは環境変数 "mmcargs" で指定することができます。

"mmcargs" のデフォルト値は次に示す値に設定されています。

```
setenv mmcargs setenv bootargs console=${console},${baudrate} root=${mmcroot} ${optargs}
```

デフォルトでは、コンソールには環境変数 "console"が、コンソールのボーレートには環境変数 "baudrate"が、ルートファイルシステムには、環境変数 "mmcroot"が設定されています。

Linux カーネル起動オプションの追加をしたい場合、環境変数 "optargs"を使用すると便利です。

次に、例として、Linux カーネルが利用可能なメモリの量を 384M に設定する方法を「図 9.5. 利用可能なメモリ量を 384M にする」に示します。

```
=> setenv optargs mem=384M
=> saveenv
=> printenv optargs
mem=384M
```

図 9.5 利用可能なメモリ量を 384M にする

10. ビルド手順

本章では、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-X1 ドキュメント・ダウンロード

http://armadillo.atmark-techno.com/armadillo-x1/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。

10.1. 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「U-Boot」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 10.1 ブートローダーをビルド

1. **ソースコードの準備**

   U-Boot のソースコードアーカイブを準備し展開します。

```
[PC ~]$ ls
uboot_2016.07-at[version].tar.gz
[PC ~]$ tar xf uboot_2016.07-at[version].tar.gz
[PC ~]$ ls
uboot_2016.07-at[version]  uboot_2016.07-at[version].tar.gz
```

2. **デフォルトコンフィギュレーションの適用**

   U-Boot ディレクトリに入り、Armadillo-X1 用のデフォルトコンフィギュレーションを適用します。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 x1_config を指定します。SD 起動用イメージを作成する場合は、x1_sd_config を指定してください。

```
[PC ~]$ cd uboot_2016.07-at[version]
[PC ~/uboot_2016.07-at[version]]$ make ARCH=arm x1_config
```

3. **ビルド**

   ビルドには **make** コマンドを利用します。

```
[PC ~/uboot_2016.07-at[version]]$ make CROSS_COMPILE=arm-linux-gnueabihf-
```

4. **イメージファイルの生成確認**

   ビルドが終了すると、U-Boot ディレクトリにイメージファイルが作成されています。

```
[PC ~/uboot_2016.07-at[version]]$ ls u-boot-x1.bin
u-boot-x1.bin
```

10.2. Linux カーネルをビルドする

ここでは、Linux カーネルのソースコードと initramfs アーカイブから、イメージファイルを作成する手順を説明します。

手順 10.2 Linux カーネルをビルド

1. **アーカイブの展開**

   Linux カーネルのソースコードアーカイブを展開します。

```
[PC ~]$ ls
initramfs_x1-[version].cpio.gz linux-3.14-x1-at[version].tar.gz
[PC ~]$ tar xf linux-3.14-x1-at[version].tar.gz
[PC ~]$ ls
initramfs_x1-[version].cpio.gz linux-3.14-x1-at[version]  linux-3.14-x1-
at[version].tar.gz
```

2. **initramfs アーカイブへのシンボリックリンク作成**

   Linux カーネルディレクトリに移動して、initramfs アーカイブへのシンボリックリンク作成します。

```
[PC ~]$ cd linux-3.14-x1-at[version]
[PC ~/linux-3.14-x1-at[version]]$ ln -s ../initramfs_x1-[version].cpio.gz
initramfs_x1.cpio.gz
```

3. **コンフィギュレーション**

   コンフィギュレーションをします。

```
[PC ~/linux-3.14-x1-at[version]]$ make ARCH=arm x1_defconfig
```

4. **ビルド**

   ビルドするには、次のようにコマンドを実行します。

```
[PC ~/linux-3.14-x1-at[version]]$ make CROSS_COMPILE=arm-linux-gnueabihf- ARCH=arm
[PC ~/linux-3.14-x1-at[version]]$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- ⏎
LOADADDR=0x80008000 uImage
```

5. **イメージファイルの生成確認**

ビルドが終了すると、arch/arm/boot/ディレクトリと、arch/arm/boot/dts/以下にイメージファイル(Linux カーネルと DTB)が作成されています。

```
[PC ~/linux-3.14-x1-at[version]]$ ls arch/arm/boot/uImage
uImage
[PC ~/linux-3.14-x1-at[version]]$ ls arch/arm/boot/dts/armadillo_x1.dtb
armadillo_x1.dtb
```

10.3. Debian GNU/Linux ルートファイルシステムをビルドする

ここでは、x1-debian-builder を使って、Debian GNU/Linux ルートファイルシステムを構築する方法を示します。

x1-debuan-builder は ATDE6 等の PC で動作している Linux 上で Armadillo-X1 用の armhf アーキテクチャに対応した Debian GNU/Linux ルートファイルシステムを構築することができるツールです。

Armadillo-X1 を一度起動した後のルートファイルシステム上には、使い方によっては ssh の秘密鍵や、動作ログ、シェルのコマンド履歴、ハードウェアの UUID に紐付く設定ファイル等が生成されています。そのまま、他の Armadillo-X1 にルートファイルシステムをコピーした場合は、鍵の流出や UUID の不一致による動作の相違が起きる可能性があります。そのため、量産等に使用するルートファイルシステムは新規に x1-debuan-builder を使って構築することをお勧めします。

10.3.1. 出荷状態のルートファイルシステムアーカイブを構築する

出荷状態のルートファイルシステムアーカイブを構築する手順を次に示します。パッケージをインターネット上から取得するため回線速度に依存しますが、40 分程度かかります。

```
[ATDE ~]$ tar xf x1-debian-builder-[VERSION].tar.gz
[ATDE ~]$ cd x1-debian-builder-[VERSION]
[ATDE ~]$ sudo ./build.sh ax1
```

図 10.1 出荷状態のルートファイルシステムアーカイブを構築する手順

10.3.2. カスタマイズされたルートファイルシステムアーカイブを構築する

x1-debian-builder-*[VERSION]*/aiotg3_resources 内のファイルを変更し、build.sh を実行することで、ルートファイルシステムをカスタマイズすることができます。

10.3.2.1. ファイル/ディレクトリを追加する

aiotg3_resources/ 以下に配置したファイルやディレクトリは resources ディレクトリを除いて、そのまま、ルートファイルシステムの直下にコピーされます。ファイルの UID と GID は共に root になります。

10.3.2.2. パッケージを変更する

aiotg3_resources/resources/packages を変更することで、ルートファイルシステムにインストールするパッケージをカスタマイズすることができます。

パッケージ名は 1 行に 1 つ書くことができます。パッケージ名は Armadillo-X1 上で "apt-get install" の引数に与えることのできる正しい名前で記載してください。

誤ったパッケージ名を指定した場合は、ビルドログに以下のようなエラーメッセージが表示されて当該のパッケージが含まれないアーカイブが生成されます。

```
E: Unable to locate package XXXXX
```

図 10.2 誤ったパッケージ名を指定した場合に起きるエラーメッセージ

パッケージに依存する他のパッケージは明記しなくても、apt によって自動的にインストールされます。また、apt や dpkg 等の Debian GNU/Linux の根幹となるパッケージも自動的にインストールされます。

packages には lua と ruby のインタプリタや、Web サーバー(lighttpd)が含まれていますが、これらが不要な場合は、それぞれの行を削除してください。

openssh-server のような「パッケージのインストールの際に、自動的に秘密鍵を生成する」パッケージは、基本的に packages には追加せず、Armadillo を起動した後に "apt-get install" を使って個別にインストールしてください。

openssh-server を packages に追加した場合、構築したルートファイルシステムアーカイブを書き込んだ全ての Armadillo に、単一の公開鍵を使ってログインすることができてしまいます。もし、意図的に、複数の Armadillo で同一の秘密鍵を利用したい場合、脆弱性となり得ることを理解して適切な対策をとった上で利用してください。

11. イメージファイルの書き換え方法

本章では、Armadillo-X1 の内蔵ストレージ(eMMC 及び QSPI 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書き換える手順について説明します。

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、評価セット付属の DVD に収録されています。

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、評価セット付属の 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、最新バージョンを利用することを推奨します。

Armadillo サイト - Armadillo-X1 ドキュメント・ダウンロード

http://armadillo.atmark-techno.com/armadillo-x1/downloads

11.1. インストールディスクを使用する

インストールディスクを使用すると、内蔵ストレージ上のすべてのイメージをまとめて書き換えることができます。 Armadillo がソフトウェアの問題により起動しなくなった場合の復旧方法としてもご使用頂けます。

内蔵ストレージに保存されている、すべてのイメージファイルが上書きされるため、既に保存されているデータやアプリケーションなどは削除されます。

特定のイメージのみ書き換えたい場合には 「11.2. 特定のイメージファイルだけを書き換える」 を参照してください。

インストールディスクは ATDE で作成します。インストールディスクの作成に使用するファイルを次に示します。

表 11.1 インストールディスク作成に使用するファイル

ファイル	ファイル名
インストールディスクイメージ	install_disk_sd_*[version]*.img

11.1.1. インストールディスクの作成

1. 512 MB 以上の SD カードを用意してください。
2. ATDE に SD カードを接続します。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。
3. SD カードがマウントされている場合、アンマウントします。

```
[PC ~]$ mount
(省略)
/dev/sdb1 on /media/atmark/B18A-3218 type vfat
(rw,nosuid,nodev,relatime,uid=1000,gid=1000,fmask=0022,dmask=0077,codepage=437,iocharse
t=utf8,shortname=mixed,showexec,utf8,flush,errors=remount-ro,uhelper=udisks2)
[PC ~]$ sudo umount /dev/sdb1
```

4. SD カードにインストールディスクイメージを書き込みます。

```
[PC ~]$ sudo dd if=install_disk_sd_[version].img of=/dev/sdb bs=4M
94+1 レコード入力
94+1 レコード出力
397410304 バイト (397 MB) コピーされました、 45.8441 秒、 8.7 MB/秒
[PC ~]$ sync
```

11.1.2. インストールの実行

1. 電源が切断されていることを確認します。接続されていた場合は、電源を切断してください。

2. USB シリアル変換アダプタのスライドスイッチを確認します。スライドスイッチが「図 4.8. スライドスイッチの設定」の 1 側に設定されている事を確認してください。

3. インストールディスクを使用して SD ブートを行います。インストールディスクを接続した、SD スロット拡張ボード接続し、「19.1.3. インターフェース仕様」を参照して SD スロット拡張ボードの SW1 を「SD BOOT」に設定してください。

4. ユーザースイッチを押しながら Armadillo に電源を投入すると SD カードからブートローダーが起動し、次に示すログが表示されます。 ユーザースイッチの位置については「3.4. Armadillo-X1 の外観」を参照してください。

```
U-Boot 2016.07-at3 (Sep 16 2016 - 15:29:22 +0900)

CPU:   Freescale i.MX7D rev1.1 996 MHz (running at 792 MHz)
CPU:   Extended Commercial temperature grade (-20C to 105C) at 45C
Reset cause: POR
       Watchdog enabled
I2C:   ready
DRAM:  512 MiB
Board Type: Armadillo-X1(0a100000)
Revison: 0001
S/N: 12
DRAM: 00001d05
XTAL: 00
X1 Addon EEPROM Detect
Atmark Techno Ext SD Slot Detect
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
SF: Detected N25Q512 with page size 256 Bytes, erase size 64 KiB, total 64 MiB
In:    serial
Out:   serial
Err:   serial
Found PFUZE300! deviceid 0x30, revid 0x11
```

```
Net:   FEC0
=>
```

5. 次のように"boot"コマンドを実行するとインストールが始まり、自動的に eMMC と QSPI が書き換えられます。

```
=> boot
switch to partitions #0, OK
mmc1(part 0) is current device
switch to partitions #0, OK
mmc1(part 0) is current device
reading boot.scr
** Unable to read file boot.scr **
reading boot.scr
** Unable to read file boot.scr **
reading uImage
9784016 bytes read in 241 ms (38.7 MiB/s)
Booting from mmc ...
reading armadillo_x1.dtb
50454 bytes read in 17 ms (2.8 MiB/s)
## Booting kernel from Legacy Image at 82000000 ...
   Image Name:   Linux-3.14.76-at3
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    9783952 Bytes = 9.3 MiB
   Load Address: 80008000
   Entry Point:  80008000
   Verifying Checksum ... OK
## Flattened Device Tree blob at 84800000
   Booting using the fdt blob at 0x84800000
   Loading Kernel Image ... OK
   Using Device Tree in place at 84800000, end 8480f515

Starting kernel ...
: (省略)
*+*+* Recovery Start!! *+*+*
```

6. 以下のようにメッセージが表示され、自動的に halt するとインストール完了です。

```
*+*+* Recovery Completed!! *+*+*

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
reboot: System halted
```

11.2. 特定のイメージファイルだけを書き換える

Armadillo-X1 が起動した状態であれば、特定のイメージファイルだけを書き換えることができます。

イメージファイルと書き込み先の対応を次に示します。

表 11.2 イメージファイルと書き込み先の対応

名称	ファイル名	ストレージ	デバイスファイル
ブートローダーイメージ	u-boot-x1-*[version]*.bin	QSPI フラッシュメモリ	/dev/mtdblock0
Linux カーネルイメージ	uImage-x1-*[version]*	eMMC	/dev/mmcblk2p1
Device Tree Blob	armadillo_x1-*[version]*.dtb		/dev/mmcblk2p1
Debian GNU/Linux ルートファイルシステム	debian-jessie-armhf_x1_*[version]*.tar.gz		/dev/mmcblk2p2

11.2.1. ブートローダーイメージの書き換え

ブートローダーイメージの書き換え方法を次に示します。MTD のブロックデバイスに直接イメージファイルを書き込むことで行います。

```
[armadillo ~]# dd if=u-boot-x1-[version].bin of=/dev/mtdblock0 ❶
282+1 records in
282+1 records out
288816 bytes (289 kB) copied, 5.4582 s, 52.9 kB/s
[armadillo ~]$ sync
```

❶　MTD のブロックデバイスの先頭からブートローダーイメージを書き込みます。

製品アップデートでリリースされた最新のブートローダーイメージに書き換えを行った場合は「図 9.4. 全ての環境変数をデフォルト値に戻す」を参照して環境変数をデフォルト値に戻してください。

新しくリリースされた最新のブートローダーイメージは、環境変数のデフォルト値が更新されることがあります。書き替え前のブートローダーによって生成された環境変数で、最新のブートローダーを動作させると不具合が起こる可能性があります。

11.2.2. Linux カーネルイメージの書き換え

Linux カーネルイメージの書き換え方法を次に示します。

```
[armadillo ~]# mount -t vfat /dev/mmcblk2p1 /mnt ❶
[armadillo ~]# cp uImage-x1-[version] /mnt/uImage ❷
[armadillo ~]# umount /mnt ❸
```

❶　eMMC の第 1 パーティションを/mnt/ディレクトリにマウントします。

❷　Linux カーネルイメージを/mnt/ディレクトリにコピーします。

❸　/mnt/ディレクトリにマウントした eMMC の第 1 パーティションをアンマウントします。

11.2.3. DTB の書き換え

DTB の書き換え方法を次に示します。

```
[armadillo ~]# mount -t vfat /dev/mmcblk2p1 /mnt ❶
[armadillo ~]# cp armadillo_x1-[version].dtb /mnt/armadillo_x1.dtb ❷
[armadillo ~]# umount /mnt ❸
```

❶　eMMC の第 1 パーティションを/mnt/ディレクトリにマウントします。

❷　DTB を/mnt/ディレクトリにコピーします。

❸　/mnt/ディレクトリにマウントした eMMC の第 1 パーティションをアンマウントします。

11.2.4. ルートファイルシステムの書き換え

eMMC 上のルートファイルシステムを書き換える手順を次に示します。

手順 11.1 eMMC 上のルートファイルシステムを書き換える

1. eMMC 上のルートファイルシステムを書き換えるには、SD ブートを行う必要があります。ブートディスクの作成方法や SD ブートの実行方法ついては「14. SD ブートの活用」を参照してください。

2. 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。

```
[armadillo ~]# ls
debian-jessie-armhf_x1_[version].tar.gz
```

3. ルートファイルシステムを eMMC の第 2 パーティションに再構築します。

```
[armadillo ~]# mount -t ext4 /dev/mmcblk2p2 /mnt ❶
[armadillo ~]# rm -rf /mnt/* ❷
[armadillo ~]# tar zxf debian-jessie-x1_[version].tar.gz -C /mnt ❸
[armadillo ~]# umount /mnt ❹
```

❶　eMMC の第 2 パーティションを/mnt/ディレクトリにマウントします。

❷　/mnt/以下のファイルを全て削除します。

❸　ルートファイルシステムアーカイブを/mnt/ディレクトリに展開します。

❹　/mnt/ディレクトリにマウントした eMMC の第 2 パーティションをアンマウントします。

12. 開発の基本的な流れ

この章では Armadillo-X1 を使ったアプリケーションソフトウェアの 開発方法について説明します。

Armadillo-X1 を使ったアプリケーションソフトウェア開発には、Ruby 等の軽量スクリプト言語を使うことができます。

新たにパケット通信、各種アドオンボードを利用したセンサーからのデータ読み出し を実装したアプリケーションプログラムを実装するときは、Ruby 等の軽量スクリプト言語を 使った開発をお勧めします。

Armadillo-X1 の出荷用のユーザーランドには、最初から Ruby インタプリタ がインストールされているので、PC と同じように開発を進めることができます。

もちろん、Ruby に限らず、Debian の提供する豊富なパッケージ群から Python や Go、 Haskel といったスクリプト言語を自由にインストールして使うことも可能です。

12.1. 軽量スクリプト言語によるセンサーデータの送信例(Ruby)

ここでは、サンプルとして Armadillo-X1 に搭載された温度センサーの値を 定期的に HTTP POST でパラメータ名 "temp" に格納した値として送信する例を示します。

温度センサーからの値の取得は sysfs から可能です。

ここで作成するアプリケーションは Armadillo-X1 で動作するクライアントと ATDE で動作するテスト用のサーバーの 2 つです。 ATDE で動作させるためのテスト用のサーバーは、典型的な HTTP プロトコルで アクセスできる Web API を持ったサービスを模擬しています。テスト用サーバーは 単に入力された POST リクエストの内容を変数に格納してコンソールに出力し、 クライアントには"Thanks!"という文字列を返します。

12.1.1. テスト用サーバーの実装

最初に ATDE6 にテスト用サーバーの動作に必要なパッケージをインストールします。

```
[PC ~]$ sudo apt-get install ruby
[PC ~]$ sudo gem install sinatra
```

図 12.1 ruby と sinatra のインストール

次にエディタで次のコードを入力して、server.rb として保存してください。

```
require 'sinatra'

post '/' do
  puts "Temperature is #{params[:temp]}"
  "Thanks!\n"
end
```

図 12.2 テスト用サーバー (server.rb)

12.1.2. テスト用サーバーの動作確認

Armadillo-X1 でクライアントアプリケーションを動かす前に、テスト用サーバーの動作確認を行います。 動作確認は Armadillo-X1 から cURL コマンドを使って、クライアントアプリケーションと同等のリクエストを送ってみます。

まず、ATDE6 の IP アドレス を確認しておきます。下記の例では、ip または ifconfig コマンドで確認すると ATDE6 の IP アドレス が 172.16.2.117 であることがわかります。

```
[PC ~]$  ip addr show eth0
      2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group
default qlen 1000
    link/ether 00:0c:29:30:b0:e0 brd ff:ff:ff:ff:ff:ff
    inet 172.16.2.117/16 brd 172.16.255.255 scope global dynamic eth0
       valid_lft 65913sec preferred_lft 65913sec
    inet6 fe80::20c:29ff:fe30:b0e0/64 scope link
       valid_lft forever preferred_lft forever
```

図 12.3 IP アドレスの確認 (ip コマンド)

```
[PC ~]$  sudo ifconfig
eth0      Link encap:イーサネット  ハードウェアアドレス 00:0c:29:30:b0:e0
          inet アドレス:172.16.2.117 ブロードキャスト:172.16.255.255  マスク:255.255.0.0
          inet6 アドレス: fe80::20c:29ff:fe30:b0e0/64 範囲:リンク
          UP BROADCAST RUNNING MULTICAST  MTU:1500  メトリック:1
          RX パケット:132712 エラー:59329 損失:0 オーバラン:0 フレーム:0
          TX パケット:13632 エラー:0 損失:0 オーバラン:0 キャリア:0
      衝突(Collisions):0 TX キュー長:1000
          RX バイト:124163876 (118.4 MiB)  TX バイト:970956 (948.1 KiB)
          割り込み:19 ベースアドレス:0x2024
```

図 12.4 IP アドレスの確認 (ifconfig コマンド)

次の例のように server.rb を実行すると、全ての IP アドレスからのリクエストを 8081 番ポートで Web サーバーとして待ち受けます。

```
[PC ~]$  ruby server.rb -p 8081 -o 0.0.0.0
[2016-03-28 16:02:15] INFO  WEBrick 1.3.1
[2016-03-28 16:02:15] INFO  ruby 2.1.5 (2014-11-13) [i386-linux-gnu]
== Sinatra (v1.4.7) has taken the stage on 4567 for development with backup from WEBrick
[2016-03-28 16:02:15] INFO  WEBrick::HTTPServer#start: pid=10849 port=8081
```

ここで、Armadillo-X1 から cURL を使ってテストデータを送ってみましょう。正しく通信できた場合は、"Thanks!" の文字列が表示されます。もし、"Connection refused" 等が表示された場合は、一旦 ATDE6 の IP アドレスに ping を送信してネットワークの設定に問題が無いか確認してください。

```
[Armadillo ~]$   curl -d "temp=30" 172.16.2.117:8081
Thanks!
```

図 12.5 curl によるテストデータの送信

正しく受信できた場合は、ATDE6 で起動しているテスト用サーバーが起動しているコンソールに下記の文字列が出力されます。

```
Temperature is 30
```

図 12.6 ATDE6 におけるテストデータの受信表示

12.2. クライアントの実装

Armadillo-X1 で動作するクライアントを実装します。下記のコードをエディタで入力して、client.rb として保存してください。ファイルは、ATDE6 上で作成しても Armadillo-X1 上で、vi 等を使って作成しても構いません。ATDE6 で作成した場合は次の手順で、Armadillo-X1 へ転送します。

```
require 'net/http'

uri = URI.parse(ARGV[0])
thermal_sys = "/sys/class/thermal/thermal_zone0/temp"
File.open(thermal_sys, "r") do |f|
  @temp=(f.read.to_f/1000).round(2)
end
response = Net::HTTP.post_form(uri, {"temp" => @temp})

puts response.body
```

図 12.7 温度送信クライアント(client.rb)

12.3. Armadillo-X1 へのファイルの転送

ATDE6 上で作成したソースコードを Armadillo-X1 に配置する方法の一例として、 ここでは、SSH を使った転送方法を説明します。

```
[armadillo ~]#  apt-get install openssh-server
```

図 12.8 Armadillo-X1 への SSH サーバーのインストール

```
[ATDE ~]$ scp client.rb atmark@[armadillo の IP アドレス]:~/
```

図 12.9 ATDE6 から Armadillo-X1 への client.rb の転送

12.4. クライアントの実行

作成した温度送信クライアントを実行します。第一引数にはテスト用サーバーが動いている ATDE6 の IP アドレスとポートを HTTP スキーマの URI で記述してください。

```
[armadillo ~]#  ruby client.rb  http://172.16.2.117:8081
Thanks!
```

図 12.10 クライアントの実行方法

正しくクライアントとの通信ができた場合、ATDE6 で動作しているサーバーのコンソールには小数点以下 2 ケタの温度が表示されます。

```
Temperature is 33.02
```

図 12.11 ATDE6 における温度データの受信表示

12.5. C 言語による開発環境

C/C++等の資産がある場合は、Armadillo 上で gcc/g++を使ってアプリケーションを コンパイルする事もできます。

12.5.1. 開発環境の準備

アプリケーションをコンパイルするために、Armadillo に gcc 等を含むツールチェーンを インストールします。 Armadillo のコンソールで次のコマンドを実行してください。

```
[armadillo ~]#  apt-get install build-essential
```

図 12.12 ツールチェーンのインストール

これで、gcc, make, gdb 等が使えるようになりました。 次に、アプリケーションのビルドに必要なライブラリとヘッダーファイルを インストールします。例えば libssl であれば次のコマンドでインストールする ことができます。

```
[armadillo ~]#  apt-get install libssl-dev
```

図 12.13 開発用パッケージのインストールの例 (libssl の場合)

例に示すように、コンパイルに必要なヘッダーファイルを含むパッケージは、 普通 -dev という名前が付いています。

必要なヘッダファイルの名前や、共有ライブラリのファイル名がわかっている 場合は、Debian プロジェクトサイトの「パッケージの内容を検索」からファイルの 含まれるパッケージの名前を探す事ができます。

Debian -- パッケージ パッケージの内容を検索 https://www.debian.org/distrib/packages#search_contents

また、パッケージの部分的な名前が分っている場合は「8.2. パッケージ管理」で紹介した、 apt-cache search コマンドを使って必要なパッケージを探す事もできます。

13. i.MX 7Dual の電源制御

本章では、パワーマネジメント IC による i.MX 7Dual の電力供給を制御する方法について説明します。

i.MX 7Dual の電源は、パワーマネジメント IC によって制御されています。パワーマネジメント IC の電圧出力を停止・開始することで、i.MX 7Dual の電源を ON または OFF にすることができます。

13.1. i.MX 7Dual 自身による制御

GPIO クラスディレクトリ以下の value ファイルに値を書き込むことによって、i.MX 7Dual 自身で電源を OFF にすることができます。

対応する、GPIO クラスディレクトリは、/sys/class/gpio/gpio255/ です。

電源を OFF にするには、次のようにコマンドを実行します。

```
[armadillo ~]# echo 255 > /sys/class/gpio/export
[armadillo ~]# echo 0 > /sys/class/gpio/gpio255/value
```

図 13.1 GPIO の export と value ファイルへの書き込みによる電源 OFF

13.2. アドオンインターフェースによる制御

アドオンインターフェース(CON7)の 55 ピン PMIC_ONOFF 信号によって、i.MX 7Dual の電源を ON または OFF にすることができます。

PMIC_ONOFF 信号を、2 秒以上 GND にショートすると、i.MX 7Dual の電源を OFF にすることができます。 PMIC_ONOFF 信号を、2 秒未満 GND にショートすると、i.MX 7Dual の電源を ON にすることができます。

13.3. RTC による制御

RTC のアラーム割り込みによって、i.MX 7Dual の電源を ON にすることができます。

アラーム割り込みは、sysfs RTC クラスディレクトリ以下の wakealarm ファイルから利用できます。

wakealarm ファイルに UNIX エポックからの経過秒数、または先頭に+を付けて現在時刻からの経過秒数を書き込むと、アラーム割り込み発生時刻を指定できます。

3600 秒後、アラーム割り込みを発生させるには、次のようにコマンドを実行します。

```
[armadillo ~]# echo +3600 > /sys/class/rtc/rtc0/wakealarm
```

図 13.2 アラーム割り込みの設定

コマンド実行後、「13.2. アドオンインターフェースによる制御」 を参照し、i.MX 7Dual の電源を OFF にします。

3600 秒後、アラーム割り込みによって i.MX 7Dual の電源が ON になります。

13.4. ユーザースイッチ 1(SW1)の操作による制御

ユーザースイッチ 1(SW1)の操作によって、i.MX 7Dual の電源を ON にすることができます。

「13.2. アドオンインターフェースによる制御」 、 「13.1. i.MX 7Dual 自身による制御」を参照し、i.MX 7Dual の電源を OFF にします。その後、ユーザースイッチ 1(SW1)を押すことで、i.MX 7Dual の電源が ON になります。

ユーザースイッチ 1(SW1)の位置については、「3.4. Armadillo-X1 の外観」を参照してください。

14. SD ブートの活用

本章では、SD カードから直接起動(以降「SD ブート」と表記します)する手順を示します。SD ブートを活用すると、SD カードを取り替えることでシステムイメージを変更することができます。本章に示す手順を実行するためには、容量が 2GByte 以上の SD カードを必要とします。以下では、例として Debian GNU/Linux 8(コードネーム jessie)を SD ブートする手順を示しますが、他の OS を SD ブートすることも可能です。

SD ブートを行った場合、ブートローダーの設定は SD カードに保存されます。

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。

```
[PC ~]$ mount
(省略)
/dev/sdb1 on /media/52E6-5897 type ext2
(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)
[PC ~]$ sudo umount /dev/sdb1
```

図 14.1 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、評価セット付属の 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、最新バージョンを利用することを推奨します。

Armadillo サイト - Armadillo-X1 ドキュメント・ダウンロード

http://armadillo.atmark-techno.com/armadillo-x1/downloads

14.1. 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 14.1 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	u-boot-x1-sd-*[version]*.bin

「表 14.2. 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 14.1. ブートディスクの作成例」に示します。

表 14.2 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	ext4	ルートファイルシステムを構築するために ext4 ファイルシステムを構築しておきます。

手順 14.1 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。

```
[PC ~]$ ls
u-boot-x1-sd-[version].bin
```

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	u-boot-x1-sd-[version].bin
フラッシュメモリ	u-boot-x1-[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。

```
[PC ~]$ sudo fdisk /dev/sdb ❶

Welcome to fdisk (util-linux 2.25.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help): o ❷
Created a new DOS disklabel with disk identifier 0x2b685734.

Command (m for help): n ❸
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): ❹

Using default response p.
Partition number (1-4, default 1): ❺
First sector (2048-7761919, default 2048): ❻
Last sector, +sectors or +size{K,M,G,T,P} (2048-7761919, default 7761919): +128M ❼

Created a new partition 1 of type 'Linux' and of size 128 MiB.

Command (m for help): n ❽
Partition type
```

```
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p):  ❾

Using default response p.
Partition number (2-4, default 2):  ❿
First sector (264192-7761919, default 264192):  ⓫
Last sector, +sectors or +size{K,M,G,T,P} (264192-7761919, default 7761919):  ⓬

Created a new partition 2 of type 'Linux' and of size 3.6 GiB.

Command (m for help): t  ⓭
Partition number (1,2, default 2): 1  ⓮
Hex code (type L to list all codes): b  ⓯

If you have created or modified any DOS 6.x partitions, please see the fdisk
documentation for additional information.
Changed type of partition 'Linux' to 'W95 FAT32'.

Command (m for help): w  ⓰
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

[PC ~]$
```

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。

```
[PC ~]$ sudo fdisk -l /dev/sdb

Disk /dev/sdb: 3.7 GiB, 3974103040 bytes, 7761920 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
Disklabel type: dos
Disk identifier: 0x2b685734

Device     Boot  Start     End Sectors  Size Id Type
/dev/sdb1         2048  264191  262144  128M  b W95 FAT32
/dev/sdb2       264192 7761919 7497728  3.6G 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。

```
[PC ~]$ sudo mkfs.vfat -F 32 /dev/sdb1  ❶
mkfs.fat 3.0.27 (2014-11-12)
[PC ~]$ sudo mkfs.ext4 -L rootfs /dev/sdb2  ❷
mke2fs 1.42.12 (29-Aug-2014)
Creating filesystem with 937216 4k blocks and 234320 inodes
Filesystem UUID: AAAAAAAA-BBBB-CCCC-DDDD-EEEEEEEEEEEE
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736

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

[PC ~]$
```

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 ext4 ファイルシステムを構築します。ボリュームラベルには "rootfs"を設定します。

5. SD ブート用のブートローダーイメージファイルを SD カードに書き込みます。

```
[PC ~]$ ls
u-boot-x1-sd-[version].bin
[PC ~]$ sudo dd if=u-boot-x1-sd-[version].bin of=/dev/sdb bs=1k seek=1
[PC ~]$ sync
```

14.2. ルートファイルシステムの構築

「14.1. 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。

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 14.3 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
Debian GNU/Linux	debian-jessie-armhf_x1_*[version]*.tar.gz	ARM(armhf)アーキテクチャ用 Debian GNU/Linux 8(コードネーム jessie)のルートファイルシステムアーカイブ

14.2.1. Debian GNU/Linux のルートファイルシステムを構築する

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 14.2 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。

```
[PC ~]$ ls
debian-jessie-armhf_x1_[version].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。

```
[PC ~]$ mkdir sd ❶
[PC ~]$ sudo mount -t ext4 /dev/sdb2 sd ❷
[PC ~]$ sudo tar zxf debian-jessie-armhf_x1_[version].tar.gz -C sd ❸
[PC ~]$ sudo umount sd ❹
[PC ~]$ rmdir 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。

14.3. Linux カーネルイメージと DTB の配置

「14.1. ブートディスクの作成」で作成したブートディスクに Linux カーネルイメージおよび DTB(Device Tree Blob)を配置します。使用するファイルを次に示します。以降、DTB(Device Tree Blob)を DTB と表記します。

表 14.4 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	uImage-x1-*[version]*
DTB	armadillo_x1-*[version]*.dtb

SD カードに Linux カーネルイメージおよび DTB 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 カーネルイメージまたは DTB を検出することができなくなる場合があります。

表 14.5 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	FAT32
圧縮形式	非圧縮
Linux カーネルイメージファイル名	uImage
DTB ファイル名	armadillo_x1.dtb

Linux カーネルイメージおよび DTB をブートディスクに配置する手順を次に示します。

手順 14.3 Linux カーネルイメージおよび DTB の配置

1. Linux カーネルイメージおよび DTB を準備しておきます。

```
[PC ~]$ ls
uImage-x1-[version]  armadillo_x1-[version].dtb
```

2. Linux カーネルイメージをブートディスクの第 1 パーティションに配置します。

```
[PC ~]$ mkdir sd ❶
[PC ~]$ sudo mount -t vfat /dev/sdb1 sd ❷
[PC ~]$ sudo cp uImage-x1-[version] sd/uImage ❸
[PC ~]$ sudo cp armadillo_x1-[version].dtb sd/armadillo_x1.dtb ❹
[PC ~]$ sudo umount sd ❺
[PC ~]$ rmdir 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 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 Linux カーネルイメージを sd/ディレクトリにコピーします。

❹ DTB を sd/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 1 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。

14.4. SD ブートの実行

「14.1. 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. ブートディスクを接続した SD スロット拡張ボードを、CON7 アドオンインターフェースに接続します。
2. 「19.1.3. インターフェース仕様」を参照して SD スロット拡張ボードの SW1 を「SD BOOT」に設定します。

準備が完了後、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、saveenv を実行すると「図 14.2. SD ブート時の saveenv メッセージ」のようにメッセージが表示されます。環境変数の保存先が"MMC"になっていることを確認してください。

```
=> saveenv
Saving Environment to MMC...
Writing to MMC(0)... done
=>
```

図 14.2 SD ブート時の saveenv メッセージ

SD カードのライトプロテクションスイッチは無効にしてください。SD カードに書き込みが出来ない場合、SD ブートを正常に行うことができません。

15. 電気的仕様

15.1. 絶対最大定格

表 15.1 絶対最大定格

項目	記号	Min.	Max.	単位	備考
電源電圧	VIN	-0.3	5.3	V	
入出力電圧(USB 信号以外)	VI,VO	-0.3	OVDD+0.3	V	OVDD=VCC_3.3V, NVCC_SD1, NVCC_SD2
入力電圧(USB 信号)	VI_USB	-0.3	3.63	V	USBx_DP, USBx_DM
入力電圧(USB_HUB 信号)	VI_USB_HUB	-0.3	5.5	V	USB_HUBx_DP, USB_HUBx_DM
入力電圧(USB_VBUS 信号)	VI_VBUS	-0.3	5.25	V	USB2_VBUS_IN
RTC バックアップ電源電圧	RTC_BAT	-0.3	3.8	V	
RF 入力(CH0, CH1)	RFin		10	dBm	
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、あらゆる使用条件や試験状況において、瞬時でも超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。

15.2. 推奨動作条件

表 15.2 推奨動作条件

項目	記号	Min.	Typ.	Max.	単位	備考
電源電圧	VIN	4.75	5	5.25	V	
RTC バックアップ電源電圧	RTC_BAT	2.4	3	3.6	V	
使用周囲温度	Ta	-20	25	70	℃	ただし結露なきこと

15.3. 入出力インターフェースの電気的仕様

表 15.3 入出力インターフェース電源の電気的仕様

項目	記号	Min.	Typ.	Max.	単位	備考
5V 電源電圧	VCC_5V USB1_VBUS USB_HUB2_VBUS	4.75	5	5.25	V	
3.3V 電源電圧	VCC_3.3V VCC_3.3V_IO	3.135	3.3	3.465	V	
SD1 信号電源電圧	NVCC_SD1	1.757	1.85	1.943	V	IOUT_MAX=50mA
		3.135	3.3	3.465		
SD2 信号電源電圧	NVCC_SD2	1.71	1.8	1.89	V	IOUT_MAX=50mA
		3.135	3.3	3.465		

表 15.4 入出力インターフェースの電気的仕様(OVDD = VCC_3.3V, NVCC_SD1, NVCC_SD2)

項目	記号	Min.	Max.	単位	備考
ハイレベル出力電圧	VOH	0.8×OVDD	OVDD	V	IOH = -1.8mA, -3.6mA, -7.2mA, -10.8mA
ローレベル出力電圧	VOL	0	0.2×OVDD	V	IOL = 1.8mA, 3.6mA, 7.2mA, 10.8mA
ハイレベル入力電圧	VIH	0.7×OVDD	OVDD+0.3	V	
ローレベル入力電圧	VIL	-0.3	0.3×OVDD	V	
ローレベル入力電圧(PMIC_ONOFF 信号)	VIL	-0.3	0.57	V	
ローレベル入力電圧(EXT_RESET_B 信号)	VIL	-0.3	0.57	V	
入力リーク電流 (no Pull-up/Pull-down)	IOZ	-5	5	μA	
Pull-up 抵抗 (5kΩ)	-	4.8	5.3	kΩ	
Pull-up 抵抗 (47kΩ)	-	45.8	49.8	kΩ	
Pull-up 抵抗 (100kΩ)	-	101	105	kΩ	
Pull-down 抵抗 (100kΩ)	-	101	108	kΩ	

15.4. 電源回路の構成

Armadillo-X1 の電源回路の構成は次のとおりです。電源入力インターフェース 1(CON10)または電源入力インターフェース 2(CON13)からの入力電圧を電源 IC で各電圧に変換し、内部回路および各インターフェースに供給しています。 各インターフェースやスイッチングレギュレータ(DC-DC)の最大出力電流値を超えないように、外部機器の接続、供給電源の設計を行ってください。

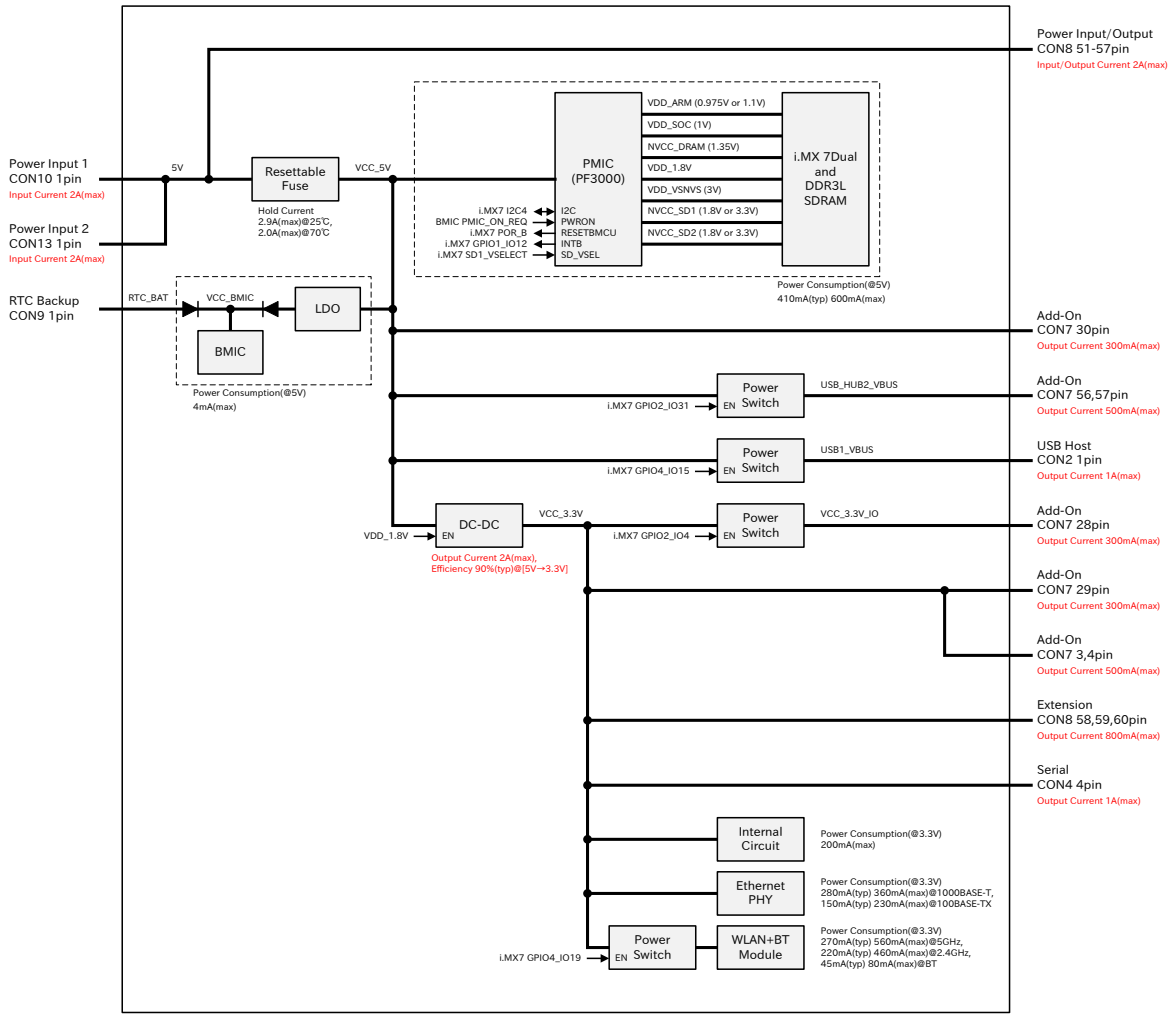


図 15.1 電源回路の構成

16. インターフェース仕様

Armadillo-X1 のインターフェース仕様について説明します。

16.1. インターフェースレイアウト

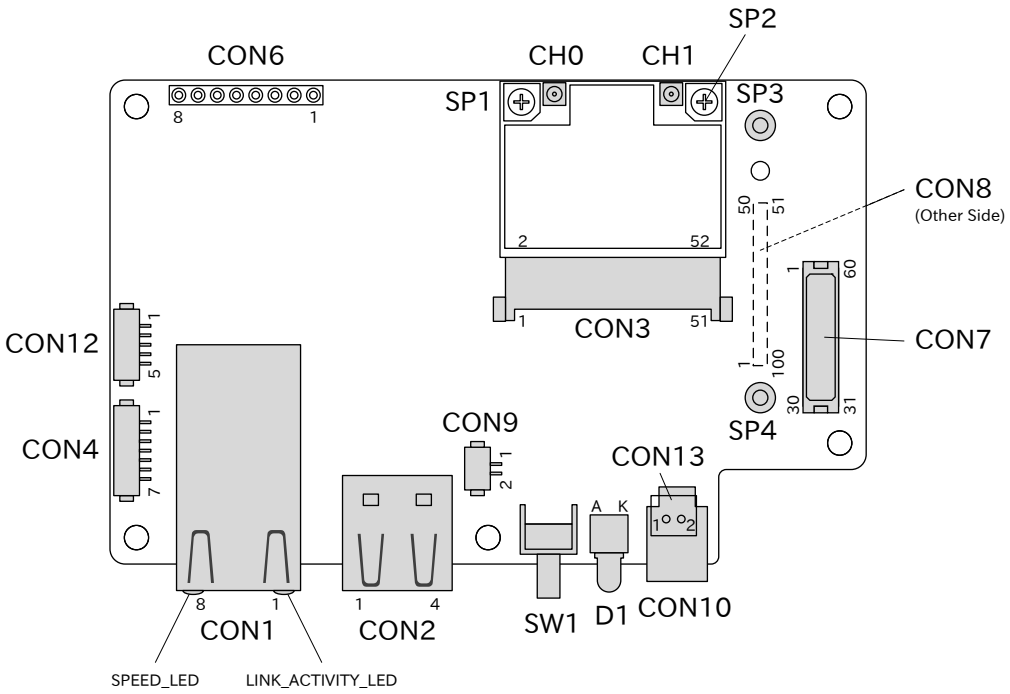


図 16.1 Armadillo-X1 インターフェースレイアウト

表 16.1 Armadillo-X1 インターフェース一覧

部品番号	インターフェース名	型番	メーカー
CON1	LAN インターフェース	9771-8813-S3L6T1	SUN JUN ELECTRONICS
CON2	USB ホストインターフェース	UBA-4R-D14T-4D(LF)(SN)	J.S.T. Mfg.
CON3	WLAN インターフェース	MM60-52B1-E1-R650	Japan Aviation Electronics Industry
CON4	シリアルインターフェース	DF13C-7P-1.25V(51)	HIROSE ELECTRIC
CON6	JTAG インターフェース	A2-8PA-2.54DSA(71)[a]	HIROSE ELECTRIC[a]
CON7	アドオンインターフェース	DF17(4.0)-60DS-0.5V(57)	HIROSE ELECTRIC
CON8	拡張インターフェース	DF40C-100DP-0.4V(51)	HIROSE ELECTRIC
CON9	RTC バックアップインターフェース	DF13C-2P-1.25V(21)	HIROSE ELECTRIC
CON10	電源入力インターフェース 1	HEC3600-016110	HOSIDEN
CON12	-	DF13C-5P-1.25V(51)	HIROSE ELECTRIC
CON13	電源入力インターフェース 2	B02B-PASK(LF)(SN)[a]	J.S.T. Mfg.[a]

部品番号	インターフェース名	型番	メーカー
CH0	WLAN+BT アンテナインターフェース	U.FL-R-SMT-1 同等品	
CH1	WLAN アンテナインターフェース	U.FL-R-SMT-1 同等品	
SW1	ユーザースイッチ	SKHHLRA010	ALPS ELECTRIC
D1	ユーザー LED	SLR-342MC3F	ROHM Semiconductor
SP1	WLAN モジュール用スタッド	NT4R1600	Japan Aviation Electronics Industry
SP2			
SP3	アドオンモジュール用スタッド	KRB-2008	Hirosugi-Keiki
SP4			

[a]コネクタは非搭載です。実装可能なコネクタ型番を記載しています。

「表 16.1. Armadillo-X1 インターフェース一覧」に記載した部品型番は、必ずしも搭載されていることを保証しているわけではありません。お手元の製品の搭載部品は、アットマークテクノ ユーザーズサイトからダウンロード可能な、納入仕様書および変更履歴表にてご確認ください。

16.2. CON1 LAN インターフェース

CON1 は 10BASE-T/100BASE-TX/1000BASE-T に対応した LAN インターフェースです。カテゴリ 5e 以上のイーサネットケーブルを接続することができます。 AUTO-MDIX 機能を搭載しており、ストレートケーブルまたはクロスケーブルを自動認識して送受信端子を切り替えます。

信号は Ethernet PHY(VSC8501XML-03/Microsemi) を経由して、i.MX 7Dual の Ethernet MAC(ENET2)に接続されています。

搭載コネクタ　　9771-8813-S3L6T1/SUN JUN ELECTRONICS

表 16.2 CON1 信号配列 (10BASE-T/100BASE-TX)

ピン番号	ピン名	I/O	説明
1	TX+	In/Out	送信データ+
2	TX-	In/Out	送信データ-
3	RX+	In/Out	受信データ+
4	-	-	
5	-	-	
6	RX-	In/Out	受信データ-
7	-	-	
8	-	-	

表 16.3 CON1 信号配列 (1000BASE-T)

ピン番号	ピン名	I/O	説明
1	TRD0+	In/Out	送受信データ 0+
2	TRD0-	In/Out	送受信データ 0-
3	TRD1+	In/Out	送受信データ 1+
4	TRD2+	In/Out	送受信データ 2+
5	TRD2-	In/Out	送受信データ 2-
6	TRD1-	In/Out	送受信データ 1-
7	TRD3+	In/Out	送受信データ 3+
8	TRD3-	In/Out	送受信データ 3-

表 16.4 LAN コネクタ LED

名称	状態	説明
LINK_ACTIVITY_LED	消灯	リンクが確立されていない
	点灯(黄色)	リンクが確立されている
	点滅(黄色)	リンクが確立されており、データを送受信している
SPEED_LED	消灯	10Mbps で接続されている
	点灯(緑色)	100Mbps で接続されている
	点灯(橙色)	1000Mbps で接続されている

16.3. CON2 USB ホストインターフェース

CON2 は USB2.0 ホストインターフェースです。信号は i.MX 7Dual の USB コントローラ(OTG1)に接続されています。

USB デバイスに供給される電源(USB1_VBUS)は、i.MX 7Dual の I2C4_SDA(GPIO4_IO15)ピンで制御が可能です。GPIO モードに設定後、High レベル出力で電源が供給され、Low レベル出力で電源が切断されます。

データ転送モード

- High Speed(480Mbps)
- Full Speed(12Mbps)
- Low Speed(1.5Mbps)

搭載コネクタ　　UBA-4R-D14T-4D/J.S.T. Mfg.

表 16.5 CON2 信号配列

ピン番号	ピン名	I/O	説明
1	USB1_VBUS	Power	USB 電源出力(USB1_VBUS)
2	USB1_DM	In/Out	USB マイナス側信号、i.MX 7Dual の USB_OTG1_DN ピンに接続
3	USB1_DP	In/Out	USB プラス側信号、i.MX 7Dual の USB_OTG1_DP ピンに接続
4	GND	Power	電源(GND)

16.4. CON3 WLAN インターフェース

CON3 は WLAN+BT コンボモジュール(AEH-AR9462/VoxMicro)用インターフェースです。

PCI Express 信号は i.MX 7Dual の PCI Express PHY(PCIe_PHY)に、USB 信号は USB HUB コントローラ(ポート 1)経由で i.MX 7Dual の USB HSIC コントローラに接続されています。

WLAN+BT コンボモジュールに供給される電源(WLAN_VDD)は、i.MX 7Dual の ECSPI1_SS0(GPIO4_IO19)ピンで制御が可能です。GPIO モードに設定後、High レベル出力で電源が供給され、Low レベル出力で電源が切断されます。

搭載コネクタ　　MM60-52B1-E1-R650/Japan Aviation Electronics Industry

表 16.6 CON3 信号配列

ピン番号	ピン名	I/O	説明
1	-	-	Reserved、i.MX 7Dual の SAI1_TXFS(GPIO6_IO14)ピンに接続、基板上で 10kΩ プルアップ(VCC_3.3V)されています
2	WLAN_VDD	Power	WLAN 電源出力(WLAN_VDD)
3	-	-	Reserved、i.MX 7Dual の SAI1_TXD(GPIO6_IO15)ピンに接続
4	GND	Power	電源(GND)

ピン番号	ピン名	I/O	説明
5	BT_DISABLE_L	Out	BT の有効/無効信号、i.MX 7Dual の EPDC1_DATA11(GPIO2_IO11)ピンに接続 (Low: BT 無効、High: BT 有効)
6	-	-	Reserved
7	CLKREQ_L	In	リファレンスクロックリクエスト、i.MX 7Dual の EPDC1_DATA12(GPIO2_IO12)ピンに接続、基板上で 10kΩ プルアップ(VCC_3.3V)されています
8	NC	-	未接続
9	GND	Power	電源(GND)
10	NC	-	未接続
11	REFCLK-	Out	差動リファレンスクロック(−)、i.MX 7Dual の PCIE_REFCLKOUT_N ピンに接続
12	NC	-	未接続
13	REFCLK+	Out	差動リファレンスクロック(+)、i.MX 7Dual の PCIE_REFCLKOUT_P ピンに接続
14	NC	-	未接続
15	GND	Power	電源(GND)
16	NC	-	未接続
17	NC	-	未接続
18	GND	Power	電源(GND)
19	NC	-	未接続
20	W_DISABLE_L	Out	WLAN の有効/無効信号、i.MX 7Dual の SAI1_TXC(GPIO2_IO13)ピンに接続 (Low: WLAN 無効、High: WLAN 有効)
21	GND	Power	電源(GND)
22	PERST_L	Out	基本リセット信号、i.MX 7Dual の SAI1_RXD(GPIO2_IO12)ピンに接続 (Low: リセット状態、High: リセット解除)
23	PERn0	In	差動レシーバ(−)、i.MX 7Dual の PCIE_RX_N ピンに接続
24	WLAN_VDD	Power	WLAN 電源出力(WLAN_VDD)
25	PERp0	In	差動レシーバ(+)、i.MX 7Dual の PCIE_RX_P ピンに接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	-	-	Reserved
29	GND	Power	電源(GND)
30	-	-	Reserved、i.MX 7Dual の EPDC1_DATA09(GPIO2_IO9)ピンに接続
31	PETn0	Out	差動トランスミッタ(−)、i.MX 7Dual の PCIE_TX_N ピンに接続
32	-	-	Reserved、i.MX 7Dual の EPDC1_DATA10(GPIO2_IO10)ピンに接続
33	PETp0	Out	差動トランスミッタ(+)、i.MX 7Dual の PCIE_TX_P ピンに接続
34	GND	Power	電源(GND)
35	GND	Power	電源(GND)
36	USB_HUB1_DM	In/Out	USB マイナス側信号、USB HUB コントローラ(ポート 1)経由で i.MX 7Dual の USB HSIC コントローラに接続
37	GND	Power	電源(GND)
38	USB_HUB1_DP	In/Out	USB プラス側信号、USB HUB コントローラ(ポート 1)経由で i.MX 7Dual の USB HSIC コントローラに接続
39	WLAN_VDD	Power	WLAN 電源出力(WLAN_VDD)
40	GND	Power	電源(GND)
41	WLAN_VDD	Power	WLAN 電源出力(WLAN_VDD)
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
47	NC	-	未接続
48	-	-	Reserved
49	NC	-	未接続

ピン番号	ピン名	I/O	説明
50	GND	Power	電源(GND)
51	NC	-	未接続
52	WLAN_VDD	Power	WLAN 電源出力(WLAN_VDD)

16.5. CON4 シリアルインターフェース

CON4 は非同期(調歩同期)シリアルインターフェースです。信号は i.MX 7Dual の UART コントローラ(UART5)に接続されています。

CON4 の 6 ピンは i.MX 7Dual の GPIO1_09 ピンに接続されており、Low レベル入力で保守モード、High レベル入力で OS 自動起動モードで起動します。

搭載コネクタ　DF13C-7P-1.25V(51)/HIROSE ELECTRIC

対向コネクタ例　DF13-7S-1.25C/HIROSE ELECTRIC(ハウジング)

DF13-2630SCFA/HIROSE ELECTRIC(コンタクト)

信号レベル　3.3V CMOS

許容電流　1A 以下(端子 1 本あたり)

表 16.7 CON4 信号配列

ピン番号	ピン名	I/O	説明	電圧グループ
1	CON4_UART_RXD	In	受信データ、i.MX 7Dual の GPIO1_IO06 ピンに接続	VCC_3.3V
2	GND	Power	電源(GND)	-
3	CON4_UART_TXD	Out	送信データ、i.MX 7Dual の GPIO1_IO07 ピンに接続	VCC_3.3V
4	VCC_3.3V	Power	電源出力(VCC_3.3V)	-
5	CON4_UART_CTS	In	送信可能、i.MX 7Dual の GPIO1_IO05 ピンに接続	VCC_3.3V
6	BOOTLOADER_EN_B	In	起動モード設定、i.MX 7Dual の GPIO1_IO09 ピンに接続、基板上で 10kΩ プルアップ(VCC_3.3V)されています (Low: 保守モード、High: OS 自動起動モード)	VCC_3.3V
7	CON4_UART_RTS	Out	送信要求、i.MX 7Dual の GPIO1_IO04 ピンに接続	VCC_3.3V

16.6. CON6 JTAG インターフェース

CON6 は ARM JTAG デバッガを接続することができる JTAG インターフェースです。信号は i.MX 7Dual のシステム JTAG コントローラ(SJC)に接続されています。コネクタを実装してご使用ください。

搭載可能コネクタ　ピンヘッダ 8 ピン(2.54mm ピッチ) : A2-8PA-2.54DSA(71)/HIROSE ELECTRIC 等

表 16.8 CON6 信号配列

ピン番号	ピン名	I/O	説明	電圧グループ
1	VCC_3.3V	Power	電源出力(VCC_3.3V)	-
2	JTAG_TRST_B	In	テストリセット、i.MX 7Dual の JTAG_TRST_B ピンに接続、基板上で 10kΩ プルアップ(VCC_3.3V)されています	VCC_3.3V
3	JTAG_TDI	In	テストデータ入力、i.MX 7Dual の JTAG_TDI ピンに接続、基板上で 10kΩ プルアップ(VCC_3.3V)されています	VCC_3.3V
4	JTAG_TMS	In	テストモード選択、i.MX 7Dual の JTAG_TMS ピンに接続、基板上で 10kΩ プルアップ(VCC_3.3V)されています	VCC_3.3V

ピン番号	ピン名	I/O	説明	電圧グループ
5	JTAG_TCK	In	テストクロック、i.MX 7Dual の JTAG_TCK ピンに接続、基板上で 10kΩ プルアップ(VCC_3.3V)されています	VCC_3.3V
6	JTAG_TDO	Out	テストデータ出力、i.MX 7Dual の JTAG_TDO ピンに接続	VCC_3.3V
7	JTAG_SRST_B	In	システムリセット、基板上で 1kΩ プルアップ(VCC_3.3V)されています (Low: リセット状態、High: リセット解除)	VCC_3.3V
8	GND	Power	電源(GND)	-

オプション品の「8 ピン JTAG 変換ケーブル[1]」(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。

16.7. CON7 アドオンインターフェース

CON7 は機能拡張用のインターフェースです。複数の機能(マルチプレクス)をもった i.MX 7Dual の信号が接続されており、USB、SD、UART、SPI、I2C、CAN、AUDIO、GPIO の機能を拡張することができます。

電源(VCC_3.3V_IO)は、i.MX 7Dual の EPDC1_DATA04(GPIO2_IO4)ピンで制御が可能です。GPIO モードに設定後、High レベル出力で電源が供給され、Low レベル出力で電源が切断されます。

USB 電源(USB_HUB2_VBUS)は、i.MX 7Dual の EPDC1_PWRSTAT(GPIO2_IO31)ピンで制御が可能です。GPIO モードに設定後、High レベル出力で電源が供給され、Low レベル出力で電源が切断されます。

搭載コネクタ　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

許容電流　0.3A 以下(端子 1 本あたり)

表 16.9 CON7 信号配列

ピン番号	ピン名	I/O	説明	電圧グループ
1	GND	Power	電源(GND)	-
2	GND	Power	電源(GND)	-
3	VCC_3.3V	Power	電源出力(VCC_3.3V)	-
4	VCC_3.3V	Power	電源出力(VCC_3.3V)	-
5	SDBOOT_EN	In	起動デバイス設定、i.MX 7Dual の BOOT_MODE0 ピンに接続、基板上で 10kΩ プルダウンされています (Low: SPI フラッシュメモリブート、High: SD ブート)	VCC_3.3V
6	NVCC_SD1	Power	SD1 信号電源出力(NVCC_SD1)	-
7	GPIO5_IO0	In/Out	拡張入出力、i.MX 7Dual の SD1_CD_B ピンに接続	NVCC_SD1
8	GPIO5_IO1	In/Out	拡張入出力、i.MX 7Dual の SD1_WP ピンに接続	NVCC_SD1
9	GPIO5_IO2	In/Out	拡張入出力、i.MX 7Dual の SD1_RESET_B ピンに接続	NVCC_SD1
10	GPIO5_IO3	In/Out	拡張入出力、i.MX 7Dual の SD1_CLK ピンに接続	NVCC_SD1
11	GPIO5_IO4	In/Out	拡張入出力、i.MX 7Dual の SD1_CMD ピンに接続	NVCC_SD1

[1]詳細については、「19.5. 8 ピン JTAG 変換ケーブル」を参照してください。

ピン番号	ピン名	I/O	説明	電圧グループ
12	GPIO5_IO5	In/Out	拡張入出力、i.MX 7Dual の SD1_DATA0 ピンに接続	NVCC_SD1
13	GPIO5_IO6	In/Out	拡張入出力、i.MX 7Dual の SD1_DATA1 ピンに接続	NVCC_SD1
14	GPIO5_IO7	In/Out	拡張入出力、i.MX 7Dual の SD1_DATA2 ピンに接続	NVCC_SD1
15	GPIO5_IO8	In/Out	拡張入出力、i.MX 7Dual の SD1_DATA3 ピンに接続	NVCC_SD1
16	GPIO4_IO16	In/Out	拡張入出力、i.MX 7Dual の ECSPI1_SCLK ピンに接続	VCC_3.3V
17	GPIO4_IO17	In/Out	拡張入出力、i.MX 7Dual の ECSPI1_MOSI ピンに接続	VCC_3.3V
18	GPIO4_IO18	In/Out	拡張入出力、i.MX 7Dual の ECSPI1_MISO ピンに接続	VCC_3.3V
19	NC	-	未接続	-
20	GPIO4_IO12	In/Out	拡張入出力、i.MX 7Dual の I2C3_SCL ピンに接続	VCC_3.3V
21	GPIO4_IO13	In/Out	拡張入出力、i.MX 7Dual の I2C3_SDA ピンに接続	VCC_3.3V
22	NC	-	未接続	-
23	NC	-	未接続	-
24	GPIO4_IO0	In/Out	拡張入出力、i.MX 7Dual の UART1_RXD ピンに接続	VCC_3.3V
25	GPIO4_IO1	In/Out	拡張入出力、i.MX 7Dual の UART1_TXD ピンに接続	VCC_3.3V
26	GND	Power	電源(GND)	-
27	GND	Power	電源(GND)	-
28	VCC_3.3V_IO	Power	電源出力(VCC_3.3V_IO)	-
29	VCC_3.3V	Power	電源出力(VCC_3.3V)	-
30	VCC_5V	Power	電源出力(VCC_5V)	
31	-	Out	Low 固定信号、1kΩ 抵抗で GND に接続されています	-
32	GPIO6_IO18	In/Out	拡張入出力、i.MX 7Dual の SAI1_MCLK ピンに接続	VCC_3.3V
33	GPIO4_IO2	In/Out	拡張入出力、i.MX 7Dual の UART2_RXD ピンに接続	VCC_3.3V
34	GPIO4_IO3	In/Out	拡張入出力、i.MX 7Dual の UART2_TXD ピンに接続	VCC_3.3V
35	GPIO4_IO6	In/Out	拡張入出力、i.MX 7Dual の UART3_RTS ピンに接続	VCC_3.3V
36	GPIO4_IO4	In/Out	拡張入出力、i.MX 7Dual の UART3_RXD ピンに接続	VCC_3.3V
37	GPIO4_IO5	In/Out	拡張入出力、i.MX 7Dual の UART3_TXD ピンに接続	VCC_3.3V
38	GPIO4_IO23	In/Out	拡張入出力、i.MX 7Dual の ECSPI2_SS0 ピンに接続	VCC_3.3V
39	GPIO4_IO22	In/Out	拡張入出力、i.MX 7Dual の ECSPI2_MISO ピンに接続	VCC_3.3V
40	GPIO4_IO21	In/Out	拡張入出力、i.MX 7Dual の ECSPI2_MOSI ピンに接続	VCC_3.3V
41	GPIO4_IO20	In/Out	拡張入出力、i.MX 7Dual の ECSPI2_SCLK ピンに接続	VCC_3.3V
42	GPIO6_IO16	In/Out	拡張入出力、i.MX 7Dual の SAI1_RXFS ピンに接続	VCC_3.3V
43	GPIO6_IO17	In/Out	拡張入出力、i.MX 7Dual の SAI1_RXC ピンに接続	VCC_3.3V
44	NC	-	未接続	-
45	NC	-	未接続	-
46	GPIO6_IO19	In/Out	拡張入出力、i.MX 7Dual の SAI2_TXFS ピンに接続	VCC_3.3V
47	GPIO6_IO20	In/Out	拡張入出力、i.MX 7Dual の SAI2_TXC ピンに接続	VCC_3.3V
48	GPIO6_IO21	In/Out	拡張入出力、i.MX 7Dual の SAI2_RXD ピンに接続	VCC_3.3V
49	GPIO6_IO22	In/Out	拡張入出力、i.MX 7Dual の SAI2_TXD ピンに接続	VCC_3.3V
50	GPIO4_IO7	In/Out	拡張入出力、i.MX 7Dual の UART3_CTS ピンに接続	VCC_3.3V
51	NC	-	未接続	-
52	NC	-	未接続	-
53	NC	-	未接続	-
54	GND	Power	電源(GND)	-
55	PMIC_ONOFF	In	パワーマネジメント IC の ON/OFF 用信号、基板上で 47kΩ プルアップ(VCC_BMIC)されています、オープンドレイン(またはオープンコレクタ)信号を入力してください (2 秒以上 Low: 電源 OFF、OFF 時に 2 秒未満 Low: 電源 ON)	VCC_BMIC
56	USB_HUB2_VBUS	Power	電源出力(USB_HUB2_VBUS)	-
57	USB_HUB2_VBUS	Power	電源出力(USB_HUB2_VBUS)	-
58	GND	Power	電源(GND)	-
59	USB_HUB2_DP	In/Out	USB プラス側信号、USB HUB コントローラ(ポート 2)経由で i.MX 7Dual の USB HSIC コントローラに接続	-
60	USB_HUB2_DM	In/Out	USB マイナス側信号、USB HUB コントローラ(ポート 2)経由で i.MX 7Dual の USB HSIC コントローラに接続	-

「18. アドオンモジュール」で紹介しているアドオンモジュール、Armadillo-X1 開発セットに付属されている SD スロット拡張ボード等を接続することが可能です。

アドオンインターフェースのマルチプレクス表は Armadillo サイトからダウンロードすることが可能ですので、拡張ボード設計の際などにご確認ください。

16.8. CON8 拡張インターフェース

CON8 は機能拡張用のインターフェースです。複数の機能(マルチプレクス)をもった i.MX 7Dual の信号が接続されており、USB、MIPI_DSI、MIPI_CSI、SD、Ethernet、LCD、CAMERA、UART、SPI、I2C、CAN、AUDIO、PWM、GPIO の機能を拡張することができます。

搭載コネクタ　DF40C-100DP-0.4V(51)/HIROSE ELECTRIC

対向コネクタ例　DF40HC(3.0)-100DS-0.4V(51)/HIROSE ELECTRIC

許容電流　0.3A 以下(端子 1 本あたり)

表 16.10 CON8 信号配列

ピン番号	ピン名	I/O	説明	電圧グループ
1	GND	Power	電源(GND)	-
2	MIPI_DSI_D0_P	Out	MIPI_DSI D-PHY 送信データ 0+、i.MX 7Dual の MIPI_DSI_D0_P に接続	-
3	MIPI_DSI_D0_N	Out	MIPI_DSI D-PHY 送信データ 0-、i.MX 7Dual の MIPI_DSI_D0_N に接続	-
4	GND	Power	電源(GND)	-
5	MIPI_DSI_CLK_P	Out	MIPI_DSI D-PHY 送信クロック+、i.MX 7Dual の MIPI_DSI_CLK_P に接続	-
6	MIPI_DSI_CLK_N	Out	MIPI_DSI D-PHY 送信クロック-、i.MX 7Dual の MIPI_DSI_CLK_N に接続	-
7	GND	Power	電源(GND)	-
8	MIPI_DSI_D1_P	Out	MIPI_DSI D-PHY 送信データ 1+、i.MX 7Dual の MIPI_DSI_D1_P に接続	-
9	MIPI_DSI_D1_N	Out	MIPI_DSI D-PHY 送信データ 1-、i.MX 7Dual の MIPI_DSI_D1_N に接続	-
10	GND	Power	電源(GND)	-
11	MIPI_CSI_D0_P	In	MIPI_CSI D-PHY 受信データ 0+、i.MX 7Dual の MIPI_CSI_D0_P に接続	-
12	MIPI_CSI_D0_N	In	MIPI_CSI D-PHY 受信データ 0-、i.MX 7Dual の MIPI_CSI_D0_N に接続	-
13	GND	Power	電源(GND)	-
14	MIPI_CSI_CLK_P	In	MIPI_CSI D-PHY 受信クロック+、i.MX 7Dual の MIPI_CSI_CLK_P に接続	-
15	MIPI_CSI_CLK_N	In	MIPI_CSI D-PHY 受信クロック-、i.MX 7Dual の MIPI_CSI_CLK_N に接続	-

ピン番号	ピン名	I/O	説明	電圧グループ
16	GND	Power	電源(GND)	-
17	MIPI_CSI_D1_P	In	MIPI_CSI D-PHY 受信データ 1+、i.MX 7Dual の MIPI_CSI_D1_P に接続	-
18	MIPI_DSI_D1_N	In	MIPI_CSI D-PHY 受信データ 1-、i.MX 7Dual の MIPI_CSI_D1_N に接続	-
19	GND	Power	電源(GND)	-
20	GPIO4_IO8	In/Out	拡張入出力、i.MX 7Dual の I2C1_SCL ピンに接続	VCC_3.3V
21	GPIO4_IO9	In/Out	拡張入出力、i.MX 7Dual の I2C1_SDA ピンに接続	VCC_3.3V
22	GPIO4_IO10	In/Out	拡張入出力、i.MX 7Dual の I2C2_SCL ピンに接続	VCC_3.3V
23	GPIO4_IO11	In/Out	拡張入出力、i.MX 7Dual の I2C2_SDA ピンに接続	VCC_3.3V
24	GPIO7_IO0	In/Out	拡張入出力、i.MX 7Dual の ENET1_RD0 ピンに接続	VCC_3.3V
25	GPIO7_IO1	In/Out	拡張入出力、i.MX 7Dual の ENET1_RD1 ピンに接続	VCC_3.3V
26	GPIO7_IO2	In/Out	拡張入出力、i.MX 7Dual の ENET1_RD2 ピンに接続	VCC_3.3V
27	GPIO7_IO3	In/Out	拡張入出力、i.MX 7Dual の ENET1_RD3 ピンに接続	VCC_3.3V
28	GPIO7_IO4	In/Out	拡張入出力、i.MX 7Dual の ENET1_RX_CTL ピンに接続	VCC_3.3V
29	GPIO7_IO5	In/Out	拡張入出力、i.MX 7Dual の ENET1_RXC ピンに接続	VCC_3.3V
30	GPIO7_IO6	In/Out	拡張入出力、i.MX 7Dual の ENET1_TD0 ピンに接続	VCC_3.3V
31	GPIO7_IO7	In/Out	拡張入出力、i.MX 7Dual の ENET1_TD1 ピンに接続	VCC_3.3V
32	GPIO7_IO8	In/Out	拡張入出力、i.MX 7Dual の ENET1_TD2 ピンに接続	VCC_3.3V
33	GPIO7_IO9	In/Out	拡張入出力、i.MX 7Dual の ENET1_TD3 ピンに接続	VCC_3.3V
34	GPIO7_IO10	In/Out	拡張入出力、i.MX 7Dual の ENET1_TX_CTL ピンに接続	VCC_3.3V
35	GPIO7_IO11	In/Out	拡張入出力、i.MX 7Dual の ENET1_TXC ピンに接続	VCC_3.3V
36	GPIO7_IO12	In/Out	拡張入出力、i.MX 7Dual の ENET1_TX_CLK ピンに接続	VCC_3.3V
37	GPIO7_IO13	In/Out	拡張入出力、i.MX 7Dual の ENET1_RX_CLK ピンに接続	VCC_3.3V
38	GPIO7_IO14	In/Out	拡張入出力、i.MX 7Dual の ENET1_CRS ピンに接続	VCC_3.3V
39	GPIO7_IO15	In/Out	拡張入出力、i.MX 7Dual の ENET1_COL ピンに接続	VCC_3.3V
40	GPIO1_IO1	In/Out	拡張入出力、i.MX 7Dual の GPIO1_IO01 ピンに接続	VCC_3.3V
41	EXT_RESET_B	In	外部リセット信号、基板上で 47kΩ プルアップ(VDD_VSNVS)されています、オープンドレイン(またはオープンコレクタ)信号を入力してください (Low: リセット状態、High: リセット解除)	VDD_VSNVS
42	PMIC_ONOFF	In	パワーマネジメント IC の ON/OFF 用信号、基板上で 47kΩ プルアップ(VCC_BMIC)されています、オープンドレイン(またはオープンコレクタ)信号を入力してください (2 秒以上 Low: 電源 OFF、OFF 時に 2 秒未満 Low: 電源 ON)	VCC_BMIC
43	GND	Power	電源(GND)	-
44	USB_HUB3_DP	In/Out	USB プラス側信号、USB HUB コントローラ(ポート 3)経由で i.MX 7Dual の USB HSIC コントローラに接続	-
45	USB_HUB3_DM	In/Out	USB マイナス側信号、USB HUB コントローラ(ポート 3)経由で i.MX 7Dual の USB HSIC コントローラに接続	-
46	GND	Power	電源(GND)	-
47	USB2_DP	In/Out	USB プラス側信号、i.MX 7Dual の USB_OTG2_DP ピンに接続	-
48	USB2_DM	In/Out	USB マイナス側信号、i.MX 7Dual の USB_OTG2_DN ピンに接続	-
49	GND	Power	電源(GND)	-
50	USB2_VBUS_IN	In	USB2_VBUS 入力検出、i.MX 7Dual の USB_OTG2_VBUS ピンに接続	-
51	VIN	Power	電源入出力(VIN)	-
52	VIN	Power	電源入出力(VIN)	-
53	VIN	Power	電源入出力(VIN)	-
54	VIN	Power	電源入出力(VIN)	-
55	VIN	Power	電源入出力(VIN)	-
56	VIN	Power	電源入出力(VIN)	-
57	VIN	Power	電源入出力(VIN)	-

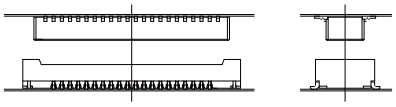
ピン番号	ピン名	I/O	説明	電圧グループ
58	VCC_3.3V	Power	電源出力(VCC_3.3V)	-
59	VCC_3.3V	Power	電源出力(VCC_3.3V)	-
60	VCC_3.3V	Power	電源出力(VCC_3.3V)	-
61	NVCC_SD2	Power	SD2 信号電源出力(NVCC_SD2)	-
62	GPIO5_IO17	In/Out	拡張入出力、i.MX 7Dual の SD2_DATA3 ピンに接続	NVCC_SD2
63	GPIO5_IO16	In/Out	拡張入出力、i.MX 7Dual の SD2_DATA2 ピンに接続	NVCC_SD2
64	GPIO5_IO15	In/Out	拡張入出力、i.MX 7Dual の SD2_DATA1 ピンに接続	NVCC_SD2
65	GPIO5_IO14	In/Out	拡張入出力、i.MX 7Dual の SD2_DATA0 ピンに接続	NVCC_SD2
66	GPIO5_IO13	In/Out	拡張入出力、i.MX 7Dual の SD2_CMD ピンに接続	NVCC_SD2
67	GPIO5_IO12	In/Out	拡張入出力、i.MX 7Dual の SD2_CLK ピンに接続	NVCC_SD2
68	GPIO5_IO11	In/Out	拡張入出力、i.MX 7Dual の SD2_RESET_B ピンに接続	NVCC_SD2
69	GPIO5_IO10	In/Out	拡張入出力、i.MX 7Dual の SD2_WP ピンに接続	NVCC_SD2
70	GPIO5_IO9	In/Out	拡張入出力、i.MX 7Dual の SD2_CD_B ピンに接続	NVCC_SD2
71	GND	Power	電源(GND)	-
72	GPIO3_IO0	In/Out	拡張入出力、i.MX 7Dual の LCD_CLK ピンに接続	VCC_3.3V
73	GPIO3_IO1	In/Out	拡張入出力、i.MX 7Dual の LCD_ENABLE ピンに接続	VCC_3.3V
74	GPIO3_IO2	In/Out	拡張入出力、i.MX 7Dual の LCD_HSYNC ピンに接続	VCC_3.3V
75	GPIO3_IO3	In/Out	拡張入出力、i.MX 7Dual の LCD_VSYNC ピンに接続	VCC_3.3V
76	GPIO3_IO4	In/Out	拡張入出力、i.MX 7Dual の LCD_RESET ピンに接続	VCC_3.3V
77	GPIO3_IO5	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT0 ピンに接続	VCC_3.3V
78	GPIO3_IO6	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT1 ピンに接続	VCC_3.3V
79	GPIO3_IO7	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT2 ピンに接続	VCC_3.3V
80	GPIO3_IO8	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT3 ピンに接続	VCC_3.3V
81	GPIO3_IO9	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT4 ピンに接続	VCC_3.3V
82	GPIO3_IO10	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT5 ピンに接続	VCC_3.3V
83	GPIO3_IO11	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT6 ピンに接続	VCC_3.3V
84	GPIO3_IO12	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT7 ピンに接続	VCC_3.3V
85	GPIO3_IO13	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT8 ピンに接続	VCC_3.3V
86	GPIO3_IO14	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT9 ピンに接続	VCC_3.3V
87	GPIO3_IO15	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT10 ピンに接続	VCC_3.3V
88	GPIO3_IO16	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT11 ピンに接続	VCC_3.3V
89	GPIO3_IO17	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT12 ピンに接続	VCC_3.3V
90	GPIO3_IO18	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT13 ピンに接続	VCC_3.3V
91	GPIO3_IO19	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT14 ピンに接続	VCC_3.3V
92	GPIO3_IO20	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT15 ピンに接続	VCC_3.3V
93	GPIO3_IO21	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT16 ピンに接続	VCC_3.3V
94	GPIO3_IO22	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT17 ピンに接続	VCC_3.3V
95	GPIO3_IO23	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT18 ピンに接続	VCC_3.3V
96	GPIO3_IO24	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT19 ピンに接続	VCC_3.3V
97	GPIO3_IO25	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT20 ピンに接続	VCC_3.3V
98	GPIO3_IO26	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT21 ピンに接続	VCC_3.3V
99	GPIO3_IO27	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT22 ピンに接続	VCC_3.3V
100	GPIO3_IO28	In/Out	拡張入出力、i.MX 7Dual の LCD_DAT23 ピンに接続	VCC_3.3V

拡張インターフェースのマルチプレクス表は Armadillo サイトからダウンロードすることが可能ですので、拡張ボード設計の際などにご確認ください。

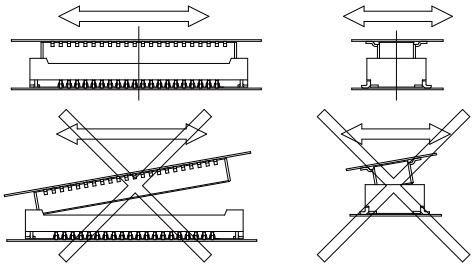
EXT_RESET_B 信号でリセットする場合、確実にリセットさせるために 1 秒以上 Low を入力してください。

コネクタ嵌合時の注意

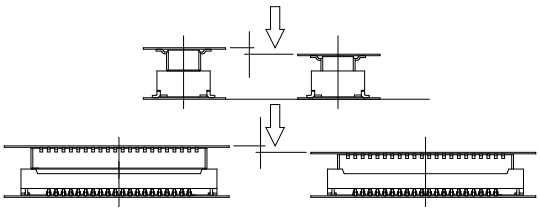
コネクタの中心を合わせて嵌合してください。



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。

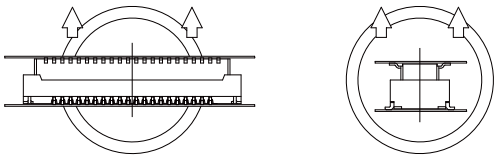


コネクタが誘い込まれると、コネクタ間の距離が近くなり、平行になって前後左右に動かなくなります。この状態からまっすぐに嵌合してください。

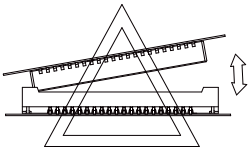


コネクタ抜去時の注意

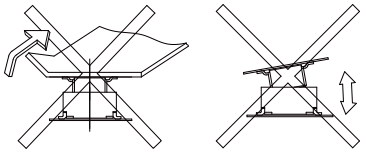
コネクタは平行に抜去してください。



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。



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。



16.9. CON9 RTC バックアップインターフェース

CON9 はリアルタイムクロック機能の外部バックアップインターフェースです。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にご使用ください。

搭載コネクタ	DF13C-2P-1.25V(21)/HIROSE ELECTRIC
対向コネクタ例	DF13-2S-1.25C/HIROSE ELECTRIC(ハウジング)
	DF13-2630SCF/HIROSE ELECTRIC(コンタクト)
許容電流	1A 以下(端子 1 本あたり)
対応バッテリー例	CR2032 WK11/Hitachi Maxell[2]等

表 16.11 CON9 信号配列

ピン番号	ピン名	I/O	説明
1	RTC_BAT	Power	リアルタイムクロックの外部バックアップ用電源入力
2	GND	Power	電源(GND)

リアルタイムクロックの平均月差は周囲温度 25℃で±90 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。

16.10. CON10 電源入力インターフェース 1

CON10 は+5V 電源入力インターフェースです。DC ジャックが実装されています。AC アダプタのジャック形状は JEITA RC-5320A 準拠(電圧区分 2)です。「図 16.2. AC アダプタの極性マーク」と同じ極性マークのある AC アダプタが使用できます。

搭載コネクタ	HEC3600-016110/HOSIDEN

[2]詳しくは、各 Armadillo 販売代理店にお問い合わせください。

図 16.2 AC アダプタの極性マーク

CON13 と共通の信号が接続されています。両方のコネクタから同時に電源供給は行わないでください。

16.11. CON12

CON12 に関する仕様は公開しておりません。

16.12. CON13 電源入力インターフェース 2

CON13 は+5V 電源入力インターフェースです。

搭載コネクタ　　S02B-PASK-2(LF)(SN)/J.S.T. Mfg.

対向コネクタ例　　PAP-02V-S/J.S.T. Mfg.(ハウジング)

SPHD-001T-P0.5/J.S.T. Mfg.(コンタクト)

表 16.12 CON13 信号配列

ピン番号	ピン名	I/O	説明
1	VIN	Power	電源入力(VIN)
2	GND	Power	電源(GND)

CON10 と共通の信号が接続されています。両方のコネクタから同時に電源供給は行わないでください。

16.13. CH0 WLAN+BT アンテナインターフェース

CON17 は WLAN+BT コンボモジュール(AEH-AR9462/VoxMicro)に搭載された WLAN+BT アンテナ用のインターフェースです。

搭載コネクタ　　U.FL-R-SMT-1/HIROSE ELECTRIC 互換品

アンテナ端子にアンテナケーブルを接続する際、無理な力を加えると破損の原因となりますので、十分にご注意ください。

16.14. CH1 WLAN アンテナインターフェース

CON17 は WLAN+BT コンボモジュール(AEH-AR9462/VoxMicro)に搭載された WLAN アンテナ用のインターフェースです。

搭載コネクタ　　U.FL-R-SMT-1/HIROSE ELECTRIC 互換品

アンテナ端子にアンテナケーブルを接続する際、無理な力を加えると破損の原因となりますので、十分にご注意ください。

16.15. SW1 ユーザースイッチ

SW1 はユーザー側で自由に使用できるタクトスイッチです。

搭載スイッチ　　SKHHLRA010/ALPS ELECTRIC

表 16.13 ユーザースイッチの接続

部品番号	説明
SW1	i.MX 7Dual の GPIO1_IO02 ピンおよび BMIC に接続(ON: Low、OFF: High)

16.16. D1 ユーザー LED

D1 はユーザー側で自由に使用できる緑色 LED です。

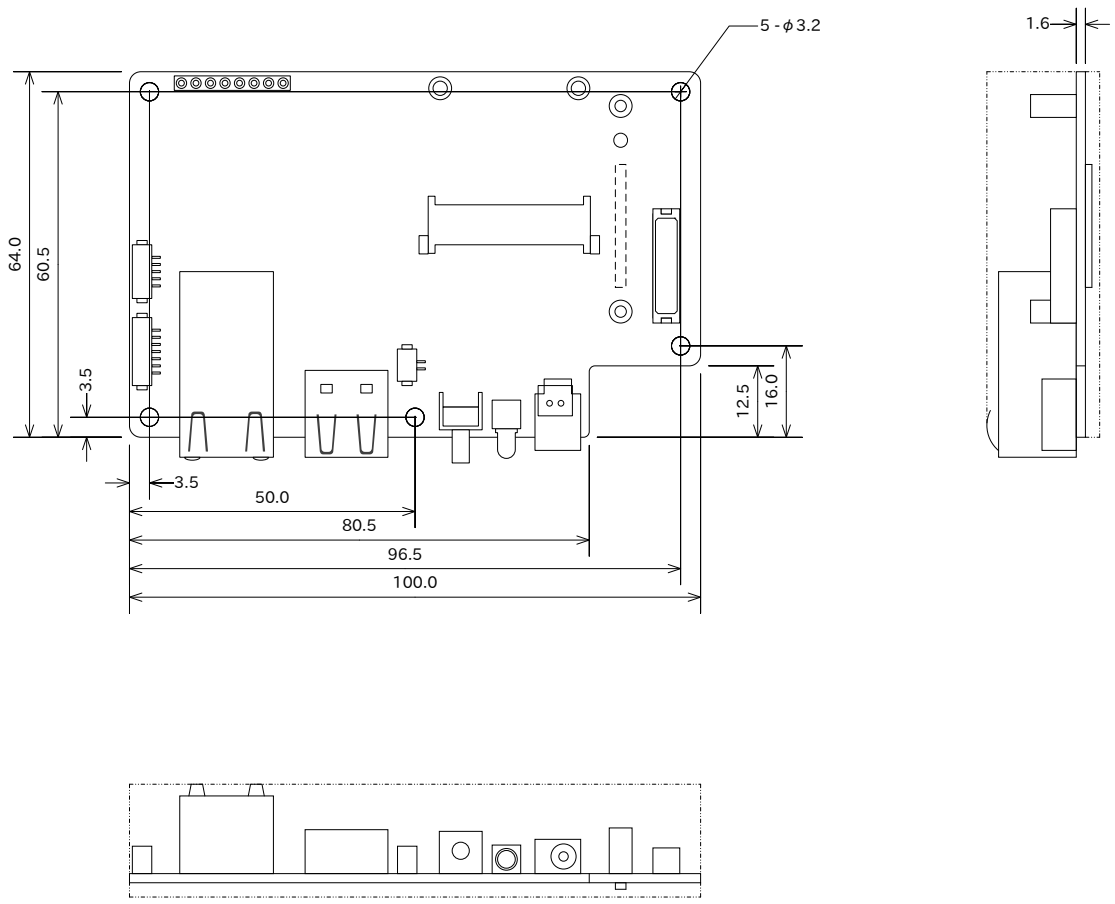
搭載 LED　　SLR-342MC3F/ROHM Semiconductor

搭載 LED スペーサー　　LDZ-805/Hirosugi-Keiki

表 16.14 ユーザー LED の接続

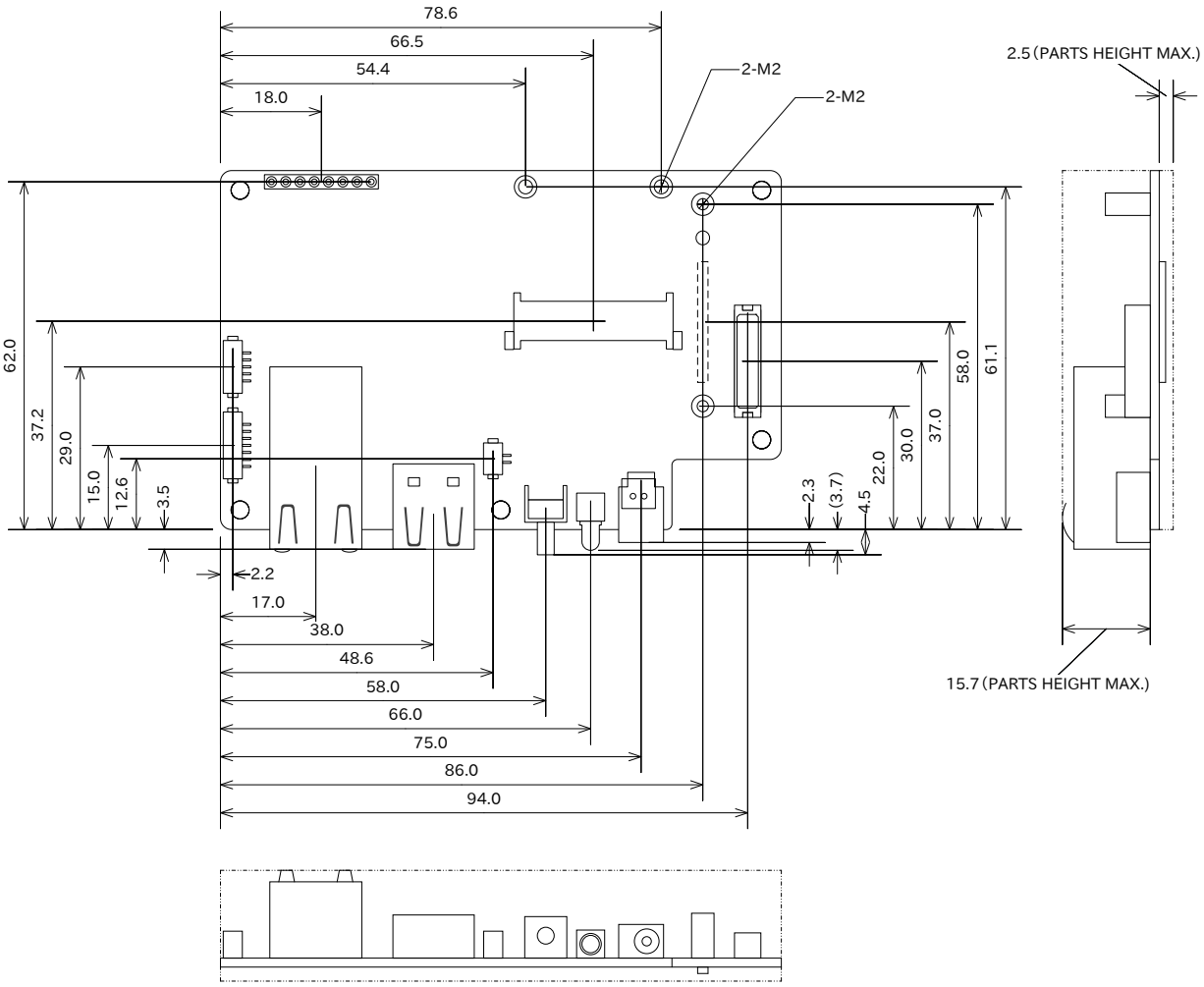
部品番号	説明
D1	i.MX 7Dual の EPDC1_DATA07(GPIO2_IO7)ピンに接続(Low: 消灯、High: 点灯)

17. 基板形状図



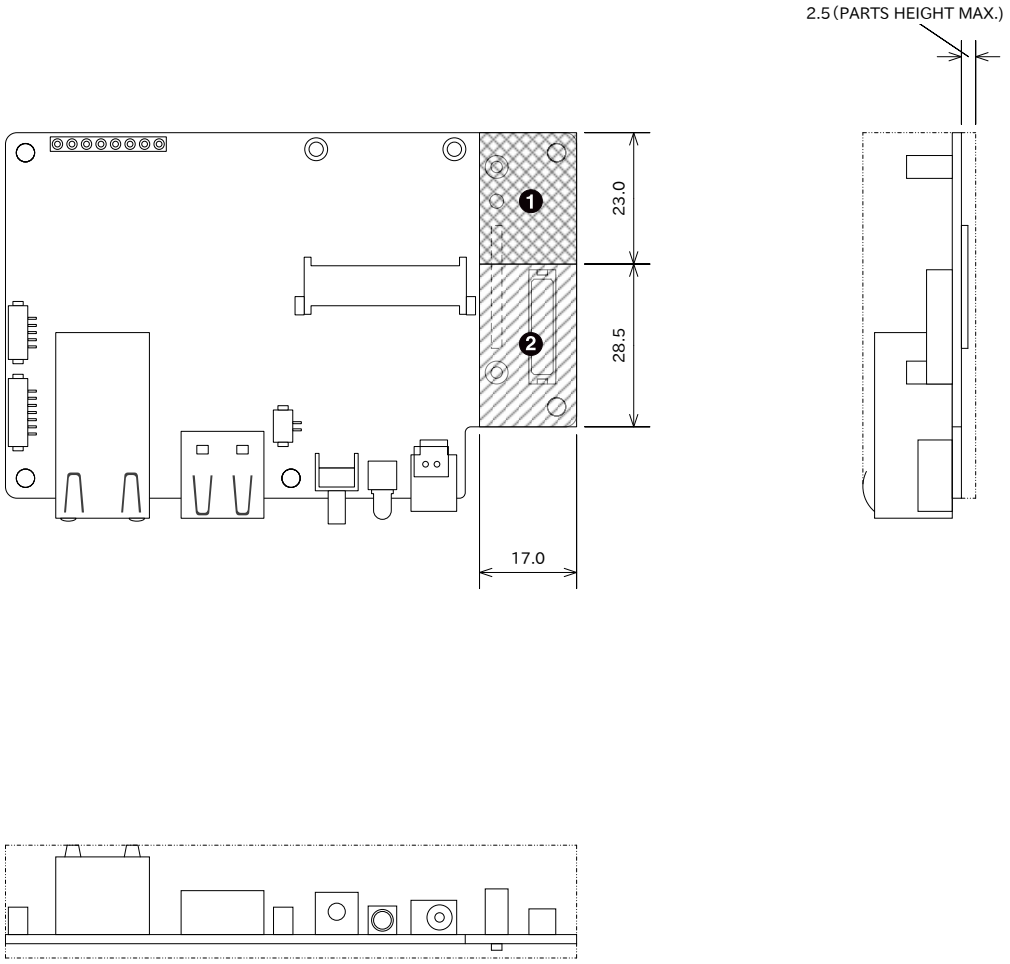
[Unit : mm]

図 17.1 基板形状および固定穴寸法



[Unit : mm]

図 17.2 コネクタ中心寸法



[Unit : mm]

❶ 最大部品高さ 2.0mm (SP3 を除く)

❷ 部品非搭載 (CON7、SP4 を除く)

図 17.3 最大部品高さ

DXF 形式の基板形状図を「アットマークテクノ ユーザーズサイト」から「購入者向けの限定公開データ」としてダウンロード可能です。

18. アドオンモジュール

本章では、Armadillo-IoT のアドオンモジュールについて説明します。

Armadillo-X1 は Armadillo-IoT シリーズでラインアップされている、アドオンモジュールを利用することが可能です。対応しているアドオンモジュールのラインアップは「表 18.1. Armadillo-X1 で利用可能な Armadillo-IoT アドオンモジュール」のとおりです。

表 18.1 Armadillo-X1 で利用可能な Armadillo-IoT アドオンモジュール

名称	型番
Armadillo-IoT RS232C アドオンモジュール RS00	OP-AGA-RS00-00
Armadillo-IoT 絶縁 RS232C/422/485 アドオンモジュール RS01	OP-AGA-RS01-00
Armadillo-IoT 絶縁 RS485 アドオンモジュール RS02	OP-AGA-RS02-00
Armadillo-IoT RN4020 アドオンモジュール BT00	OP-AGA-BT00-00
Armadillo-IoT EnOcean アドオンモジュール EN00[a]	OP-AGA-EN00-00
Armadillo-IoT Wi-SUN アドオンモジュール WS00	OP-AGA-WS00-00
Armadillo-IoT 絶縁デジタル入出力/アナログ入力アドオンモジュール DA00	OP-AGA-DA00-00

[a]発売予定

アドオンモジュールの回路図/部品表、DXF 形式の基板形状図を「アットマークテクノ ユーザーズサイト」から「購入者向けの限定公開データ」としてダウンロード可能です。

18.1. Armadillo-IoT RS232C アドオンモジュール RS00

18.1.1. 概要

Armadillo-IoT RS232C アドオンモジュール RS00(以降、RS232C アドオンモジュールと記載します)は、RS232C レベルのシリアルを 1 ポート追加することができます。また、CON1 は Armadillo-X1 のアドオンインターフェース(CON7)に実装されている 0.5mm ピッチのコネクタを 2.54 ピッチに変換するテストインターフェースを備えています。

RS232C アドオンモジュールの仕様は次のとおりです。

表 18.2 RS232C アドオンモジュールの仕様

シリアル(UART)	Texas Instruments 製 MAX3243E 搭載 最大データ転送レート: 250kbps フロー制御ピンあり(CTS、RTS、DTR、DSR、DCD、RI)
電源電圧	DC 3.3V±5%
使用温度範囲	-20℃～70℃
基板サイズ	40 x 60mm(突起部を除く)

18.1.2. ブロック図

RS232C アドオンモジュールのブロック図は次のとおりです。

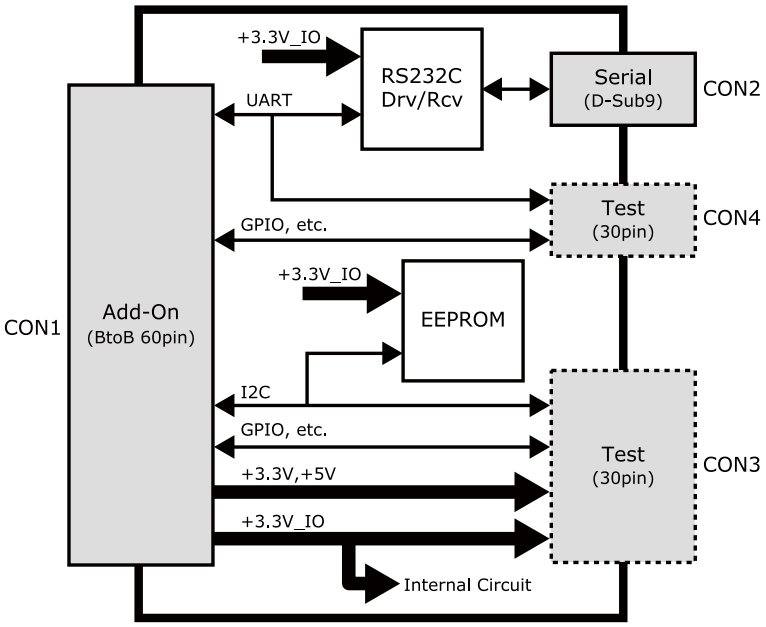


図 18.1 RS232C アドオンモジュール ブロック図

18.1.3. インターフェース仕様

RS232C アドオンモジュールのインターフェース仕様について説明します。

18.1.3.1. RS232C アドオンモジュール インターフェースレイアウト

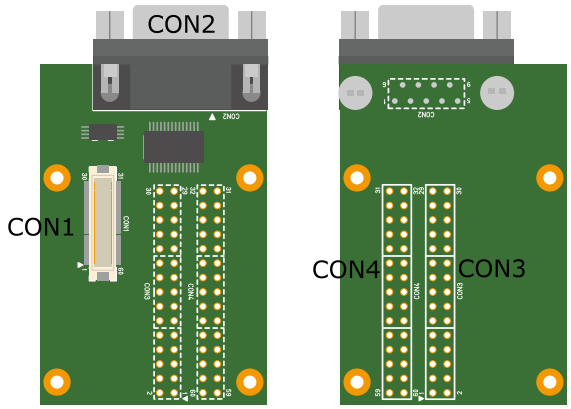


図 18.2 RS232C アドオンモジュール インターフェースレイアウト

表 18.3 搭載コネクタ、スイッチ型番一覧[a]

部品番号	インターフェース名	型番	メーカー
CON1	アドオンインターフェース	DF17(4.0)-60DP-0.5V(57)	HIROSE ELECTRIC
CON2	シリアル(UART)インターフェース	XM2C-0942-132L	OMRON
CON3	テストインターフェース	A1-30PA-2.54DSA(71)	HIROSE ELECTRIC
CON4	テストインターフェース		

[a]色のついたセルの部品は実装していません。実装例を記載しています。

CON3、CON4 は開発用途でご使用ください。

18.1.3.2. CON1 アドオンインターフェース

CON1 は Armadillo-X1 アドオンインターフェース(CON7)との接続コネクタです。

・ 許容電流: 0.3A(端子 1 本あたり)

表 18.4 CON1 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)
2	GND	Power	電源(GND)
3	ADDIO3	In/Out	拡張入出力、CON3 の 3 ピンに接続
4	ADDIO4	In/Out	拡張入出力、CON3 の 4 ピンに接続
5	ADDIO5	In/Out	拡張入出力、CON3 の 5 ピンに接続
6	ADDIO6	In/Out	拡張入出力、CON3 の 6 ピンに接続
7	ADDIO7	In/Out	拡張入出力、CON3 の 7 ピンに接続
8	ADDIO8	In/Out	拡張入出力、CON3 の 8 ピンに接続
9	ADDIO9	In/Out	拡張入出力、CON3 の 9 ピンに接続
10	ADDIO10	In/Out	拡張入出力、CON3 の 10 ピンに接続
11	ADDIO11	In/Out	拡張入出力、CON3 の 11 ピンに接続
12	ADDIO12	In/Out	拡張入出力、CON3 の 12 ピンに接続
13	ADDIO13	In/Out	拡張入出力、CON3 の 13 ピンに接続
14	ADDIO14	In/Out	拡張入出力、CON3 の 14 ピンに接続
15	ADDIO15	In/Out	拡張入出力、CON3 の 15 ピンに接続
16	ADDIO16	In/Out	拡張入出力、CON3 の 16 ピンに接続
17	ADDIO17	In/Out	拡張入出力、CON3 の 17 ピンに接続
18	ADDIO18	In/Out	拡張入出力、CON3 の 18 ピンに接続
19	ADDIO19	In/Out	拡張入出力、CON3 の 19 ピンに接続
20	ADDIO20	In/Out	拡張入出力、CON3 の 20 ピン、EEPROM の SCL ピンに接続
21	ADDIO21	In/Out	拡張入出力、CON3 の 21 ピン、EEPROM の SDA ピンに接続
22	ADDIO22	In/Out	拡張入出力、CON3 の 22 ピンに接続
23	ADDIO23	In/Out	拡張入出力、CON3 の 23 ピンに接続
24	ADDIO24	In/Out	拡張入出力、CON3 の 24 ピンに接続
25	ADDIO25	In/Out	拡張入出力、CON3 の 25 ピンに接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	+3.3V_IO	Power	電源(+3.3V_IO)
29	+3.3V	Power	電源(+3.3V)
30	+5V	Power	電源(+5V)
31	DETECT	In	EEPROM のアドレスピン、CON4 の 31 ピンに接続
32	ADDIO32	In/Out	拡張入出力、CON4 の 32 ピンに接続
33	ADDIO33	In/Out	拡張入出力、CON4 の 33 ピンに接続
34	ADDIO34	In/Out	拡張入出力、CON4 の 34 ピンに接続
35	ADDIO35	In/Out	拡張入出力、CON4 の 35 ピンに接続
36	ADDIO36	In/Out	拡張入出力、CON4 の 36 ピンに接続
37	ADDIO37	In/Out	拡張入出力、CON4 の 37 ピンに接続
38	ADDIO38	In/Out	送信要求 CON4 の 38 ピン、レベル変換 IC を経由して CON2 の 7 ピンに接続

ピン番号	ピン名	I/O	説明
39	ADDIO39	In/Out	送信可能 CON4 の 39 ピン、レベル変換 IC を経由して CON2 の 8 ピンに接続
40	ADDIO40	In/Out	送信データ CON4 の 40 ピン、レベル変換 IC を経由して CON2 の 3 ピンに接続
41	ADDIO41	In/Out	受信データ CON4 の 41 ピン、レベル変換 IC を経由して CON2 の 2 ピンに接続
42	ADDIO42	In/Out	拡張入出力、CON4 の 42 ピンに接続
43	ADDIO43	In/Out	拡張入出力、CON4 の 43 ピンに接続
44	ADDIO44	In/Out	拡張入出力、CON4 の 44 ピンに接続
45	ADDIO45	In/Out	拡張入出力、CON4 の 45 ピンに接続
46	ADDIO46	In/Out	被呼表示 CON4 の 46 ピン、レベル変換 IC を経由して CON2 の 9 ピンに接続
47	ADDIO47	In/Out	キャリア検出 CON4 の 47 ピン、レベル変換 IC を経由して CON2 の 1 ピンに接続
48	ADDIO48	In/Out	データセットレディ CON4 の 48 ピン、レベル変換 IC を経由して CON2 の 6 ピンに接続
49	ADDIO49	In/Out	データ端末レディ CON4 の 49 ピン、レベル変換 IC を経由して CON2 の 4 ピンに接続
50	ADDIO50	In/Out	拡張入出力、CON4 の 50 ピンに接続
51	ADDIO51	In/Out	拡張入出力、CON4 の 51 ピンに接続
52	ADDIO52	In/Out	拡張入出力、CON4 の 52 ピンに接続
53	ADDIO53	In/Out	拡張入出力、CON4 の 53 ピンに接続
54	GND	Power	電源(GND)
55	PMIC_ONOFF	Out	パワーマネジメント IC の ON/OFF 用信号、CON4 の 55 ピンに接続
56	USB_VBUS	Power	USB 電源、CON4 の 56 ピンに接続
57	USB_VBUS	Power	USB 電源、CON4 の 57 ピンに接続
58	GND	Power	電源(GND)
59	EXT_USB_HS_DP	In/Out	USB プラス側信号、CON4 の 59 ピンに接続
60	EXT_USB_HS_DM	In/Out	USB マイナス側信号、CON4 の 60 ピンに接続

抵抗を取り外すことにより、RS232C レベル変換 IC、EEPROM への配線を切り離すことが可能です。詳細につきましては、回路図をご参照ください。

18.1.3.3. CON2 シリアルインターフェース

CON2 は非同期(調歩同期)シリアルインターフェースです。

- 信号入出力レベル: RS232C レベル
- 最大データ転送レート: 250kbps
- フロー制御: CTS、RTS、DTR、DSR、DCD、RI

表 18.5 CON2 信号配列

ピン番号	信号名	I/O	機能
1	DCD	In	キャリア検出、レベル変換 IC を経由して CON1 の 47 ピンに接続
2	RXD	In	受信データ、レベル変換 IC を経由して CON1 の 41 ピンに接続
3	TXD	Out	送信データ、レベル変換 IC を経由して CON1 の 40 ピンに接続
4	DTR	Out	データ端末レディ、レベル変換 IC を経由して CON1 の 49 ピンに接続
5	GND	Power	電源(GND)
6	DSR	In	データセットレディ、レベル変換 IC を経由して CON1 の 48 ピンに接続

ピン番号	信号名	I/O	機能
7	RTS	Out	送信要求、レベル変換 IC を経由して CON1 の 38 ピンに接続
8	CTS	In	送信可能、レベル変換 IC を経由して CON1 の 39 ピンに接続
9	RI	In	被呼表示、レベル変換 IC を経由して CON1 の 46 ピンに接続

18.1.3.4. CON3 テストインターフェース

CON3 はベースボードのアドオンインターフェースに接続されている信号線を確認するための、テスト用インターフェースです。アドオンインターフェースの信号線がスルーで接続されています。

表 18.6 CON3 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)
2	GND	Power	電源(GND)
3	ADDIO3	In/Out	CON1 の 3 ピンに接続
4	ADDIO4	In/Out	CON1 の 4 ピンに接続
5	ADDIO5	In/Out	CON1 の 5 ピンに接続
6	ADDIO6	In/Out	CON1 の 6 ピンに接続
7	ADDIO7	In/Out	CON1 の 7 ピンに接続
8	ADDIO8	In/Out	CON1 の 8 ピンに接続
9	ADDIO9	In/Out	CON1 の 9 ピンに接続
10	ADDIO10	In/Out	CON1 の 10 ピンに接続
11	ADDIO11	In/Out	CON1 の 11 ピンに接続
12	ADDIO12	In/Out	CON1 の 12 ピンに接続
13	ADDIO13	In/Out	CON1 の 13 ピンに接続
14	ADDIO14	In/Out	CON1 の 14 ピンに接続
15	ADDIO15	In/Out	CON1 の 15 ピンに接続
16	ADDIO16	In/Out	CON1 の 16 ピンに接続
17	ADDIO17	In/Out	CON1 の 17 ピンに接続
18	ADDIO18	In/Out	CON1 の 18 ピンに接続
19	ADDIO19	In/Out	CON1 の 19 ピンに接続
20	ADDIO20	In/Out	CON1 の 20 ピンに接続
21	ADDIO21	In/Out	CON1 の 21 ピンに接続
22	ADDIO22	In/Out	CON1 の 22 ピンに接続
23	ADDIO23	In/Out	CON1 の 23 ピンに接続
24	ADDIO24	In/Out	CON1 の 24 ピンに接続
25	ADDIO25	In/Out	CON1 の 25 ピンに接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	+3.3V_IO	Power	電源(+3.3V_IO)
29	+3.3V	Power	電源(+3.3V)
30	+5V	Power	電源(+5V)

18.1.3.5. CON4 テストインターフェース

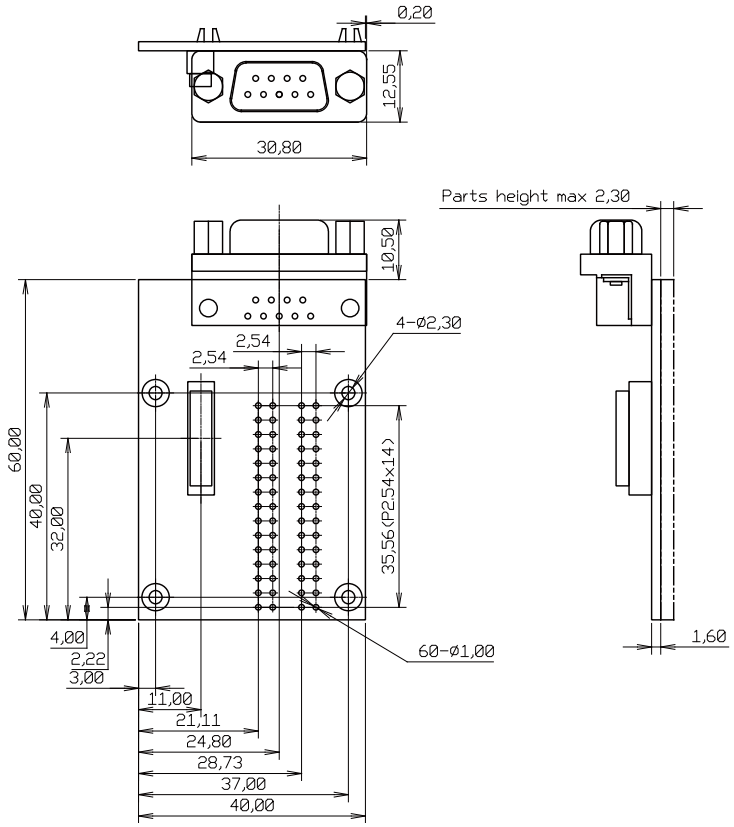
CON4 はベースボードのアドオンインターフェースに接続されている信号線を確認するための、テスト用インターフェースです。アドオンインターフェースの信号線がスルーで接続されています。

表 18.7 CON4 信号配列

ピン番号	ピン名	I/O	説明
31	DETECT	In	CON1 の 31 ピンに接続
32	ADDIO32	In/Out	CON1 の 32 ピンに接続
33	ADDIO33	In/Out	CON1 の 33 ピンに接続
34	ADDIO34	In/Out	CON1 の 34 ピンに接続

ピン番号	ピン名	I/O	説明
35	ADDIO35	In/Out	CON1 の 35 ピンに接続
36	ADDIO36	In/Out	CON1 の 36 ピンに接続
37	ADDIO37	In/Out	CON1 の 37 ピンに接続
38	ADDIIO38	In/Out	CON1 の 38 ピンに接続
39	ADDIO39	In/Out	CON1 の 39 ピンに接続
40	ADDIO40	In/Out	CON1 の 40 ピンに接続
41	ADDIO41	In/Out	CON1 の 41 ピンに接続
42	ADDIO42	In/Out	CON1 の 42 ピンに接続
43	ADDIO43	In/Out	CON1 の 43 ピンに接続
44	ADDIO44	In/Out	CON1 の 44 ピンに接続
45	ADDIO45	In/Out	CON1 の 45 ピンに接続
46	ADDIO46	In/Out	CON1 の 46 ピンに接続
47	ADDIO47	In/Out	CON1 の 47 ピンに接続
48	ADDIO48	In/Out	CON1 の 48 ピンに接続
49	ADDIO49	In/Out	CON1 の 49 ピンに接続
50	ADDIO50	In/Out	CON1 の 50 ピンに接続
51	ADDIO51	In/Out	CON1 の 51 ピンに接続
52	ADDIO50	In/Out	CON1 の 52 ピンに接続
53	ADDIO53	In/Out	CON1 の 53 ピンに接続
54	GND	Power	電源(GND)
55	PMIC_ONOFF	Out	CON1 の 55 ピンに接続
56	USB_VBUS	Power	CON1 の 56 ピンに接続
57	USB_VBUS	Power	CON1 の 57 ピンに接続
58	GND	Power	電源(GND)
59	EXT_USB_HS_DP	In/Out	CON1 の 59 ピンに接続
60	EXT_USB_HS_DM	In/Out	CON1 の 60 ピンに接続

18.1.4. 基板形状図



[Unit : mm]

図 18.3 RS232C アドオンモジュール基板形状

18.2. Armadillo-IoT 絶縁 RS232C/422/485 アドオンモジュール RS01

18.2.1. 概要

Armadillo-IoT 絶縁 RS232C/422/485 アドオンモジュール RS01(以降、絶縁シリアルアドオンモジュールと記載します)は、電気的に絶縁された RS232C/RS422/RS485 のシリアルインターフェースを 1 ポート追加することができます。

絶縁シリアルアドオンモジュールの仕様は次のとおりです。

表 18.8 絶縁シリアルアドオンモジュールの仕様

シリアル(UART)	Exar 製 XR3160E 搭載 RS232C/RS422/RS485 x 1 最大データ転送レート: 1Mbps
スイッチ	RS232C/RS422/RS485 切替用ディップスイッチ
絶縁耐圧	2kV
電源電圧	DC 3.3V±5%
使用温度範囲	-20℃～70℃
基板サイズ	40 x 60mm(突起部を除く)

18.2.2. ブロック図

絶縁シリアルアドオンモジュールのブロック図は次のとおりです。

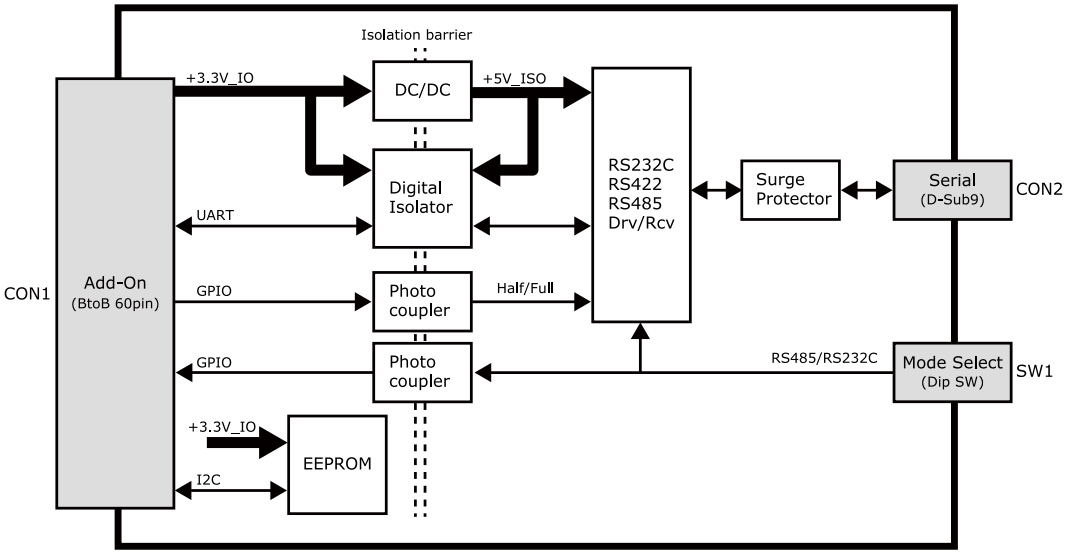


図 18.4 絶縁シリアルアドオンモジュール ブロック図

18.2.3. インターフェース仕様

絶縁シリアルアドオンモジュールのインターフェース仕様について説明します。

18.2.3.1. インターフェースレイアウト

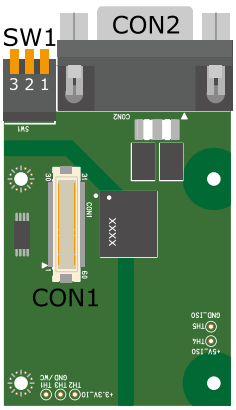


図 18.5 絶縁シリアルアドオンモジュール インターフェースレイアウト

表 18.9 搭載コネクタ、スイッチ型番一覧

部品番号	インターフェース名	型番	メーカー
CON1	アドオンインターフェース	DF17(4.0)-60DP-0.5V(57)	HIROSE ELECTRIC
CON2	シリアル(UART)インターフェース	XM2C-0942-132L	OMRON
SW1	設定スイッチ	A6ER-3104	OMRON

絶縁シリアルアドオンモジュールの固定穴(TH6、TH7)の PAD 部分は GND に接続されています。固定穴(TH8、TH9)はキリ穴で GND に接続されていません。

D-Sub コネクタ(CON2)の金属フレームは GND_ISO に接続されています。

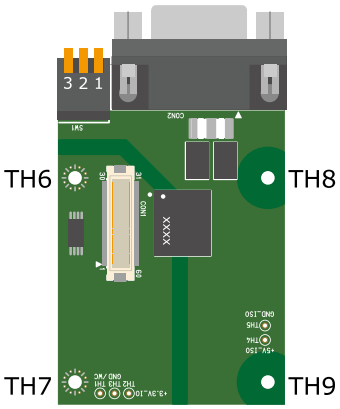


図 18.6 絶縁シリアルアドオンモジュールの固定穴

18.2.3.2. CON1 アドオンインターフェース

CON1 は Armadillo-X1 アドオンインターフェース(CON7)との接続コネクタです。

・ 許容電流: 0.3A(端子 1 本あたり)

表 18.10 CON1 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)
2	GND	Power	電源(GND)
3	NC	-	未接続
4	NC	-	未接続
5	NC	-	未接続
6	NC	-	未接続
7	NC	-	未接続
8	NC	-	未接続
9	NC	-	未接続
10	NC	-	未接続
11	NC	-	未接続
12	NC	-	未接続
13	NC	-	未接続
14	NC	-	未接続
15	NC	-	未接続
16	NC	-	未接続
17	NC	-	未接続
18	NC	-	未接続
19	NC	-	未接続
20	EEPROM_SCL	In/Out	EEPROM の SCL ピンに接続
21	EEPROM_SDA	In/Out	EEPROM の SDA ピンに接続
22	NC	-	未接続

ピン番号	ピン名	I/O	説明
23	NC	-	未接続
24	NC	-	未接続
25	NC	-	未接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	+3.3V_IO	Power	電源(+3.3V_IO)
29	NC	-	未接続
30	NC	-	未接続
31	DETECT	In	EEPROM のアドレスピンに接続
32	GPIO0	In	半二重/全二重通信の切替信号入力 (Low: 半二重、High: 全二重)
33	NC	-	未接続
34	NC	-	未接続
35	NC	-	未接続
36	NC	-	未接続
37	NC	-	未接続
38	UART_RTS	In	デジタルアイソレータを経由して RS232C/422/485 トランシーバに接続
39	UART_CTS	Out	デジタルアイソレータを経由して RS232C/422/485 トランシーバに接続
40	UART_TXD	In	デジタルアイソレータを経由して RS232C/422/485 トランシーバに接続
41	UART_RXD	Out	デジタルアイソレータを経由して RS232C/422/485 トランシーバに接続
42	GPIO2	Out	RS232C、RS422/RS485 の切替信号出力、フォトカプラを経由して SW1 に接続 (Low: RS232C、High: RS422/RS485)
43	GPIO3	In	デジタルアイソレータのイネーブルピンに接続
44	NC	-	未接続
45	NC	-	未接続
46	NC	-	未接続
47	NC	-	未接続
48	NC	-	未接続
49	NC	-	未接続
50	NC	-	未接続
51	NC	-	未接続
52	NC	-	未接続
53	NC	-	未接続
54	GND	Power	電源(GND)
55	NC	-	未接続
56	NC	-	未接続
57	NC	-	未接続
58	GND	Power	電源(GND)
59	NC	-	未接続
60	NC	-	未接続

18.2.3.3. CON2 シリアルインターフェース

CON2 は電気的に絶縁されたシリアルインターフェースです。設定スイッチ(SW1)で RS232C と RS422/RS485 の切替が可能です。

- 最大データ転送レート: 1Mbps
- フロー制御: CTS、RTS(RS232C)
- 通信方式: 半二重、全二重(RS422/RS485)

SW1.1 を ON にすると RS232C に設定されます。信号配列は次のとおりです。

表 18.11 CON2 信号配列(RS232C に設定時)

ピン番号	信号名	I/O	機能
1	NC	-	未接続
2	RXD	In	受信データ RS232C/422/485 トランシーバ、デジタルアイソレータを経由して CON1 の 41 ピンに接続
3	TXD	Out	送信データ RS232C/422/485 トランシーバ、デジタルアイソレータを経由して CON1 の 40 ピンに接続
4	NC	-	未接続
5	GND_ISO	Power	電源(GND_ISO)
6	NC	-	未接続
7	RTS	Out	送信要求 RS232C/422/485 トランシーバ、デジタルアイソレータを経由して CON1 の 38 ピンに接続
8	CTS	In	送信可能 RS232C/422/485 トランシーバ、デジタルアイソレータを経由して CON1 の 39 ピンに接続
9	NC	-	未接続

SW1.1 を OFF にすると RS422/RS485 に設定されます。半二重/全二重の切替は GPIO で行います。

RS422/RS485 全二重に設定時の接続は次のとおりです。

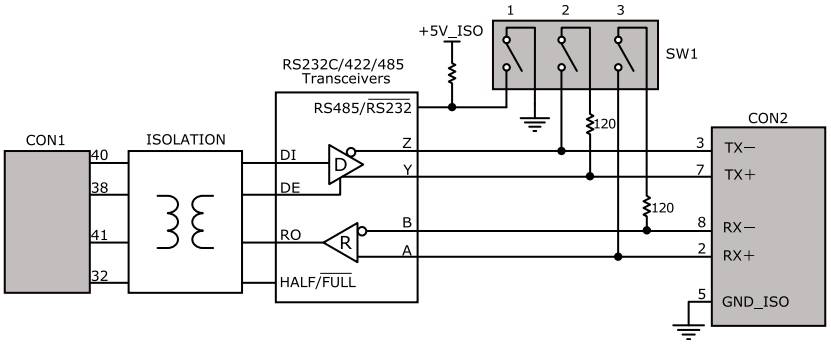


図 18.7 RS422/RS485 全二重に設定時の接続

表 18.12 CON2 信号配列(RS422/RS485 全二重に設定時)

ピン番号	信号名	I/O	機能
1	NC	-	未接続
2	RX +	In	受信データ(+) RS232C/422/485 トランシーバの A ピンに接続
3	TX−	Out	送信データ(−) RS232C/422/485 トランシーバの Z ピンに接続
4	NC	-	未接続
5	GND_ISO	Power	電源(GND_ISO)
6	NC	-	未接続
7	TX +	Out	送信データ(+) RS232C/422/485 トランシーバの Y ピンに接続
8	RX−	In	受信データ(−) RS232C/422/485 トランシーバの B ピンに接続
9	NC	-	未接続

RS422/RS485 半二重に設定時の接続は次のとおりです。

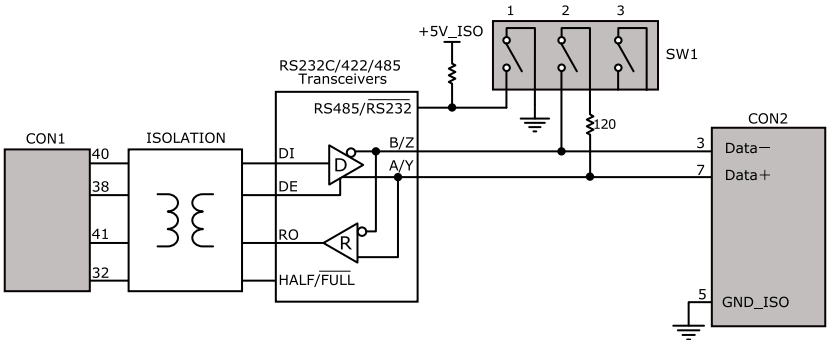


図 18.8 RS422/RS485 半二重に設定時の接続

表 18.13 CON2 信号配列(RS422/RS485 半二重に設定時)

ピン番号	信号名	I/O	機能
1	NC	-	未接続
2	Reserved	-	未接続
3	DATA−	In/Out	送受信データ(−) RS232C/422/485 トランシーバの B/Z ピンに接続
4	NC	-	未接続
5	GND_ISO	Power	電源(GND_ISO)
6	NC	-	未接続
7	DATA +	In/Out	送受信データ(+) RS232C/422/485 トランシーバの A/Y ピンに接続
8	Reserved	-	未接続
9	NC	-	未接続

18.2.3.4. SW1 設定スイッチ

SW1 は RS232C と RS422/RS485 の切替、終端抵抗(120Ω)の ON/OFF を行うためのディップスイッチです。

表 18.14 SW1 機能

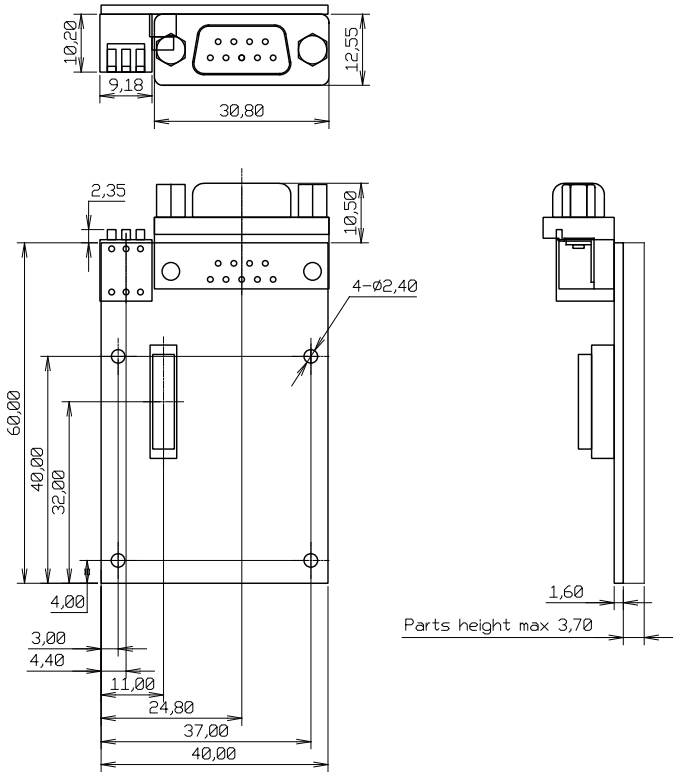
SW1	ON	OFF
1	RS232C	RS422/RS485
2	TX 終端抵抗(120Ω) ON	TX 終端抵抗(120Ω) OFF
3	RX 終端抵抗(120Ω) ON	RX 終端抵抗(120Ω) OFF

設定スイッチ(SW1)は電源を切断した状態で操作してください。

RS232C で使用する場合、終端抵抗(120Ω)は必ず OFF にしてください。

終端は RS422/RS485 の信号線の最遠端で行います。Armadillo-X1 が最遠端になる場合は終端抵抗を ON にしてください。

18.2.4. 基板形状図



[Unit : mm]

図 18.9 絶縁シリアルアドオンモジュール基板形状

18.2.5. 使用方法

絶縁シリアルアドオンモジュールのシリアルインターフェース(CON2)は、設定スイッチ(SW1)で RS232C と RS422/RS485 の切替が可能です。

RS232C で使用する場合

シリアルインターフェース(CON2)を RS232C で使用する場合は、SW1.1 を ON にします。

図 18.10 RS232C で使用する場合の設定スイッチ(SW1)の状態

RS232C で使用する場合の、外部機器との接続例は次のとおりです。

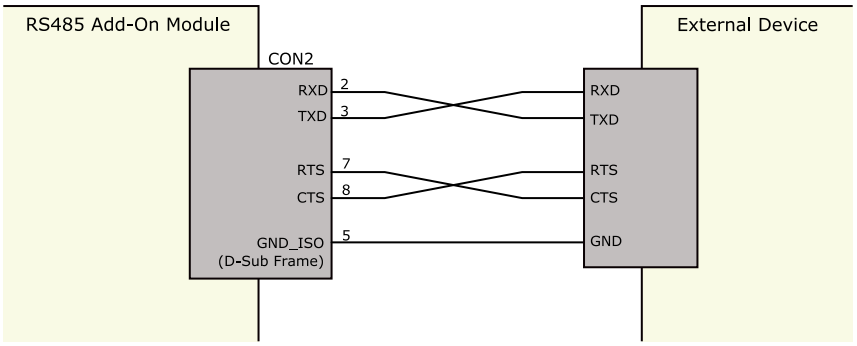


図 18.11 外部機器との接続例(RS232C で使用する場合)

RS422/RS485 で使用する場合

シリアルインターフェース(CON2)を RS422/RS485 で使用する場合は、SW1.1 を OFF にします。

図 18.12 RS422/RS485 で使用する場合の設定スイッチ(SW1)の状態

半二重と全二重の切替はアドオンインターフェース(CON1)の 32 ピンから行います。Low レベルを入力することで半二重、High レベルを入力することで全二重に設定されます。

表 18.15 半二重と全二重の切替

入力レベル	通信方式
Low	半二重
High	全二重

RS422/RS485 半二重で使用する場合の、外部機器との接続例は次のとおりです。

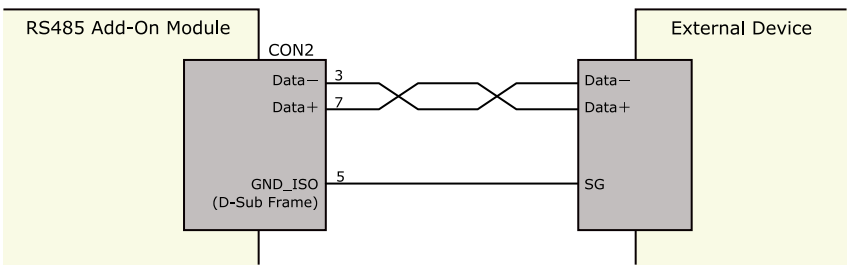


図 18.13 外部機器との接続例(RS422/RS485 半二重で使用する場合)

RS422/RS485 全二重で使用する場合の、外部機器との接続例は次のとおりです。

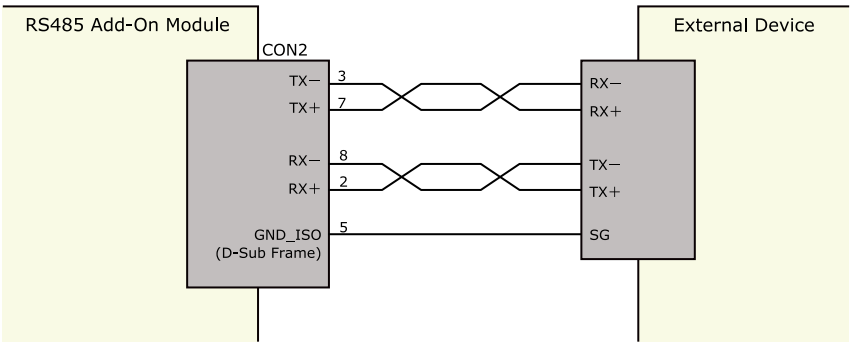


図 18.14 外部機器との接続例(RS422/RS485 全二重で使用する場合)

ESD/雷サージ

接続ケーブルが屋外に露出するような設置環境では、ケーブルに侵入した雷サージ等のストレスによりインターフェース回路が破壊される場合があります。ストレスへの耐性を向上させるには、シールド付きケーブルを使用すること、GND_ISO(D-Sub コネクタの金属フレーム)とアース間にアレスタ、バリスタ等の保護素子を接続することが効果的です。

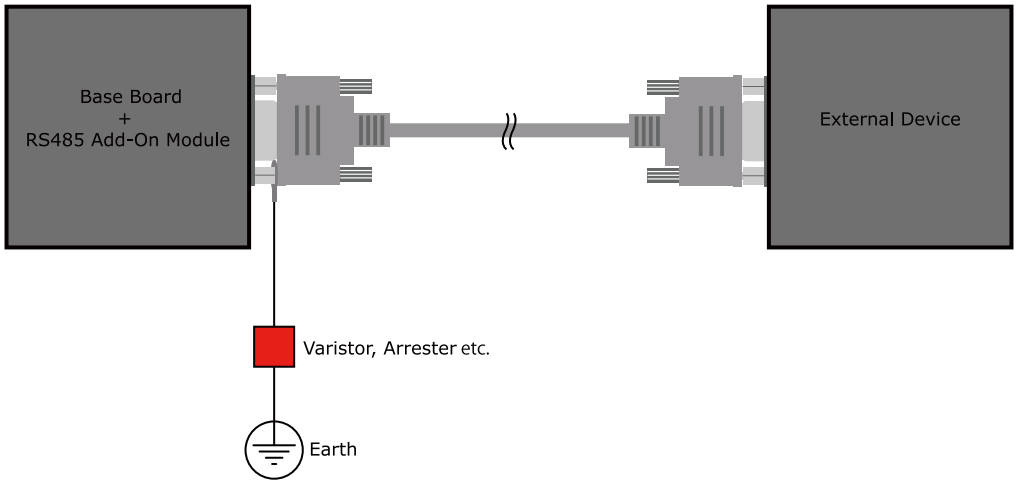


図 18.15 保護素子の接続例

シリアルインターフェース(CON2)の 5 ピン(GND_ISO)と D-Sub コネクタの金属フレームは基板上で接続されており、切り離すことはできません。

信号品質の低下、故障を防ぐため、配線、接地などの設置環境に十分にご配慮ください。

18.3. Armadillo-IoT 絶縁 RS485 アドオンモジュール RS02

18.3.1. 概要

Armadillo-IoT 絶縁 RS485 アドオンモジュール RS02(以降、絶縁 RS485 アドオンモジュールと記載します)は、電気的に絶縁された RS422/RS485 のシリアルインターフェースを 1 ポート追加することができます。

絶縁 RS485 アドオンモジュールの仕様は次のとおりです。

表 18.16 絶縁 RS485 アドオンモジュールの仕様

シリアル(UART)	Texas Instruments 製 ISO3086T 搭載 RS422/RS485 x 1 最大データ転送レート: 4Mbps
スイッチ	設定用ディップスイッチ
絶縁耐圧	2kV
電源電圧	DC 3.3V±5%
使用温度範囲	-20℃～70℃
基板サイズ	40 x 63mm(突起部を除く)

18.3.2. ブロック図

絶縁 RS485 アドオンモジュールのブロック図は次のとおりです。

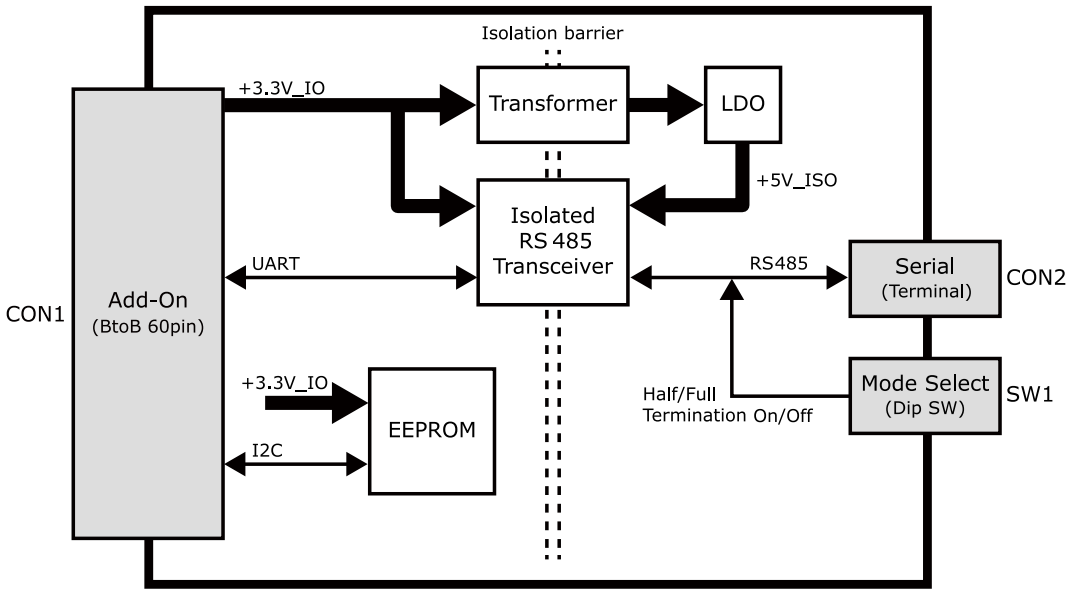


図 18.16 絶縁 RS485 アドオンモジュール ブロック図

18.3.3. インターフェース仕様

絶縁 RS485 アドオンモジュールのインターフェース仕様について説明します。

18.3.3.1. インターフェースレイアウト

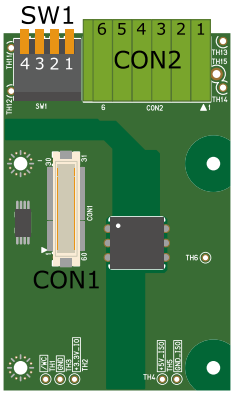


図 18.17 絶縁 RS485 アドオンモジュール インターフェースレイアウト

表 18.17 搭載コネクタ、スイッチ型番一覧

部品番号	インターフェース名	型番	メーカー
CON1	アドオンインターフェース	DF17(4.0)-60DP-0.5V(57)	HIROSE ELECTRIC
CON2	シリアル(UART)インターフェース	XW4C-06D1-H1	OMRON
SW1	設定スイッチ	A6ER-4104	OMRON

絶縁 RS485 アドオンモジュールの固定穴(TH7、TH8)の PAD 部分は GND に接続されています。固定穴(TH9、TH10)はキリ穴で GND に接続されていません。

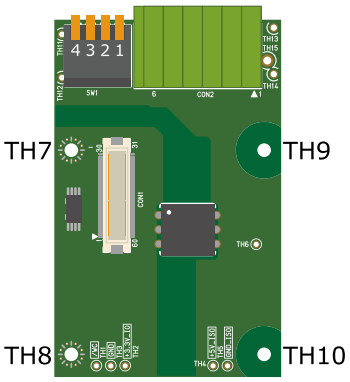


図 18.18 絶縁 RS485 アドオンモジュールの固定穴

18.3.3.2. CON1 アドオンインターフェース

CON1 は Armadillo-X1 アドオンインターフェース(CON7)との接続コネクタです。

・ 許容電流: 0.3A(端子 1 本あたり)

表 18.18 CON1 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)

ピン番号	ピン名	I/O	説明
2	GND	Power	電源(GND)
3	NC	-	未接続
4	NC	-	未接続
5	NC	-	未接続
6	NC	-	未接続
7	NC	-	未接続
8	NC	-	未接続
9	NC	-	未接続
10	NC	-	未接続
11	NC	-	未接続
12	NC	-	未接続
13	NC	-	未接続
14	NC	-	未接続
15	NC	-	未接続
16	NC	-	未接続
17	NC	-	未接続
18	NC	-	未接続
19	NC	-	未接続
20	EEPROM_SCL	In/Out	EEPROM の SCL ピンに接続
21	EEPROM_SDA	In/Out	EEPROM の SDA ピンに接続
22	NC	-	未接続
23	NC	-	未接続
24	NC	-	未接続
25	NC	-	未接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	+3.3V_IO	Power	電源(+3.3V_IO)
29	NC	-	未接続
30	NC	-	未接続
31	DETECT	In	EEPROM のアドレスピンに接続
32	RS485_DE	In	RS485 トランシーバの DE ピンに接続
33	RS485_RE_N	In	RS485 トランシーバの RE_N ピンに接続
34	NC	-	未接続
35	NC	-	未接続
36	NC	-	未接続
37	NC	-	未接続
38	NC	-	未接続
39	NC	-	未接続
40	UART_TXD	In	RS485 トランシーバの D ピンに接続
41	UART_RXD	Out	RS485 トランシーバの R ピンに接続
42	NC	-	未接続
43	NC	-	未接続
44	NC	-	未接続
45	NC	-	未接続
46	NC	-	未接続
47	NC	-	未接続
48	NC	-	未接続
49	NC	-	未接続
50	NC	-	未接続
51	NC	-	未接続
52	NC	-	未接続
53	NC	-	未接続
54	GND	Power	電源(GND)
55	NC	-	未接続

ピン番号	ピン名	I/O	説明
56	NC	-	未接続
57	NC	-	未接続
58	GND	Power	電源(GND)
59	NC	-	未接続
60	NC	-	未接続

18.3.3.3. CON2 シリアルインターフェース

CON2 は電気的に絶縁されたシリアルインターフェースです。設定スイッチ(SW1)で半二重/全二重の切替、終端抵抗の ON/OFF が可能です。

- 最大データ転送レート: 4Mbps
- 通信方式: 半二重、全二重

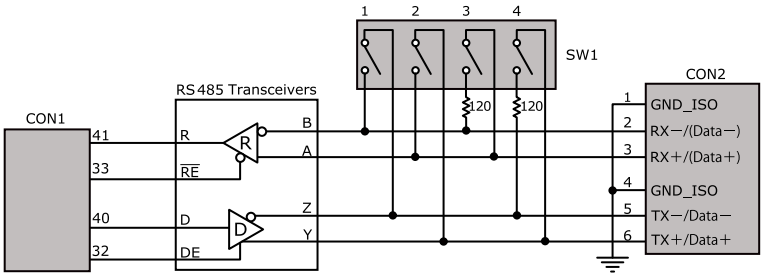


図 18.19 RS485 トランシーバ周辺回路

SW1.1、SW1.2 を ON にすると半二重に設定されます。

表 18.19 CON2 信号配列(半二重に設定時)

ピン番号	信号名	I/O	機能
1	GND_ISO	Power	電源(GND_ISO)
2	(Data-)	In/Out	送受信データ(−)、CON2 の 5 ピンと共通 RS485 トランシーバの B ピン、Z ピンに接続
3	(Data+)	In/Out	送受信データ(+)、CON2 の 6 ピンと共通 RS485 トランシーバの A ピン、Y ピンに接続
4	GND_ISO	Power	電源(GND_ISO)
5	Data-	In/Out	送受信データ(−)、CON2 の 2 ピンと共通 RS485 トランシーバの B ピン、Z ピンに接続
6	Data+	In/Out	送受信データ(+)、CON2 の 3 ピンと共通 RS485 トランシーバの A ピン、Y ピンに接続

SW1.1、SW1.2 を OFF にすると全二重に設定されます。

表 18.20 CON2 信号配列(全二重に設定時)

ピン番号	信号名	I/O	機能
1	GND_ISO	Power	電源(GND_ISO)
2	RX-	In	受信データ(−) RS485 トランシーバの B ピンに接続
3	RX+	In	受信データ(+) RS485 トランシーバの A ピンに接続
4	GND_ISO	Power	電源(GND_ISO)
5	TX-	Out	送信データ(−) RS485 トランシーバの Z ピンに接続

ピン番号	信号名	I/O	機能
6	TX+	Out	送信データ(+) RS485 トランシーバの Y ピンに接続

18.3.3.4. SW1 設定スイッチ

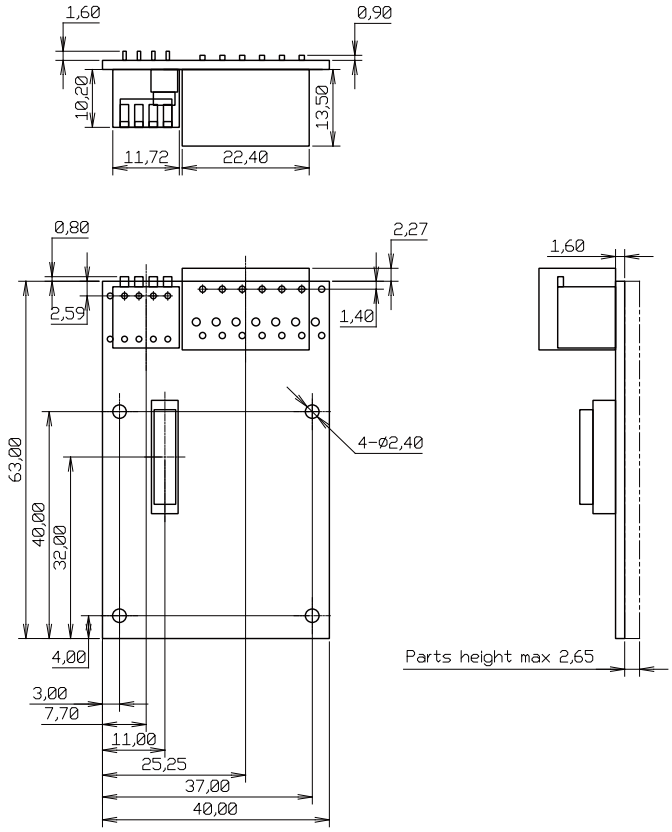
SW1 は半二重/全二重の切替、終端抵抗(120Ω)の ON/OFF を行うためのディップスイッチです。

表 18.21 SW1 機能

SW1	ON	OFF
1	半二重	全二重
2	半二重	全二重
3	RX 終端抵抗(120Ω) ON	RX 終端抵抗(120Ω) OFF
4	TX 終端抵抗(120Ω) ON	TX 終端抵抗(120Ω) OFF

終端は RS485 の信号線の最遠端で行います。Armadillo-X1 が最遠端になる場合は終端抵抗を ON にしてください。

18.3.4. 基板形状図



[Unit : mm]

図 18.20 絶縁 RS485 アドオンモジュール基板形状

18.3.5. 使用方法

シリアルインターフェース(CON2)に実装されている端子台に接続可能な電線は次のとおりです。

表 18.22 端子台に接続可能な電線

単線		0.2～1.5mm^2
撚線		0.2～1.5mm^2
棒端子	スリーブなし	0.25～1.5mm^2
	スリーブあり	0.25～0.75mm^2
AWG		24～16

電線を直接接続する場合、先端加工は次のとおりです。電線むき長さ L は 10±1mm となります。

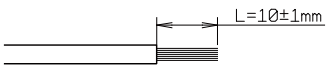


図 18.21 電線の先端加工

電線の先端を予備半田しないでください。正しい接続ができなくなります。

棒端子を使用する場合、使用する棒端子に合わせて電線加工を行ってください。棒端子のサイズは次のとおりです。

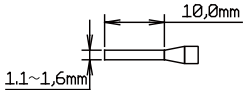


図 18.22 棒端子のサイズ

端子台に電線を接続する際、端子台に過度な力をかけないでください。端子台が破損する恐れがあります。

絶縁 RS485 アドオンモジュールのシリアルインターフェース(CON2)は、設定スイッチ(SW1)で半二重/全二重の切替が可能です。

半二重で使用する場合

シリアルインターフェース(CON2)を半二重で使用する場合は、SW1.1、SW1.2 を ON にします。

表 18.23 半二重で使用する場合の設定スイッチ(SW1)

SW1	機能	設定
1	半二重/全二重選択	ON
2		ON
3	RX 終端抵抗	OFF
4	TX 終端抵抗	ON/OFF[a]

[a]終端抵抗は必要に応じて設定してください。

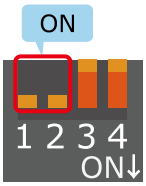


図 18.23 半二重で使用する場合の設定スイッチ(SW1)の状態

半二重で使用する場合の、外部機器との接続例は次のとおりです。

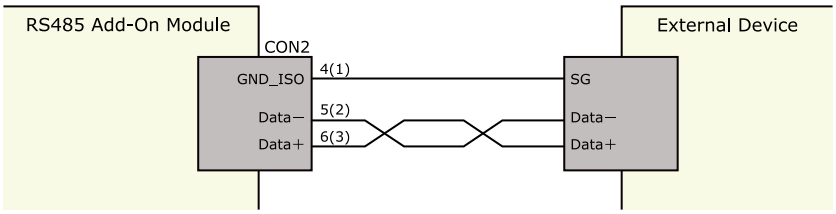


図 18.24 外部機器との接続例(半二重で使用する場合)

全二重で使用する場合

シリアルインターフェース(CON2)を全二重で使用する場合は、SW1.1、SW1.2 を OFF にします。

表 18.24 全二重で使用する場合の設定スイッチ(SW1)

SW1	機能	設定
1	半二重/全二重選択	OFF
2		OFF
3	RX 終端抵抗	ON/OFF[a]
4	TX 終端抵抗	ON/OFF[a]

[a]終端抵抗は必要に応じて設定してください。

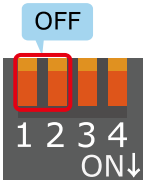


図 18.25 全二重で使用する場合の設定スイッチ(SW1)の状態

全二重で使用する場合の、外部機器との接続例は次のとおりです。

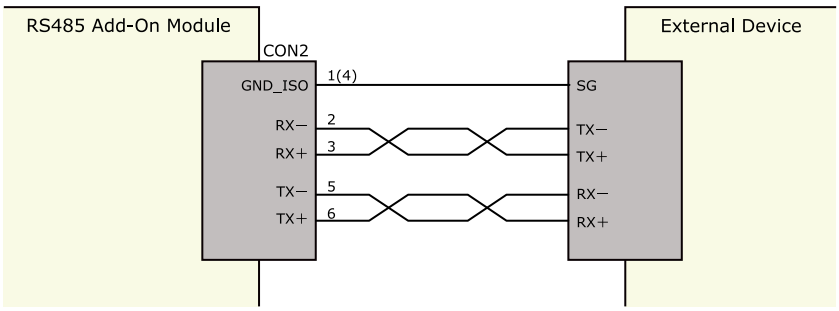


図 18.26 外部機器との接続例(全二重で使用する場合)

雷サージ

雷サージ対策部品は基板上に実装されておりません。

接続ケーブルが屋外に露出するような設置環境では、ケーブルに侵入した雷サージ等のストレスによりインターフェース回路が破壊される場合があります。ストレスへの耐性を向上させるために、各端子とアース間にアレスタ、バリスタ等の保護素子を接続することを推奨します。

信号品質の低下、故障を防ぐため、配線、接地などの設置環境に十分にご配慮ください。

18.4. Armadillo-IoT RN4020 アドオンモジュール BT00

18.4.1. 概要

Armadillo-IoT RN4020 アドオンモジュール BT00(以降、RN4020 アドオンモジュールと記載します)は、Microchip Technology 製 Bluetooth Low Energy モジュール RN4020 を搭載しています。

RN4020 アドオンモジュールの仕様は次のとおりです。

表 18.25 RN4020 アドオンモジュールの仕様

搭載モジュール	Microchip Technology 製 RN4020 Bluetooth 4.1/LE 同時接続数: 1[a]
電源電圧	DC 3.3V±5%
使用温度範囲	-20℃～70℃
基板サイズ	40 x 50mm(突起部を除く)

[a]アドバタイジングを含む、ブロードキャストされるパケットは複数同時受信可能です

18.4.2. Bluetooth SIG 認証(ロゴ認証)に関して

Bluetooth 対応製品を販売するには、Bluetooth SIG によって認証取得、製品登録および申告を行うことが定められています。

RN4020 はモジュールとして Bluetooth SIG 認証を取得済みです。認証取得済みの Bluetooth モジュールを自社製品に組み込む場合、QDID を使用して、Bluetooth SIG 製品登録および準拠申告を行うことができます(有償)。

詳しくは Bluetooth SIG の web サイトをご参照ください。

Bluetooth 認証および申告プロセス

https://www.bluetooth.org/ja-jp/test-qualification/qualification-overview

18.4.3. ブロック図

RN4020 アドオンモジュールのブロック図は次のとおりです。

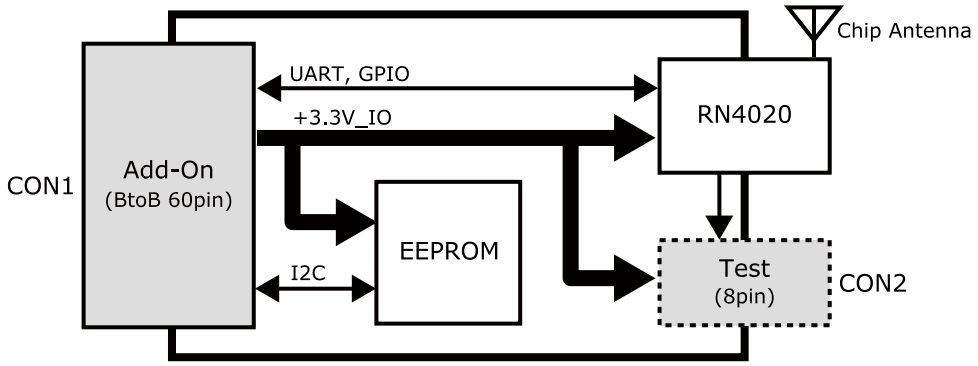


図 18.27 RN4020 アドオンモジュール ブロック図

18.4.4. インターフェース仕様

RN4020 アドオンモジュールのインターフェース仕様について説明します。

18.4.4.1. RN4020 アドオンモジュール インターフェースレイアウト

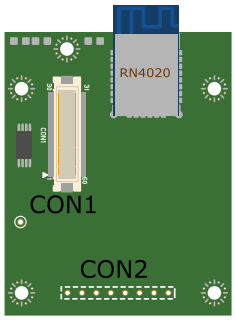


図 18.28 RN4020 アドオンモジュール インターフェースレイアウト

表 18.26 搭載コネクタ、スイッチ型番一覧[a]

部品番号	インターフェース名	型番	メーカー
CON1	アドオンインターフェース	DF17(4.0)-60DP-0.5V(57)	HIROSE ELECTRIC
CON2	テストインターフェース	A2-8PA-2.54DSA(71)	HIROSE ELECTRIC

[a]色のついたセルの部品は実装していません。実装例を記載しています。

CON2 は開発用途でご使用ください。

18.4.4.2. CON1 アドオンインターフェース

CON1 は Armadillo-X1 アドオンインターフェース(CON7)との接続コネクタです。

- 許容電流: 0.3A(端子 1 本あたり)

表 18.27 CON1 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)
2	GND	Power	電源(GND)
3	NC	-	未接続
4	NC	-	未接続
5	NC	-	未接続
6	NC	-	未接続
7	NC	-	未接続
8	NC	-	未接続
9	NC	-	未接続
10	NC	-	未接続
11	NC	-	未接続
12	NC	-	未接続
13	NC	-	未接続
14	NC	-	未接続
15	NC	-	未接続
16	NC	-	未接続
17	NC	-	未接続
18	NC	-	未接続
19	NC	-	未接続
20	EEPROM_SCL	In/Out	EEPROM の SCL ピンに接続
21	EEPROM_SDA	In/Out	EEPROM の SDA ピンに接続
22	NC	-	未接続
23	NC	-	未接続
24	NC	-	未接続
25	NC	-	未接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	+3.3V_IO	Power	電源(+3.3V_IO)
29	NC	-	未接続
30	NC	-	未接続
31	DETECT	In	EEPROM のアドレスピンに接続
32	NC	-	未接続
33	NC	-	未接続
34	NC	-	未接続
35	NC	-	未接続
36	NC	-	未接続
37	NC	-	未接続
38	UART_RTS	In	RN4020 の 14 ピンに接続
39	UART_CTS	Out	RN4020 の 18 ピンに接続
40	UART_TXD	In	RN4020 の 6 ピンに接続
41	UART_RXD	Out	RN4020 の 5 ピンに接続
42	GPIO2	Out	RN4020 の 15 ピンに接続
43	GPIO3	Out	RN4020 の 7 ピンに接続
44	NC	-	未接続
45	NC	-	未接続
46	GPIO6	Out	RN4020 の 8 ピンに接続
47	NC	-	未接続
48	NC	-	未接続
49	NC	-	未接続
50	NC	-	未接続
51	NC	-	未接続
52	NC	-	未接続

ピン番号	ピン名	I/O	説明
53	NC	-	未接続
54	GND	Power	電源(GND)
55	NC	-	未接続
56	NC	-	未接続
57	NC	-	未接続
58	GND	Power	電源(GND)
59	NC	-	未接続
60	NC	-	未接続

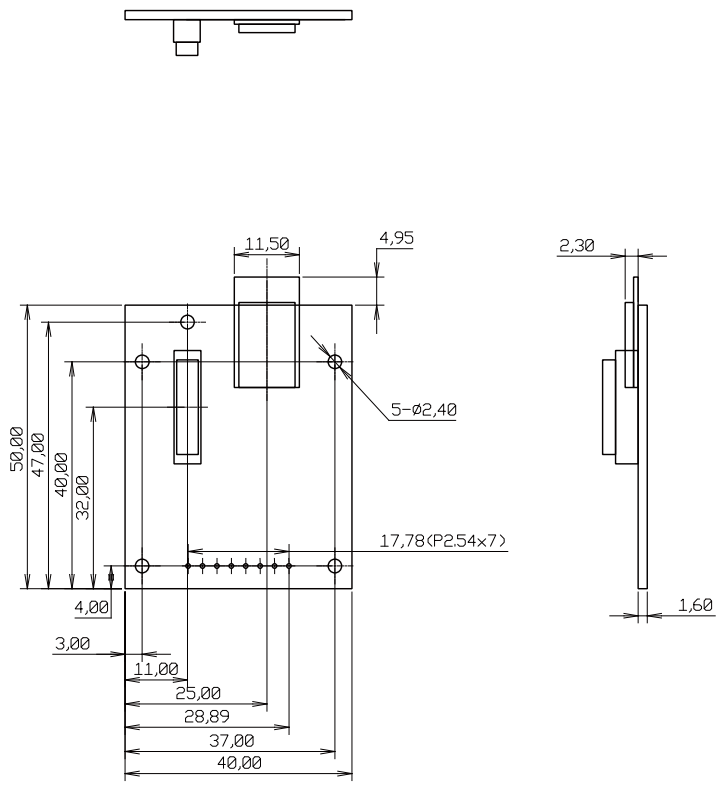
18.4.4.3. CON2 テストインターフェース

CON2 は RN4020 の信号線を確認するためのテスト用インターフェースです。RN4020 の信号線の一部がスルーで接続されています。

表 18.28 CON2 信号配列

ピン番号	信号名	I/O	機能
1	SPI_MODE	In/Out	RN4020 の 17 ピンに接続
2	+3.3_IO	Power	電源(+3.3V_IO)
3	GND	Power	電源(GND)
4	LED1_PIO1_SCK	In/Out	RN4020 の 10 ピンに接続
5	LED2_PIO2_SS	In/Out	RN4020 の 11 ピンに接続
6	LED3_PIO3_MOSI	In/Out	RN4020 の 12 ピンに接続
7	PIO4_MISO	In/Out	RN4020 の 13 ピンに接続
8	AIO0	In/Out	RN4020 の 4 ピンに接続

18.4.5. 基板形状図



[Unit : mm]

図 18.29 RN4020 アドオンモジュール基板形状

18.5. Armadillo-IoT EnOcean アドオンモジュール EN00

18.5.1. 概要

Armadillo-IoT EnOcean アドオンモジュール EN00(以降、EnOcean アドオンモジュールと記載します)は、ROHM 製の BP35A3 を搭載した EnOcean モジュールです。

EnOcean アドオンモジュールの仕様は次のとおりです。

表 18.29 EnOcean アドオンモジュールの仕様

EnOcean	ROHM 製 BP35A3 搭載
電源電圧	DC 3.3V±5%
使用温度範囲	-20℃～70℃
基板サイズ	40 x 50mm(突起部を除く)

18.5.2. ブロック図

EnOcean アドオンモジュールのブロック図は次のとおりです。

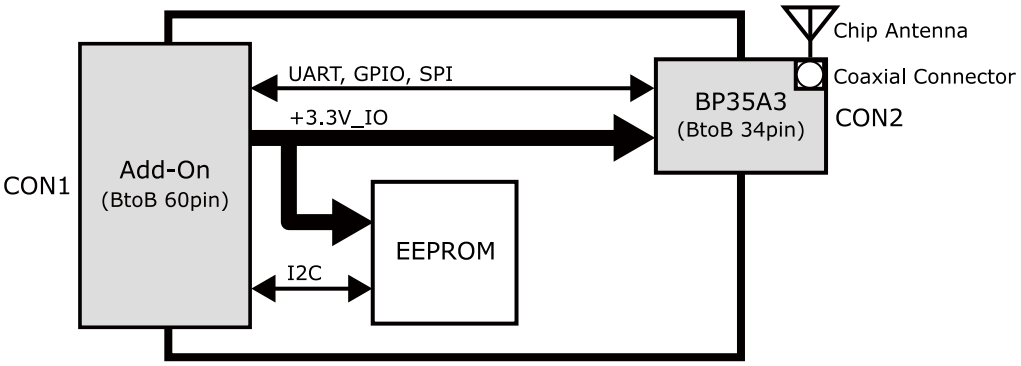


図 18.30 EnOcean アドオンモジュール ブロック図

18.5.3. インターフェース仕様

EnOcean アドオンモジュールのインターフェース仕様について説明します。

18.5.3.1. EnOcean アドオンモジュール インターフェースレイアウト

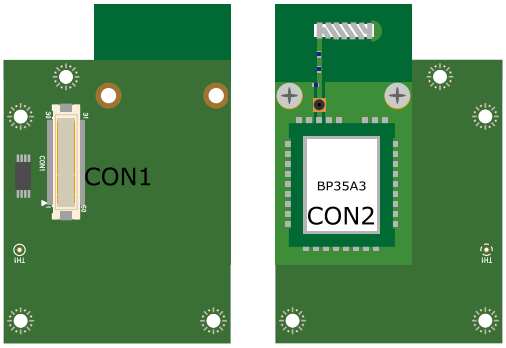


図 18.31 EnOcean アドオンモジュール インターフェースレイアウト

表 18.30 搭載コネクタ、スイッチ型番一覧

部品番号	インターフェース名	型番	メーカー
CON1	アドオンインターフェース	DF17(4.0)-60DP-0.5V(57)	HIROSE ELECTRIC
CON2	EnOcean モジュールインターフェース	AXK6F34347YG-E	Panasonic

18.5.3.2. CON1 アドオンインターフェース

CON1 は Armadillo-X1 アドオンインターフェース(CON7)との接続コネクタです。

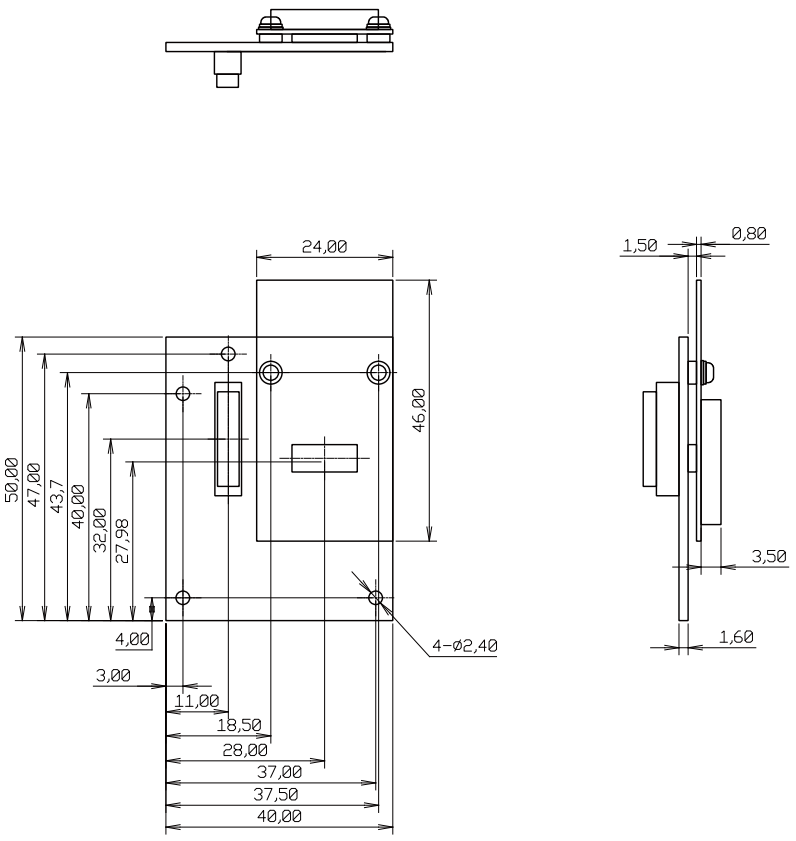
・ 許容電流: 0.3A(端子 1 本あたり)

表 18.31 CON1 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)
2	GND	Power	電源(GND)
3	NC	-	未接続
4	NC	-	未接続
5	NC	-	未接続
6	NC	-	未接続
7	NC	-	未接続

ピン番号	ピン名	I/O	説明
8	NC	-	未接続
9	NC	-	未接続
10	NC	-	未接続
11	NC	-	未接続
12	NC	-	未接続
13	NC	-	未接続
14	NC	-	未接続
15	NC	-	未接続
16	NC	-	未接続
17	NC	-	未接続
18	NC	-	未接続
19	NC	-	未接続
20	EEPROM_SCL	In/Out	EEPROM の SCL ピンに接続
21	EEPROM_SDA	In/Out	EEPROM の SDA ピンに接続
22	NC	-	未接続
23	NC	-	未接続
24	NC	-	未接続
25	NC	-	未接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	+3.3V_IO	Power	電源(+3.3V_IO)
29	NC	-	未接続
30	NC	-	未接続
31	DETECT	In	EEPROM のアドレスピンに接続
32	PROG_EN	In	BP35A3 の 15 ピンに接続
33	NC	-	未接続
34	NC	-	未接続
35	SPI_SCLK	In	BP35A3 の 13 ピンに接続
36	SPI_MISO	Out	BP35A3 の 11 ピンに接続
37	SPI_MOSI	In	BP35A3 の 12 ピンに接続
38	NC	-	未接続
39	NC	-	未接続
40	UART_TXD	In	BP35A3 の 17 ピンに接続
41	UART_RXD	Out	BP35A3 の 16 ピンに接続
42	GPIO2	Out	BP35A3 の 5 ピンに接続
43	NC	-	未接続
44	NC	-	未接続
45	NC	-	未接続
46	NC	-	未接続
47	NC	-	未接続
48	NC	-	未接続
49	NC	-	未接続
50	SPI_SS	In	BP35A3 の 14 ピンに接続
51	NC	-	未接続
52	NC	-	未接続
53	NC	-	未接続
54	GND	Power	電源(GND)
55	NC	-	未接続
56	NC	-	未接続
57	NC	-	未接続
58	GND	Power	電源(GND)
59	NC	-	未接続
60	NC	-	未接続

18.5.4. 基板形状図



[Unit : mm]

図 18.32 EnOcean アドオンモジュール基板形状

18.6. Armadillo-IoT Wi-SUN アドオンモジュール WS00

18.6.1. 概要

Armadillo-IoT Wi-SUN アドオンモジュール WS00(以降、Wi-SUN アドオンモジュールと記載します)は、ROHM 製の BP35A1 を搭載した Wi-SUN モジュールです。

Wi-SUN アドオンモジュールの仕様は次のとおりです。

表 18.32 Wi-SUN アドオンモジュールの仕様

Wi-SUN	ROHM 製 BP35A1 搭載
電源電圧	DC 3.3V±5%
使用温度範囲	-20℃～70℃
基板サイズ	40 x 49mm(突起部を除く)

18.6.2. ブロック図

Wi-SUN アドオンモジュールのブロック図は次のとおりです。

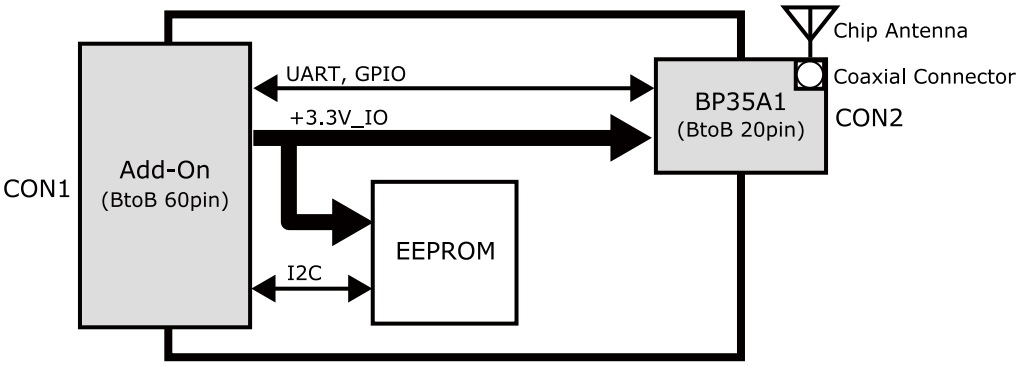


図 18.33 Wi-SUN アドオンモジュール ブロック図

18.6.3. インターフェース仕様

Wi-SUN アドオンモジュールのインターフェース仕様について説明します。

18.6.3.1. Wi-SUN アドオンモジュール インターフェースレイアウト

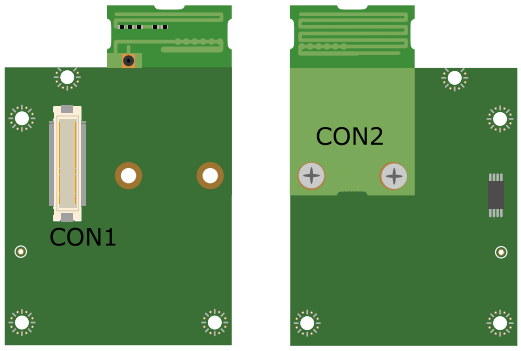


図 18.34 Wi-SUN アドオンモジュール インターフェースレイアウト

表 18.33 搭載コネクタ、スイッチ型番一覧

部品番号	インターフェース名	型番	メーカー
CON1	アドオンインターフェース	DF17(4.0)-60DP-0.5V(57)	HIROSE ELECTRIC
CON2	Wi-SUN モジュールインターフェース	20P3.0-JMCS-G-B-TF(N)	J.S.T. Mfg.

18.6.3.2. CON1 アドオンインターフェース

CON1 は Armadillo-X1 アドオンインターフェース(CON7)との接続コネクタです。

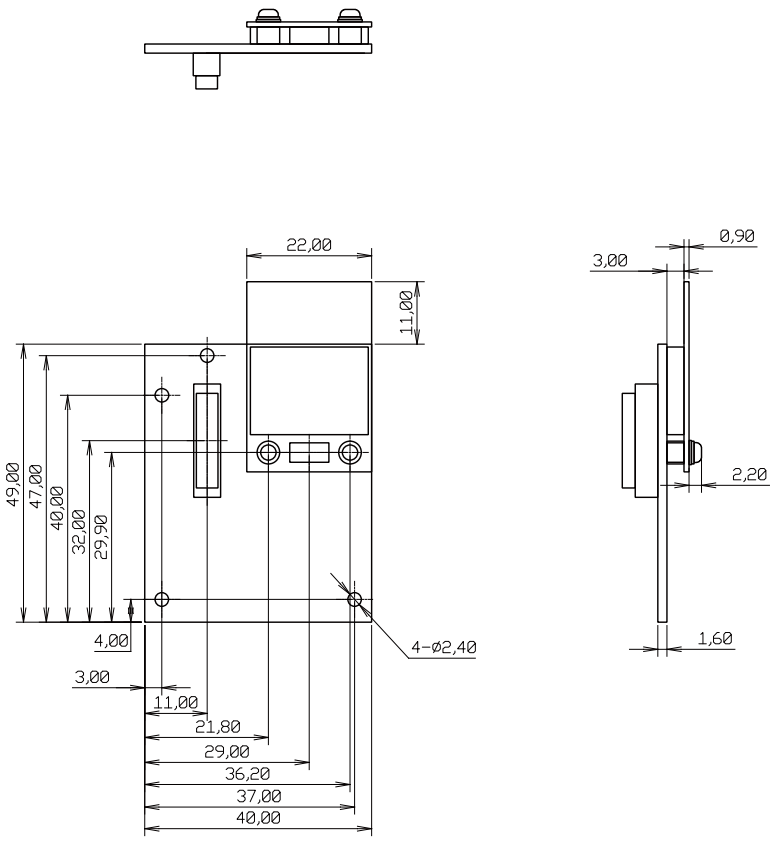
・ 許容電流: 0.3A(端子 1 本あたり)

表 18.34 CON1 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)
2	GND	Power	電源(GND)
3	NC	-	未接続
4	NC	-	未接続
5	NC	-	未接続
6	NC	-	未接続
7	NC	-	未接続

ピン番号	ピン名	I/O	説明
8	NC	-	未接続
9	NC	-	未接続
10	NC	-	未接続
11	NC	-	未接続
12	NC	-	未接続
13	NC	-	未接続
14	NC	-	未接続
15	NC	-	未接続
16	NC	-	未接続
17	NC	-	未接続
18	NC	-	未接続
19	NC	-	未接続
20	EEPROM_SCL	In/Out	EEPROM の SCL ピンに接続
21	EEPROM_SDA	In/Out	EEPROM の SDA ピンに接続
22	NC	-	未接続
23	NC	-	未接続
24	NC	-	未接続
25	NC	-	未接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	+3.3V_IO	Power	電源(+3.3V_IO)
29	NC	-	未接続
30	NC	-	未接続
31	DETECT	In	EEPROM のアドレスピンに接続
32	NC	-	未接続
33	NC	-	未接続
34	NC	-	未接続
35	NC	-	未接続
36	NC	-	未接続
37	NC	-	未接続
38	UART_RTS	In	BP35A1 の 14 ピンに接続
39	UART_CTS	Out	BP35A1 の 15 ピンに接続
40	UART_TXD	In	BP35A1 の 4 ピンに接続
41	UART_RXD	Out	BP35A1 の 3 ピンに接続
42	GPIO2	Out	BP35A1 の 6 ピンに接続
43	GPIO3	Out	BP35A1 の 5 ピンに接続
44	NC	-	未接続
45	NC	-	未接続
46	NC	-	未接続
47	NC	-	未接続
48	NC	-	未接続
49	NC	-	未接続
50	NC	-	未接続
51	NC	-	未接続
52	NC	-	未接続
53	NC	-	未接続
54	GND	Power	電源(GND)
55	NC	-	未接続
56	NC	-	未接続
57	NC	-	未接続
58	GND	Power	電源(GND)
59	NC	-	未接続
60	NC	-	未接続

18.6.4. 基板形状図



[Unit : mm]

図 18.35 Wi-SUN アドオンモジュール基板形状

18.7. Armadillo-IoT 絶縁デジタル入出力/アナログ入力アドオンモジュール DA00

18.7.1. 概要

Armadillo-IoT 絶縁デジタル入出力/アナログ入力アドオンモジュール DA00(以降、絶縁 IO アドオンモジュールと記載します)は、電気的に絶縁されたデジタル入力 2 ポート、デジタル出力 2 ポートと 0〜5V のアナログ入力 2 ポートを追加することができます。

絶縁 IO アドオンモジュールの仕様は次のとおりです。

表 18.35 絶縁 IO アドオンモジュールの仕様

デジタル入力	入力点数	2 点
	定格入力電圧	DC 3.3～48V
	許容入力電圧	DC 3.15～52.8V
	入力インピーダンス	1kΩ
	入力電流	3.8mA Typ.(ON 時)
	応答時間	1ms 以内
	ON 電圧	ショート(または 0.6V 以下)
	OFF 電圧	オープン(または 3.15V 以上)
	絶縁耐圧	2kV
デジタル出力	出力点数	2 点
	定格電圧	48V
	応答時間	2ms 以内
	出力形式	無極性
	絶縁耐圧	2kV
アナログ入力	AD コンバータ	Microchip 製 MCP3202 搭載
	入力点数	2 点
	入力電圧	0～5V
	入力インピーダンス	10MΩ
	分解能	12bit
	精度	±1%
電源電圧		DC 3.3V±5%
使用温度範囲		-20℃～70℃
基板サイズ		40 x 63mm(突起部を除く)

18.7.2. ブロック図

絶縁 IO アドオンモジュールのブロック図は次のとおりです。

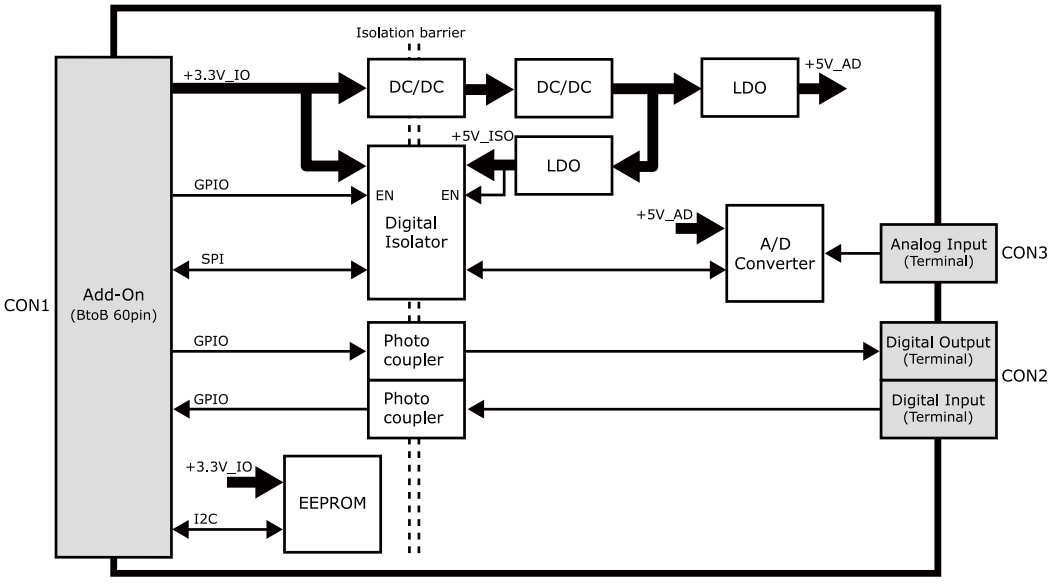


図 18.36 絶縁 IO アドオンモジュール ブロック図

18.7.3. インターフェース仕様

絶縁 IO アドオンモジュールのインターフェース仕様について説明します。

18.7.3.1. 絶縁 IO アドオンモジュール インターフェースレイアウト

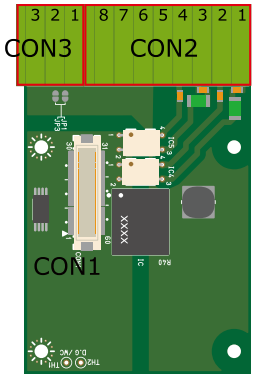


図 18.37 絶縁 IO アドオンモジュール インターフェースレイアウト

表 18.36 搭載コネクタ、スイッチ型番一覧

部品番号	インターフェース名	型番	メーカー
CON1	アドオンインターフェース	DF17(4.0)-60DP-0.5V(57)	HIROSE ELECTRIC
CON2	デジタル入出力インターフェース	XW4C-08D1-H1	OMRON
CON3	アナログ入力インターフェース	XW4C-03D1-H1	OMRON

絶縁 IO アドオンモジュールの固定穴(TH5、TH6)の PAD 部分は GND に接続されています。固定穴(TH3、TH4)はキリ穴で GND に接続されていません。

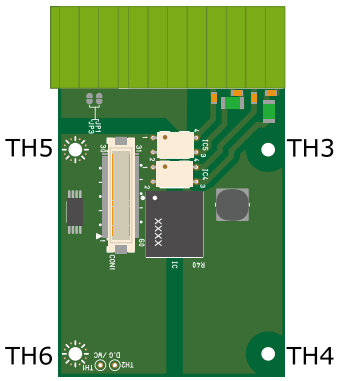


図 18.38 絶縁 IO アドオンモジュールの固定穴

18.7.3.2. CON1 アドオンインターフェース

CON1 は Armadillo-X1 アドオンインターフェース(CON7)との接続コネクタです。

- 許容電流: 0.3A(端子 1 本あたり)

表 18.37 CON1 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)

ピン番号	ピン名	I/O	説明
2	GND	Power	電源(GND)
3	NC	-	未接続
4	NC	-	未接続
5	NC	-	未接続
6	NC	-	未接続
7	NC	-	未接続
8	NC	-	未接続
9	NC	-	未接続
10	NC	-	未接続
11	NC	-	未接続
12	NC	-	未接続
13	NC	-	未接続
14	NC	-	未接続
15	NC	-	未接続
16	NC	-	未接続
17	NC	-	未接続
18	NC	-	未接続
19	NC	-	未接続
20	EEPROM_SCL	In/Out	EEPROM の SCL ピンに接続
21	EEPROM_SDA	In/Out	EEPROM の SDA ピンに接続
22	NC	-	未接続
23	NC	-	未接続
24	DO1	In	CON2 の DO1 制御ピンに接続 (Low: DO1 オープン、High: DO1 ショート)
25	DO2	In	CON2 の DO2 制御ピンに接続 (Low: DO2 オープン、High: DO2 ショート)
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	+3.3V_IO	Power	電源(+3.3V_IO)
29	NC	-	未接続
30	NC	-	未接続
31	DETECT	In	EEPROM のアドレスピンに接続
32	NC	-	未接続
33	NC	-	未接続
34	NC	-	未接続
35	ADC_CLK	In	デジタルアイソレータを経由して AD コンバーターに接続
36	ADC_DOUT	Out	デジタルアイソレータを経由して AD コンバーターに接続
37	ADC_DIN	In	デジタルアイソレータを経由して AD コンバーターに接続
38	NC	-	未接続
39	NC	-	未接続
40	NC	-	未接続
41	NC	-	未接続
42	NC	-	未接続
43	ISOLATOR_VE1	In	デジタルアイソレータのイネーブルピンに接続
44	NC	-	未接続
45	NC	-	未接続
46	NC	-	未接続
47	DI2	Out	デジタル入力 2
48	DI1	Out	デジタル入力 1
49	NC	-	未接続
50	CS*/SHDN	In	デジタルアイソレータを経由して AD コンバーターに接続
51	NC	-	未接続
52	NC	-	未接続
53	NC	-	未接続

ピン番号	ピン名	I/O	説明
54	GND	Power	電源(GND)
55	NC	-	未接続
56	NC	-	未接続
57	NC	-	未接続
58	GND	Power	電源(GND)
59	NC	-	未接続
60	NC	-	未接続

18.7.3.3. CON2 デジタル入出力インターフェース

CON2 は入力を 2 点、出力を 2 点もつデジタル入出力インターフェースです。

デジタル入力部はフォトカプラによる絶縁入力(電流シンク出力)となっています。入力部を駆動するための電源を内蔵しており、外部電源の接続は不要です。

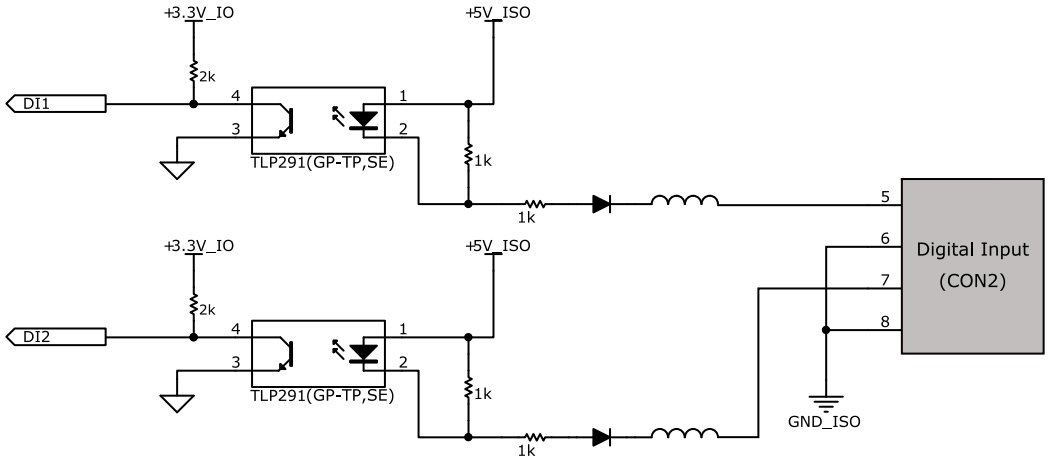


図 18.39 CON2 デジタル入力部

デジタル出力部はフォトリレーによる絶縁出力(無極性)となっています。出力部を駆動するためには外部に電源が必要となります。出力 1 点につき最大電流 200mA(定格 48V)まで駆動可能です。

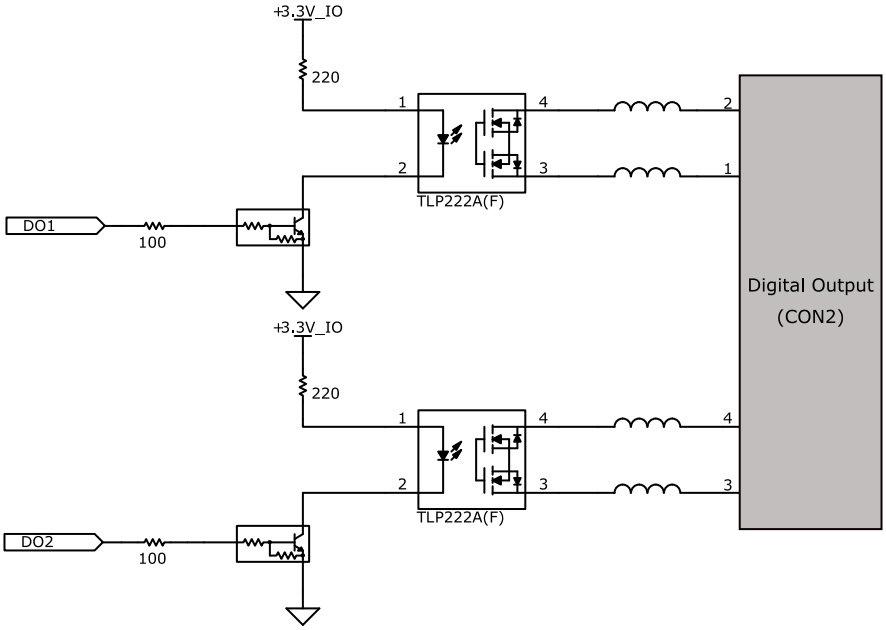


図 18.40 CON2 デジタル出力部

表 18.38 CON2 信号配列

ピン番号	信号名	I/O	機能
1	DO1A	-	デジタル出力 1A
2	DO1B	-	デジタル出力 1B
3	DO2A	-	デジタル出力 2A
4	DO2B	-	デジタル出力 2B
5	DI1	In	デジタル入力 1
6	GND_ISO	Power	電源(GND_ISO)
7	DI2	In	デジタル入力 2
8	GND_ISO	Power	電源(GND_ISO)

18.7.3.4. CON3 アナログ入力インターフェース

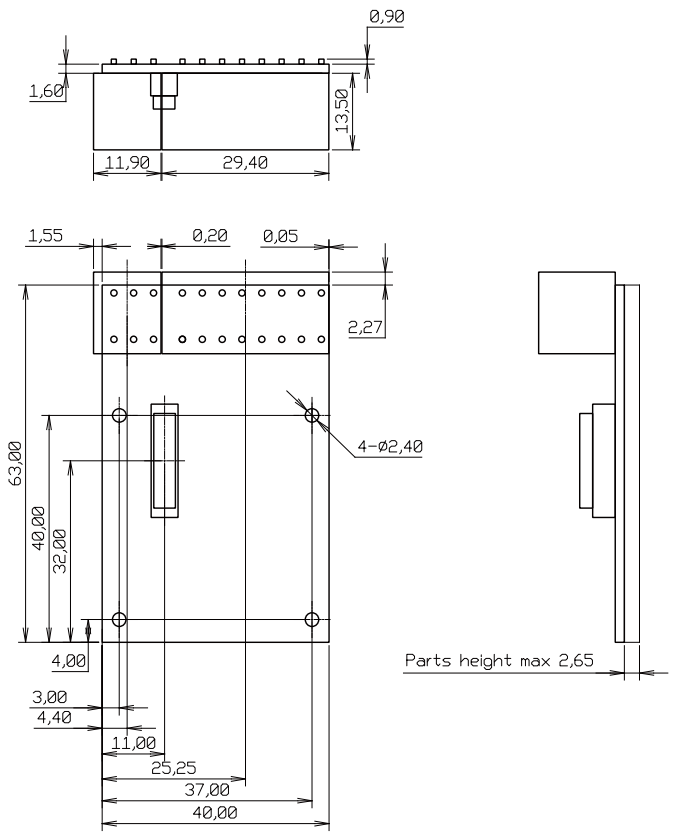
アナログ入力部はデジタルアイソレータによる絶縁入力となっています。入力レンジは 0〜5V で、シングルエンド入力(2CH)もしくは疑似差動入力(1CH)が可能です。

- 入力電圧: 0〜5V
- 入力インピーダンス: 10MΩ
- 分解能: 12bit
- 精度: 1%

表 18.39 CON3 信号配列

ピン番号	信号名	I/O	機能
1	ADC_CH0	In	アナログ入力 CH0
2	GND_ISO	Power	電源(GND_ISO)
3	ADC_CH1	In	アナログ入力 CH1

18.7.4. 基板形状図



[Unit : mm]

図 18.41 絶縁 IO アドオンモジュール基板形状

18.7.5. 使用方法

デジタル入出力インターフェース(CON2)、アナログ入力インターフェース(CON3)に実装されている端子台に接続可能な電線は次のとおりです。

表 18.40 端子台に接続可能な電線

単線		0.2～1.5mm^2
撚線		0.2～1.5mm^2
棒端子	スリーブなし	0.25～1.5mm^2
	スリーブあり	0.25～0.75mm^2
AWG		24～16

電線を直接接続する場合、先端加工は次のとおりです。電線むき長さ L は 10±1mm となります。

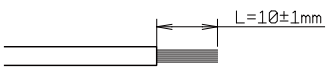


図 18.42 電線の先端加工

電線の先端を予備半田しないでください。正しい接続ができなくなります。

棒端子を使用する場合、使用する棒端子に合わせて電線加工を行ってください。棒端子のサイズは次のとおりです。

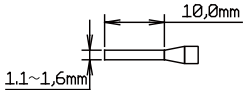


図 18.43 棒端子のサイズ

端子台に電線を接続する際、端子台に過度な力をかけないでください。端子台が破損する恐れがあります。

デジタル入力

デジタル入力は 2 点あり、CON2 の 5 ピン(DI1)、6 ピン(GND_ISO)の組み合わせ、CON2 の 7 ピン(DI2)、8 ピン(GND_ISO)の組み合わせで使用します。デジタル入力には、無電圧接点、有電圧接点を接続可能です。

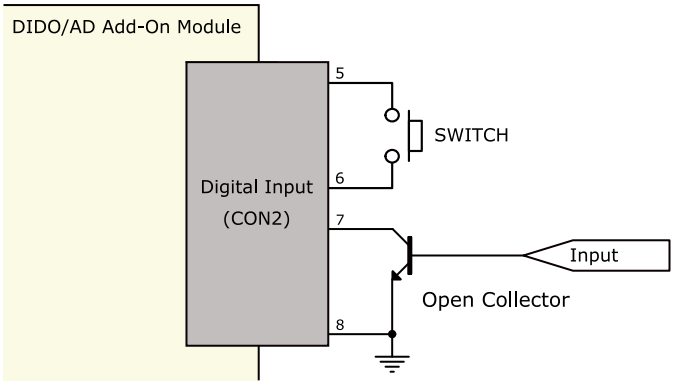


図 18.44 デジタル入力接続例

デジタル出力

デジタル出力は 2 点あり、CON2 の 1 ピン(DO1A)、2 ピン(DO1B)の組み合わせ、CON2 の 3 ピン(DO2A)、4 ピン(DO2B)の組み合わせで使用します。

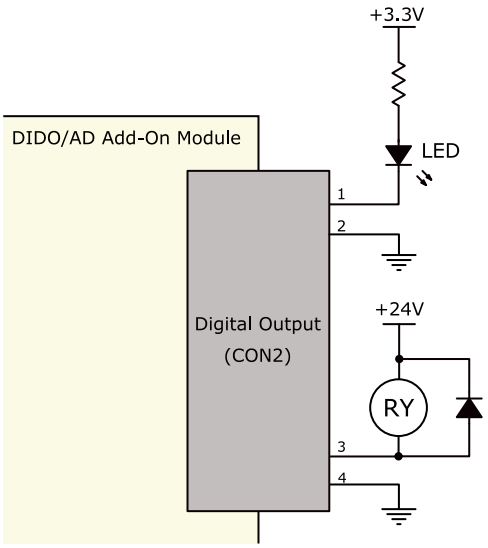


図 18.45 デジタル出力接続例

過電流、過電圧保護のためのヒューズ等は基板上に実装されておりません。必要に応じて外部で対策を行ってください。

アナログ入力

アナログ入力は、シングルエンド入力と疑似差動入力が可能です。シングルエンド入力で使用する場合は、CON3 の 1 ピン(ADC_CH0)、2 ピン(GND_ISO)の組み合わせ、CON3 の 3 ピン(ADC_CH1)、2 ピン(GND_ISO)の組み合わせで使用します。疑似差動入力で使用する場合は、CON3 の 1 ピン(ADC_CH0)、2 ピン(GND_ISO)、3 ピン(ADC_CH1)の組み合わせで使用します。

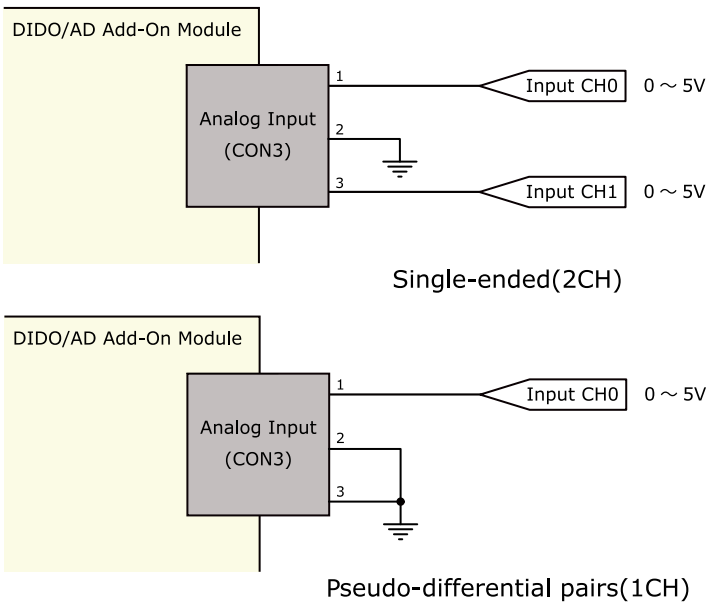


図 18.46 アナログ入力接続例

ESD/雷サージ

接続ケーブルが屋外に露出するような設置環境では、ケーブルに侵入した雷サージ等のストレスによりインターフェース回路が破壊される場合があります。ストレスへの耐性を向上させるには、各端子とアース間にアレスタ、バリスタ等の保護素子を接続することが効果的です。

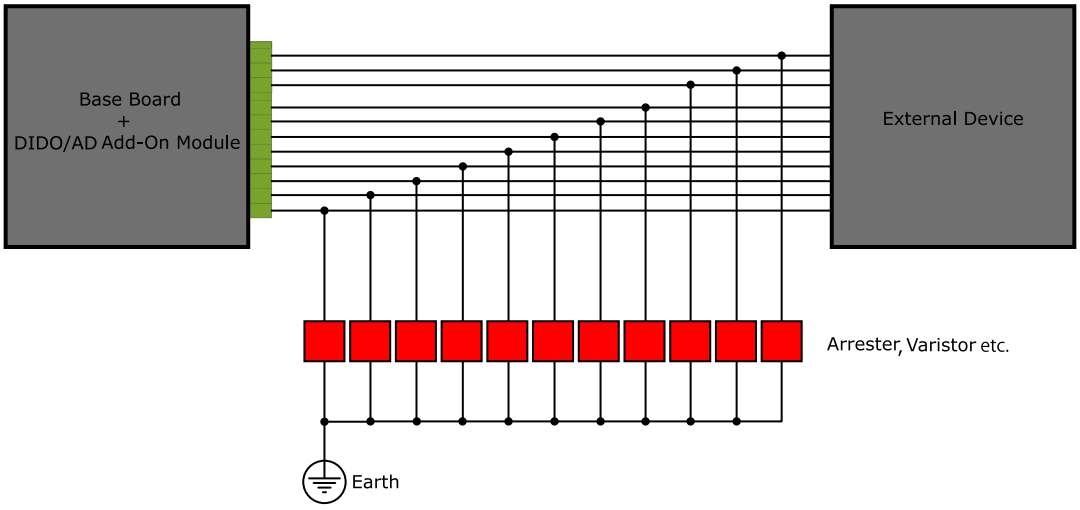


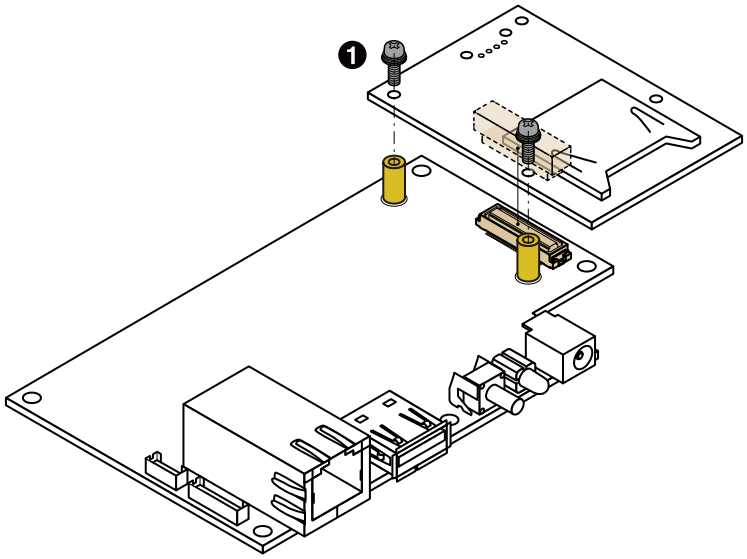
図 18.47 保護素子の接続例

絶縁 IO アドオンモジュールの電源を再投入する場合は 10 秒以上の間隔をあけてください。コンデンサに蓄えられた電荷が抜ける前に電源を再投入すると、絶縁 IO アドオンモジュールの電源シーケンスが守られず、故障の原因となる可能性があります。

信号品質の低下、故障を防ぐため、配線、接地などの設置環境に十分にご配慮ください。

18.8. 組み立て

Armadillo-IoT シリーズのアドオンモジュールは、Armadillo-X1 の CON7 に接続することが可能です。接続方法は次のとおりです。



❶ なべ小ねじ スプリングワッシャー、小径平ワッシャー付(M2、L=6mm)×2

図 18.48 アドオンモジュールの接続

19. オプション品

本章では、Armadillo-X1 関連のオプション品について説明します。

表 19.1 Armadillo-X1 関連のオプション品

名称	型番	備考
SD スロット拡張ボード	-	Armadillo-X1 開発セットに付属
100 ピンコネクタ ピッチ変換基板	-	Armadillo-X1 開発セットに付属
100 ピンコネクタ 延長ケーブル	-	Armadillo-X1 開発セットに付属
USB シリアル変換アダプタ	SA-SCUSB-00	Armadillo-X1 開発セットに付属
8 ピン JTAG 変換ケーブル	OP-JC8P25-00	
AC アダプタ(5V/2.0A EIAJ#2)	OP-AC5V4-10	Armadillo-X1 開発セットに付属
アンテナ固定金具	-	Armadillo-X1 開発セットに付属
WLAN+BT コンボモジュール 外付けアンテナセット	-	Armadillo-X1 開発セットに付属
920MHz 帯 外付けアンテナセット 02	OP-ANT-920-02K	

外付けアンテナセット以外のオプション品は、試作・開発用の製品です。
これらは外観や仕様を予告なく変更する場合があります。

19.1. SD スロット拡張ボード

19.1.1. 概要

SD スロット拡張ボードは、SD インターフェースを 1 ポート追加することができます。SD スロット拡張ボードの仕様は次のとおりです。

表 19.2 SD スロット拡張ボードの仕様

SD	UHS-I(SDR50、最大クロック周波数 100MHz)対応
スイッチ	起動デバイス設定用スライドスイッチ
電源電圧	DC 3.3V±5%
使用温度範囲	-20℃～70℃
基板サイズ	40 x 62mm(突起部を除く)

19.1.2. ブロック図

SD スロット拡張ボードのブロック図は次のとおりです。

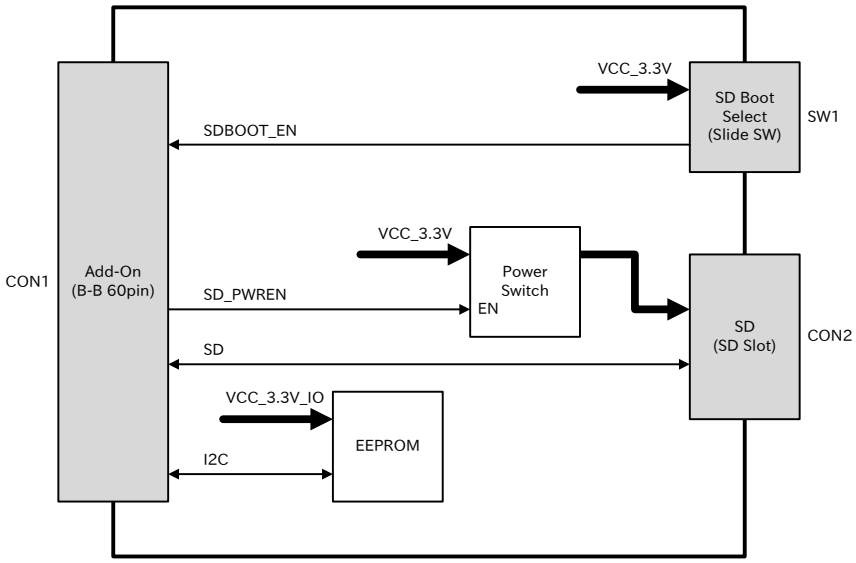


図 19.1 SD スロット拡張ボード ブロック図

19.1.3. インターフェース仕様

SD スロット拡張ボードのインターフェース仕様について説明します。

19.1.3.1. インターフェースレイアウト

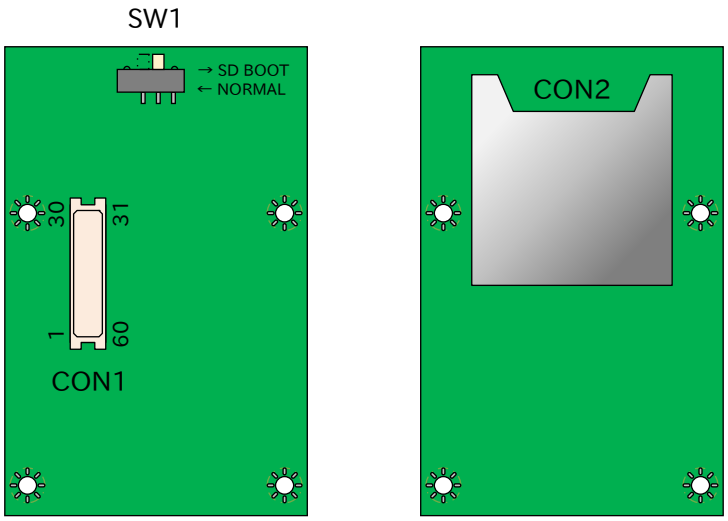


図 19.2 SD スロット拡張ボード インターフェースレイアウト

表 19.3 搭載コネクタ、スイッチ型番一覧

部品番号	インターフェース名	型番	メーカー
CON1	アドオンインターフェース	DF17(4.0)-60DP-0.5V(57)	HIROSE ELECTRIC
CON2	SD インターフェース	CIM-K03NS	MITSUMI ELECTRIC
SW1	起動デバイス設定スイッチ	CSS-1210TB	NIDEC COPAL ELECTRONICS

19.1.3.2. CON1 アドオンインターフェース

CON1 は Armadillo-X1 アドオンインターフェース(CON7)との接続コネクタです。

・ 許容電流: 0.3A 以下(端子 1 本あたり)

表 19.4 CON1 信号配列

ピン番号	ピン名	I/O	説明
1	GND	Power	電源(GND)
2	GND	Power	電源(GND)
3	VCC_3.3V	Power	電源(VCC_3.3V)
4	VCC_3.3V	Power	電源(VCC_3.3V)
5	SDBOOT_EN	Out	起動デバイス設定、SW1 に接続 (Low: SPI フラッシュメモリブート、High: SD ブート)
6	NVCC_SD1	Power	SD1 信号電源(NVCC_SD1)
7	SD_CD_B	Out	カード検出、CON2 に接続、基板上で 15kΩ プルアップ(NVCC_SD1)されています (Low: カード挿入、High: カード未挿入)
8	SD_WP	Out	ライトプロテクト検出、CON2 に接続、基板上で 15kΩ プルアップ(NVCC_SD1)されています (Low: 書き込み可能、High: 書き込み不可能)
9	SD_PWREN	In	SD カード電源制御、基板上で 1kΩ プルアップ(NVCC_SD1)されています (Low: 電源切断、High: 電源供給)
10	SD_CLK	In	SD クロック、CON2 の 5 ピンに接続
11	SD_CMD	In/Out	SD コマンド/レスポンス、CON2 の 2 ピンに接続
12	SD_DAT0	In/Out	SD データバス(bit0)、CON2 の 7 ピンに接続
13	SD_DAT1	In/Out	SD データバス(bit1)、CON2 の 8 ピンに接続
14	SD_DAT2	In/Out	SD データバス(bit2)、CON2 の 9 ピンに接続
15	SD_DAT3	In/Out	SD データバス(bit3)、CON2 の 1 ピンに接続
16	NC	-	未接続
17	NC	-	未接続
18	NC	-	未接続
19	NC	-	未接続
20	EEPROM_SCL	In/Out	EEPROM の SCL ピンに接続、基板上で 4.7kΩ プルアップ(VCC_3.3V_IO)されています
21	EEPROM_SDA	In/Out	EEPROM の SDA ピンに接続、基板上で 4.7kΩ プルアップ(VCC_3.3V_IO)されています
22	NC	-	未接続
23	NC	-	未接続
24	NC	-	未接続
25	NC	-	未接続
26	GND	Power	電源(GND)
27	GND	Power	電源(GND)
28	VCC_3.3V_IO	Power	電源(VCC_3.3V_IO)
29	NC	-	未接続
30	NC	-	未接続
31	EEPROM_E0	In	EEPROM のアドレスピンに接続
32	NC	-	未接続
33	NC	-	未接続
34	NC	-	未接続
35	NC	-	未接続
36	NC	-	未接続
37	NC	-	未接続
38	NC	-	未接続
39	NC	-	未接続
40	NC	-	未接続
41	NC	-	未接続
42	NC	-	未接続
43	NC	-	未接続

ピン番号	ピン名	I/O	説明
44	NC	-	未接続
45	NC	-	未接続
46	NC	-	未接続
47	NC	-	未接続
48	NC	-	未接続
49	NC	-	未接続
50	NC	-	未接続
51	NC	-	未接続
52	NC	-	未接続
53	NC	-	未接続
54	GND	Power	電源(GND)
55	NC	-	未接続
56	NC	-	未接続
57	NC	-	未接続
58	GND	Power	電源(GND)
59	NC	-	未接続
60	NC	-	未接続

19.1.3.3. CON2 SD インターフェース

CON2 は、UHS-I(SDR50、最大クロック周波数: 100MHz)に対応した SD インターフェースです。

SD カードに供給される電源(SD_VDD)は、SD_PWREN ピンで制御が可能です。High レベル出力で電源が供給され、Low レベル出力で電源が切断されます。

表 19.5 CON2 信号配列

ピン番号	ピン名	I/O	説明
1	SD_DAT3	In/Out	SD データバス(bit3)、CON1 の 15 ピンに接続、基板上で 15kΩ プルアップ(NVCC_SD1)されています
2	SD_CMD	In/Out	SD コマンド/レスポンス、CON1 の 11 ピンに接続、基板上で 15kΩ プルアップ(NVCC_SD1)されています
3	GND	Power	電源(GND)
4	SD_VDD	Power	電源(SD_VDD)
5	SD_CLK	Out	SD クロック、CON1 の 10 ピンに接続
6	GND	Power	電源(GND)
7	SD_DAT0	In/Out	SD データバス(bit0)、CON1 の 12 ピンに接続、基板上で 15kΩ プルアップ(NVCC_SD1)されています
8	SD_DAT1	In/Out	SD データバス(bit1)、CON1 の 13 ピンに接続、基板上で 15kΩ プルアップ(NVCC_SD1)されています
9	SD_DAT2	In/Out	SD データバス(bit2)、CON1 の 14 ピンに接続、基板上で 15kΩ プルアップ(NVCC_SD1)されています

障害や破損を引き起こす場合がありますので、コネクタに過大な外力を加えないようにしてください。

19.1.3.4. SW1 起動デバイス設定スイッチ

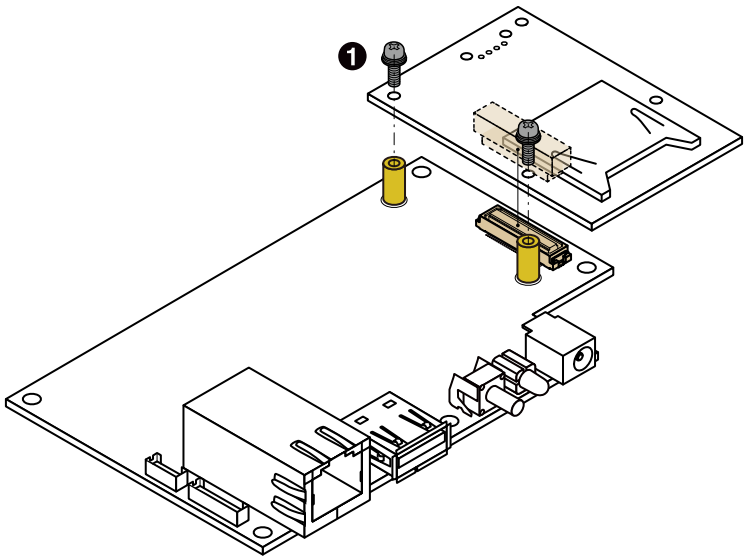
SW1 は起動デバイスの設定を行うスライドスイッチです。スイッチの切り替え位置は、SW1 付近の基板シルクで確認してください。

表 19.6 スライドスイッチの機能

部品番号	機能	動作
SW1	起動デバイス設定	[→ SD BOOT]: CON2 に挿入された SD カードのブートローダーを起動します。 [← NORMAL]: SPI フラッシュメモリのブートローダーを起動します。

19.1.4. 組み立て

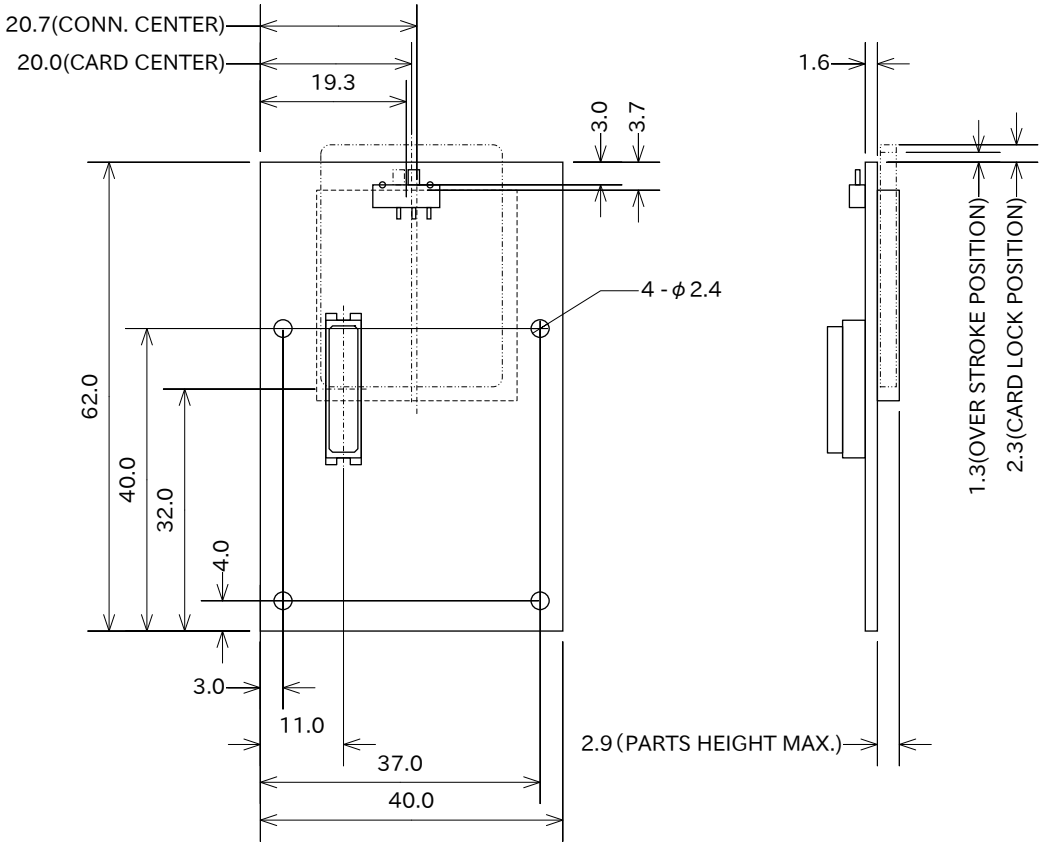
SD スロット拡張ボードは、Armadillo-X1 の CON7 に接続することが可能です。接続方法は次のとおりです。



❶　なべ小ねじ スプリングワッシャー、小径平ワッシャー付(M2、L=6mm)×2

図 19.3 SD スロット拡張ボードの接続

19.1.5. 基板形状図



[Unit : mm]

図 19.4 SD スロット拡張ボードの基板形状

19.2. 100 ピンコネクタ ピッチ変換基板

19.2.1. 概要

100 ピンコネクタ ピッチ変換基板は、Armadillo-X1 の拡張インターフェース(CON8)を 2.54mm ピッチに変換するための基板です。拡張ボード開発の事前評価等に、100 ピンコネクタ 延長ケーブルと合わせて使用することが可能です。

19.2.2. インターフェースレイアウト

100 ピンコネクタ ピッチ変換基板のインターフェースレイアウトは次のとおりです。

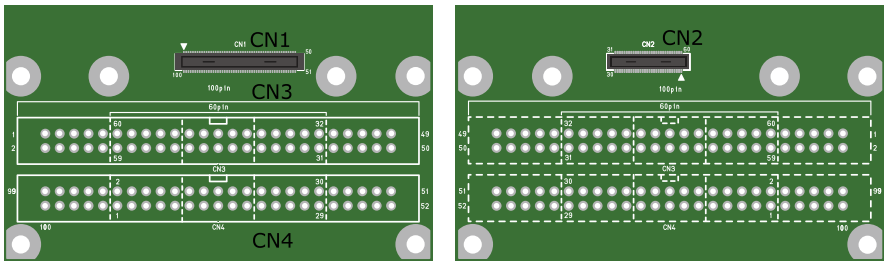


図 19.5 100 ピンコネクタ ピッチ変換基板 インターフェースレイアウト図

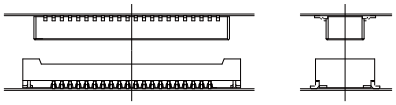
表 19.7 インターフェース内容

部品番号	形状	型番	メーカー	備考
CN1	B-B コネクタ 100 ピン(0.4mm ピッチ)	DF40HC(3.0)-100DS-0.4V(51)	HIROSE ELECTRIC	
CN2	B-B コネクタ 60 ピン (0.4mm ピッチ)	DF40HC(3.0)-60DS-0.4V(51)	HIROSE ELECTRIC	Armadillo-X1 では使用しません
CN3	ピンヘッダ 50 ピン (2.54mm ピッチ)	XG4C-5031	OMRON	コネクタ非搭載、実装可能なコネクタ型番を記載しています
CN4				

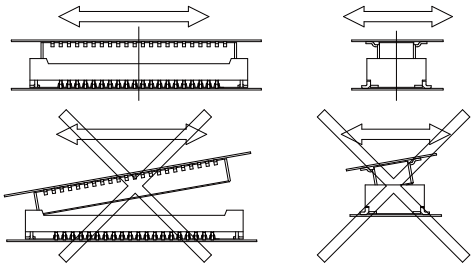
コネクタは 1 ピン(△印)を合わせて接続してください。

コネクタ嵌合時の注意

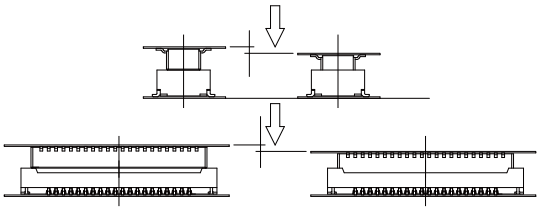
コネクタの中心を合わせて嵌合してください。



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。

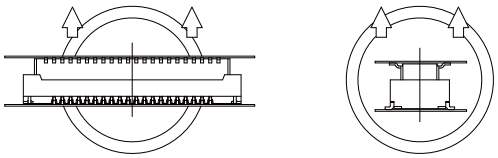


コネクタが誘い込まれると、コネクタ間の距離が近くなり、平行になって前後左右に動かなくなります。この状態からまっすぐに嵌合してください。

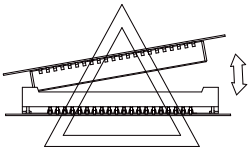


コネクタ抜去時の注意

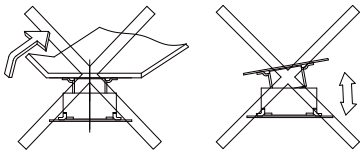
コネクタは平行に抜去してください。



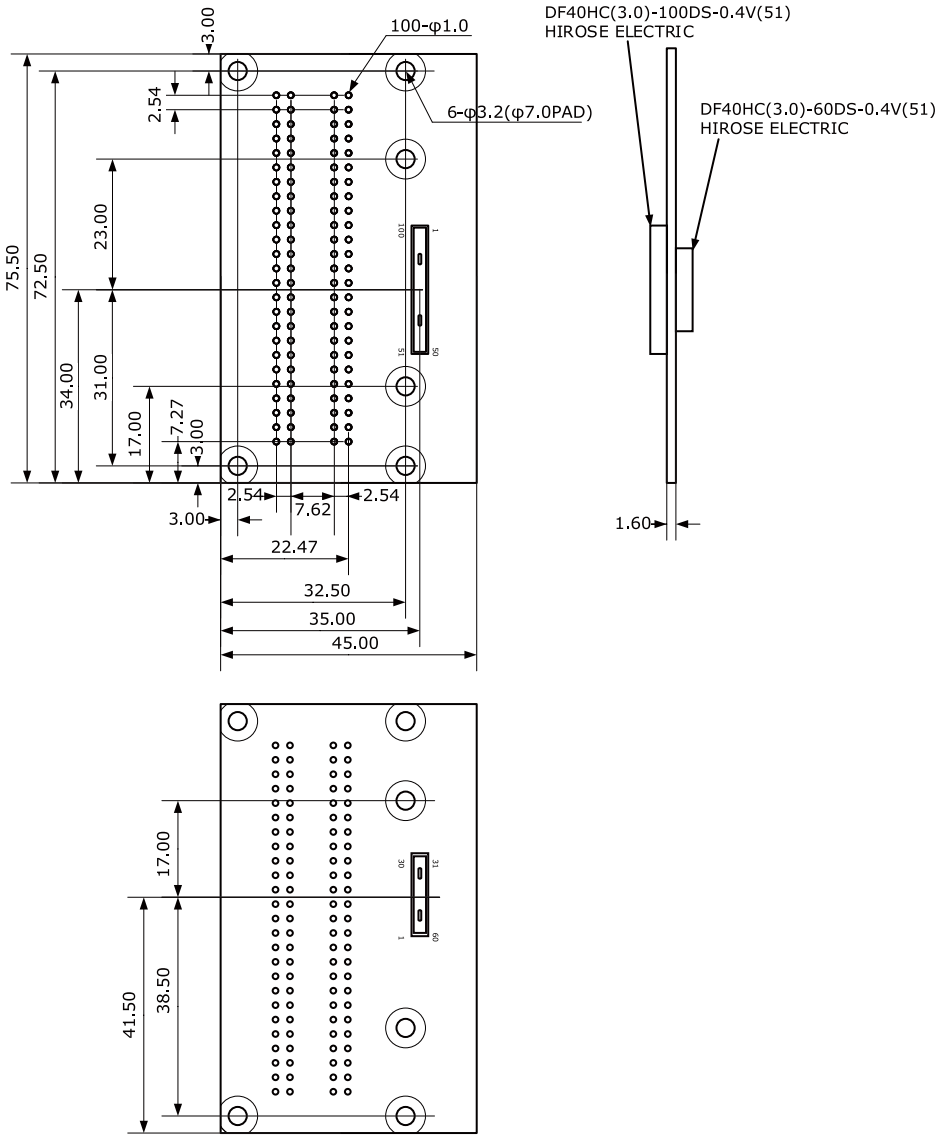
平行に抜去することが困難な場合、コネクタ幅の狭い方向から斜めに抜去してください。



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。



19.2.3. 基板形状図



[Unit : mm]

図 19.6 100 ピンコネクタ ピッチ変換基板 形状図

19.3. 100 ピンコネクタ 延長ケーブル

19.3.1. 概要

100 ピンコネクタ 延長ケーブルは、Armadillo-X1 の拡張インターフェース(CON8)の延長ケーブルです。拡張ボード開発の事前評価等に、100 ピンコネクタ ピッチ変換基板と合わせて使用することが可能です。

19.3.2. インターフェースレイアウト

100 ピンコネクタ 延長ケーブルのインターフェースレイアウトは次のとおりです。

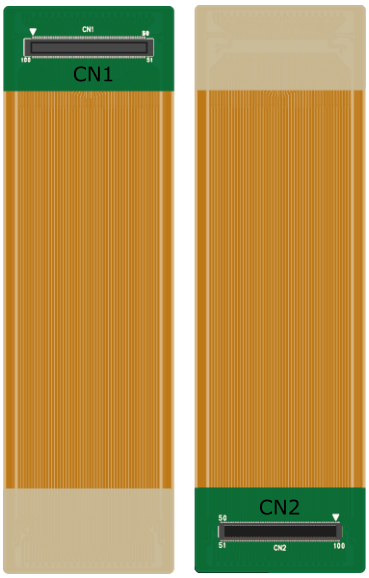


図 19.7 100 ピンコネクタ延長ケーブル インターフェースレイアウト図

表 19.8 インターフェース内容

部品番号	形状	型番	メーカー
CN1	B-B コネクタ 100 ピン(0.4mm ピッチ)	DF40HC(3.0)-100DS-0.4V(51)	HIROSE ELECTRIC
CN2	B-B コネクタ 100 ピン(0.4mm ピッチ)	DF40C(3.0)-100DP-0.4V(51)	HIROSE ELECTRIC

19.3.3. 組み立て

100 ピンコネクタ 延長ケーブルは、Armadillo-X1 の拡張インターフェース(CON8)に接続可能です。組み立て方法は次のとおりです。

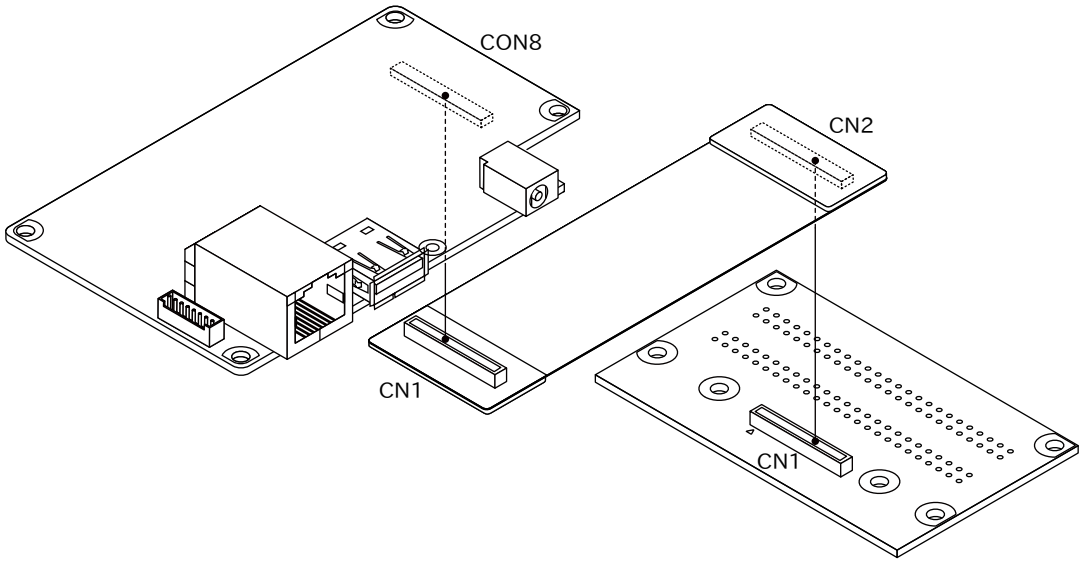
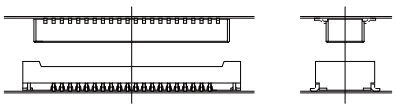


図 19.8 Armadillo-X1 CON8 に 100 ピンコネクタ 延長ケーブルを接続

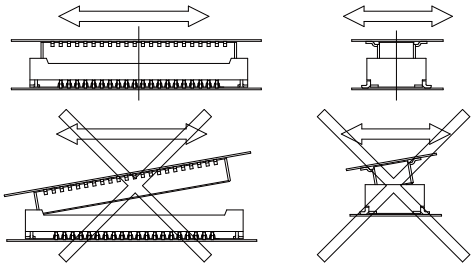
コネクタは 1 ピン(△印)を合わせて接続してください。

コネクタ嵌合時の注意

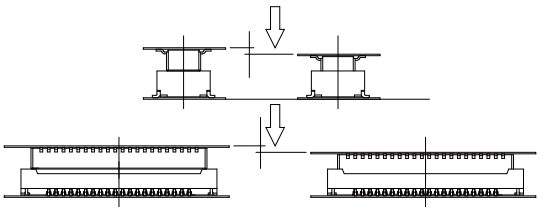
コネクタの中心を合わせて嵌合してください。



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。

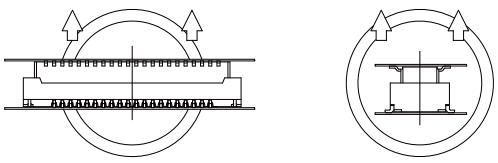


コネクタが誘い込まれると、コネクタ間の距離が近くなり、平行になって前後左右に動かなくなります。この状態からまっすぐに嵌合してください。

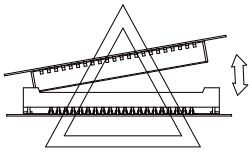


コネクタ抜去時の注意

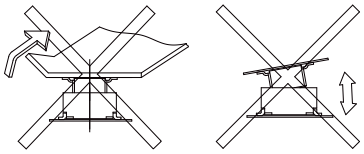
コネクタは平行に抜去してください。



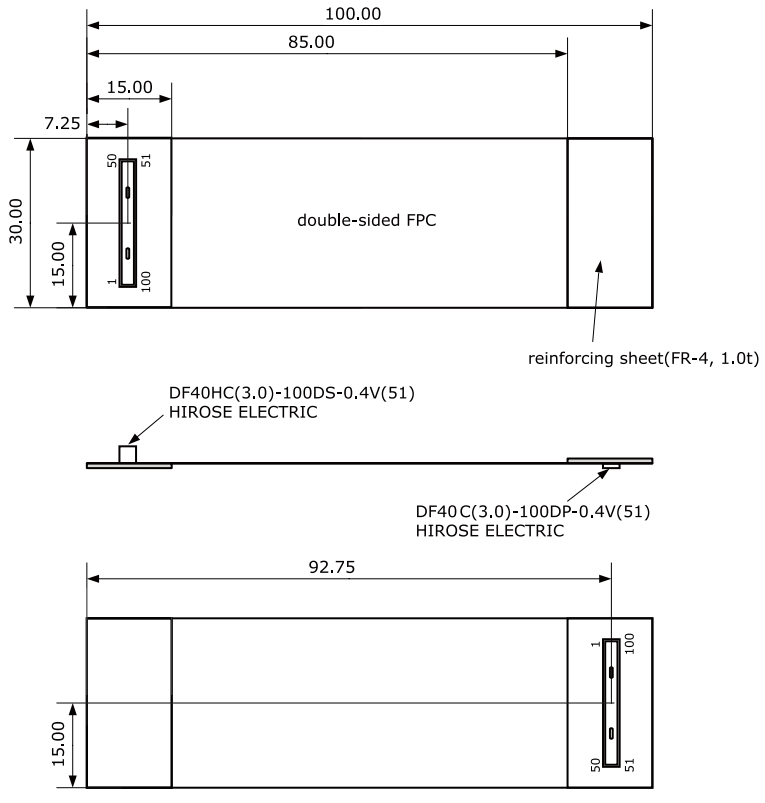
平行に抜去することが困難な場合、コネクタ幅の狭い方向から斜めに抜去してください。



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。



19.3.4. ケーブル形状図

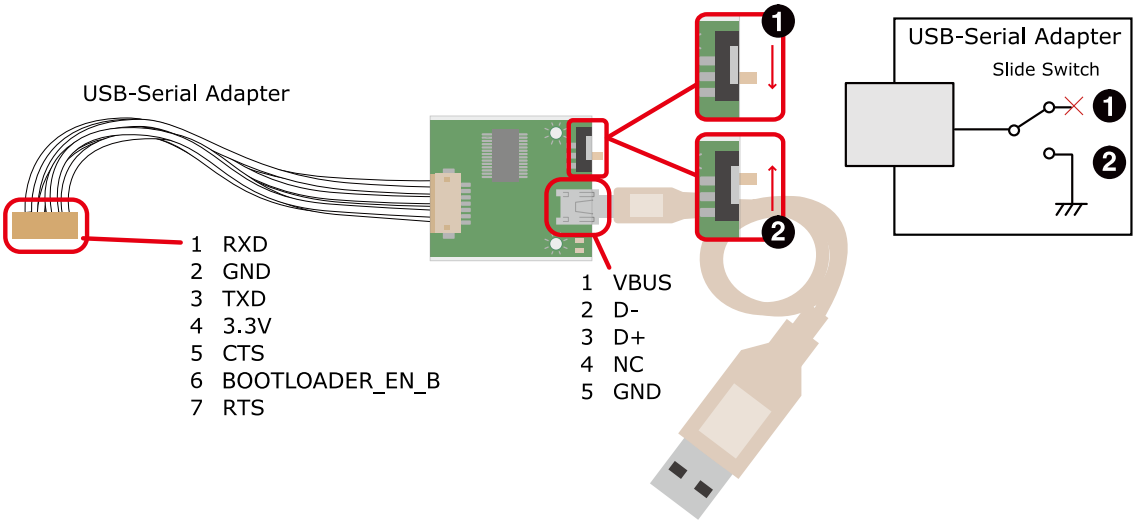


[Unit : mm]

図 19.9 100 ピンコネクタ 延長ケーブル 形状図

19.4. USB シリアル変換アダプタ

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-X1 のシリアルインターフェース(CON4)に接続して使用することが可能です。スライドスイッチが実装されており、信号線の接続先を切替することができます。



❶　OS 自動起動モード

❷　保守モード

図 19.10 USB シリアル変換アダプタの配線

19.5. 8 ピン JTAG 変換ケーブル

8 ピン JTAG 変換ケーブルは JTAG インターフェースを ARM 標準コネクタ(20 ピン、2.54mm ピッチ)に変換するケーブルです。Armadillo-X1 の JTAG インターフェース(CON6)に接続して使用することが可能です。

8 ピン JTAG 変換ケーブルの接続図、参考回路を以下に示します。

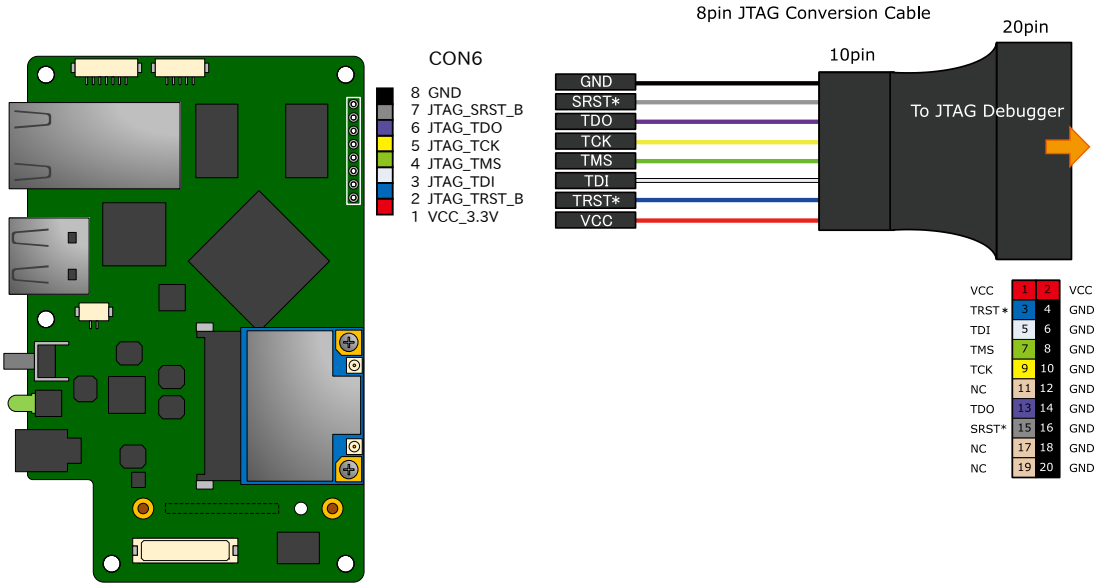


図 19.11 8 ピン JTAG 変換ケーブルの接続図

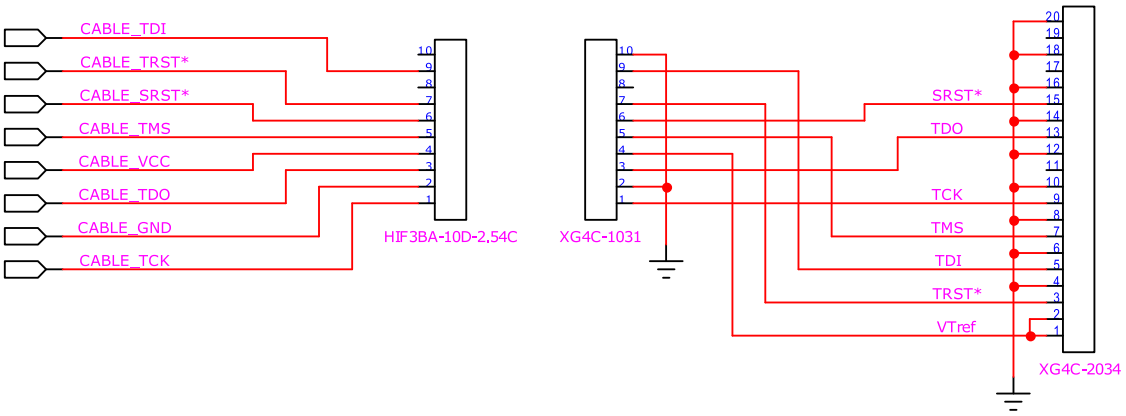
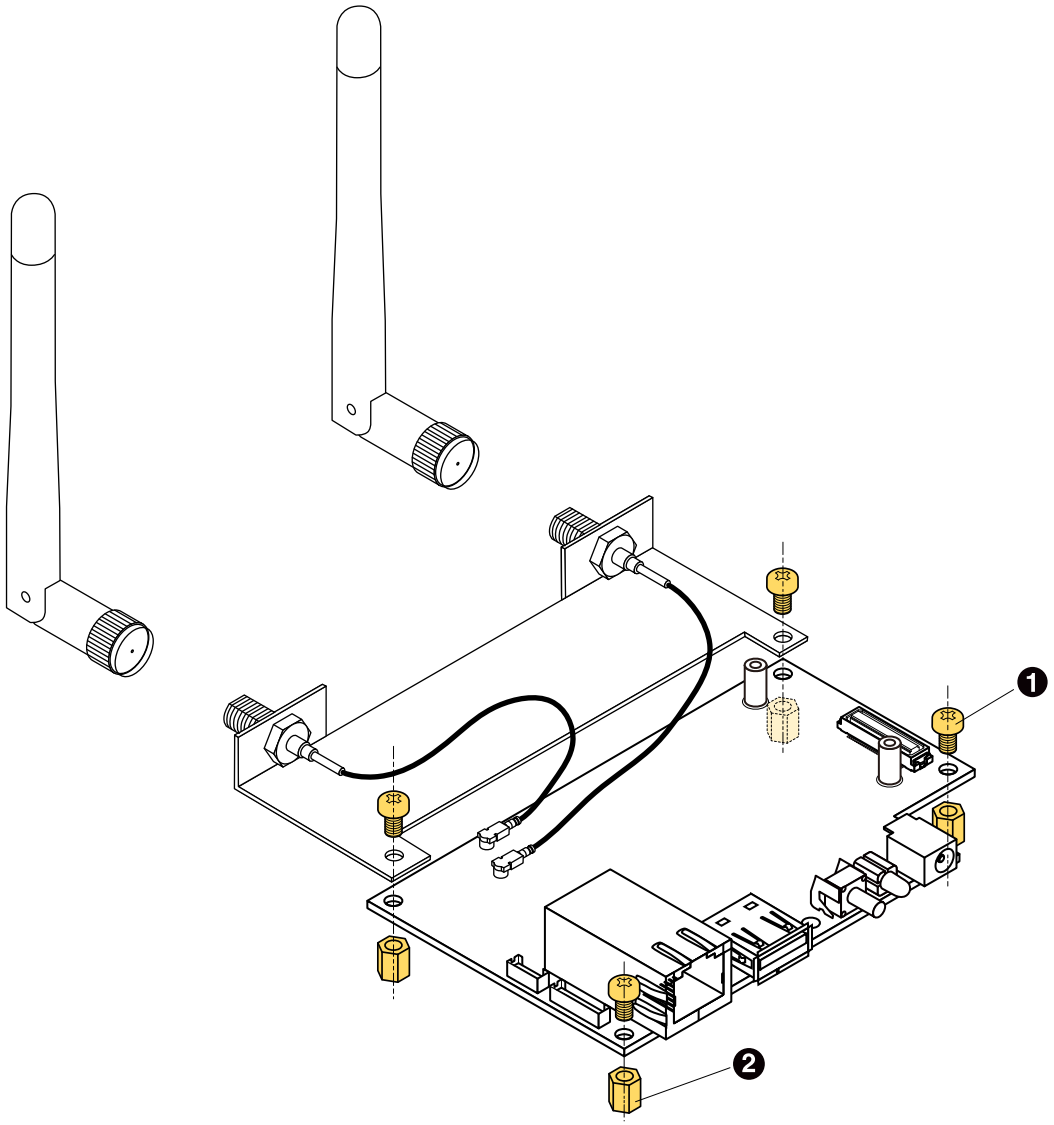


図 19.12 8 ピン JTAG 変換ケーブルの参考回路

19.6. アンテナ固定金具

19.6.1. 組み立て

アンテナ固定金具の接続方法は次のとおりです。



❶ なべ小ねじ(M3、L=5mm)×4

❷ 金属スペーサ 両メネジ六角(M3、L=8mm、平径=5.5mm)×4

図 19.13 アンテナ固定金具の接続

19.7. WLAN+BT コンボモジュール 外付けアンテナセット

19.7.1. 概要

WLAN+BT コンボモジュール 外付けアンテナセットは、 WLAN+BT コンボモジュール(AEH-AR9462/VoxMicro)対応のアンテナセットです。全長 109mm のアンテナ(WAND2DBI-SMA-2NB/OxfordTEC)とケーブル長 140mm のアンテナケーブル(U.FL to RP-SMA)がセットになっています。

19.7.2. 組み立て

アンテナケーブルは、Armadillo-X1 に搭載された WLAN+BT コンボモジュールの U.FL コネクタ(CH0、CH1)に取り付けます。

アンテナ端子に外付けアンテナケーブルを接続する際、無理な力を加えると破損の原因となりますので十分に注意してください。

外付けアンテナケーブルを引き抜く際は、専用の引き抜き治具(U.FL-LP-N-2/ヒロセ電機 等)を用いて行うことを推奨します。引き抜き治具を用いずに引き抜いた場合に、コネクタの変形やケーブルの断線等の原因となります。

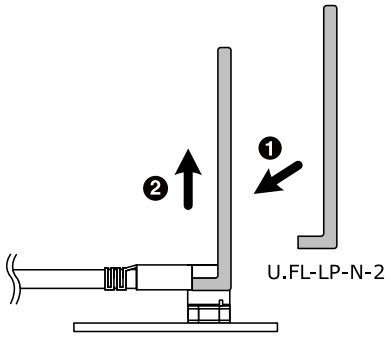


図 19.14 外付けアンテナケーブルの引き抜き方法

19.8. 920MHz 帯 外付けアンテナセット 02

19.8.1. 概要

920MHz 帯 外付けアンテナセット 02 は Wi-SUN アドオンモジュール(OP-AGA-WS00-00) と EnOcean アドオンモジュール(OP-AGA-EN00-00) 対応のアンテナセットです。ケーブル長 200mm のアンテナケーブルと全長 86.3mm のアンテナがセットになっています。

19.8.2. 組み立て

アドオンモジュールのアンテナ端子にアンテナケーブルを取り付けます。

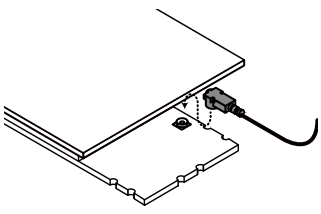


図 19.15 Wi-SUN アドオンモジュール(OP-AGA-WS00-00)のアンテナケーブル取り付け

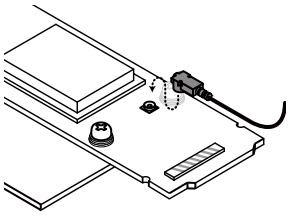


図 19.16 EnOcean アドオンモジュール(OP-AGA-WS00-00)のアンテナケーブル取り付け

アンテナ端子に外付けアンテナケーブルを接続する際、無理な力を加えると破損の原因となりますので十分に注意してください。

外付けアンテナケーブルを引き抜く際は、専用の引き抜き治具(U.FL-LP-N-2/ヒロセ電機 等)を用いて行うことを推奨します。引き抜き治具を用いずに引き抜いた場合に、コネクタの変形やケーブルの断線等の原因となります。

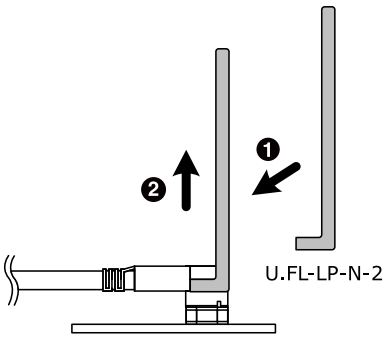
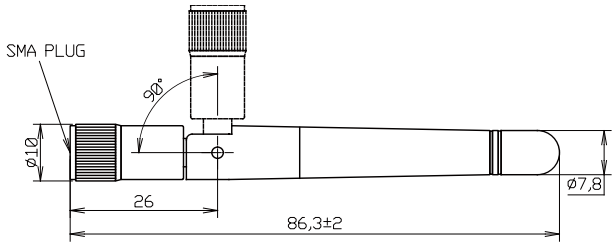


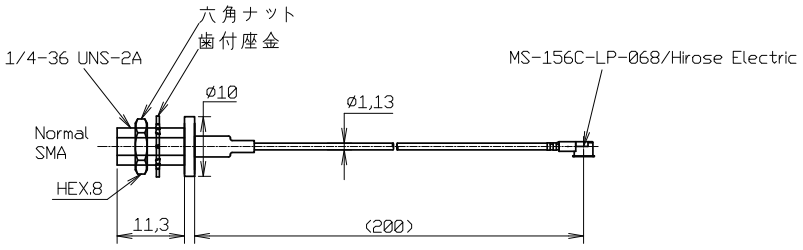
図 19.17 外付けアンテナケーブルの引き抜き方法

19.8.3. 形状図



[Unit : mm]

図 19.18 アンテナ形状



[Unit : mm]

図 19.19 アンテナケーブル形状

20. 設計情報

本章では、Armadillo-X1 の機能拡張や信頼性向上のための設計情報について説明します。

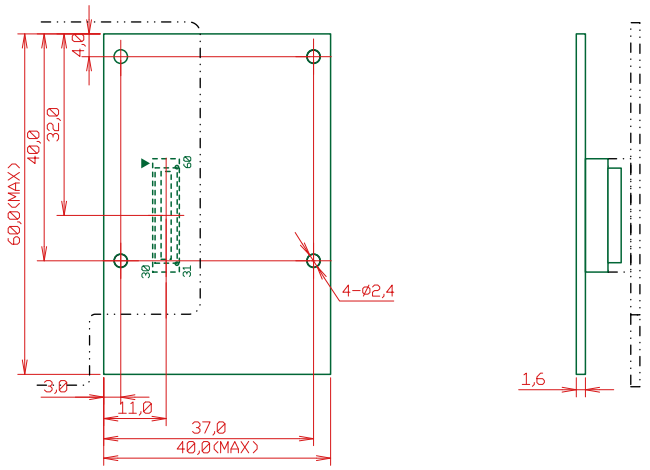
20.1. アドオンモジュールの設計

Armadillo-X1 のアドオンインターフェース(CON7)には、複数の機能(マルチプレクス)をもった i.MX 7Dual の信号線が接続されており、様々な機能拡張を行うことができます。

アドオンモジュールを設計する際の基板形状および部品の搭載制限について説明します。

20.1.1. 基板形状

アドオンモジュールの推奨基板寸法は、「図 20.1. アドオンモジュール推奨基板寸法」のとおりです[1]。



[Unit : mm]

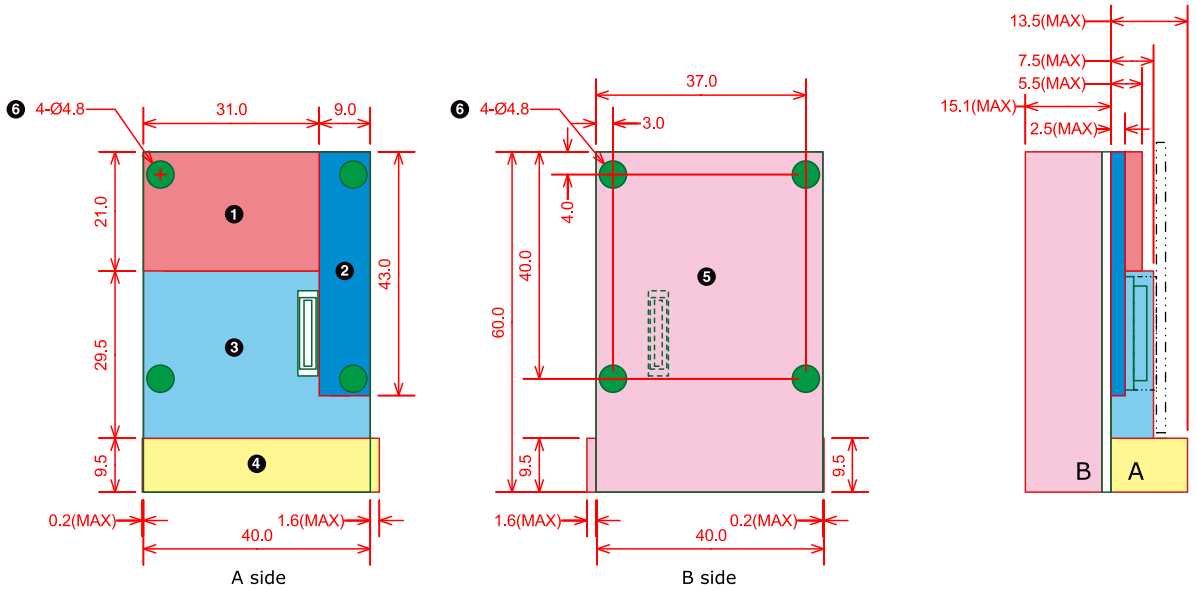
図 20.1 アドオンモジュール推奨基板寸法

Armadillo-X1 の固定穴は GND に接続されています。絶縁等で GND 分離が必要な場合はキリ穴で設計してください。

20.1.2. 部品の搭載制限

アドオンモジュールの部品搭載制限は、「図 20.2. アドオンモジュールの部品搭載制限」のとおりです。

[1]接続コネクタの実装面を A 面、裏面を B 面とし、B 面側から見た図となります。



[Unit : mm]

❶ 最大部品高さ 5.5mm(A 面)

❷ 最大部品高さ 2.5mm(A 面)

❸ 最大部品高さ 7.5mm(A 面)

❹ 最大部品高さ 13.5mm(A 面)

❺ 最大部品高さ 15.1mm(B 面、基板厚さを含む)

❻ 部品搭載禁止領域(A 面、B 面)

図 20.2 アドオンモジュールの部品搭載制限

20.1.3. 接続コネクタ

Armadillo-X1 との接続コネクタは、HIROSE ELECTRIC 製 DF17(4.0)-60DP-0.5V(57)を搭載してください。ピン配置は「図 20.3. アドオンモジュールに実装する接続コネクタのピン配置」のとおりです。

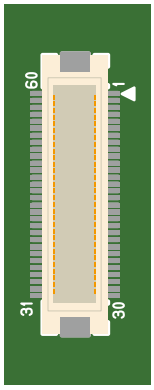


図 20.3 アドオンモジュールに実装する接続コネクタのピン配置

ピン機能については、Armadillo サイトからダウンロード可能な『Armadillo-X1 ベースボード マルチプレクス表』をご確認ください。

アドオンインターフェース(CON7)の PMIC_ONOFF 信号は、2 秒以上 GND にショートすると、パワーマネジメント IC は電圧出力を停止します。 2 秒未満 PMIC_ONOFF 信号を GND にショートすると、パワーマネジメント IC は電圧出力を開始します。

20.2. ESD/雷サージ

ESD 耐性を向上させるための情報を以下に記載します。

Armadillo-X1 を組み込んだ機器、または Armadillo-X1 と LAN ケーブル等で接続された機器を屋外に設置する場合には、以下の点にご注意ください。

Armadillo-X1 に接続されたケーブルが屋外に露出するような設置環境では、ケーブルに侵入した雷サージ等のストレスによりインターフェース回路が破壊される場合があります。ストレスへの耐性を向上させるには、Armadillo-X1 と外部機器同士の GND 接続を強化することおよびシールド付のケーブルを使用することが効果的です。

21. Howto

本章では、Armadillo-X1 のソフトウェアをカスタマイズをする方法などについて説明します。

21.1. Device Tree とは

Device Tree とは、ハードウェア情報を記述したデータ構造体です。ハードウェアの差分を Device Tree に記述することによって、1 つの Linux カーネルイメージを複数のハードウェアで利用することができるようになります。

Device Tree に対応しているメリットの 1 つは、ハードウェアの変更に対するソフトウェアの変更が容易になることです。例えば、拡張インターフェース(CON8)に接続する拡張基板を作成した場合、主に C 言語で記述された Linux カーネルのソースコードを変更する必要は無く、やりたいことをより直感的に記述できる DTS(Device Tree Source)の変更で対応できます。

ただし、Device Tree は「データ構造体」であるため、ハードウェアの制御方法などの「処理」を記述することができない点に注意してください。Device Tree には、CPU アーキテクチャ、RAM の容量、各種デバイスのベースアドレスや割り込み番号などのハードウェアの構成情報のみが記述されます。

Armadillo-X1 での Device Tree のカスタマイズ例については、「21.6. 拡張インターフェースを使う」を参照してください。

Device Tree のより詳細な情報については、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/devicetree/`)、devicetree.org で公開されている「Device Tree Specification」を参照してください。

DeviceTree: The Devicetree Specification

https://www.devicetree.org/

21.2. イメージをカスタマイズする

コンフィギュレーションを変更して Linux カーネルイメージをカスタマイズする方法を説明します。

手順 21.1 イメージをカスタマイズ

1. **Linux カーネルアーカイブの展開**

   Linux カーネルのソースコードアーカイブと、initramfs アーカイブを準備し、Linux カーネルのソースコードアーカイブを展開します。

```
[PC ~]$ ls
initramfs_x1-[version].cpio.gz linux-3.14-x1-at[version].tar.gz
[PC ~]$ tar xf linux-3.14-x1-at[version].tar.gz
[PC ~]$ ls
initramfs_x1-[version].cpio.gz linux-3.14-x1-at[version]  linux-3.14-x1-
at[version].tar.gz
```

2. **initramfs アーカイブへのシンボリックリンク作成**

Linux カーネルディレクトリに移動して、initramfs アーカイブへのシンボリックリンク作成します。

```
[PC ~]$ cd linux-3.14-x1-at[version]
[PC ~/linux-3.14-x1-at[version]]$ ln -s ../initramfs_x1-[version].cpio.gz
initramfs_x1.cpio.gz
```

3. **コンフィギュレーション**

コンフィギュレーションをします。

```
[PC ~/linux-3.14-x1-at[version]]$ make ARCH=arm x1_defconfig
[PC ~/linux-3.14-x1-at[version]]$ make ARCH=arm menuconfig
```

4. **カーネルコンフィギュレーションの変更**

カーネルコンフィギュレーションを変更後、"Exit"を選択して「Do you wish to save your new kernel configuration ? <ESC><ESC> to continue.」で"Yes"とし、カーネルコンフィギュレーションを確定します。

```
 .config - Linux/arm 3.14.38-at1 Kernel Configuration
--------------------------------------------------------------------------------
              Linux/arm 3.14.38-at1 Kernel Configuration
 Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty
 submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
 includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to
 exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
--------------------------------------------------------------------------------

     -*- Patch physical to virtual translations at runtime
         General setup  --->
     [*] Enable loadable module support  --->
     [*] Enable the block layer  --->
         System Type  --->
         Bus support  --->
         Kernel Features  --->
         Boot options  --->
         CPU Power Management  --->
         Floating point emulation  --->
--------------------------------------------------------------------------------
       <Select>    < Exit >    < Help >    < Save >    < Load >
```

Linux Kernel Configuration メニューで"/"キーを押下すると、カーネルコンフィギュレーションの検索を行うことができます。カーネルコンフィギュレーションのシンボル名(の一部)を入力して"Ok"を選択すると、部分一致するシンボル名を持つカーネルコンフィギュレーションの情報が一覧されます。

5. **ビルド**

   ビルドするには、次のようにコマンドを実行します。

```
[PC ~/linux-3.14-x1-at[version]]$ make CROSS_COMPILE=arm-linux-gnueabihf- ARCH=arm
[PC ~/linux-3.14-x1-at[version]]$ make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf-
LOADADDR=0x80008000 uImage
```

6. **イメージファイルの生成確認**

   ビルドが終了すると、arch/arm/boot/ディレクトリと arch/arm/boot/dts/以下に、イメージファイル(Linux カーネルと DTB)が作成されています。

```
[PC ~/linux-3.14-x1-at[version]]$ ls arch/arm/boot/uImage
uImage
[PC ~/linux-3.14-x1-at[version]]$ ls arch/arm/boot/dts/armadillo_x1.dtb
armadillo_x1.dtb
```

21.3. ルートファイルシステムへの書き込みと電源断からの保護機能

Armadillo-X1 のルートファイルシステムは、標準で eMMC に配置されます。 Linux が稼動している間は、ログや、設定ファイル、各種アプリケーション によるファイルへの書き込みが発生します。もし、停電等で終了処理を実行できずに 電源を遮断した場合は RAM 上に残ったキャッシュが eMMC に書き込まれずに、 ファイルシステムの破綻やファイルの内容が古いままになる状況が発生します。

また、eMMC 内部の NAND Flash Memory には消去回数に上限があるため、書き込み 回数を制限することを検討する必要がある場合もあります。

そこで、Armadillo-X1 では、overlayfs を利用して、eMMC への書き込み保護を行う機能を提供しています。

21.3.1. 保護機能の使用方法

eMMC への書き込み保護を使うには、kernel の起動オプションに"overlay=50%" ("=50%" は省略可、"overlay"のみ書くと RAM の 50%を使用)というパラメタを追加するだけです。

パラメタを追加すると、debian の起動前に initramfs によってルートファイルシステムが upper=RAM ディスク(tmpfs)、 lower=eMMC(ext4)とした overlayfs に切り替えられて、Debian が起動します。

overlayfs の機能によって、起動後のルートファイルシステムに対する差分は、 全て RAM ディスク(/overlay/ramdisk にマウント) に記録されるようになりま す。そのため、起動後の情報は保存されませんが、電源を遮断した場合でも、 eMMC は起動前と変らない状態のまま維持されています。

kernel の起動オプションの指定を行うには Armadillo-X1 を保守モードで起動し、次のようにコマンドを実行してください。

```
=> setenv optargs overlay
=> saveenv
```

21.3.2. 保護機能を使用する上での注意事項

overlayfs は差分を ファイル単位で管理するため、予想以上に RAM ディスクを消費する場合があります。 単に、新しいファイルやディレクトリを作れば、その分 RAM ディスクが消費されるのは想像に難くないと思います。

しかし、「lower=eMMC に既に存在していたファイルの書き換え」をする場合は、upper=RAM ディスク に対象のファイル全体をコピーして書き換え」ます。

具体的に、問題になりそうな例を紹介します。 例えば、sqlite は DB 毎に 1 つのファイルでデータ格納します。ここで、1GB の DB を作って eMMC に保存した後、 overlayfs による保護を有効にして起動した後に、たった 10 バイトのレコードを 追加しただけで RAM ディスクは 1GB + 10 バイト消費されます。実際には、 Armadillo に 1GB も RAM は無いので、追記を開始した時点で RAM ディスクが不足します。

overlayfs による、eMMC への書き込み保護を行う場合、 必ず実際の運用状態でのテストを行い、RAM ディスクが不足しないか確認してください。 動作中に書き込むファイルを必要最小限に留めると共に、追記を行う大きなファイルを作らない実装の検討を行ってください。

Armadillo-X1 の eMMC の記録方式は出荷時に SLC に設定しており、MLC 方式の eMMC よりも消去回数の上限が高くなっています。 そのため、開発するシステムの構成によっては eMMC への書き込み保護機能を必要としない可能性があります。

eMMC への書き込み保護機能を有効にすると、eMMC を安全に使用できるというメリットがありますが、その分、使用できる RAM サイズが減る、システム構成が複雑になる、デメリットもあります。 開発・運用したいシステムの構成、eMMC への書き込み保護機能のメリット・デメリットを十分に考慮・評価したうえで、保護機能を使用する、しないの判断を行ってください。

21.4. AR9462 モジュールを使って 2.4GHz 帯で通信する使用例

2 つの機器をサンプルに AR9462 モジュールを使って 2.4GHz 帯で通信するときの使い方について説明します。

21.4.1. 「BVMCN1101AA」の信号を受信する

Braveridge 社製のビーコン「BVMCN1101AA」を例にビーコン信号を受信する方法を説明します。

「BVMCN1101AA」のアドバタイジング・パケットを受信するためには、bluetoothctl コマンドを使います。 [bluetooth]のプロンプトが表示されたら、scan on で信号を受信できます。 ご利用の環境によっては、ほかの機器からの信号も受信されます。

```
[armadillo ~]# bluetoothctl
[NEW] Controller [AA:AA:AA:AA:AA:AA] armadillo [default]
[bluetooth]# scan on
Discovery started
[CHG] Controller [AA:AA:AA:AA:AA:AA] Discovering: yes
[NEW] Device [BB:BB:BB:BB:BB:BB] BBAAEdit
[CHG] Device [BB:BB:BB:BB:BB:BB] RSSI: -67
[CHG] Device [BB:BB:BB:BB:BB:BB] RSSI: -72
```

スキャンを中止するには、scan off を実行します。

```
[bluetooth]# scan off
Discovery stopped
[CHG] Controller [AA:AA:AA:AA:AA:AA] Discovering: no
```

bluetoothctl を終了するには、exit を実行します。

```
[bluetooth]# exit
```

21.4.2. 「CC2650」を操作する

TEXAS INSTRUMENTS 社製のセンサータグ「CC2650」を例にセンサータグを gatttool で操作する方法を説明します。

手順 21.2 「CC2650」の操作手順

1. hcitool lescan を実行して、「CC2650」の MAC アドレスを確認します。 ご利用の環境によっては、他の機器も検出されます。

   ```
   [armadillo ~]# hcitool lescan
   LE Scan ...
   [CC:CC:CC:CC:CC:CC] (unknown)
   [CC:CC:CC:CC:CC:CC] CC2650 SensorTag # <- この MAC アドレスを確認します。
   ```

2. 「CC2650」に接続します。

   ```
   [armadillo ~]# gatttool -b [CC:CC:CC:CC:CC:CC] -I
   [[CC:CC:CC:CC:CC:CC]][LE]> connect
   Attempting to connect to [CC:CC:CC:CC:CC:CC]
   Connection successful
   ```

3. プライマリサービスを確認するには、primary を実行します。

```
[CC:CC:CC:CC:CC:CC][LE]> primary
attr handle: 0x0001, end grp handle: 0x0007 uuid: 00001800-0000-1000-8000-00805f9b34fb
attr handle: 0x0008, end grp handle: 0x000b uuid: 00001801-0000-1000-8000-00805f9b34fb
attr handle: 0x000c, end grp handle: 0x001e uuid: 0000180a-0000-1000-8000-00805f9b34fb
attr handle: 0x001f, end grp handle: 0x0026 uuid: f000aa00-0451-4000-b000-000000000000
attr handle: 0x0027, end grp handle: 0x002e uuid: f000aa20-0451-4000-b000-000000000000
attr handle: 0x002f, end grp handle: 0x0036 uuid: f000aa40-0451-4000-b000-000000000000
attr handle: 0x0037, end grp handle: 0x003e uuid: f000aa80-0451-4000-b000-000000000000
attr handle: 0x003f, end grp handle: 0x0046 uuid: f000aa70-0451-4000-b000-000000000000
attr handle: 0x0047, end grp handle: 0x004b uuid: 0000ffe0-0000-1000-8000-00805f9b34fb
attr handle: 0x004c, end grp handle: 0x0050 uuid: f000aa64-0451-4000-b000-000000000000
# <- ここを詳しく調べます
attr handle: 0x0051, end grp handle: 0x0058 uuid: f000ac00-0451-4000-b000-000000000000
attr handle: 0x0059, end grp handle: 0x0060 uuid: f000ccc0-0451-4000-b000-000000000000
attr handle: 0x0061, end grp handle: 0xffff uuid: f000ffc0-0451-4000-b000-000000000000
```

4. 0x004c～0x0050 でハンドルされているプロファイルの UUID を確認するには、char-desc を実行します。

```
[CC:CC:CC:CC:CC:CC][LE]> char-desc 4c 50
handle: 0x004c, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x004d, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x004e, uuid: f000aa65-0451-4000-b000-000000000000
handle: 0x004f, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0050, uuid: f000aa66-0451-4000-b000-000000000000
```

5. プロファイルの情報を読み取るには、char-read-hnd を実行します。

```
[CC:CC:CC:CC:CC:CC][LE]> char-read-hnd 50
Characteristic value/descriptor: 00
```

6. プロファイルの情報を設定するには、char-write-cmd を実行します。

```
[CC:CC:CC:CC:CC:CC][LE]> char-write-cmd 50 01 # <- 「CC2650」のブザーが鳴り、LED が光ります
[CC:CC:CC:CC:CC:CC][LE]> char-write-cmd 50 01 # <- 「CC2650」のブザーが止まり、LED が消えます
```

7. 操作を終了するには、exit を実行します。

```
[CC:CC:CC:CC:CC:CC][LE]> exit
```

21.5. ssh で Armadillo-X1 に接続する

1. ssh-server をインストールする

```
[armadillo ~]# apt-get update
[armadillo ~]# apt-get install -y ssh
```

2. ssh で root のログインを禁止する

   /etc/ssh/sshd_config 内の PermitRootLogin を no に設定します。

```
[armadillo ~]# vi /etc/ssh/sshd_config
...
# Authentication:
LoginGraceTime 120
PermitRootLogin without-password  # -> no に変更
StrictModes yes
...
```

21.6. 拡張インターフェースを使う

拡張インターフェース(CON8)の使用例を説明します。

拡張インターフェースは、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。Ethernet、USB、SPI、GPIO などに使用可能な信号やパワーマネジメント IC の ON/OFF 用信号などが接続されています。

ここでは、拡張インターフェースのハードウェア構成例と、対応する Device Tree を示します。

21.6.1. Ethernet

拡張インターフェースの Ethernet の使用例を説明します。

例として、Ethernet PHY には Microsemi 製 VSC8501XML-03 を使います。これは Armadillo-X1 本体に搭載されているものと同じです。

拡張インターフェースの Ethernet を使うためには、Device Tree とブートローダーをカスタマイズする必要があります。VSC8501 のドライバは工場出荷イメージで有効になっているため、Linux カーネルイメージのカスタマイズは不要です。

21.6.1.1. ハードウェア構成

ブロック図を「図 21.1. 拡張インターフェース Ethernet ブロック図」に、信号配列を「表 21.1. 拡張インターフェース Ethernet 信号配列」に示します。

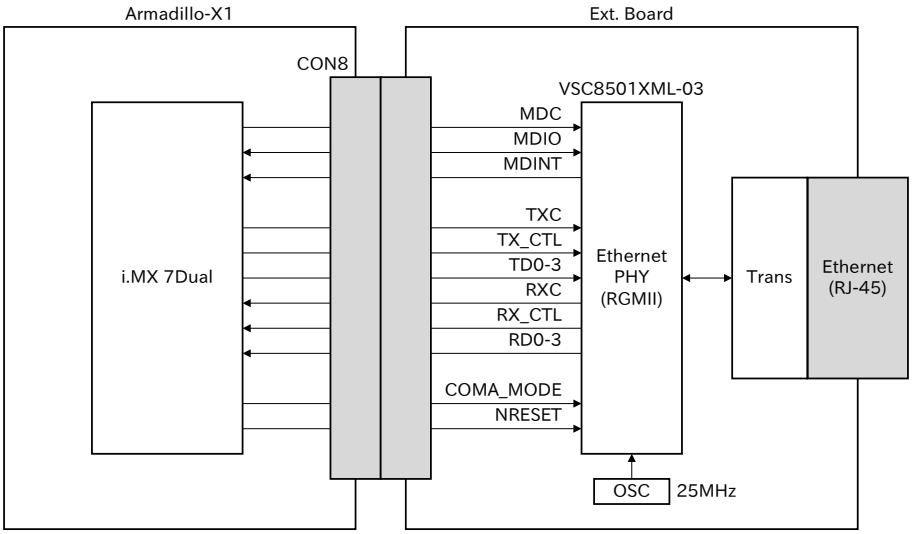


図 21.1 拡張インターフェース Ethernet ブロック図

表 21.1 拡張インターフェース Ethernet 信号配列

CON8 ピン番号	信号名	I/O	説明
24	RD0	In	受信データ 0 i.MX 7Dual の ENET1_RGMII_RD0 ピンに接続
25	RD1	In	受信データ 1 i.MX 7Dual の ENET1_RGMII_RD1 ピンに接続
26	RD2	In	受信データ 2 i.MX 7Dual の ENET1_RGMII_RD2 ピンに接続
27	RD3	In	受信データ 3 i.MX 7Dual の ENET1_RGMII_RD3 ピンに接続
28	RX_CTL	In	受信制御 i.MX 7Dual の ENET1_RGMII_RX_CTL ピンに接続
29	RXC	In	受信クロック i.MX 7Dual の ENET1_RGMII_RXC ピンに接続
30	TD0	Out	送信データ 0 i.MX 7Dual の ENET1_RGMII_TD0 ピンに接続
31	TD1	Out	送信データ 1 i.MX 7Dual の ENET1_RGMII_TD1 ピンに接続
32	TD2	Out	送信データ 2 i.MX 7Dual の ENET1_RGMII_TD2 ピンに接続
33	TD3	Out	送信データ 3 i.MX 7Dual の ENET1_RGMII_TD3 ピンに接続
34	TX_CTL	Out	送信制御 i.MX 7Dual の ENET1_RGMII_TX_CTL ピンに接続
35	TXC	Out	送信クロック i.MX 7Dual の ENET1_RGMII_TXC ピンに接続
36	GPIO7_IO12	In	VSC8501 の MDINT ピンに接続 i.MX 7Dual の ENET1_TX_CLK ピンに接続
37	GPIO7_IO13	Out	VSC8501 の NRESET ピンに接続 i.MX 7Dual の ENET1_RX_CLK ピンに接続
38	GPIO7_IO14	Out	VSC8501 の COMA_MODE ピンに接続 i.MX 7Dual の ENET1_CRS ピンに接続
69	MDC	Out	MDIO クロック i.MX 7Dual の ENET1_MDC ピンに接続
70	MDIO	In/Out	MDIO データ i.MX 7Dual の ENET1_MDIO ピンに接続

21.6.1.2. Device Tree のカスタマイズ

拡張インターフェースの Ethernet インターフェースを有効化した DTS(Device Tree Source)を用意します。「21.6.1.1. ハードウェア構成」に合わせて作成されたサンプルは、Linux カーネルのソースコードの arch/arm/boot/dts/armadillo_x1-fec1.dts です。

DTS では MDINT ピンなどの VSC8501 制御信号が定義されていますが、Linux カーネルのドライバからはアクセスしません。GPIO クラスディレクトリなどからアクセスされないようにする目的で定義しています。

DTS をビルドして DTB(Device Tree Blob)を生成します。「10.2. Linux カーネルをビルドする」に従ってビルドすると、arch/arm/boot/dts/armadillo_x1-fec1.dtb が生成されます。

生成した DTB(Device Tree Blob)を eMMC にインストールする方法を次に示します。

```
[armadillo ~]# mount -t vfat /dev/mmcblk2p1 /mnt ❶
[armadillo ~]# cp armadillo_x1-fec1.dtb /mnt/ ❷
[armadillo ~]# umount /mnt ❸
```

❶ eMMC の第 1 パーティションを/mnt/ディレクトリにマウントします。

❷ DTB を/mnt/ディレクトリにコピーします。

❸ /mnt/ディレクトリにマウントした eMMC の第 1 パーティションをアンマウントします。

21.6.1.3. ブートローダーのカスタマイズ

VSC8501 の初期化はブートローダーが行います。工場出荷イメージでは拡張インターフェースに接続された VSC8501 の初期化に対応していないため、カスタマイズが必要です。

次に示す手順で、U-Boot のビルドを行い、VSC8501 の初期化に対応したブートローダーイメージを生成します。

手順 21.3 VSC8501 の初期化に対応したブートローダーイメージの生成

1. **ソースコードの準備**

   U-Boot のソースコードアーカイブを準備し展開します。

   ```
   [PC ~]$ ls
   uboot_2016.07-at[version].tar.gz
   [PC ~]$ tar xf uboot_2016.07-at[version].tar.gz
   [PC ~]$ ls
   uboot_2016.07-at[version]  uboot_2016.07-at[version].tar.gz
   ```

2. **コンフィギュレーションの適用**

   ```
   [PC ~]$ cd uboot_2016.07-at[version]
   [PC ~/uboot_2016.07-at[version]]$ make ARCH=arm x1_fec1_en_config
   ```

3. **ビルド**

   ビルドには **make** コマンドを利用します。

```
[PC ~/uboot_2016.07-at[version]]$ make CROSS_COMPILE=arm-linux-gnueabihf-
```

4. **イメージファイルの生成確認**

   ビルドが終了すると、U-Boot ディレクトリにイメージファイルが作成されています。

```
[PC ~/uboot_2016.07-at[version]]$ ls u-boot-x1.bin
u-boot-x1.bin
```

生成したブートローダーイメージを QSPI フラッシュメモリにインストールする方法を次に示します。

```
[armadillo ~]# dd if=u-boot-x1.bin of=/dev/mtdblock0 ❶
282+1 records in
282+1 records out
288816 bytes (289 kB) copied, 5.4582 s, 52.9 kB/s
[armadillo ~]$ sync
```

❶ MTD のブロックデバイスの先頭からブートローダーイメージを書き込みます。

21.6.1.4. 動作確認

まず、「21.6.1.2. Device Tree のカスタマイズ」でインストールした DTB で起動するために、保守モードで次のようにコマンドを実行します。

```
=> setenv fdt_file armadillo_x1-fec1.dtb
=> saveenv
```

DTB の指定を出荷状態に戻すには、次のようにコマンドを実行します。

```
=> setenv fdt_file armadillo_x1.dtb
=> saveenv
```

正しく DTB の指定ができた場合は、起動ログに次のような内容が表示されます。

```
GPIO line 204 (VSC8501_MDINT) hogged as input
GPIO line 205 (VSC8501_NRESET) hogged as output/high
GPIO line 206 (VSC8501_COMA_MODE) hogged as output/low
:(省略)
fec 30be0000.ethernet eth0: registered PHC device 0
```

拡張インターフェースの Ethernet インターフェースのネットワークデバイス名は eth0 です。「6.2. ネットワーク」を参照してネットワーク設定を行ってください。

Armadillo-X1 本体の Ethernet インターフェースのネットワークデバイス名は eth1 に変更されます。

MAC アドレスは、出荷時に割り当て済みです。

21.6.2. USB OTG

拡張インターフェースの USB OTG(On-the-Go)の使用例を説明します。

拡張インターフェースの USB OTG を使うためには、Device Tree をカスタマイズする必要があります。

21.6.2.1. ハードウェア構成

ブロック図を「図 21.2. 拡張インターフェース USB OTG ブロック図」に、信号配列を「表 21.2. 拡張インターフェース USB OTG 信号配列」に示します。

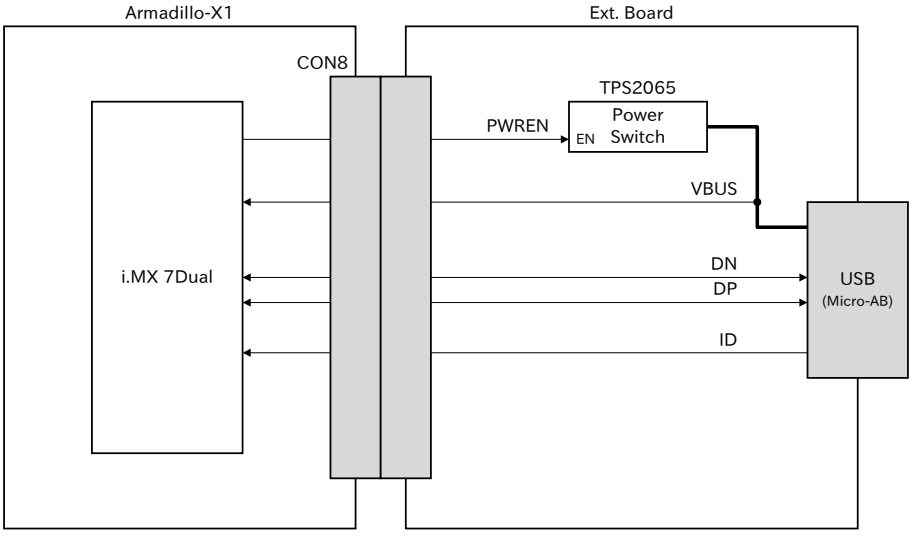


図 21.2 拡張インターフェース USB OTG ブロック図

表 21.2 拡張インターフェース USB OTG 信号配列

CON8 ピン番号	信号名	I/O	説明
47	DP	In/Out	USB プラス側信号 i.MX 7Dual の USB_OTG2_DP ピンに接続

CON8 ピン番号	信号名	I/O	説明
48	DN	In/Out	USB マイナス側信号 i.MX 7Dual の USB_OTG2_DN ピンに接続
50	VBUS	In	VBUS 監視 i.MX 7Dual の USB_OTG2_VBUS ピンに接続
68	ID	In	ID 信号 i.MX 7Dual の SD2_RESET_B ピンに接続
100	GPIO3_IO28	Out	TPS2065 の EN ピンに接続 i.MX 7Dual の LCD_DAT23 ピンに接続

21.6.2.2. Device Tree のカスタマイズ

拡張インターフェースの USB OTG インターフェースを有効化した DTS(Device Tree Source)を用意します。「21.6.2.1. ハードウェア構成」に合わせて作成されたサンプルは、Linux カーネルのソースコードの arch/arm/boot/dts/armadillo_x1-usbotg2.dts です。

DTS をビルドして DTB(Device Tree Blob)を生成します。「10.2. Linux カーネルをビルドする」に従ってビルドすると、arch/arm/boot/dts/armadillo_x1-usbotg2.dtb が生成されます。

生成した DTB(Device Tree Blob)を eMMC にインストールする方法を次に示します。

```
[armadillo ~]# mount -t vfat /dev/mmcblk2p1 /mnt ❶
[armadillo ~]# cp armadillo_x1-usbotg2.dtb /mnt/ ❷
[armadillo ~]# umount /mnt ❸
```

❶ eMMC の第 1 パーティションを/mnt/ディレクトリにマウントします。

❷ DTB を/mnt/ディレクトリにコピーします。

❸ /mnt/ディレクトリにマウントした eMMC の第 1 パーティションをアンマウントします。

21.6.2.3. 動作確認

まず、「21.6.2.2. Device Tree のカスタマイズ」でインストールした DTB で起動するために、保守モードで次のようにコマンドを実行します。

```
=> setenv fdt_file armadillo_x1-usbotg2.dtb
=> saveenv
```

DTB の指定を出荷状態に戻すには、次のようにコマンドを実行します。

```
=> setenv fdt_file armadillo_x1.dtb
=> saveenv
```

正しく DTB の指定ができた場合は、起動ログに次のような内容が表示されます。

```
ci_hdrc ci_hdrc.2: Device No Response
:(省略)
USB_OTG2_VBUS: 5000 mV
```

Armadillo に B プラグが接続された場合は USB ホストとして動作します。例として USB フラッシュメモリが接続された場合の使用方法については「6.3. ストレージ」を参照してください。

Armadillo に A プラグが接続された場合は USB デバイスとして動作します。工場出荷イメージで有効になっているガジェットドライバは、CDC Composite Device(Ethernet and ACM)です。ネットワークデバイス名は `usb0`、TTY デバイスファイル名は`/dev/ttyGS0` です。

21.6.3. I2C

拡張インターフェースの I2C の使用例を説明します。

例として、I2C デバイスには NXP セミコンダクターズ製の温度センサ(LM75B)を使います。

拡張インターフェースの I2C を使うためには、Device Tree をカスタマイズする必要があります。LM75B のドライバは工場出荷イメージで有効になっているため、Linux カーネルイメージのカスタマイズは不要です。

21.6.3.1. ハードウェア構成

ブロック図を「図 21.3. 拡張インターフェース I2C ブロック図」に、信号配列を「表 21.3. 拡張インターフェース I2C 信号配列」に示します。

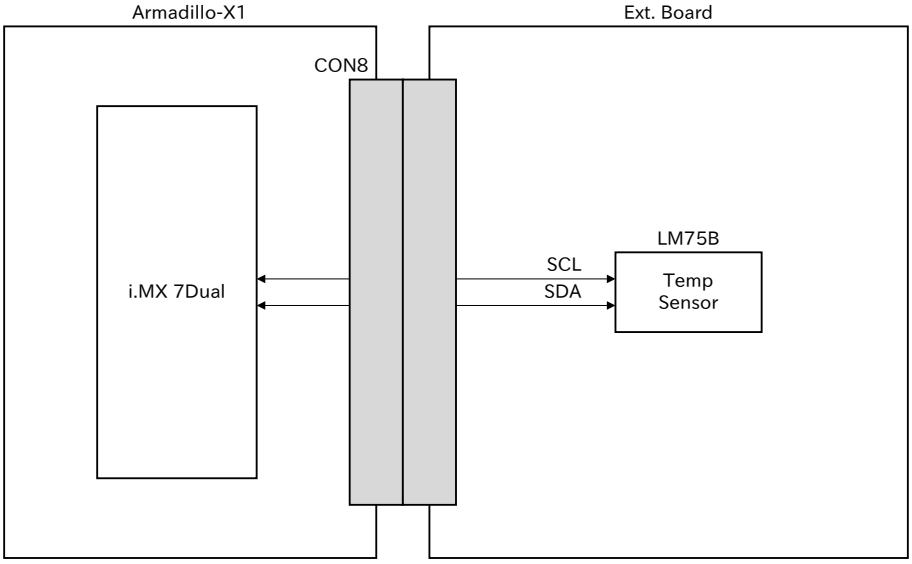


図 21.3 拡張インターフェース I2C ブロック図

表 21.3 拡張インターフェース I2C 信号配列

CON8 ピン番号	信号名	I/O	説明
20	SCL	In/Out	I2C クロック i.MX 7Dual の I2C1_SCL ピンに接続
21	SDA	In/Out	I2C データ i.MX 7Dual の I2C1_SDA ピンに接続

21.6.3.2. Device Tree のカスタマイズ

拡張インターフェースの I2C1 インターフェースを有効化した DTS(Device Tree Source)を用意します。「21.6.3.1. ハードウェア構成」に合わせて作成されたサンプルは、Linux カーネルのソースコードの `arch/arm/boot/dts/armadillo_x1-i2c1_lm75b.dts` です。

DTS をビルドして DTB(Device Tree Blob)を生成します。「10.2. Linux カーネルをビルドする」に従ってビルドすると、arch/arm/boot/dts/armadillo_x1-i2c1_lm75b.dtb が生成されます。

生成した DTB(Device Tree Blob)を eMMC にインストールする方法を次に示します。

```
[armadillo ~]# mount -t vfat /dev/mmcblk2p1 /mnt ❶
[armadillo ~]# cp armadillo_x1-i2c1_lm75b.dtb /mnt/ ❷
[armadillo ~]# umount /mnt ❸
```

❶ eMMC の第 1 パーティションを/mnt/ディレクトリにマウントします。

❷ DTB を/mnt/ディレクトリにコピーします。

❸ /mnt/ディレクトリにマウントした eMMC の第 1 パーティションをアンマウントします。

21.6.3.3. 動作確認

まず、「21.6.3.2. Device Tree のカスタマイズ」でインストールした DTB で起動するために、保守モードで次のようにコマンドを実行します。

```
=> setenv fdt_file armadillo_x1-i2c1_lm75b.dtb
=> saveenv
```

DTB の指定を出荷状態に戻すには、次のようにコマンドを実行します。

```
=> setenv fdt_file armadillo_x1.dtb
=> saveenv
```

正しく DTB の指定ができた場合は、起動ログに次のような内容が表示されます。

```
i2c i2c-0: IMX I2C adapter registered
:(省略)
lm75 0-0048: hwmon0: sensor 'lm75b'
```

/sys/class/i2c-adapter/i2c-0/0-0048/temp1_input ファイルの値を読み出すことによって、温度を取得することができます。

```
[armadillo ~]# cat /sys/class/i2c-adapter/i2c-0/0-0048/temp1_input
25000 ❶
```

❶ 温度はミリ℃ の単位で表示されます。この例では 25℃ を示しています。

21.6.4. SPI

拡張インターフェースの SPI の使用例を説明します。

例として、SPI デバイスには Microchip 製 AD コンバーター(MCP3202)を使います。

拡張インターフェースの SPI を使うためには、Device Tree をカスタマイズする必要があります。MCP3202 のドライバは工場出荷イメージで有効になっているため、Linux カーネルイメージのカスタマイズは不要です。

21.6.4.1. ハードウェア構成

ブロック図を「図 21.4. 拡張インターフェース SPI ブロック図」に、信号配列を「表 21.4. 拡張インターフェース SPI 信号配列」に示します。

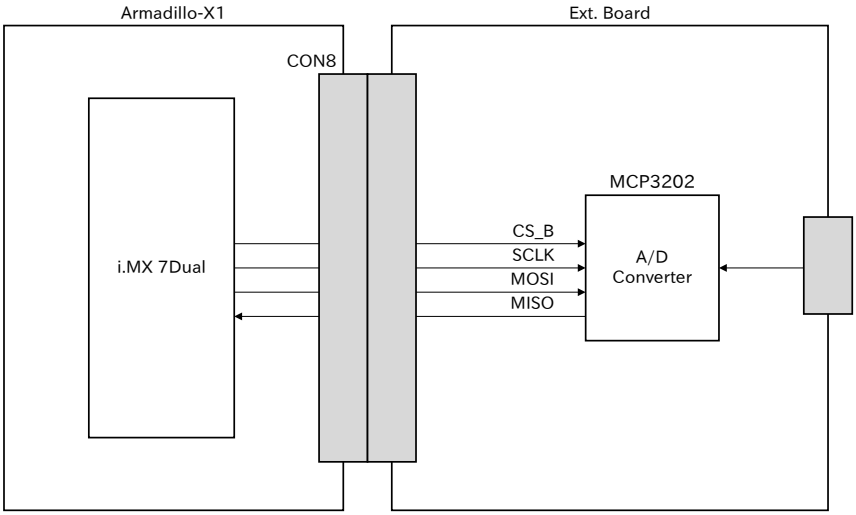


図 21.4 拡張インターフェース SPI ブロック図

表 21.4 拡張インターフェース SPI 信号配列

CON8 ピン番号	信号名	I/O	説明
72	MISO	In	マスター入力／スレーブ出力 i.MX 7Dual の LCD_CLK ピンに接続
73	MOSI	Out	マスター出力／スレーブ入力 i.MX 7Dual の LCD_ENABLE ピンに接続
74	SCLK	Out	シリアルクロック i.MX 7Dual の LCD_HSYNC ピンに接続
75	GPIO3_IO3	Out	スレーブセレクト i.MX 7Dual の LCD_VSYNC ピンに接続

21.6.4.2. Device Tree のカスタマイズ

拡張インターフェースの ECSPI4 インターフェースを有効化した DTS(Device Tree Source)を用意します。「21.6.4.1. ハードウェア構成」に合わせて作成されたサンプルは、Linux カーネルのソースコードの arch/arm/boot/dts/armadillo_x1-ecspi4_mcp3202.dts です。

DTS をビルドして DTB(Device Tree Blob)を生成します。「10.2. Linux カーネルをビルドする」に従ってビルドすると、arch/arm/boot/dts/armadillo_x1-ecspi4_mcp3202.dtb が生成されます。

生成した DTB(Device Tree Blob)を eMMC にインストールする方法を次に示します。

```
[armadillo ~]# mount -t vfat /dev/mmcblk2p1 /mnt ❶
[armadillo ~]# cp armadillo_x1-ecspi4_mcp3202.dtb /mnt/ ❷
[armadillo ~]# umount /mnt ❸
```

❶ eMMC の第 1 パーティションを/mnt/ディレクトリにマウントします。

❷ DTB を/mnt/ディレクトリにコピーします。

❸ /mnt/ディレクトリにマウントした eMMC の第 1 パーティションをアンマウントします。

21.6.4.3. 動作確認

まず、「21.6.4.2. Device Tree のカスタマイズ」でインストールした DTB で起動するために、保守モードで次のようにコマンドを実行します。

```
=> setenv fdt_file armadillo_x1-ecspi4_mcp3202.dtb
=> saveenv
```

DTB の指定を出荷状態に戻すには、次のようにコマンドを実行します。

```
=> setenv fdt_file armadillo_x1.dtb
=> saveenv
```

正しく DTB の指定ができた場合は、起動ログに次のような内容が表示されます。

```
MCP3202_VREF: 5000 mV
:(省略)
spi_imx 30630000.ecspi: probed
```

/sys/bus/iio/devices/iio:device1/ディレクトリ以下にあるファイルの値を読み出すことによって、入力電圧を算出することができます。

ファイル	説明
in_voltage0_raw	シングルエンド入力 CH0 の AD 変換値
in_voltage1_raw	シングルエンド入力 CH1 の AD 変換値
in_voltage_scale	シングルエンド入力の最小入力電圧変動
in_voltage0-voltage1_raw	疑似差動入力の AD 変換値
in_voltage-voltage_scale	疑似差動入力の最小入力電圧変動

シングルエンド入力 CH0 への入力電圧を算出する例を次に示します。

```
[armadillo ~]# cat /sys/bus/iio/devices/iio:device1/in_voltage0_raw
2048
[armadillo ~]# cat /sys/bus/iio/devices/iio:device1/in_voltage_scale
1.220703125
```

この例では、シングルエンド入力 CH0 への入力電圧は、2.5V (2048 × 1.220703125 [mV])である事がわかります。

awk コマンドを利用して、次のように電源電圧を表示することができます。

```
[armadillo ~]# adin_raw=`cat /sys/bus/iio/devices/iio:device1/
in_voltage0_raw`
[armadillo ~]# adin_scale=`cat /sys/bus/iio/devices/iio:device1/
in_voltage_scale`
[armadillo ~]# echo $adin_raw $adin_scale | awk '{printf ("%d",$1*$2)}'
2500
```

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/

改訂履歴

バージョン	年月日	改訂内容
1.0.0	2016/09/20	・初版発行
1.0.1	2016/10/27	・「表 3.4. 仕様」に消費電力(参考値)を追加 ・AX1100-U00Z, AX1100-U01Z に対応 ・誤記修正
1.1.0	2016/12/07	・「7. Linux カーネル仕様」に「7.3.20. パワーマネジメント」を追加 ・「図 6.37. 電源電圧の計算式」の誤記修正

Armadillo-X1 製品マニュアル
Version 1.1.0
2016/12/08