

Armadillo-640 Node-RED 開発ガイド

Version 1.1.0
2024/01/29

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

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

Armadillo-640 Node-RED 開発ガイド

株式会社アットマークテクノ

製作著作 © 2023-2024 Atmark Techno, Inc.

Version 1.1.0
2024/01/29

目次

1. はじめに	6
1.1. Node-RED について	6
1.2. 本書について	6
1.2.1. 本書で扱うこと	6
1.2.2. 本書で扱わないこと	6
1.2.3. 本書で必要となる知識と想定する読者	6
1.2.4. フォント	7
1.2.5. コマンド入力例	7
1.2.6. アイコン	7
1.2.7. ユーザー限定コンテンツ	8
1.2.8. 本書および関連ファイルのバージョンについて	8
2. ユーザー登録	9
2.1. 購入製品登録	9
3. Armadillo のセットアップ	10
3.1. 作業の前に	10
3.1.1. 開発セット内容物の確認	10
3.1.2. 開発に必要なもの	10
3.1.3. 接続方法	10
3.1.4. USB シリアル変換アダプタの接続方法	11
3.2. Node-RED コンテナをインストールする	12
3.3. インターフェースレイアウト	12
3.4. ネットワークに接続する	13
3.4.1. Armadillo の有線 LAN に固定 IP アドレスを設定する	17
3.5. Node-RED に接続する	19
4. 開発を行う	20
4.1. Node-RED に接続する	20
4.2. Node-RED コンテナのログを表示する	21
4.3. Node-RED で利用可能なノードの一覧	21
4.4. フローを作成する	21
4.4.1. LED を制御する	21
4.4.2. CPU の測定温度を取得する	25
4.4.3. RS-485 modbus RTU 読み出しを行う	28
4.4.4. CPU の測定温度のグラフをダッシュボードに表示する	30
4.4.5. 外部プログラムを実行する	39
4.4.6. Node-RED を終了する	41
4.5. ユーザデータを削除する	44

目次

3.1. 初回起動時の Node-RED 画面	10
3.2. Armadillo-640 の接続例	11
3.3. CON9-USB シリアル変換アダプタ接続図	12
3.4. インターフェースレイアウト	13
3.5. パスワード登録画面	15
3.6. パスワード登録完了画面	16
3.7. ログイン画面	16
3.8. 状態一覧を選択	17
3.9. LAN 情報	17
3.10. 初回起動時の Node-RED 画面	19
4.1. 初回起動時の Node-RED 画面	20
4.2. Node-RED の画面領域	21
4.3. [inject] ノードのプロパティ内容	22
4.4. [trigger] ノードのプロパティ内容	23
4.5. [write file] ノードのプロパティ内容	24
4.6. LED を 1 秒間隔で点滅するフロー	24
4.7. [inject] ノードのプロパティ内容	25
4.8. [read file] ノードのプロパティ内容	26
4.9. [function] ノードのプロパティ内容	27
4.10. CPU の測定温度を 1 秒間隔で取得するフロー	27
4.11. [modbus-client] ノードのプロパティ内容	29
4.12. [Modbus-Read] ノードのプロパティ内容	30
4.13. RS-485 を使用した Modbus RTU 読み出し用フロー	30
4.14. [ダッシュボード]を選択する	31
4.15. ダッシュボード編集画面	31
4.16. ダッシュボードに [Tab1] を追加	32
4.17. [Tab1] プロパティ内容	33
4.18. ダッシュボード編集画面に [Armadillo] タブが追加	34
4.19. ダッシュボード編集画面に [Group1] グループが追加	35
4.20. [Group1] プロパティ内容	36
4.21. ダッシュボード編集画面に [chart] ノードが追加	37
4.22. [inject] ノードのプロパティ内容	38
4.23. CPU の測定温度のグラフをダッシュボードに表示するフロー	38
4.24. CPU の測定温度のグラフのダッシュボード	39
4.25. [exec] ノードのプロパティ内容	40
4.26. 外部プログラムを実行するフロー	40
4.27. [inject] ノードのプロパティ内容	42
4.28. [exit] ノードのプロパティ内容	43
4.29. Node-RED を任意のタイミングで終了するフロー	43

表目次

- 1.1. 使用しているフォント 7
- 1.2. 表示プロンプトと実行環境の関係 7
- 1.3. コマンド入力例での省略表記 7
- 3.1. 開発に必要なもの 10
- 3.2. インターフェース内容 13
- 3.3. シリアル通信設定 18
- 4.1. LED 信号配列 22

1. はじめに

このたびは Armadillo をご利用いただき、ありがとうございます。

本書では、Armadillo-640 上での Node-RED を用いた開発手法を説明します。

Armadillo-640 の詳細な説明に関しては、製品マニュアルに記載しておりますので、以下の URL よりダウンロードしてください。

Armadillo サイト - Armadillo-640 ドキュメントダウンロード

<https://armadillo.atmark-techno.com/armadillo-640/resources/documents>

1.1. Node-RED について

Node-RED は、IoT アプリケーション開発に適した、オープンソースのローコード開発ツールです。「ノード」と呼ばれる機能ブロックを繋ぎ合わせてアプリケーションを作ります。ビジュアルプログラミング環境のため、Python や JavaScript といったプログラミングの知識がなくてもアプリケーションを作れます。PLC やリモート I/O などの産業機器からのデータ収集や、それらの制御を行うアプリケーション用のノードもあり、製造業 DX の内製ツールとしても注目されています。Armadillo-640 ですぐに使えるようにした Node-RED コンテナを提供します。



Node-RED は、OpenJS Foundation の米国およびその他の国における登録商標または商標です。

1.2. 本書について

1.2.1. 本書で扱うこと

本書では Armadillo-640 へ Node-RED コンテナをインストールし、Armadillo-640 上で Node-RED を操作し開発する方法を説明します。

1.2.2. 本書で扱わないこと

Node-RED の基本的な説明は記載しておりません。Node-RED の使用方法は各種書籍・Web サイトを参照ください。

また、Armadillo をご利用になられる際の注意事項は、Armadillo 製品マニュアル「注意事項」の章をご一読ください。

1.2.3. 本書で必要となる知識と想定する読者

Node-RED を使用したことがあり、使い方を把握しているエンジニアを対象読者として想定しております。

1.2.4. 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

1.2.5. 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 の root ユーザで実行
[PC /]\$	作業用 PC の一般ユーザで実行
[ATDE ~]#	ATDE 上の root ユーザで実行
[ATDE ~]\$	ATDE 上の一般ユーザで実行
[armadillo /]#	Armadillo 上 Linux の root ユーザで実行
[armadillo /]\$	Armadillo 上 Linux の一般ユーザで実行
[container /]#	Podman コンテナ内で実行
⇒	Armadillo 上 U-Boot 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。適宜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

1.2.6. 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。



用語の説明や補足的な説明を記載します。

1.2.7. ユーザー限定コンテンツ

アットマークテクノ Armadillo 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

限定コンテンツを取得するには、「2. ユーザー登録」を参照してください。

1.2.8. 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-640 ドキュメントダウンロード

<https://armadillo.atmark-techno.com/armadillo-640/resources/documents>

Armadillo サイト - Armadillo-640 ソフトウェアダウンロード

<https://armadillo.atmark-techno.com/armadillo-640/resources/software>

2. 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 Armadillo 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 Armadillo サイト」をご覧ください。

アットマークテクノ Armadillo サイト

<https://armadillo.atmark-techno.com/>

2.1. 購入製品登録

ユーザー登録完了後に、購入製品登録することで、「購入者向けの限定公開データ」をダウンロードすることができるようになります。

購入製品登録の詳しい手順は以下の URL をご参照ください。

Armadillo-640 購入製品登録

<https://armadillo.atmark-techno.com/armadillo-640/register>

3. Armadillo のセットアップ

Armadillo-640 上で Node-RED を使用するためのセットアップ方法を説明します。Armadillo-640 に Node-RED 対応のインストールディスクをインストールし、開発用パソコンと Armadillo-640 を有線 LAN で繋がる様にします。セットアップが完了すると Node-RED の起動画面が表示されます。

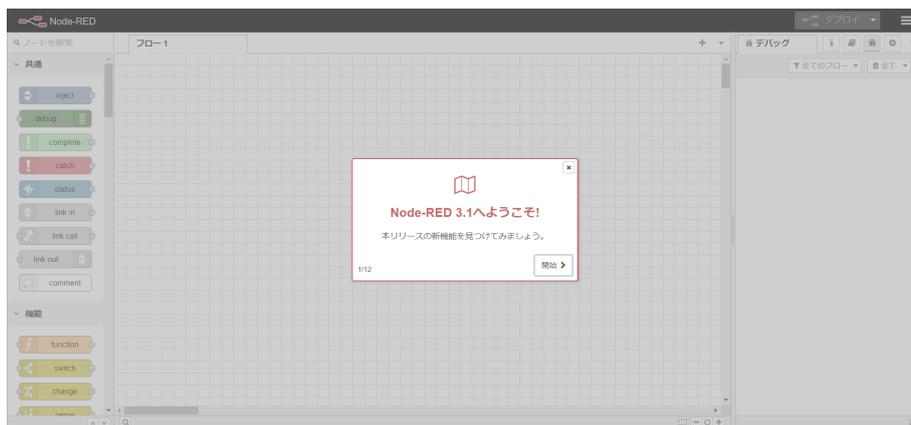


図 3.1 初回起動時の Node-RED 画面

3.1. 作業の前に

3.1.1. 開発セット内容物の確認

開発セットに同梱されている内容物は、同梱の内容物一覧でご確認いただけます。お使いになる前に、内容物がすべて揃っていることをご確認ください。万一、内容物の不足または部品の破損等がございましたら、ご購入の販売代理店までご連絡ください。

3.1.2. 開発に必要なもの

Armadillo-640 上で Node-RED を使用するためには以下のものがが必要です。

表 3.1 開発に必要なもの

品目	説明
Armadillo-640 開発セット	本製品です。
開発用パソコン	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、microSD ポートを持つパソコンです。
有線 LAN ケーブル	本ガイドでは Armadillo-640 と開発用パソコンを有線 LAN で接続する前提で記載しています。
512MB 以上の microSD カード	Node-RED の開発環境をインストールするために使用します。開発用パソコンの SD ポートが microSD ではなく、SD/miniSD ポートの場合、microSD へ変換するアダプターも必要となります。

3.1.3. 接続方法

Armadillo-640 と周辺装置の接続例を「図 3.2. Armadillo-640 の接続例」に示します。

開発用パソコンとのファイル転送やインターネットへの接続は、LAN を介して行いますので LAN ケーブルを接続してください。

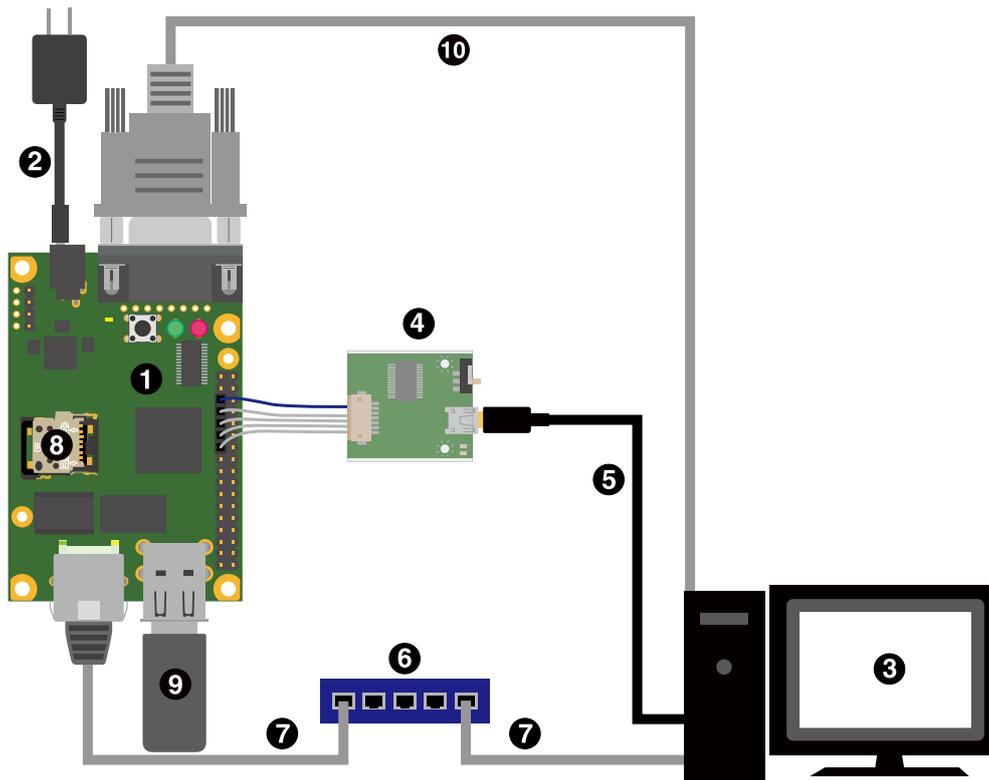


図 3.2 Armadillo-640 の接続例

- ❶ Armadillo-640
- ❷ AC アダプタ (5V/2A)
- ❸ 作業用 PC
- ❹ USB シリアル変換アダプタ
- ❺ USB2.0 ケーブル(A-miniB タイプ)
- ❻ LAN HUB
- ❼ Ethernet ケーブル
- ❽ microSD カード
- ❾ USB メモリ
- ❿ シリアルクロスケーブル

3.1.4. USB シリアル変換アダプタの接続方法

USB シリアル変換アダプタは、青色のケーブルを 1 ピンとして、Armadillo-640 の CON9 1,3,5,7,9 ピンと接続します。

USB シリアル変換アダプタを接続するピンの隣だけ、CON9,CON14 を囲っているシルクが太くなっているのをそれを目印にして、下図のように接続してください。

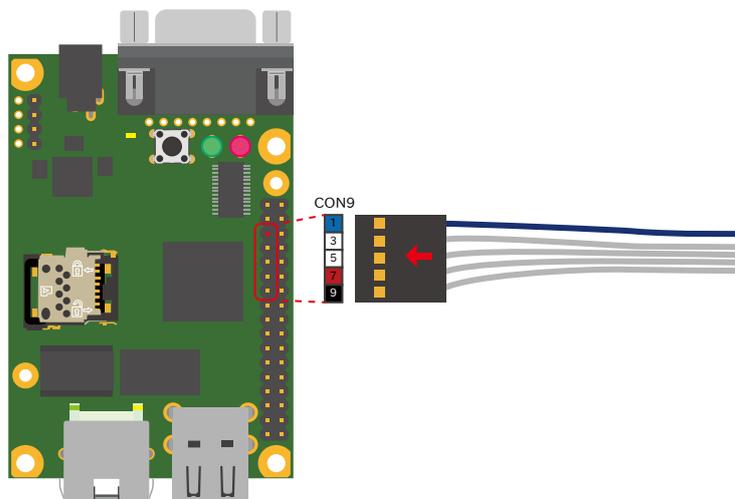


図 3.3 CON9-USB シリアル変換アダプタ接続図

3.2. Node-RED コンテナをインストールする

SWUpdate の機能を使用して SWU で Armadillo Base OS を最新版にアップデートし、Node-RED コンテナをインストールします。SWUpdate で使用する SWU イメージは以下から取得可能です。

- ・ Armadillo Base OS

Armadillo-640 Armadillo Base OS [<https://armadillo.atmark-techno.com/resources/software/armadillo-640/baseos>] から「Armadillo-640 用 SWU イメージファイル」をダウンロードしてください。

- ・ Node-RED コンテナ

Armadillo-640 Node-RED コンテナ [<https://armadillo.atmark-techno.com/resources/software/armadillo-640/node-red-container>] から「Armadillo-640 用 SWU イメージファイル」をダウンロードしてください。

上記で取得した SWU イメージを USB メモリに配置します。USB メモリを Armadillo-640 に接続すると自動的にアップデートが始まります。アップデート終了後に Armadillo-640 は自動で再起動します。

3.3. インターフェースレイアウト

Armadillo-640 のインターフェースレイアウトです。各インターフェースの配置場所等を確認してください。

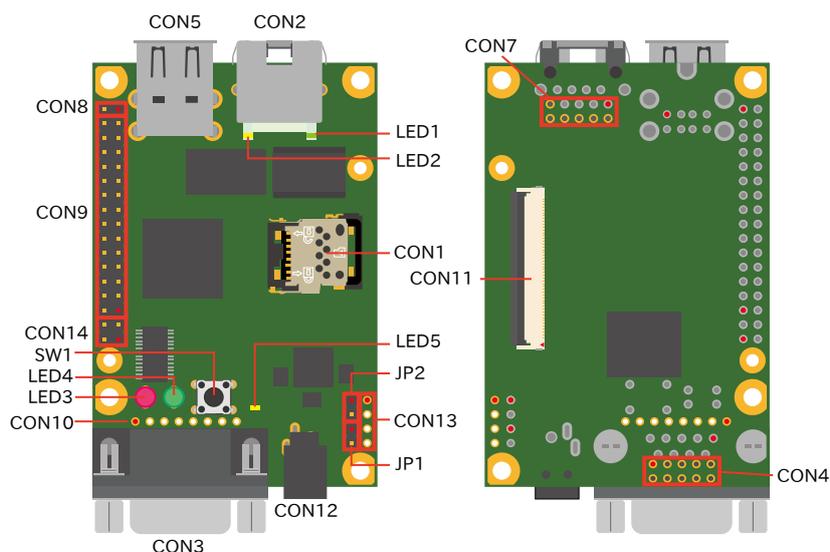


図 3.4 インターフェースレイアウト

表 3.2 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	SD インターフェース	microSD スロット(ピンジタイプ)	
CON2	LAN インターフェース	RJ-45 コネクタ	
CON3	シリアルインターフェース	D-Sub9 ピン(オス)	
CON4	シリアルインターフェース	ピンヘッダ 10 ピン(2.54mm ピッチ)	コネクタ非搭載
CON5	USB インターフェース	Type-A コネクタ(2 ポートスタック)	
CON7	LAN インターフェース	ピンヘッダ 10 ピン(2.54mm ピッチ)	コネクタ非搭載
CON8	拡張インターフェース	ピンヘッダ 2 ピン(2.54mm ピッチ)	
CON9	拡張インターフェース	ピンヘッダ 28 ピン(2.54mm ピッチ)	
CON10	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
CON11	LCD 拡張インターフェース	FFC コネクタ 50 ピン(0.5mm ピッチ)	挿抜寿命: 20 回 ^[a]
CON12	電源入力インターフェース	DC ジャック	対応プラグ: EIAJ#2
CON13	電源入力インターフェース	ピンヘッダ 4 ピン(2.54mm ピッチ)	コネクタ非搭載
CON14	拡張インターフェース	ピンヘッダ 4 ピン(2.54mm ピッチ)	
JP1	起動デバイス設定ジャンパ	ピンヘッダ 2 ピン(2.54mm ピッチ)	
JP2	起動デバイス設定ジャンパ	ピンヘッダ 2 ピン(2.54mm ピッチ)	
LED1	LAN スピード LED	LED(緑色,面実装)	
LED2	LAN リンクアクティビティ LED	LED(黄色,面実装)	
LED3	ユーザー LED(赤)	LED(赤色,φ3mm)	
LED4	ユーザー LED(緑)	LED(緑色,φ3mm)	
LED5	ユーザー LED(黄)	LED(黄色,面実装)	
SW1	ユーザースイッチ	タクトスイッチ(h=17mm)	

^[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。

3.4. ネットワークに接続する

Node-RED の開発をするパソコンと Armadillo のネットワークを接続するために Armadillo-640 の IP アドレスを取得または設定します。

Armadillo の有線 LAN は、初期状態で DHCP の設定となっておりますので、DHCP が稼働している有線 LAN に接続した場合は、なんらかの IP アドレスが付与された状態になっています。

DHCP で IP アドレスが付与される環境の場合、ABOS Web を使用することで IP アドレスの確認が可能です。

1. Armadillo-640 を「3.2. Node-RED コンテナをインストールする」でセットアップを済ませた場合は ABOS Web が起動しています。再起動がまだの場合は電源を再投入します。
2. 開発用パソコンで Web ブラウザーを起動し <https://armadillo.local:58080> にアクセスします。



ABOS Web が動作する Armadillo が、同じ LAN 上に複数ある場合、ABOS Web に接続する URL のホスト名部分 (armadillo.local) は、2 台目は armadillo-2.local、3 台目は armadillo-3.local と、違うものが自動的に割り当てられます。この場合どの URL がどの Armadillo かを判別するのは難しいので固定 IP アドレスを設定し、IP アドレスで指定できるようにする方法があります。固定 IP アドレスの設定方法は「3.4.1. Armadillo の有線 LAN に固定 IP アドレスを設定する」を参照ください。

3. 初回接続時は、ABOS Web のパスワードを設定する必要がありますので、設定します。



図 3.5 パスワード登録画面

4. "初回ログイン"のパスワード登録画面で、"パスワード" フィールドと "パスワード(確認)" フィールドに、登録したいパスワードを入力してから、"登録" ボタンをクリックしてください。パスワード登録完了画面が表示されたら、パスワード登録の完了です。



図 3.6 パスワード登録完了画面

5. ログインします。



図 3.7 ログイン画面

6. 左のメニューから「状態一覧」を選択します。



図 3.8 状態一覧を選択

7. 「LAN 情報」の「IP アドレス」の欄に有線 LAN の IP アドレスが表示されていますのでメモしておいてください。



図 3.9 LAN 情報

3.4.1. Armadillo の有線 LAN に固定 IP アドレスを設定する

DHCP サーバーが存在しない環境で開発を行う場合、Armadillo-640 に固定の IP アドレスを付与して開発を行います。

Armadillo にシリアルコンソールを接続してコマンドを入力する必要があります。シリアルコンソールは USB シリアル変換アダプタを用いて接続します。USB シリアル変換アダプタの接続方法は「3.1.4. USB シリアル変換アダプタの接続方法」を参照ください。

1. Armadillo-640 と開発用パソコンを開発セットに同梱されています USB(A オス-microB)ケーブルで接続します。Armadillo-640 側は「図 3.2. Armadillo-640 の接続例」に示す「④USB シリアル変換アダプタ」に microB を、開発用パソコン側は Type-A を接続してください。
2. 開発用パソコンでシリアルコンソールを開きます。対応しているソフトウェアは、Windows OS であれば TeraTerm、Linux であれば minicom などがありますので、インストールしてご利用ください。
3. ソフトウェアのシリアル設定を「表 3.3. シリアル通信設定」に示します。

表 3.3 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

1. Armadillo-640 の AC アダプターをコンセントに繋ぎ電源を投入します。
2. シリアルコンソールに以下のようにログインプロンプトが表示されましたら、root ユーザーでログインします。特に設定していない場合の root ユーザーの初期パスワードは root です。

```
Welcome to Alpine Linux 3.18
Kernel 5.10.197-0-at on an armv7l (/dev/ttyxc2)

armadillo login:
```

3. "Wired connection 1" に固定 IP アドレスを設定します。

```
[armadillo ~]# nmcli connection modify "Wired connection 1" \
  ipv4.method manual ipv4.addresses 192.0.2.10/24 ipv4.gateway 192.0.2.1
```

4. 有線 LAN eth0 を再起動します。

```
[armadillo ~]# nmcli connectio down "Wired connection 1"
[armadillo ~]# nmcli connectio up "Wired connection 1"
```

5. 設定された IP アドレスは ip addr コマンドで確認できます。

```
[armadillo ~]# ip addr show eth0
2: eth0: ... 中略...
   inet 192.0.2.10/24 brd 192.0.2.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
   ... 後略 ...
```

- 設定を永続化するために、`persist_file` コマンドを実行してください。

```
[armadillo ~]# persist_file /etc/NetworkManager/system-connections/"Wired connection 1.nmconnection"
```



3.5. Node-RED に接続する

パソコンの Web ブラウザから、「3.4. ネットワークに接続する」 で取得した IP アドレスを使用して、`http://<ip アドレス>:1880/` にアクセスしてください。

Node-RED の起動画面が表示されたらセットアップは終了です。

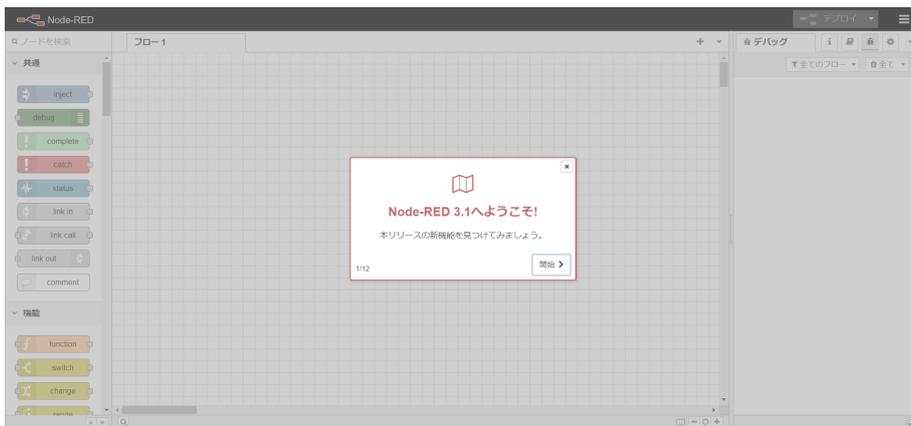


図 3.10 初回起動時の Node-RED 画面

4. 開発を行う

4.1. Node-RED に接続する

「3. Armadillo のセットアップ」を完了している場合は Armadillo を起動すると、自動的に Node-RED コンテナが起動します。Node-RED への接続には「3.4. ネットワークに接続する」で取得した IP アドレスを使用します。

パソコンの Web ブラウザから、`http://<ip アドレス>:1880/` にアクセスしてください。

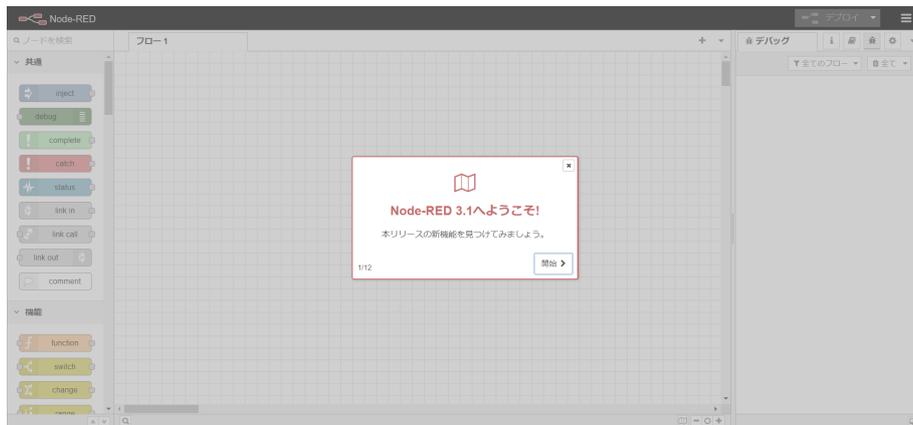


図 4.1 初回起動時の Node-RED 画面

Node-RED には大きく 3 つの領域があります。

- ・ パレット

使用可能なコアノード、カスタムノードの一覧を表示します。

- ・ ワークスペース

Node-RED では一つのアプリケーションの流れをフローという単位で表します。ワークスペース上にノードを繋げてフローを作成します。パレットから必要なノードをワークスペースにドラッグ、ドロップで配置します。

- ・ サイドバー

選択したノードの情報や、デバッグメッセージを表示します。



図 4.2 Node-RED の画面領域

4.2. Node-RED コンテナのログを表示する

パソコンの Web ブラウザから、`http://<ip アドレス>:1880/node-red.log` にアクセスすると Node-RED コンテナのログが表示されます。

このログファイルは `/var/app/rollback/volumes/node-red/log` に保存されています。ログのサイズが 5MB を超えると新しいログファイルに移行します。古いログファイルを表示したい場合は、`http://<ip アドレス>:1880/node-red.log.1` にアクセスしてください。

4.3. Node-RED で利用可能なノードの一覧

「3. Armadillo のセットアップ」でインストールした Node-RED コンテナには Node-RED のコアノードの他にインストールされているカスタムノードがあります。インストール済みのカスタムノードは以下になります。

- ・ Dashboard ノード

Dashboard を表示するノードです。

- ・ exit ノード

Node-RED を終了するためのノードです。

4.4. フローを作成する

Node-RED では一つのアプリケーションの流れをフローという単位で表します。各ノードを使用したフローの作成方法について紹介します。

4.4.1. LED を制御する

ここでは、LED を 1 秒間隔で点滅するフローを作成します。すべて Node-RED のコアノードを使用します。Armadillo-640 の LED は以下がありますが、今回対象にするのはユーザー LED(緑)です。ユーザー LED(緑)の場所については「3.3. インターフェースレイアウト」をご確認ください。

ユーザー LED(緑)は `/sys/class/leds/green/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。0 を書き込むと消灯、0 以外の値 (1~255) を書き込むと点灯します。

表 4.1 LED 信号配列

部品番号	名称(色)	説明
LED3	ユーザー LED(赤)	i.MX6ULL の GPIO1_IO05 ピンに接続、(Low: 消灯、High: 点灯)
LED4	ユーザー LED(緑)	i.MX6ULL の GPIO1_IO08 ピンに接続、(Low: 消灯、High: 点灯)
LED5	ユーザー LED(黄)	i.MX6ULL の UART1_CTS_B ピンに接続、(Low: 消灯、High: 点灯)

1. パレットから [inject] ノードをワークスペースにドラッグ、ドロップします。
2. [inject] ノードのプロパティを編集します。[inject] は 2 秒間隔で実行します。

名前: LED Blink
 繰り返し: 指定した時間間隔
 時間間隔: 2 秒

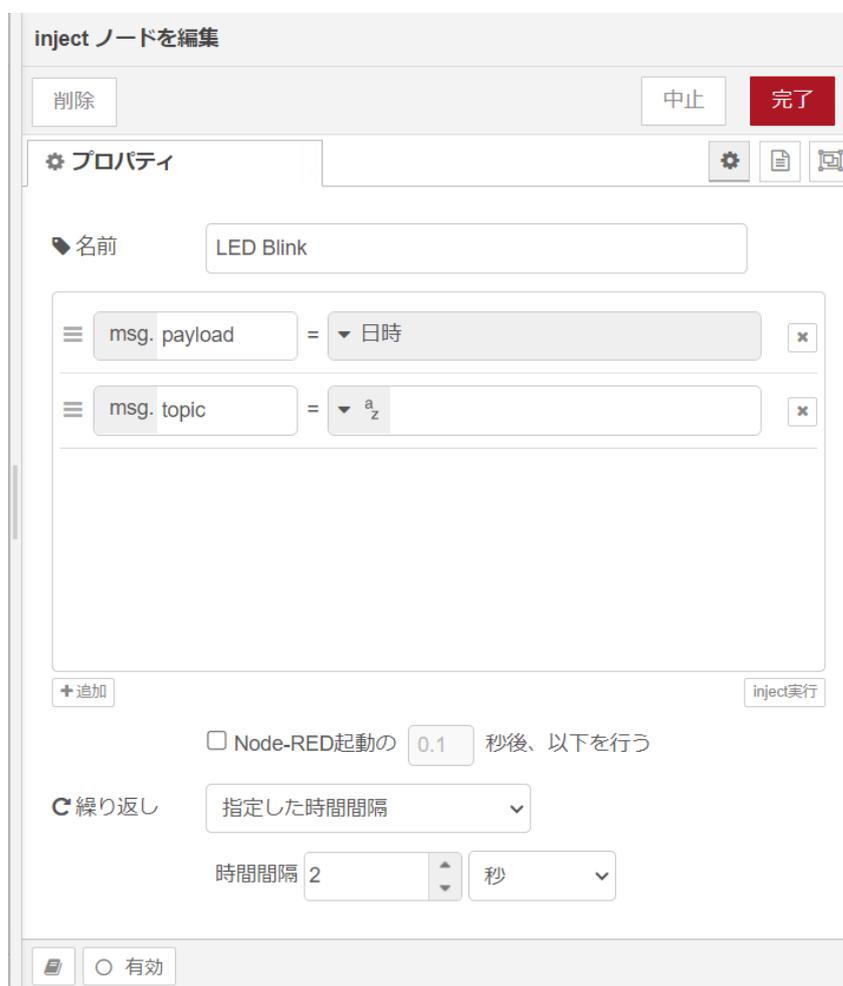


図 4.3 [inject] ノードのプロパティ内容

3. [trigger] ノードをドラッグ、ドロップします。
4. [trigger] ノードのプロパティを編集します。[trigger] は 1 を送信して 1 秒待機後、0 を送信します。これにより点滅動作を実現します。

送信データ : 1
 送信後の処理 : 指定した時間待機
 1 秒
 再送信データ : 0



図 4.4 [trigger] ノードのプロパティ内容

5. [LED Blink] ノードの右側にある端子をクリックし、[trigger] ノードの左側の端子を選択して放します。
6. [write file] ノードをドラッグ、ドロップします。
7. [write file] ノードのプロパティを編集します。/sys/class/leds/green/brightness ファイルへ [trigger] からの送信データを書き込みます。

ファイル名: /sys/class/leds/green/brightness
 動作: ファイルを上書き

文字コード: デフォルト
 名前: green LED

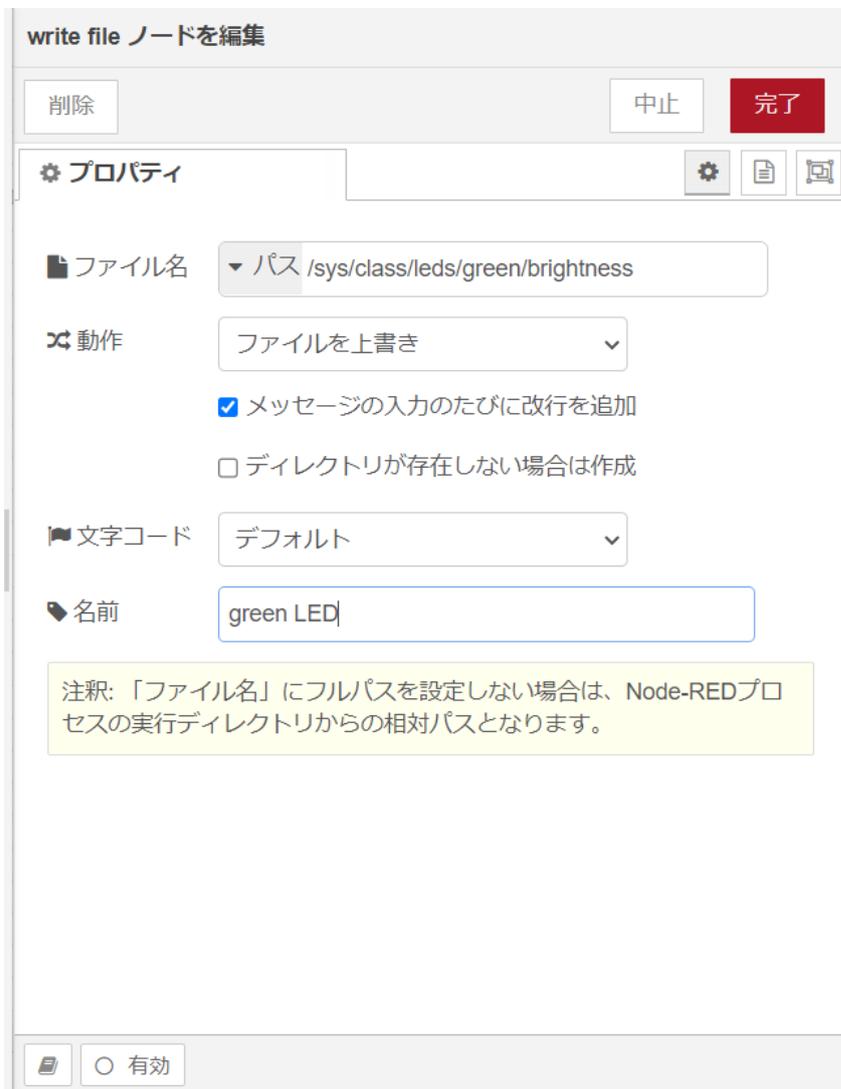


図 4.5 [write file] ノードのプロパティ内容

8. [trigger] ノードの右側にある端子をクリックし、[green LED] ノードの左側の端子を選択して放します。
9. [debug] ノードをドラッグ、ドロップします。
10. [green LED] ノードの右側にある端子をクリックし、[debug] ノードの左側の端子を選択して放します。

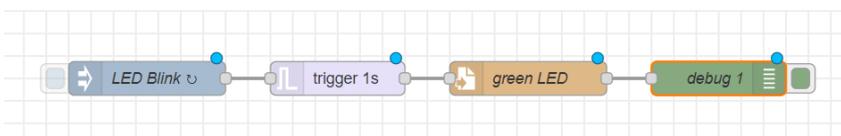


図 4.6 LED を 1 秒間隔で点滅するフロー

11. 画面右上の [デプロイ] を押します。ユーザー LED(緑) が1秒毎に点滅を繰り返す動きをすれば成功です。

4.4.2. CPU の測定温度を取得する

ここでは、Armadillo-640 の CPU の測定温度を 1 秒間隔で取得するフローを作成します。全て Node-RED のコアノードを使用します。/sys/class/thermal/thermal_zone0/temp ファイルの値を読み出すことによって測定温度を取得することができます。温度はミリ°C の単位で表示されるため、°C 単位への変換も行います。

1. パレットから [inject] ノードをワークスペースにドラッグ、ドロップします。
2. [inject] ノードのプロパティを編集します。[inject] は 1 秒間隔で実行します。

繰り返し: 指定した時間間隔
時間間隔: 1 秒



図 4.7 [inject] ノードのプロパティ内容

3. [read file] ノードをドラッグ、ドロップします。

- [read file] ノードのプロパティを編集します。/sys/class/thermal/thermal_zone0/temp ファイルの値を読み出します。

ファイル名: /sys/class/thermal/thermal_zone0/temp
 出力形式: 文字列
 文字コード: デフォルト
 名前: CPU temp



図 4.8 [read file] ノードのプロパティ内容

- [Get CPU temp] ノードの右側にある端子をクリックし、[CPU temp] ノードの左側の端子を選択して放します。
- [function] ノードをドラッグ、ドロップします。
- [function] ノードのプロパティを編集します。CPU の測定温度がミリ°C の単位のため、°C の単位へ変換します。msg.payload で渡された値を 1000 で割り、msg.payload に戻します。

```

名前: CPU temp calc
コード:
msg.payload = msg.payload / 1000;
return msg;
    
```

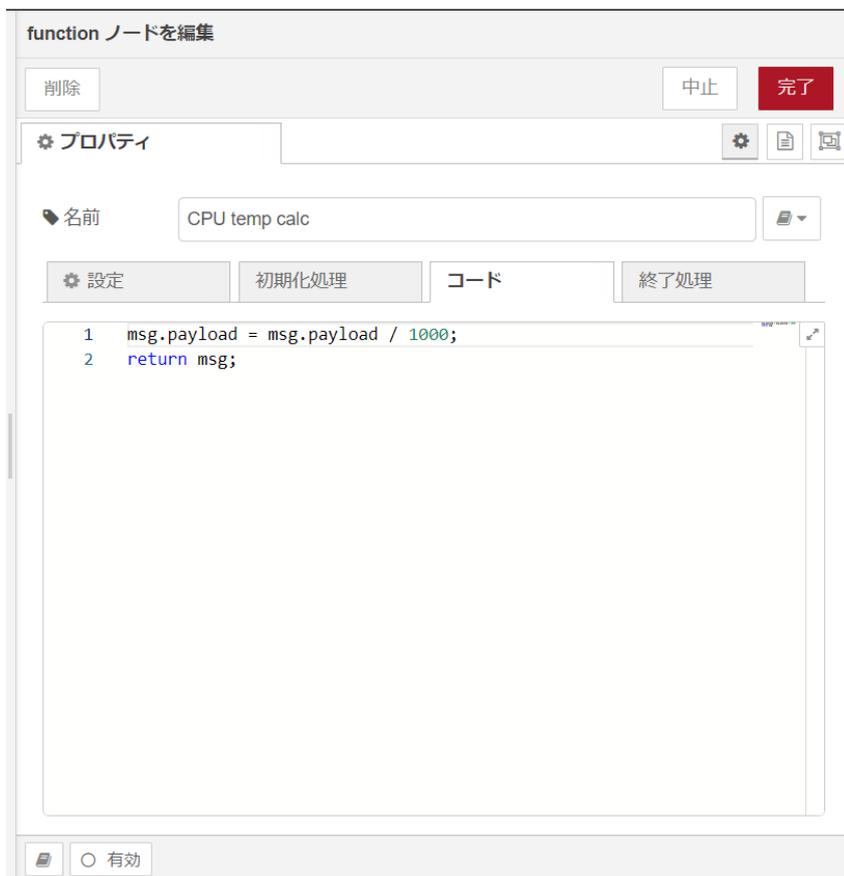


図 4.9 [function] ノードのプロパティ内容

8. [CPU temp] ノードの右側にある端子をクリックし、[CPU temp calc] ノードの左側の端子を選択して放します。
9. [debug] ノードをドラッグ、ドロップします。
10. [CPU temp calc] ノードの右側にある端子をクリックし、[debug] ノードの左側の端子を選択して放します。

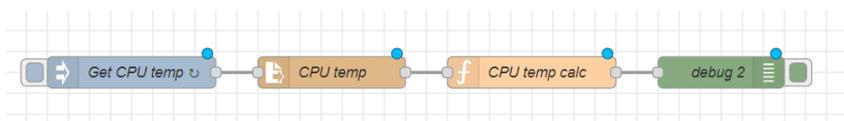


図 4.10 CPU の測定温度を 1 秒間隔で取得するフロー

11. 画面右上の [デプロイ] を押します。デバッグメッセージに 1 秒毎に CPU の測定温度が表示されます。

4.4.3. RS-485 modbus RTU 読み出しを行う

ここでは、USB RS-485 変換器を使用した Modbus RTU 読み出し用フローを作成します。Node-RED のコアノードのほかに modbus-client ノード、Modbus-Read ノード、modbus-response ノードを使用します。RS-485 シリアルインターフェースのデバイスファイルは、`/dev/ttyUSB0` を使用します。USB の場所については「3.3. インターフェースレイアウト」をご確認ください。今回は以下の RS-485 通信対応デバイスを想定した場合の設定内容となります。実際に動作確認する場合は、使用する RS-485 通信対応デバイスの設定に変更してください。

```
ユニット ID: 1  
通信プロトコル: Modbus RTU  
ボーレート: 9600  
読み出しアドレス: 0x00  
ファンクションコード: 1
```

1. パレットから [Modbus-Read] ノードをワークスペースにドラッグ、ドロップします。
2. [Modbus-Read] ノードのプロパティを編集します。
3. 先に Server を設定する必要があります。[新規に modbus-client を追加] の右隣の編集ボタンを押して [modbus-client] を作成します。

```
名前: RS485 slave device  
Type: Serial  
Serial port: /dev/ttyUSB0  
Serial type: RTU  
Baud rate: 9600  
Unit-Id: 1
```

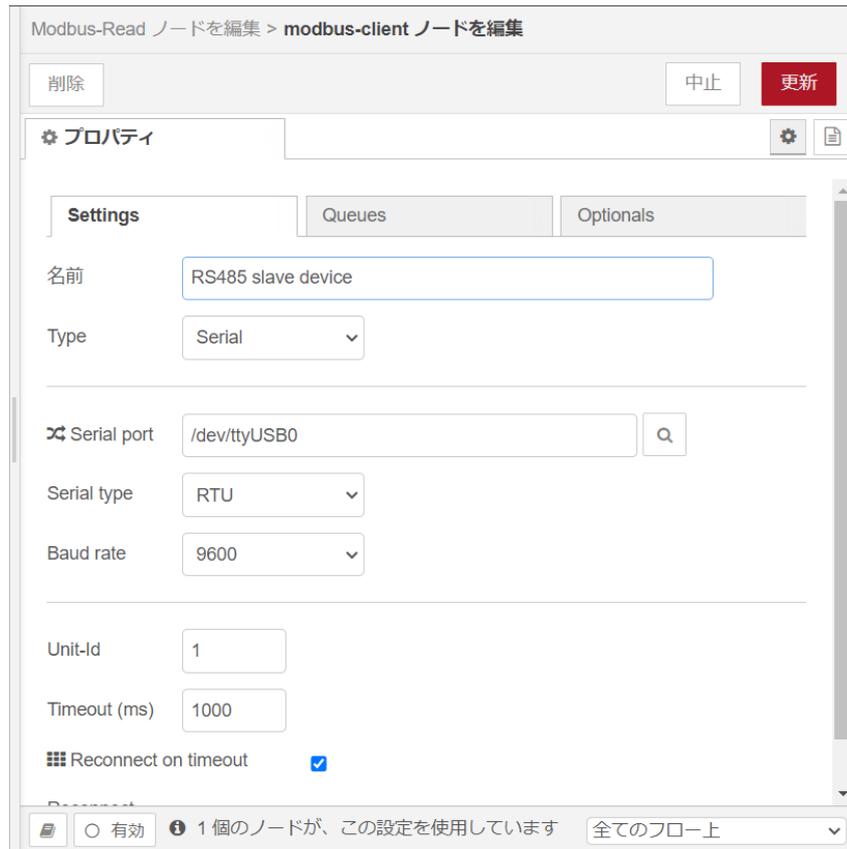


図 4.11 [modbus-client] ノードのプロパティ内容

4. [Modbus-Read] ノードのプロパティを編集します。

```

名前: slave device
トピック:
Unit-Id:
FC: FC 1:Read Coil Status
Address: 0
Quantity: 1
Poll Rate: 1 Second(s)
Server: RS485 slave device
    
```

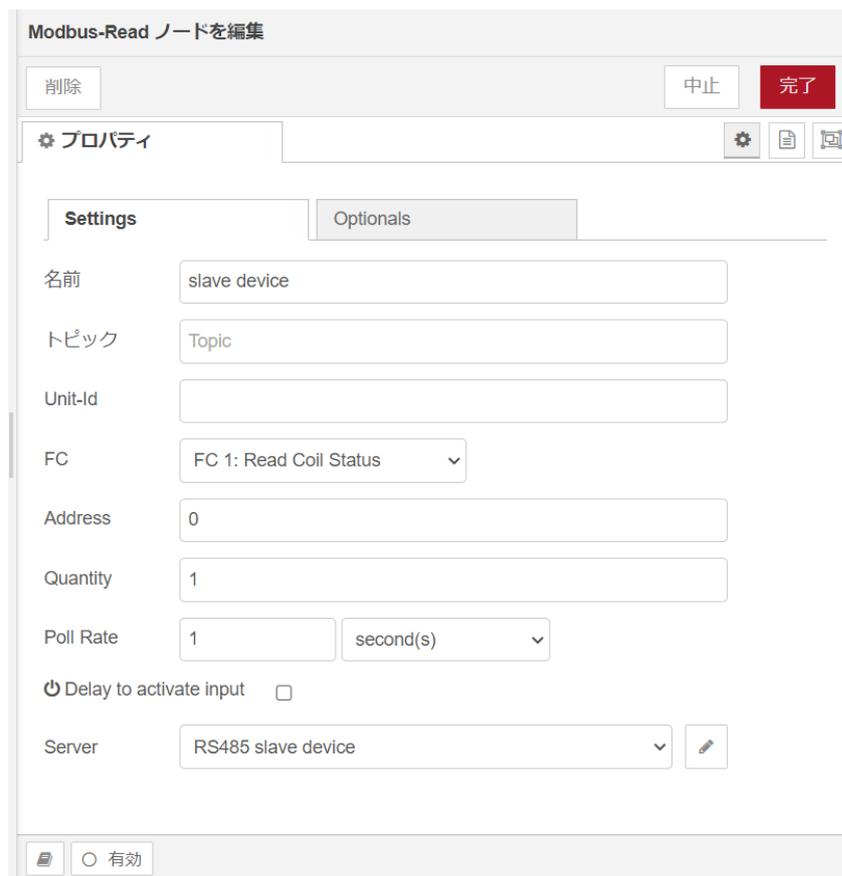


図 4.12 [Modbus-Read] ノードのプロパティ内容

5. [Modbus-Response] ノードをドラッグ、ドロップします。
6. [slave device] ノードの右側にある端子をクリックし、[Modbus-Response] ノードの左側の端子を選択して放します。

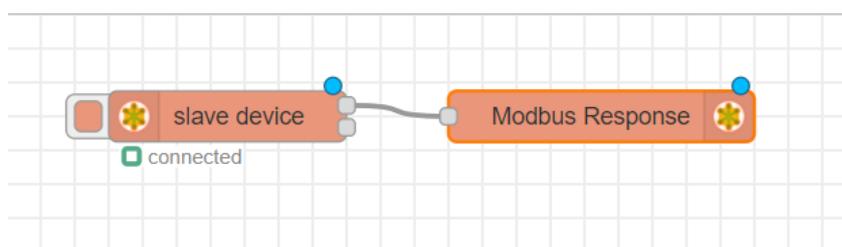


図 4.13 RS-485 を使用した Modbus RTU 読み出し用フロー

7. 画面右上の [デプロイ] を押します。読み出しに成功すると 1 秒毎に読みだした値が更新され [Modbus-Response] ノードの下に表示されます。

4.4.4. CPU の測定温度のグラフをダッシュボードに表示する

「4.4.2. CPU の測定温度を取得する」 で取得した CPU 温度をダッシュボードにグラフとして表示するフローを作成します。

1. サイドバーの右端にある三角形のボタンを押し、[ダッシュボード] を選択します。

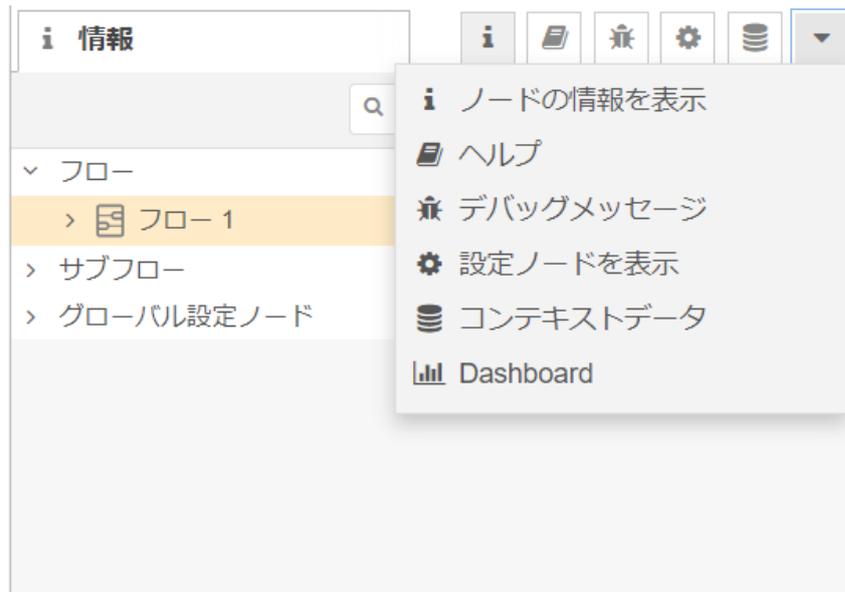


図 4.14 [ダッシュボード]を選択する

2. [ダッシュボード] 編集画面が表示されることを確認します。



図 4.15 ダッシュボード編集画面

3. [+タブ] ボタンを押して [Tab1] を追加します。



図 4.16 ダッシュボードに [Tab1] を追加

4. [Tab1] プロパティを編集します。

名前: Armadillo
アイコン: dashboard



図 4.17 [Tab1] プロパティ内容

5. [Armadillo] タブが追加されました。

名前: Armadillo
 アイコン: dashboard



図 4.18 ダッシュボード編集画面に [Armadillo] タブが追加

6. [Armadillo] 横の [+グループ] ボタンを押して [Group1] を追加します。

名前: Armadillo
アイコン: dashboard



図 4.19 ダッシュボード編集画面に [Group1] グループが追加

7. [Group1] プロパティを編集します。

名前: Group 1 タブ: Armadillo 種類: 幅: 15 <input type="checkbox"/> グループ名を表示する



dashboard group ノードを編集

削除 中止 更新

プロパティ

名前 Group 1

タブ Armadillo

種類 ウィジェット用のCSSクラス名 (オプション)

幅 15

グループ名を表示する

有効 1 個のノードが、この設定を使用しています

図 4.20 [Group1] プロパティ内容

- [chart] ノードをワークスペースにドラッグ、ドロップします。ダッシュボード編集画面の [Group1] グループに追加されたことを確認します。



図 4.21 ダッシュボード編集画面に [chart] ノードが追加

9. [chart] ノードのプロパティを編集します。

グループ: [Armadillo] Group 1
サイズ: 自動
ラベル: CPU temp
種類: 折れ線グラフ
X 軸: 直近 1 時間
X 軸ラベル: HH:mm:ss
Y 軸: 最小 20 最大 50
凡例: 非表示 補完: 直線



図 4.22 [inject] ノードのプロパティ内容

10. [CPU temp calc] ノードの右側にある端子をクリックし、[CPU temp] ノードの左側の端子を選択して放します。

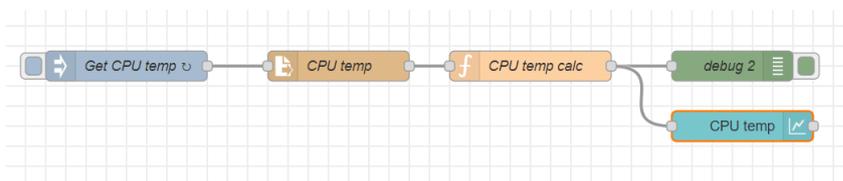


図 4.23 CPU の測定温度のグラフをダッシュボードに表示するフロー

11. 画面右上の [デプロイ] を押します。ダッシュボード編集画面の [テーマ] タブの右側にある四角に矢印が重なったボタンを選択すると、ダッシュボードが表示されます。

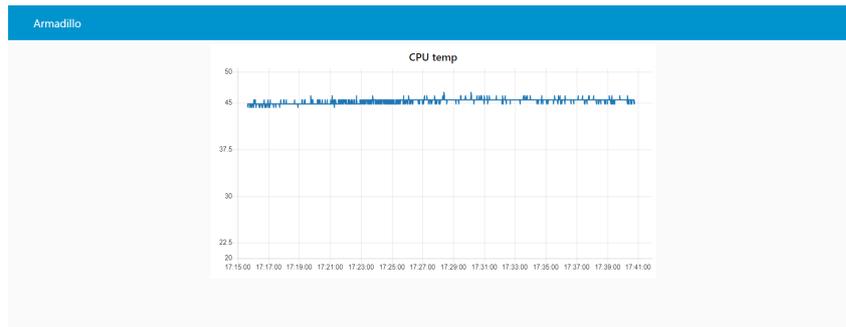


図 4.24 CPU の測定温度のグラフのダッシュボード

12. ダッシュボードの CPU 温度のグラフは一秒毎に更新されます。

4.4.5. 外部プログラムを実行する

ここでは、外部プログラムを実行しその結果を取得するフローを作成します。外部プログラムは [exec] ノードで実行することができます。ここでの外部プログラムとはシステムコマンドやユーザー自身が作成したプログラムのことを指します。

例として date コマンドを実行するフローを作成します。

1. パレットから [inject] ノードをワークスペースにドラッグ、ドロップします。プロパティはデフォルトから変更ありません。
2. [exec] ノードをドラッグ、ドロップします。
3. [exec] ノードのプロパティを編集します。

```
コマンド: date  
引数: なし
```



図 4.25 [exec] ノードのプロパティ内容

4. [infect] ノードの右側にある端子をクリックし、[exec] ノードの左側の端子を選択して放します。
5. [debug] ノードをドラッグ、ドロップします。
6. [exec] ノードの右側の一番上にある端子をクリックし、[debug] ノードの左側の端子を選択して放します。

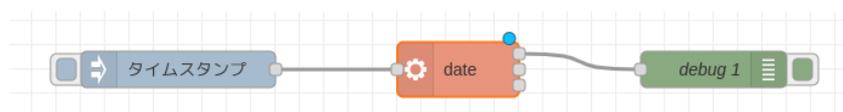


図 4.26 外部プログラムを実行するフロー

7. 画面右上の [デプロイ] を押します。

8. [inject] ノードの左の四角を押すと [exec] ノードに設定した date コマンドが実行され標準出力の結果がデバッグメッセージに表示されます。

[exec] ノードの右の端子は上からそれぞれ「標準出力」「標準エラー出力」「返却コード」となっており、取得したい出力によって使い分けることができます。

4.4.6. Node-RED を終了する

ここでは、Node-RED を任意のタイミングで終了するフローを作成します。Node-RED のコアノードのほかに exit ノードを使用します。



Node-RED 終了後、Node-RED を再起動するためには Armadillo-640 の電源を再投入する必要があります。

1. パレットから [inject] ノードをワークスペースにドラッグ、ドロップします
2. [inject] ノードのプロパティを編集します。

名前: Finish Node-RED
繰り返し: なし



図 4.27 [inject] ノードのプロパティ内容

3. [exit] ノードをドラッグ、ドロップします。
4. [exit] ノードのプロパティを編集します。

```
Name: exit  
Exit code: 0
```

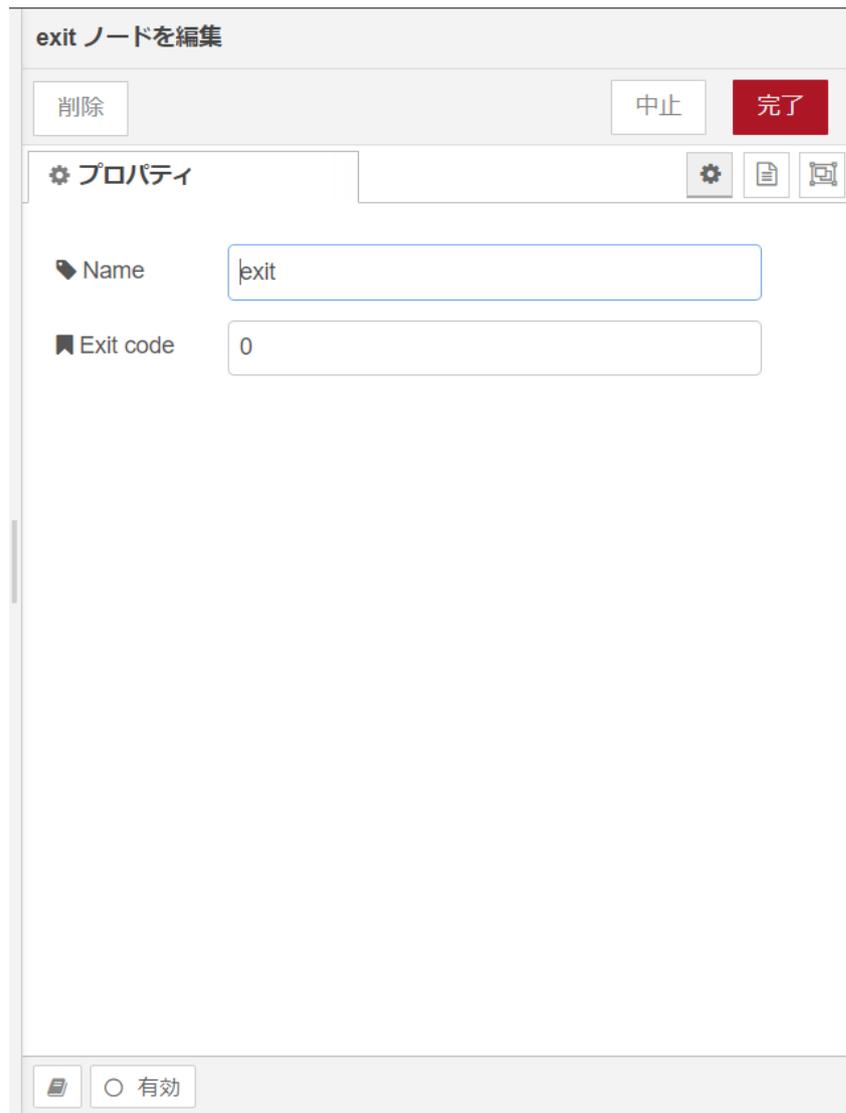


図 4.28 [exit] ノードのプロパティ内容

5. [Finish Node-RED] ノードの右側にある端子をクリックし、[exit] ノードの左側の端子を選択して放します。

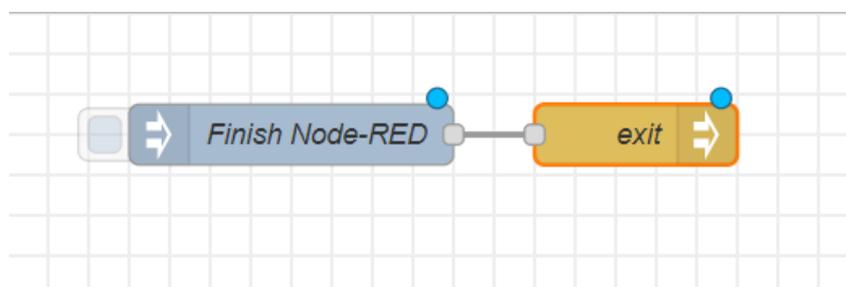


図 4.29 Node-RED を任意のタイミングで終了するフロー

6. 画面右上の [デプロイ] を押します。[Finish Node-RED] ノードの左側にあるボタンを押すと Node-RED が終了します。

4.5. ユーザデータを削除する

Node-RED に関するユーザデータは `/var/app/rollback/volumes/node-red/root` に保存されます。当該ディレクトリを削除した場合はユーザデータは全て削除されます。この場合 Node-RED のブラウザからインストールしたカスタムノードは削除されます。

改訂履歴

バージョン	年月日	改訂内容
1.0.0	2023/12/26	・ 初版発行
1.1.0	2024/01/29	・ 「4.4.5. 外部プログラムを実行する」 を追加

