

RZ/A2M グループ

RZ/A2M Azure RTOS Package for GR-MANGO クイックスタートガイド

1. 要旨

本書は、RZ/A2M を搭載した株式会社コア製の GR-MANGO ボードと e2 studio 環境で動作する RZ/A2M Software Package for GR-MANGO のクイックスタートガイドになります。

本書は、以下 Web サイトに公開するサンプルプロジェクトを実行する手順について記載しています。

https://github.com/renesas-rz/rza2_gcc_azure_rtos_bsp

2. 準備

2.1 ツール

RZ/A2M Software Package for GR-MANGO は、以下の環境で動作致します。ご確認ください。

ツール:

- IDE: e2 studio 2021-04 Windows 64-bit product version or later
- Tool Chain: GNU ARM Embedded Toolchain 6-2017-q2-update

Tool Chain は IDE に同梱されています。個別では以下のサイトから入手可能です。

<https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-rm/downloads/6-2017-q2-update>

e2 studio のインストール方法は以下文書を参照ください。

— [e2 studio Integrated Development Environment User's Manual: Getting Started](#)

ターゲットボード:

GR-MANGO

ICE (In-circuit emulator):

GR-MANGO は ARM Mbed DAPLink をサポートしており、ICE を使用せず、プログラムのダウンロード、デバッグが可能です。DAPLink は以下の機能をサポートしています。

- drag-and-drop programming (MSC)
- a virtual serial port (CDC)
- CMSIS-DAP based debugging (HID)

DAPLink の詳細は以下のページを参照ください。

<https://os.mbed.com/handbook/DAPLink>

ブートローダー:

本パッケージには、ブートローダーがテーブルデータとして実装されています。ソースコードを入手したい場合、以下より入手ください。

<https://www.mxic.com.tw/en-us/support/technical-documentation/Pages/Serial-NOR-Flash.aspx>

2.2 USBシリアルポート接続

GR-MANGO の CN1 は、シリアルポート接続を提供します。最初に接続した時、適切なドライバを PC が自動的に見つけてインストールします。シリアルポートに割り当てられた COM ポート番号は、Windows™ デバイスマネージャーで確認することが可能です。

2.3 シリアルターミナル

— シリアルターミナルソフト(PuTTY、HyperTerminal、Tera Term など)の設定を以下に記載します。

ボーレート:	115200
データビット:	8
パリティ:	無し
ストップビット:	1
フロー制御:	無し
COM ポート:	Windows™ デバイスマネージャー参照

3. サンプルプロジェクトの立ち上げ

3.1 開発環境へのインポート

本パッケージは、アーカイブファイルで提供しておりビルドする環境を e2 studio にインポートすることができます。本章の手順に従うことでの、ユーザが各サンプルプロジェクトのビルド環境をインポートすることができます。

- パッケージを入手します。
- パッケージを展開します。
- 使用するプロジェクトの zip ファイルを短いパスのフォルダに展開します。
- スタートメニューから e2 studio を起動します。
- 各サンプルプロジェクトのサブディレクトリの上にあるトップディレクトリをワークスペースディレクトリに設定します(図 3-1)。

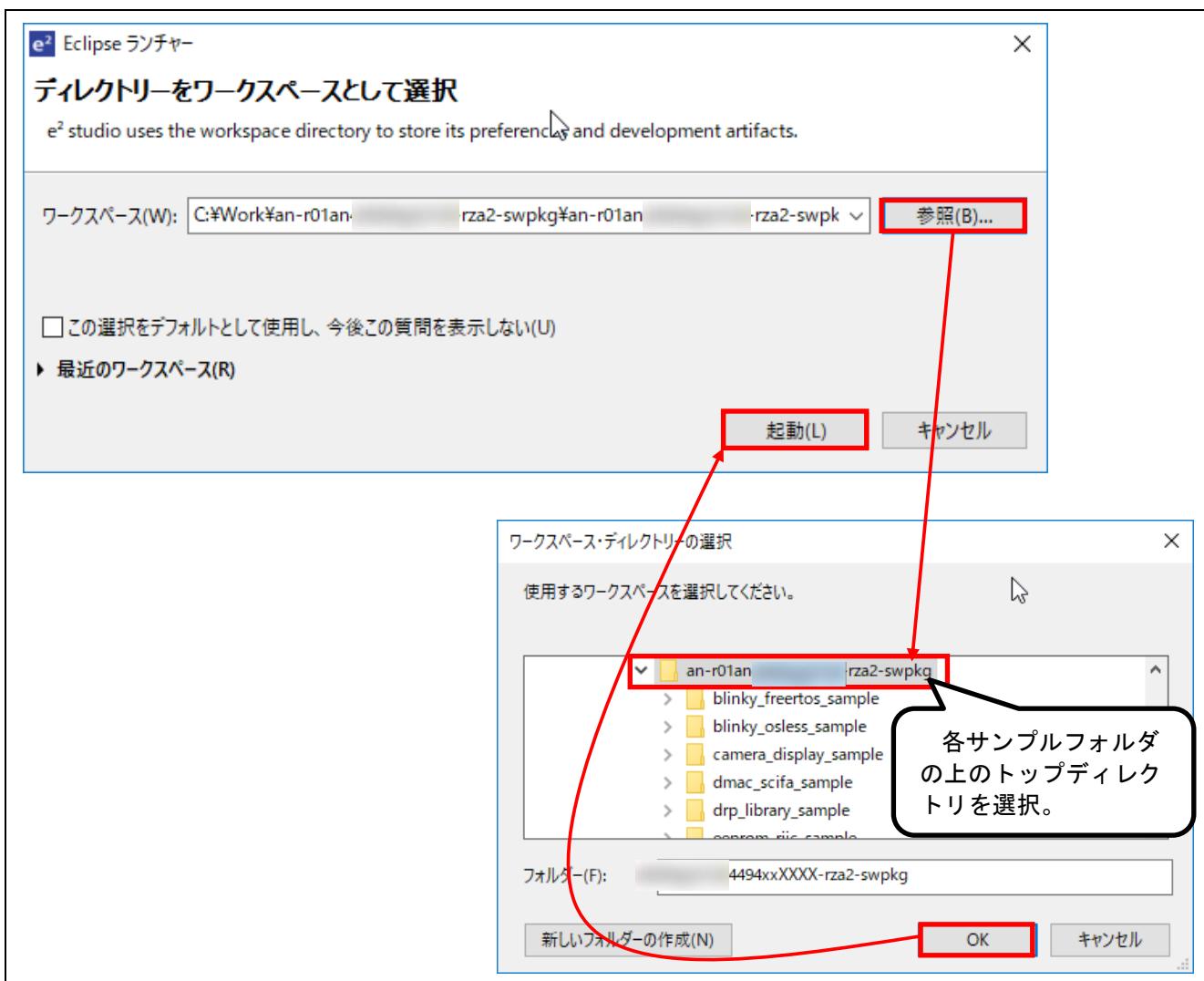


図 3-1 e2 studio の起動

— e2 studio の Welcome to e2 studio 画面で "ワークベンチ" をクリックします(図 3-2)。

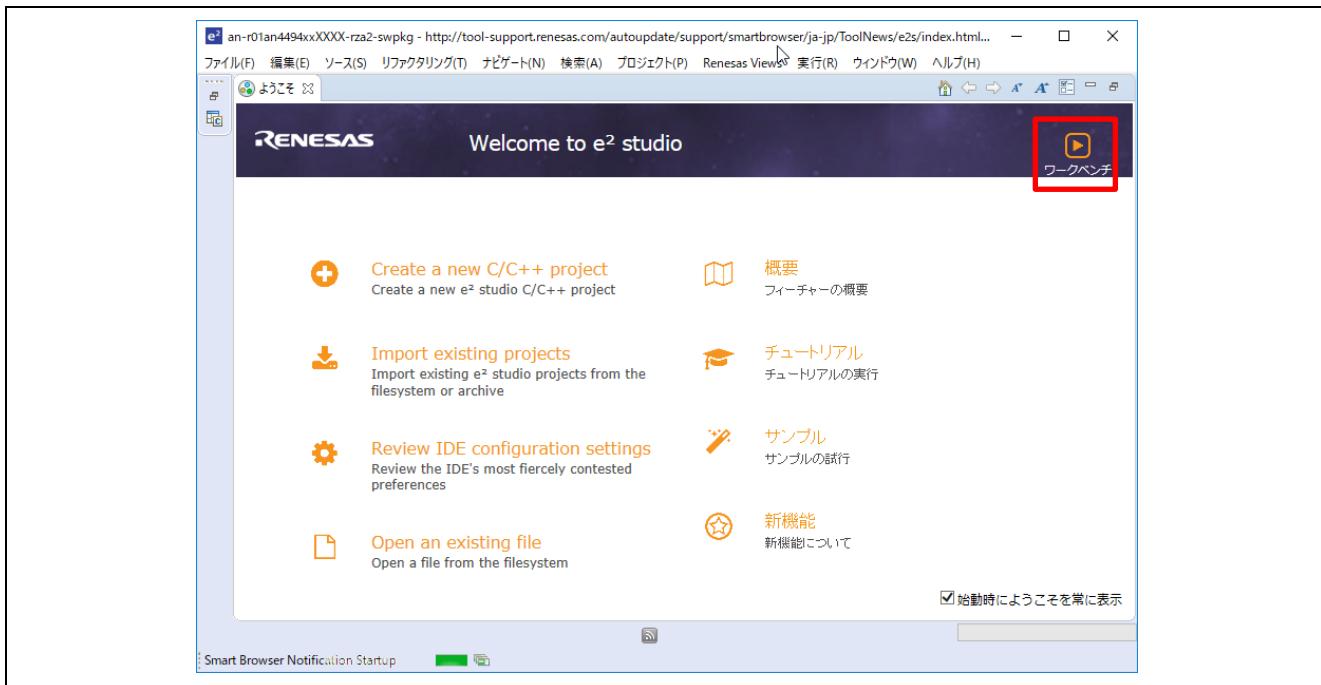


図 3-2 ワークベンチの切り替え

— プロジェクトエクスプローラーウィンドウで右クリックし "インポート" を選択します(図 3-3)。

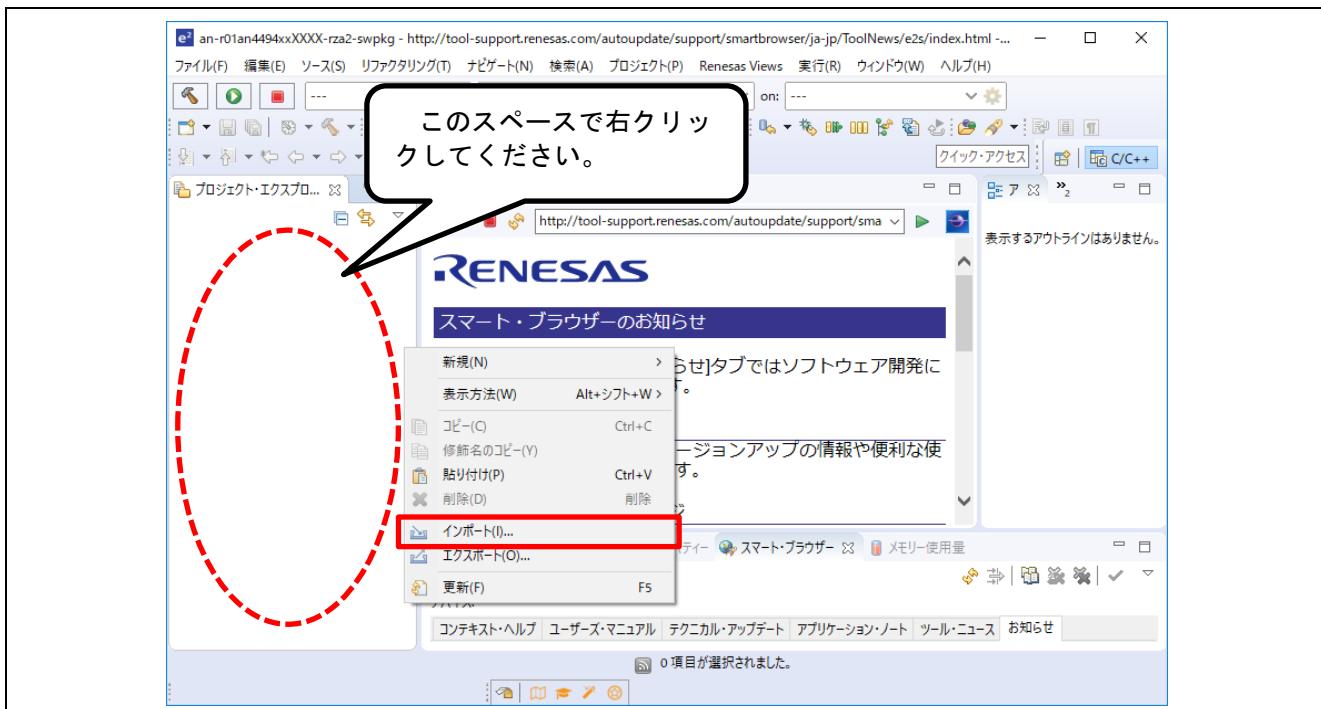


図 3-3 インポートの選択

- インポートダイアログで "一般" → "既存のプロジェクトワークスペースへ" を選択し、"次へ" のボタンをクリックします(図 3-4)。

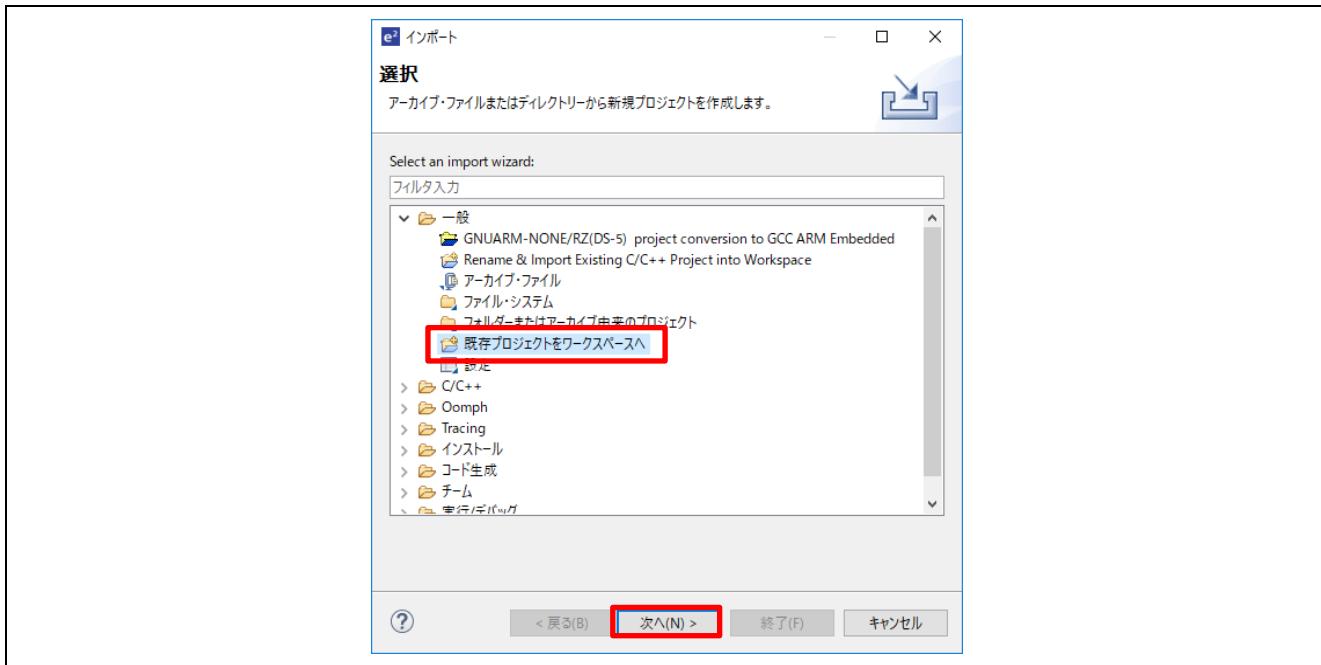


図 3-4 インポートダイアログ

- "ルートディレクトリの選択" の右にある "参照" ボタンを選択すると、"フォルダ参照" ダイアログが表示されます。そのまま "OK" ボタンを押してください(図 3-5)。

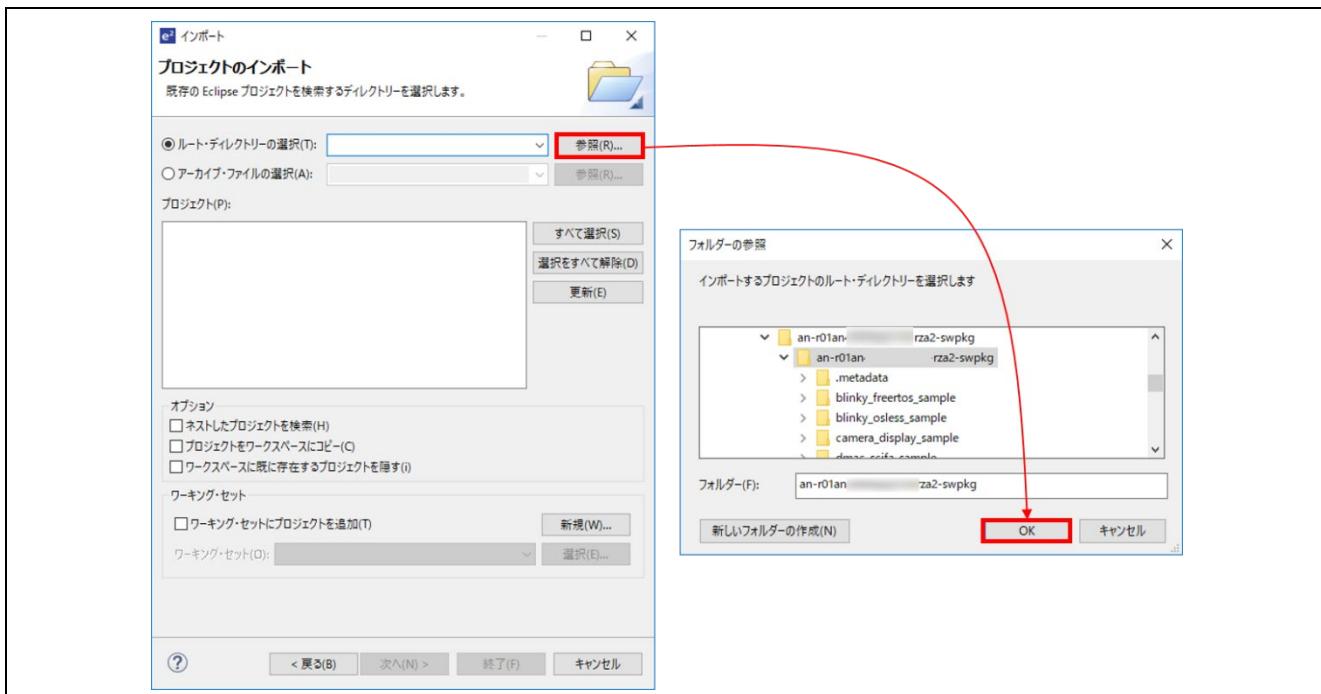


図 3-5 ルートディレクトリに選択

- ターゲットプロジェクトがチェックされていることを確認し "終了" ボタンをクリックします(図 3-6)。
 (注:次の図のプロジェクトは参考になります。実際にご使用になりたいターゲットプロジェクト名を選択してください。)

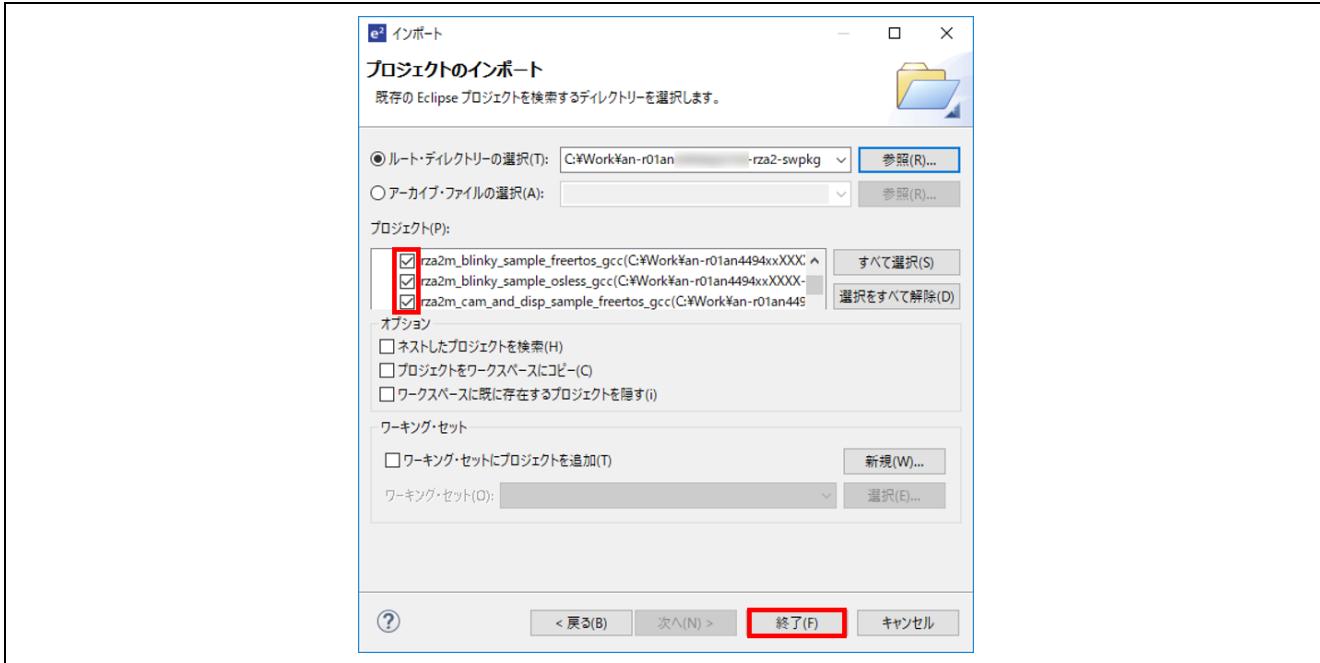


図 3-6 ターゲットプロジェクトのインポート

- ターゲットプロジェクトのインポートが完了すると、プロジェクトエクスプローラーで確認することができます(図 3-7)。

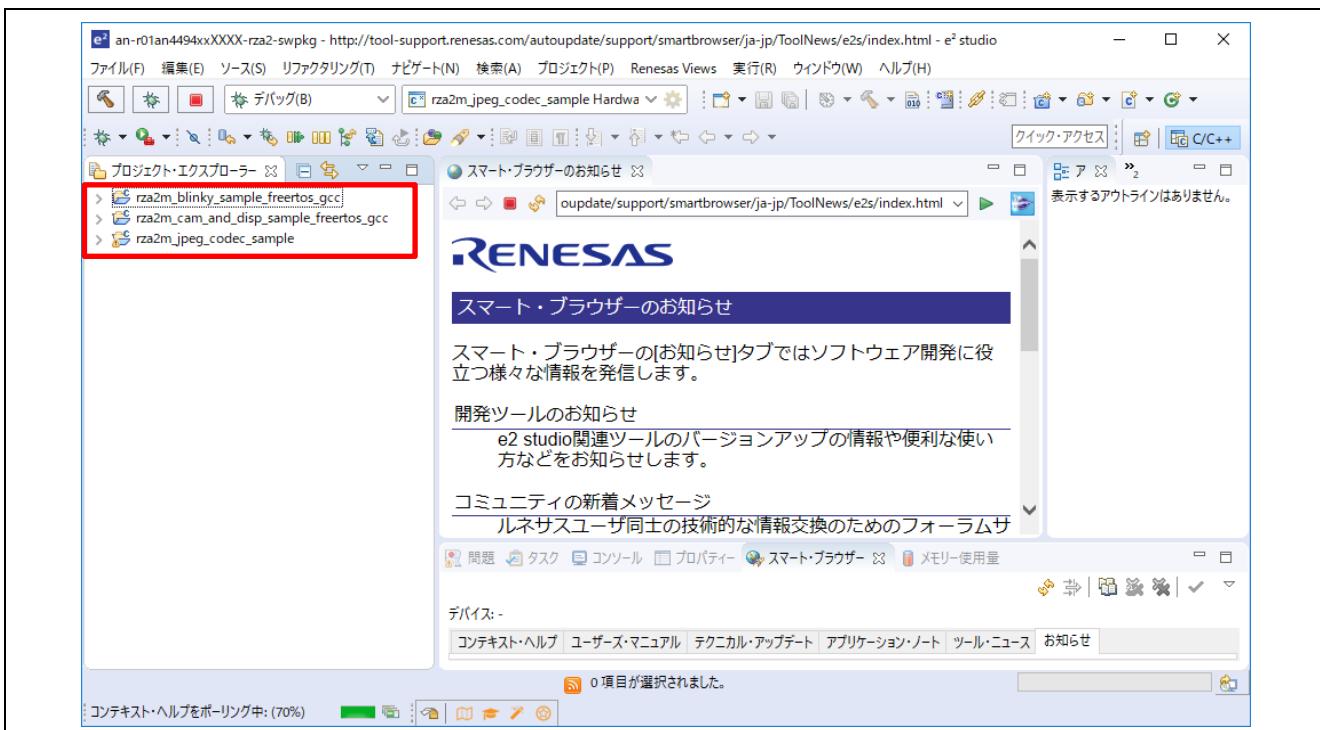


図 3-7 プロジェクトエクスプローラーでの確認

3.2 プロジェクトのビルドと評価ボードへのダウンロード

- ターゲットプロジェクトを選択しビルドボタン(トンカチアイコン)の横にある矢印をクリックし、ドロップダウンメニューから"Hardware Debug"を選択するとビルドが開始されます(図 3-8)。次回からは、ビルドボタン(トンカチアイコン)でビルドが可能になります。

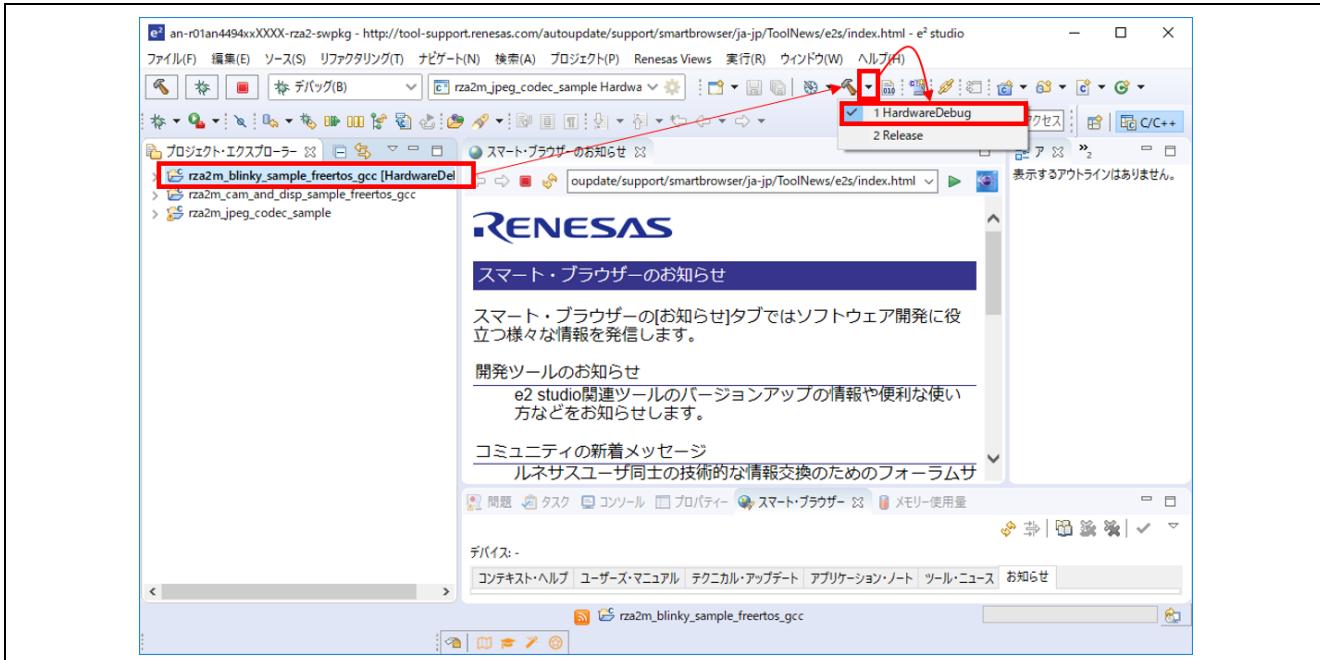


図 3-8 ターゲットプロジェクトのビルド

- ビルドが開始されコンソールウィンドウでそのステータスを確認することができます(図 3-9)。
(注:ワークスペースフォルダまでのパスの長さに注意してください。パスが長すぎるとビルドエラーになる場合があります。)

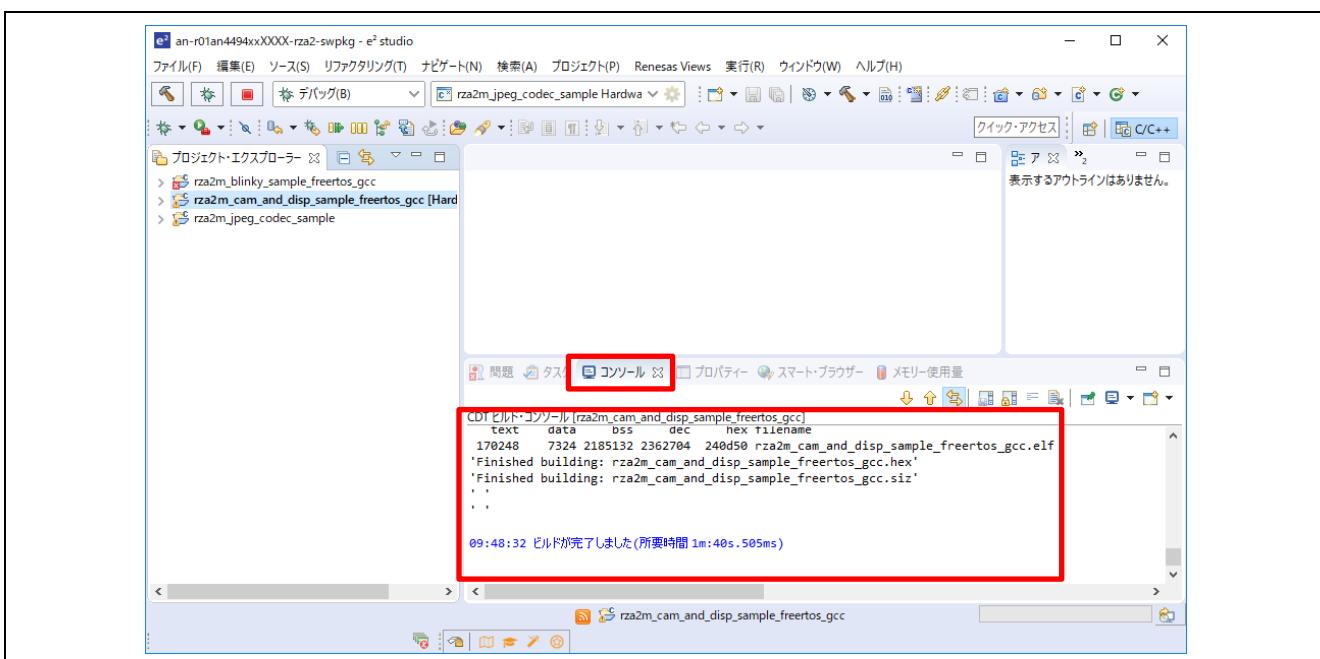


図 3-9 ビルドステータスの確認

- ビルドが完了したら、対象プロジェクトの"HardwareDebug"ディレクトリ内にバイナリファイルが生成されます。GR-MANGO の CN1 と PC を USB ケーブルで接続すると、PC は GR-MANGO をMBED ドライブとして認識します。生成したバイナリファイルを MBED ドライブへドラッグ&ドロップすることで、プログラムをダウンロードできます。

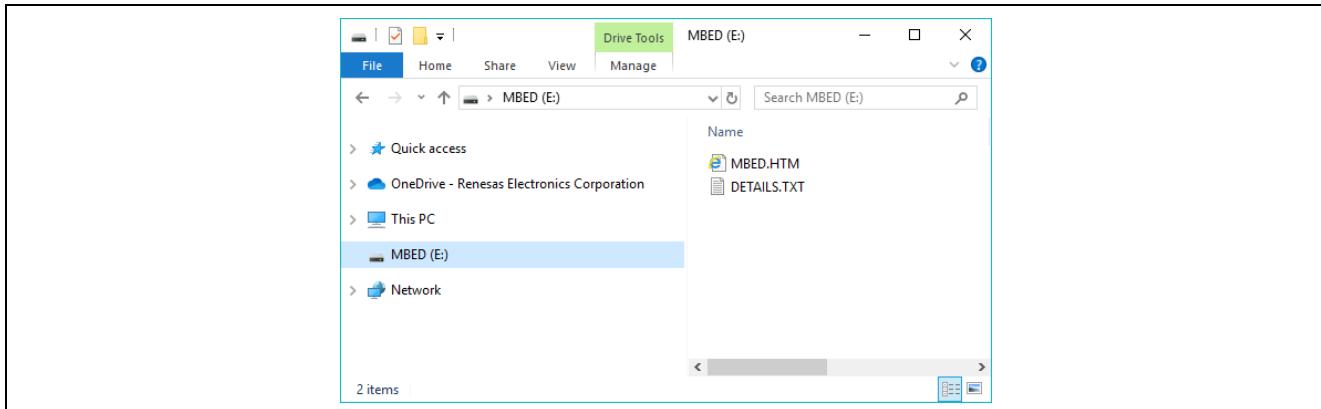


図 3-10 プログラムのダウンロード

- ダウンロード後、GR-MANGO 上の Reset ボタンを押下し、プログラムを実行します。

3.3 CMSIS-DAP によるデバッグ

GR-MANGO は ARM Mbed DAPLink をサポートしており、OpenOCD によるデバッグが可能です。GR-MANGO での OpenOCD デバッグ方法は、以下を参照してください。

<https://os.mbed.com/teams/Renesas/wiki/How-to-debug-with-e2-studio>

4. ドライバ、ミドルウェアの追加

本章では、本パッケージ同梱のプロジェクトに対して、ドライバ、ミドルウェアを追加する方法を記載します。

RZ/A2M Software Package for GR-MANGO では、ドライバ、ミドルウェアをコンポーネントとして管理しており、e2 studio 上からコンポーネントを追加することができます。

各ドライバ、ミドルウェアの使用例は、RZ/A2M Simple Applications Package for GR-MANGO(R01AN5595)の各サンプルプロジェクトをご参照ください。

各ドライバ、ミドルウェアを e2 studio にインストールする方法など、スマート・コンフィギュレータの使用法は、[RZ/A2M スマート・コンフィグレータ ユーザーガイド: e² studio 編\(R20AN0583\)](#)を参照ください。

4.1 コンポーネントの追加方法

- e2 studio のプロジェクトエクスプローラーウィンドウで、対象プロジェクトのプロジェクトツリーを開き、その中にある.scfg ファイルをダブルクリックします。
- コンポーネントタブに移動し、コンポーネントの追加ボタンから、対象のコンポーネントを追加します。
- 対象のコンポーネントを追加後、コードの生成ボタンをクリックします。

以上で、対象プロジェクトの generate¥sc_drivers と .settings¥smartconfigurator にコンポーネントが追加されます(図 4-1)。コンポーネント追加後は、3.2 章に示す手順に従い、ビルドを行ってください。

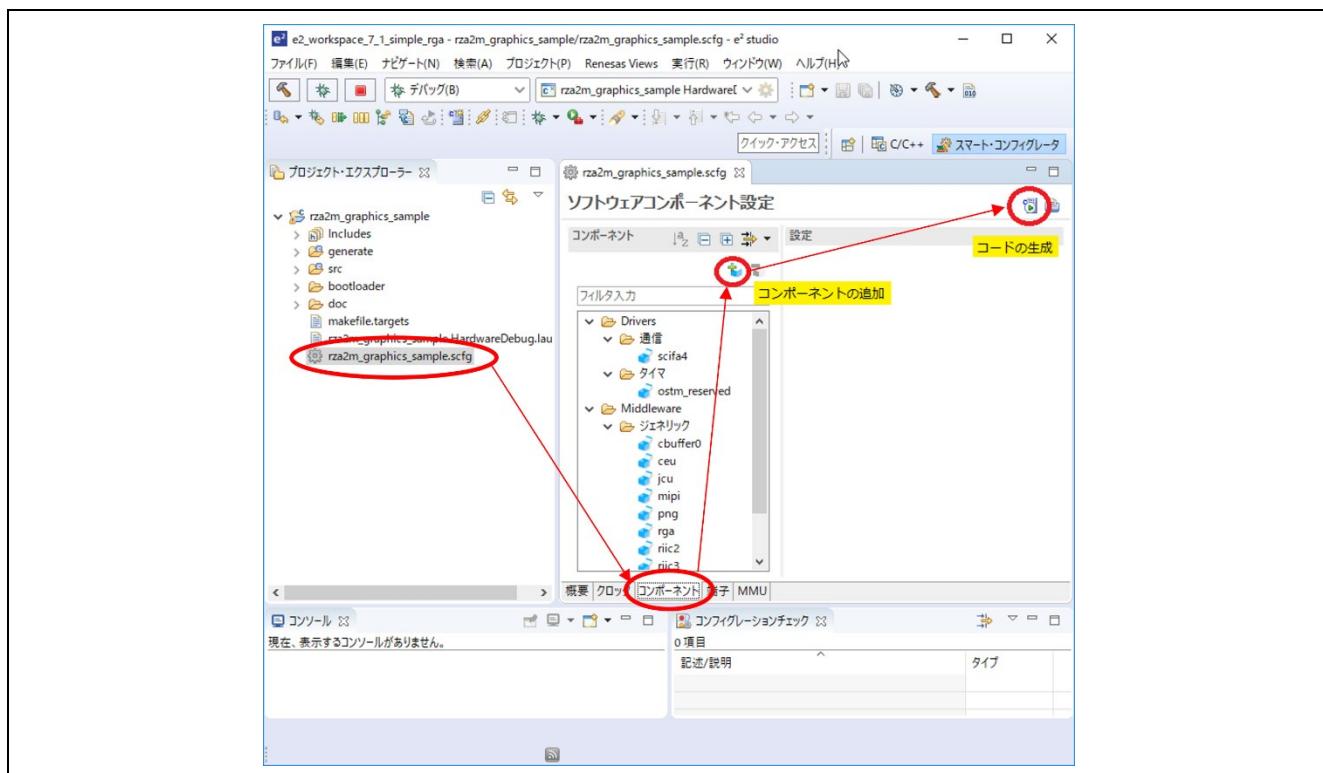


図 4-1 コンポーネントの追加方法

4.2 各コンポーネントの実装方法

各コンポーネントのプロジェクトへの実装は以下の手順で行います。

- 4.1 章の方法でコンポーネントの追加
- Smart Configurator によるコンポーネントの設定
- コンポーネントのコード生成
- コンポーネントのドキュメントで API 関数名や API 記載ファイル名を確認
- コンポーネントを使用しているプロジェクトを確認し、使用箇所を API 関数名や API 記載ファイル名で検索
- 検索した使用箇所を参考にプロジェクトに実装する

5. ThreadX デバッグ機能

この章では e² studio の ThreadX デバッグ機能について説明します。

この機能により、生成されたスレッド、キュー、タイマ等の一覧と状態をプログラム停止中に確認することができます。

5.1 使用方法

- 1 ThreadX を使用するプログラムをボードにダウンロードします。
- 2 ダウンロードしたプログラムを実行します。
- 3 プログラムを一時停止します。
- 4 「Renesas Views」メニューの「パートナーOS」を選択し、「RTOS リソース」を選択します(図 5-1)。

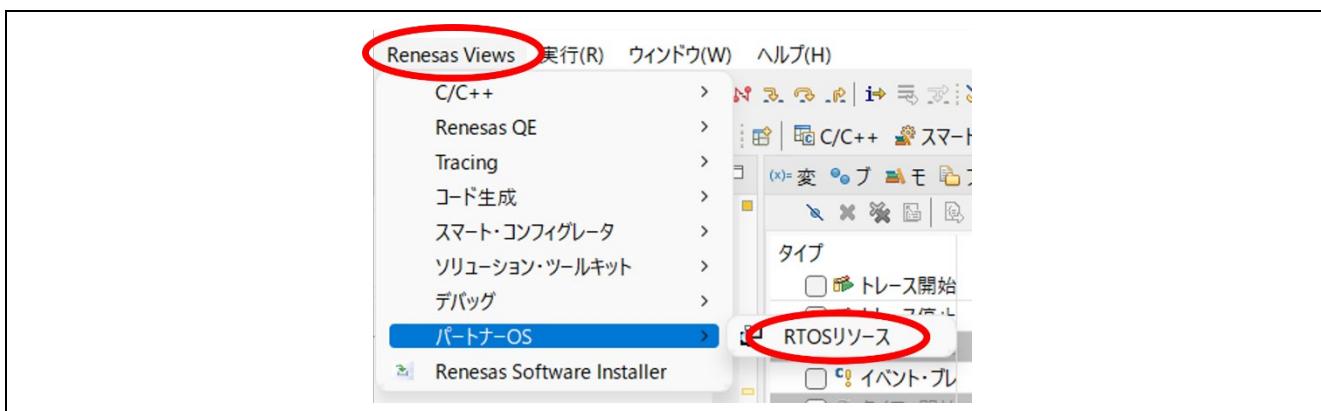


図 5-1 ThreadX RTOS デバッグ機能の使用方法

5 RTOS リソースの tab が表示されますので、[OS 選択][OS:]から "ThreadX"を選択し、[OK]ボタンを押します。

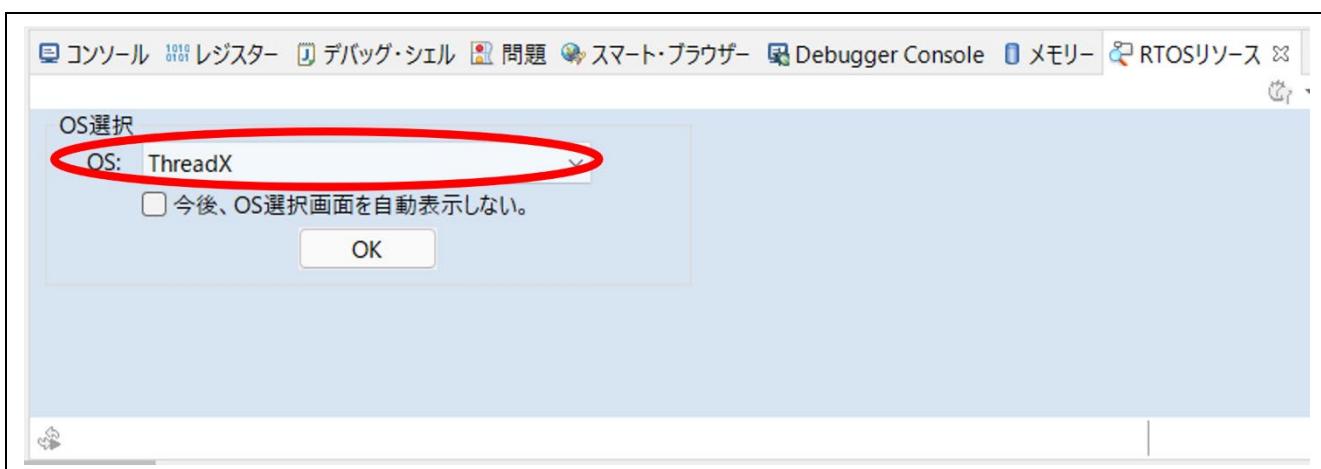


図 5-2 RTOS リソース表示(OS の選択)

6 カーネルオブジェクトのリソースが表示されます。各リソースの表示は、RTOS リソース tab の内側にあるリソース名の tab をクリックすることで、表示を切り替えることができます。

The screenshot shows the 'RTOSリソース' tab in the debugger interface. The table lists various resources and their current states:

No.	Name	Entry	Status	SuspendedFactor(ControlBlock*)	OwnedTX_MUTEX*(top)	Priority	RunCount
1	System...	_tx_tim...	SUSPE...			0	1183
2	thread 0	thread...	SLEEP			1	43
3	thread 1	thread...	READY	queue_0		16	845471
4	thread 2	thread...	RUNNI...	queue_0		16	845480
5	thread 3	thread...	SEMAP...	semaphore_0		8	1184
6	thread 4	thread...	SLEEP	semaphore_0		8	1184
7	EVENT

Run : thread 2 entry (No.:4) OS : ThreadX

図 5-3 RTOS リソース表示(リソースの表示)

6. Azure RTOS サンプルプロジェクト

このパッケージに含まれる各サンプルプロジェクトは、RZ/A2M を搭載した株式会社コア製の GR-MANGO ボードと e2 studio 環境で動作します。本章では、各サンプルプロジェクトを実行する手順について記載しています。

6.1 Azure RTOS コンポーネントのビルド

このパッケージには、NetX を除く Azure RTOS のすべてのコンポーネントのプロジェクトが含まれています。

Note : NetX は IPv4 専用の Azure RTOS コンポーネントです。パッケージのサンプルプロジェクトは IPv4/IPv6 デュアルスタックの NetXDuo を使用していますが、NetXDuo を NetX に置き換えることも可能です。

Azure RTOS のコンポーネントは、各サンプルプロジェクトを実行する前に e2 studio 環境にインポートし、ビルドしてライブラリファイルを作成しておく必要があります。

以下の手順で各コンポーネントプロジェクトをビルドし、ライブラリファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、ビルド対象のコンポーネントプロジェクトを選択します。
2. 選択したプロジェクトが、[Debug]ビルド構成になっていることを確認します。

Note : 下図は"filex"コンポーネントプロジェクトのビルド構成が[Debug]になっている場合です。
Azure RTOS コンポーネントのビルド構成は、[Debug]または[Release]のどちらかを選択します。サンプルプロジェクトを[Release]ビルド構成で作成したい場合は、すべての Azure RTOS コンポーネントのビルド構成を[Release]に設定してビルドを実行してください。

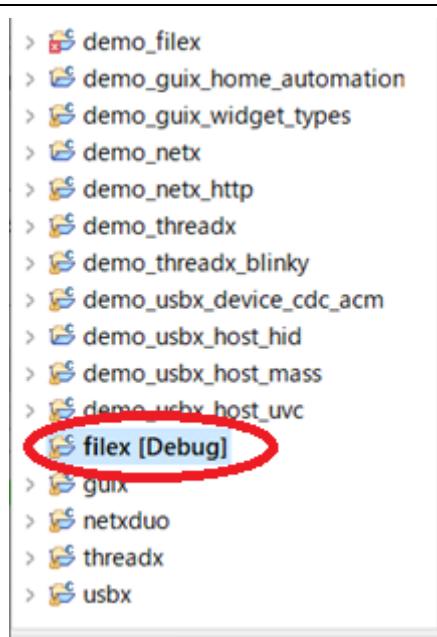


図 6-1 "filex"コンポーネントを選択

3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。

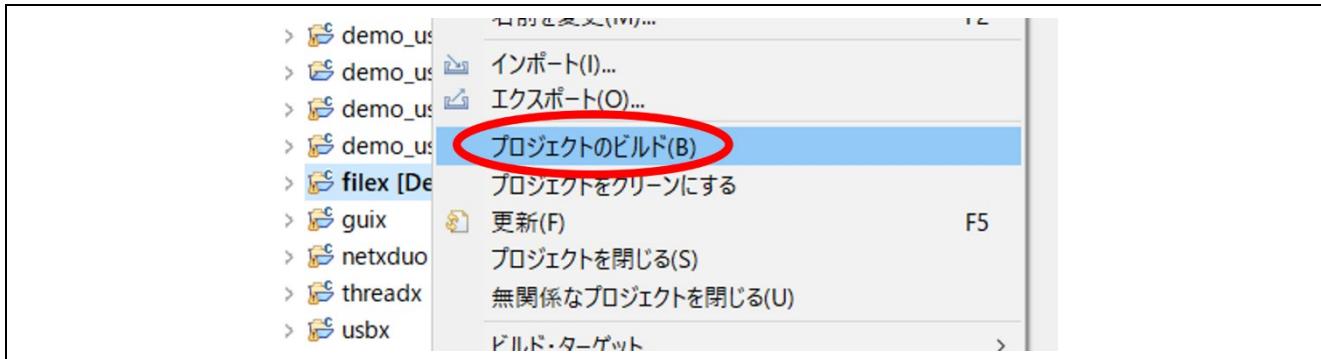


図 6-2 "filex"コンポーネントの[プロジェクトのビルド]を選択

4. プロジェクトのビルトが"0 errors"(正常終了)を表示して終了することを確認します。

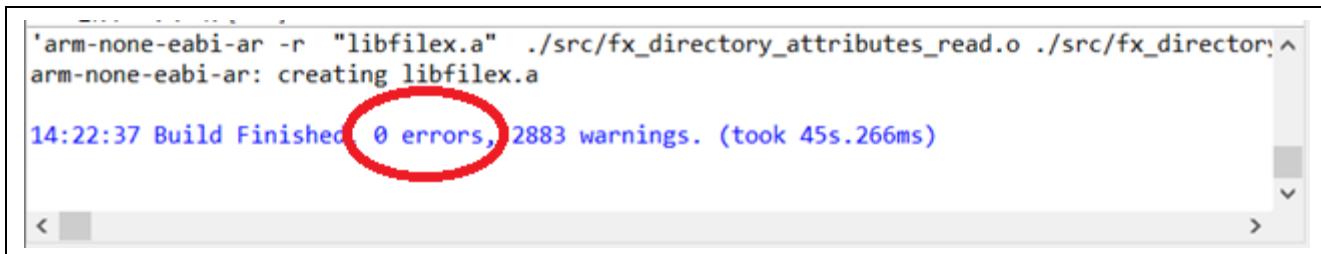


図 6-3 "filex"コンポーネントが正常終了することを確認

- すべての Azure RTOS コンポーネント、filex, guix, netxduo, threadx, usbx に対して、1.から 4.を繰り返します。

6.2 サンプルプロジェクト

このパッケージには、Azure RTOS の各コンポーネントに対応したサンプルプロジェクトと、いくつかの RZ/A2M の周辺機能を利用したサンプルプロジェクトが用意されています。サンプルプロジェクトは、各コンポーネントおよび各周辺機能の使い方のサンプルとして使用することができます。

次項から、各サンプルプロジェクトについて説明します。

6.2.1 demo_filex

demo_filex は FileX の機能を確認するサンプルプロジェクトです。

本サンプルプロジェクトは、GR-MANGO の Micro SD Card Slot にメディアが挿入されると、メディアに "TEST.TXT" というファイルを書き込みながら、状況をコンソールに出力します。

6.2.1.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- FileX

6.2.1.2 ターゲットボードとの接続

ターゲットボードと PC を、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

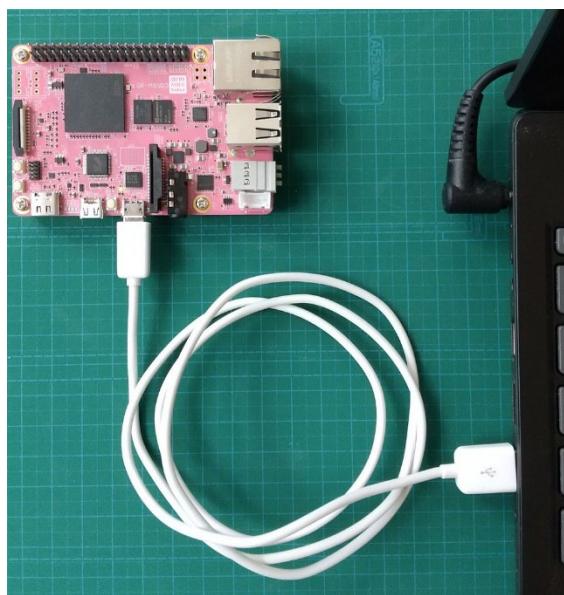


図 6-4 ターゲットボードと PC の接続

6.2.1.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、ビルド対象のサンプルプロジェクト"demo_filex"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。

Note : 下図は"demo_filex"サンプルプロジェクトのビルド構成が[HardwareDebug]になっている場合です。サンプルプロジェクトのビルド構成は、[HardwareDebug]または[Release]のどちらかを選択します。サンプルプロジェクトを[Release]ビルド構成で作成したい場合は、ビルド構成を[Release]に変更してビルドを実行してください。

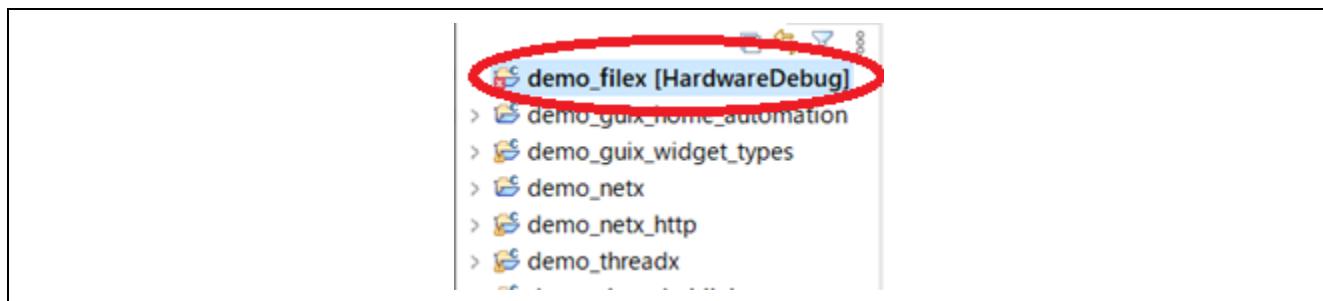


図 6-5 ビルド構成の選択

3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.1.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。

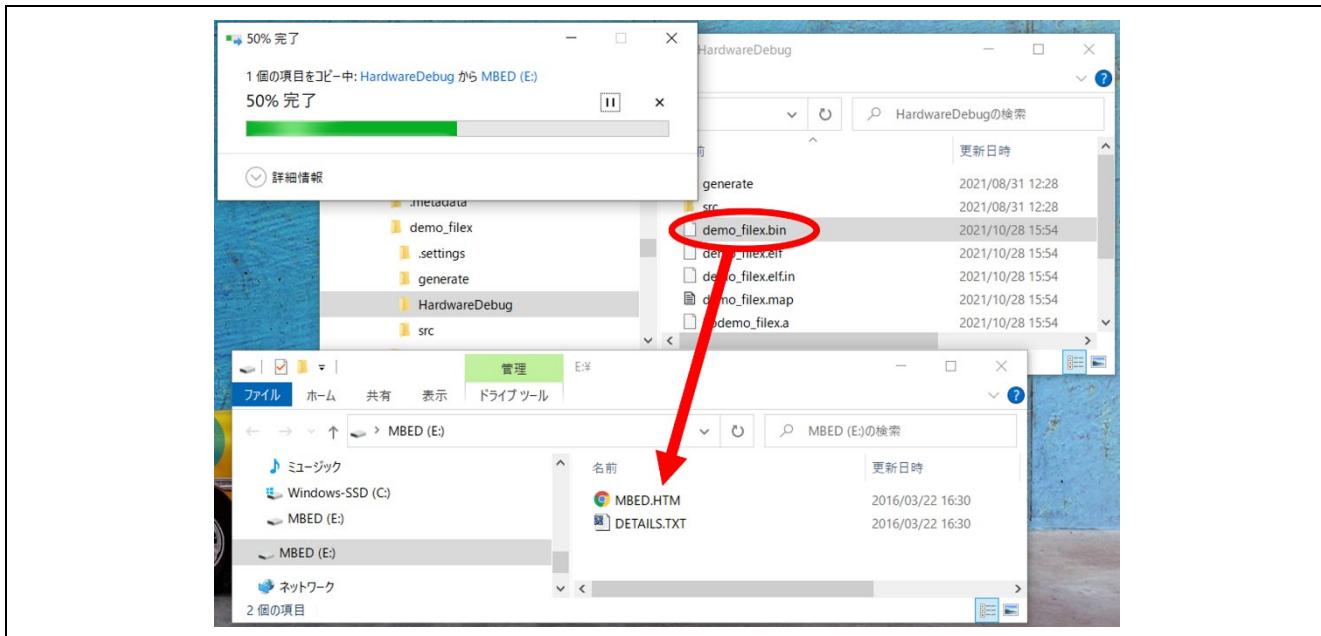


図 6-6 .bin ファイルをコピーする

4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。

Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。

6. PC 上のターミナルソフト(e.g. Tera term)を起動し、ボード接続時に認識した COM ポートに接続します。
7. PC とボードを接続している USB ケーブルを抜き、再度接続します。
8. ターミナルソフトに、以下のような表示が行われることを確認してください。

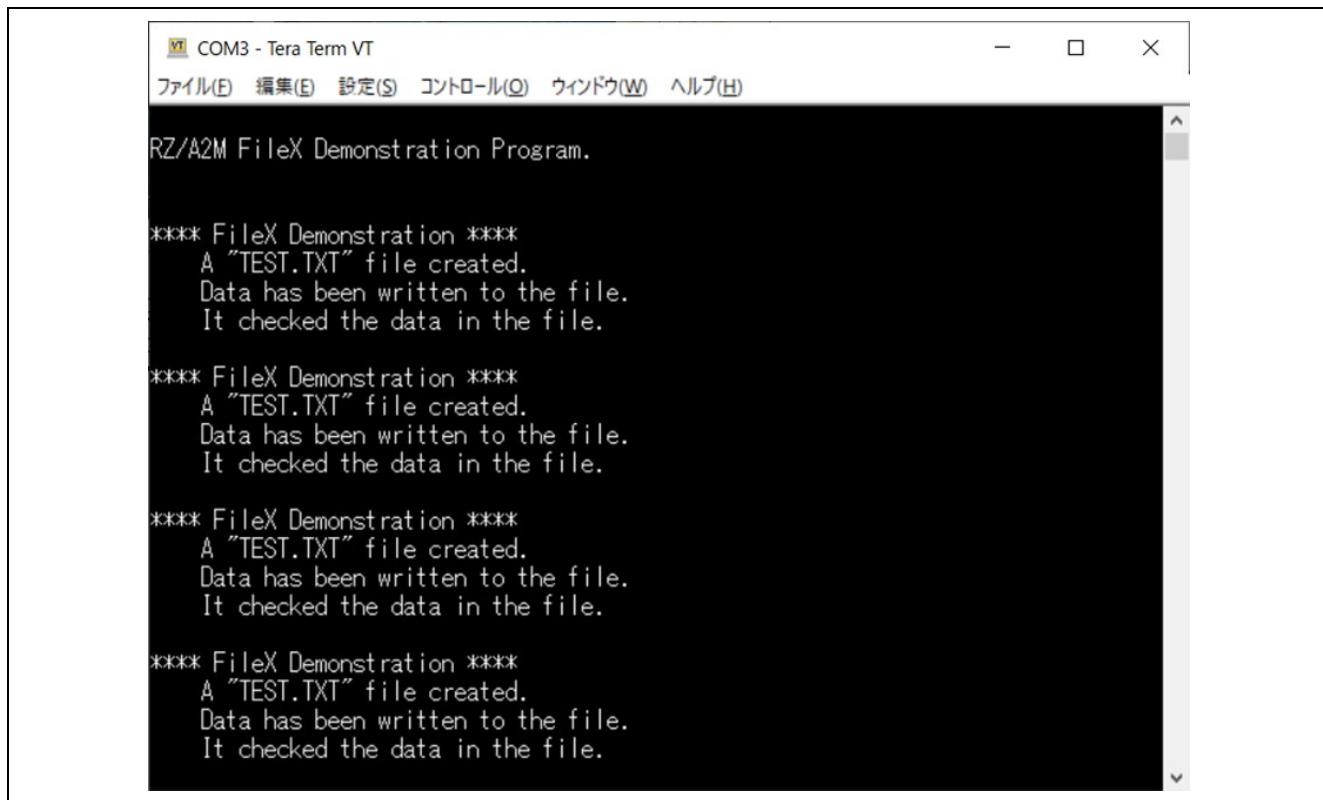


図 6-7 "demo_filex"からのメッセージ

6.2.2 demo_guix_home_automation

demo_guix_home_automation は GUIX および USBX のサンプルプロジェクトです。USB ポートは GR-MANGO USB コネクタ Type-A ポート(CN5)の下段を使用します。また GR-MANGO の HDMI ポートとモニタを接続して表示を確認します。

6.2.2.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- GUIX
- USBX

6.2.2.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタ、マウスを、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。

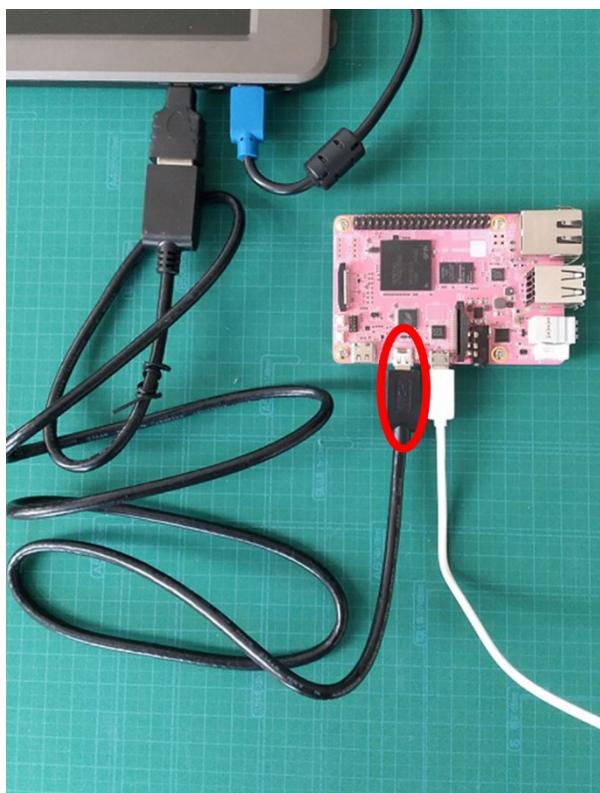


図 6-8 HDMI モニタとの接続

3. ボードの CN5 コネクタの下段に USB マウスを接続します。



図 6-9 USB マウスの接続

6.2.2.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト "demo_guix_home_automation" を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.2.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタに、以下のような表示が行われることを確認してください。



図 6-10 "demo_guix_home_automation"の表示画面

8. マウスによって、画面上のボタン等の操作ができるることを確認してください。

6.2.3 demo_guix_widget_types

demo_guix_widget_types は GUIX および USBX のサンプルプロジェクトです。USB ポートは GR-MANGO USB コネクタ Type-A ポート(CN5)の下段を使用します。また GR-MANGO の HDMI ポートとモニタを接続して表示を確認します。

6.2.3.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- GUIX
- USBX

6.2.3.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタ、マウスを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。
3. ボードの CN5 コネクタの下段に USB マウスを接続します。

6.2.3.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、**でビルド対象のサンプルプロジェクト "demo_guix_widget_types"**を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから**[プロジェクトのビルド]**を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.3.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタに、以下のような表示が行われることを確認してください。

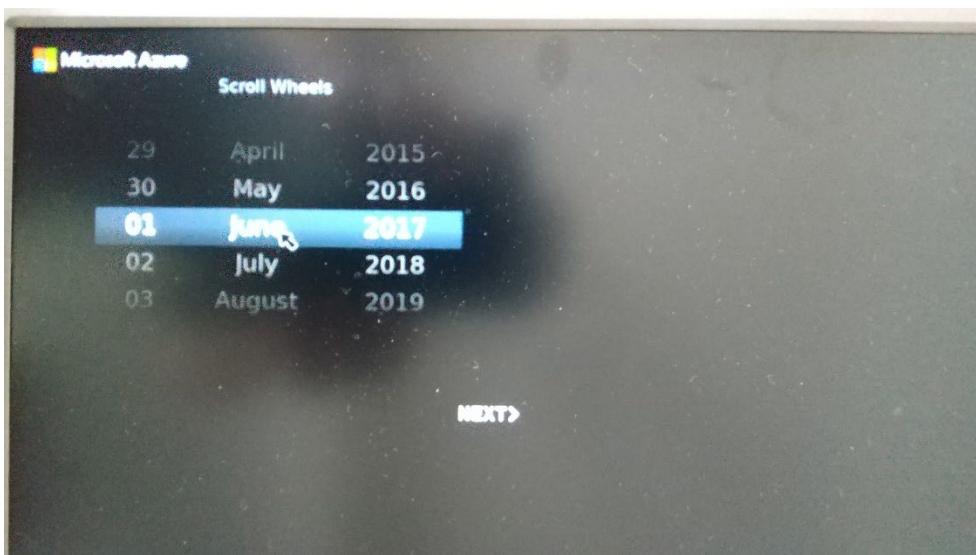


図 6-11 "demo_guix_widget_types" の表示画面

8. マウスによって、画面上のボタン等の操作ができるることを確認してください。

6.2.4 demo_guix_washing_machine

demo_guix_washing_machine は GUIX および USBX のサンプルプロジェクトです。USB ポートは GR-MANGO USB コネクタ Type-A ポート(CN5)の下段を使用します。また GR-MANGO の HDMI ポートとモニタを接続して表示を確認します。

6.2.4.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- GUIX
- USBX

6.2.4.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタ、マウスを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。
3. ボードの CN5 コネクタの下段に USB マウスを接続します。

6.2.4.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、**でビルド対象のサンプルプロジェクト "demo_guix_washing_machine"**を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから**[プロジェクトのビルド]**を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.4.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタに、以下のような表示が行われることを確認してください。



図 6-12 "demo_guix_washing_machine"の表示画面

8. マウスによって、画面上のボタン等の操作ができるることを確認してください。

6.2.5 demo_netx

demo_netx は NetXDuo のサンプルプロジェクトです。PC からの ping リクエストに対して応答を返します。

Note : ボードの IP アドレスは 192.168.2.120、ネットマスクは 255.255.255.0 に設定しています。動作させるネットワーク環境が設定されている IP アドレスに合わない場合は、nx_ip_create()で設定している IP アドレスを変更するか、PC 側の IP アドレスを変更してください。

6.2.5.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- NetXDuo

6.2.5.2 ターゲットボードとの接続

ターゲットボードと PC を、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. PC の Ethernet コネクタとボードの CN8 コネクタを Ethernet ケーブルで接続します。

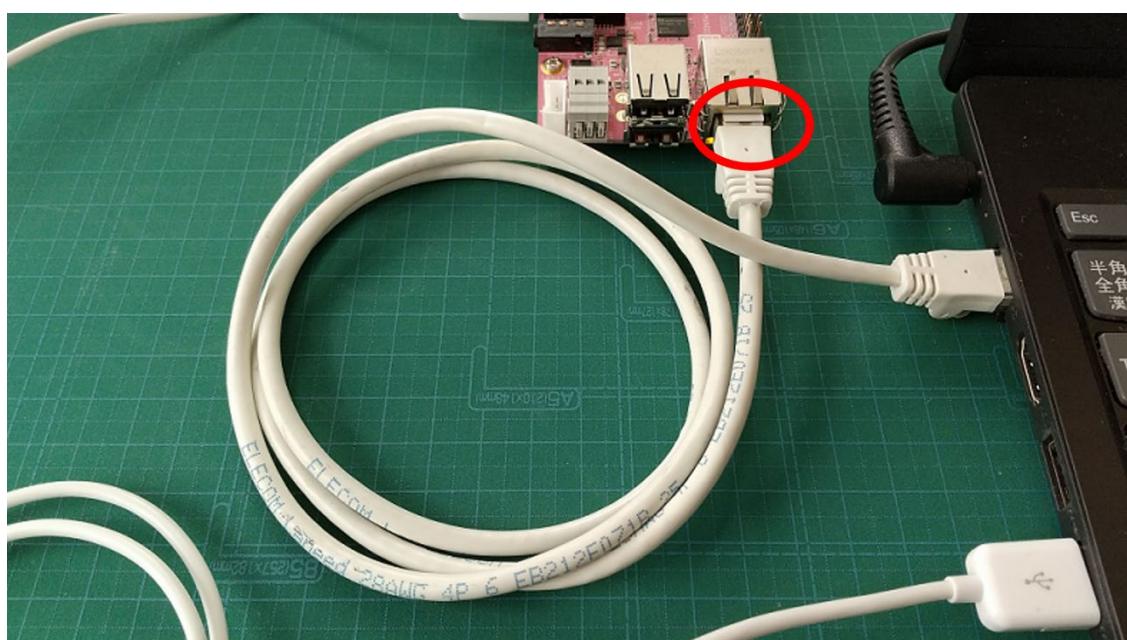


図 6-13 Ethernet の接続

6.2.5.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、ビルド対象のサンプルプロジェクト "demo_netx" を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.5.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。

Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。

6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. PC のコマンドプロンプトを起動し、ボードに設定している IP アドレスに ping コマンドを実行します。
8. ping コマンドが以下のような応答になることを確認します。

```
C:\>ping 192.168.2.120

Pinging 192.168.2.120 with 32 bytes of data:
Reply from 192.168.2.120: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.2.120:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

図 6-14 "demo_netx"の ping 応答

6.2.5.5 NetX への変更

本サンプルプログラムは NetXDuo を使用しています。もしアプリケーションが IPv4 専用で良ければ、NetXDuo の代わりに NetX を使用することができます。

NetX を使用する場合は、別途 NetX 用のライブラリプロジェクトを作成し、NetXDuo のライブラリの代わりに NetX のライブラリをアプリケーションと一緒にリンクしてアプリケーションを作成します。

6.2.6 demo_netx_http

demo_netx_http は NetXDuo の HTTP サーバのサンプルプロジェクトです。ブラウザによる HTTP リクエストに応答します。

Note : ボードの IP アドレスは 192.168.2.120、ネットマスクは 255.255.255.0 に設定しています。動作させるネットワーク環境が設定されている IP アドレスに合わない場合は、nx_ip_create()で設定している IP アドレスを変更するか、PC 側の IP アドレスを変更してください。

6.2.6.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- NetXDuo

6.2.6.2 ターゲットボードとの接続

ターゲットボードと PC を、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. PC の Ethernet コネクタとボードの CN8 コネクタを Ethernet ケーブルで接続します。

6.2.6.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト”demo_netx_http”を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが”0 errors”(正常終了)を表示して終了することを確認します。

6.2.6.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. PC のブラウザを起動し、ボードに設定している IP アドレスにアクセスします。
8. ブラウザに以下のような表示が行われることを確認します。

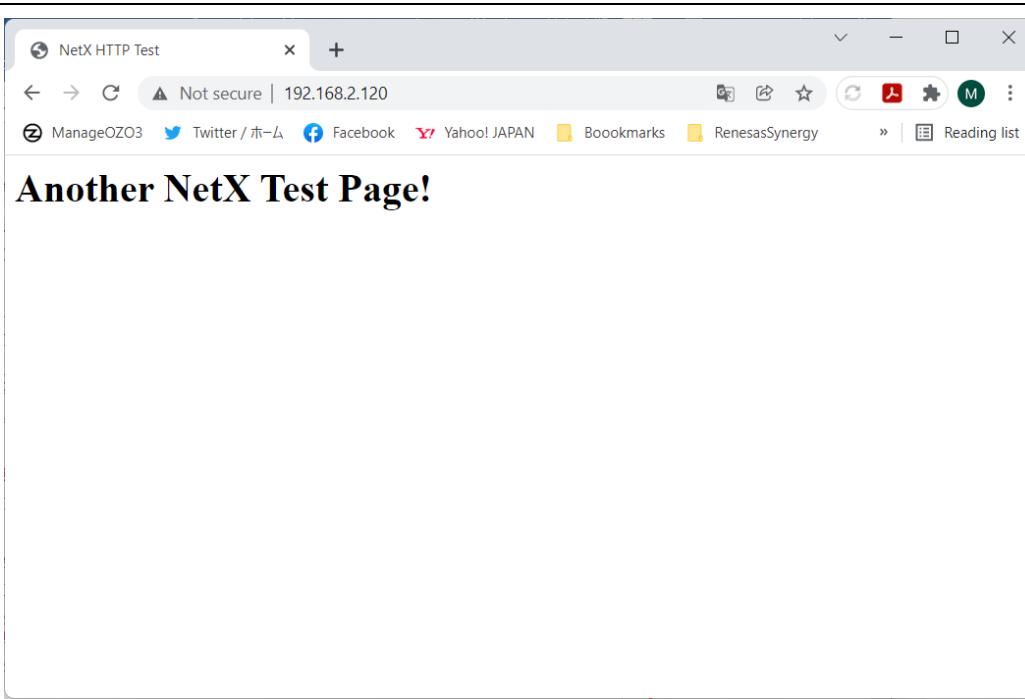


図 6-15 "demo_netx_http" のブラウザ表示

6.2.6.5 NetX への変更

本サンプルプログラムは NetXDuo を使用しています。もしアプリケーションが IPv4 専用で良ければ、NetXDuo の代わりに NetX を使用することができます。

NetX を使用する場合は、別途 NetX 用のライブラリプロジェクトを作成し、NetXDuo のライブラリの代わりに NetX のライブラリをアプリケーションと一緒にリンクしてアプリケーションを作成します。

6.2.7 demo_threadx

demo_threadx は ThreadX の動作を確認するサンプルプロジェクトです。

本サンプルプロジェクトは、複数のスレッドが切り替わりながら、メッセージをコンソールに出力します。

6.2.7.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX

6.2.7.2 ターゲットボードとの接続

ターゲットボードと PC を、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

6.2.7.3 サンプルプロジェクトのビルド

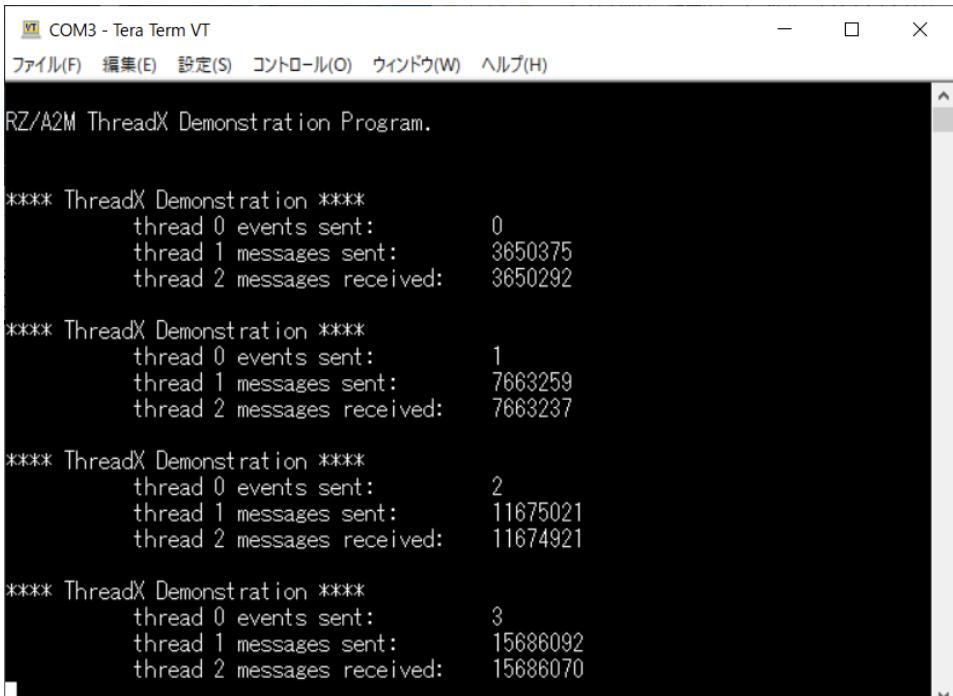
以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、"でビルド対象のサンプルプロジェクト" "demo_threadx"を選択します。
2. 選択したプロジェクトが、[Hardware Debug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.7.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. ターミナルソフトに、以下のような表示が行われることを確認してください。



The screenshot shows a terminal window titled "COM3 - Tera Term VT". The window displays the output of the "demo_threadx" program. The output is as follows:

```
RZ/A2M ThreadX Demonstration Program.

**** ThreadX Demonstration ****
    thread 0 events sent:      0
    thread 1 messages sent:   3650375
    thread 2 messages received: 3650292

**** ThreadX Demonstration ****
    thread 0 events sent:      1
    thread 1 messages sent:   7663259
    thread 2 messages received: 7663237

**** ThreadX Demonstration ****
    thread 0 events sent:      2
    thread 1 messages sent:   11675021
    thread 2 messages received: 11674921

**** ThreadX Demonstration ****
    thread 0 events sent:      3
    thread 1 messages sent:   15686092
    thread 2 messages received: 15686070
```

図 6-16 "demo_threadx"のメッセージ

6.2.8 demo_threadx_blinky

demo_threadx_blinky は ThreadX の動作を確認するサンプルプロジェクトです。

本サンプルプロジェクトは、ボード上の LED を 0.5 秒間隔で点滅させます。

6.2.8.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX

6.2.8.2 ターゲットボードとの接続

ターゲットボードと PC を、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

6.2.8.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト”demo_threadx_blinky”を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが”0 errors”(正常終了)を表示して終了することを確認します。

6.2.8.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。

Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。

6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. ボード上の LED が 0.5 秒間隔で点滅することを確認します。

6.2.9 demo_usbx_device_cdc_acm

demo_usbx_device_cdc_acm は USBX device の CDC ACM 機能を確認するサンプルプロジェクトです。

本サンプルプロジェクトは、USB の CDC ACM 機能として動作し、Windows にはシリアルポートとして認識され、入力された文字をエコーバックします。また入力された文字が改行コードだった場合、コンソールにメッセージを出力します。

6.2.9.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- USBX

6.2.9.2 ターゲットボードとの接続

ターゲットボードと PC を、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

2. PC とボードの CN6 コネクタ(USB Type-C コネクタ)を USB ケーブルで接続します。

6.2.9.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、"でビルド対象のサンプルプロジェクト" "demo_usbx_device_cdc_acm"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.9.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC 上のターミナルソフト(e.g. Teraterm)を起動し、ボード接続時に認識した COM ポートに接続します。
7. PC とボードを接続している USB ケーブルを抜き、再度接続します。
8. ボードが再度起動すると、コンソールとは別の UART(CDC ACM)として PC に認識されます。
9. PC 上のターミナルソフトを起動し、コンソールとは別の UART に接続します。
10. 別の UART に接続したターミナルソフトに文字を入力し、入力された文字がエコーバックされることを確認します。
11. 入力された文字に改行コードがあった場合、コンソールにメッセージが出力されることを確認します。

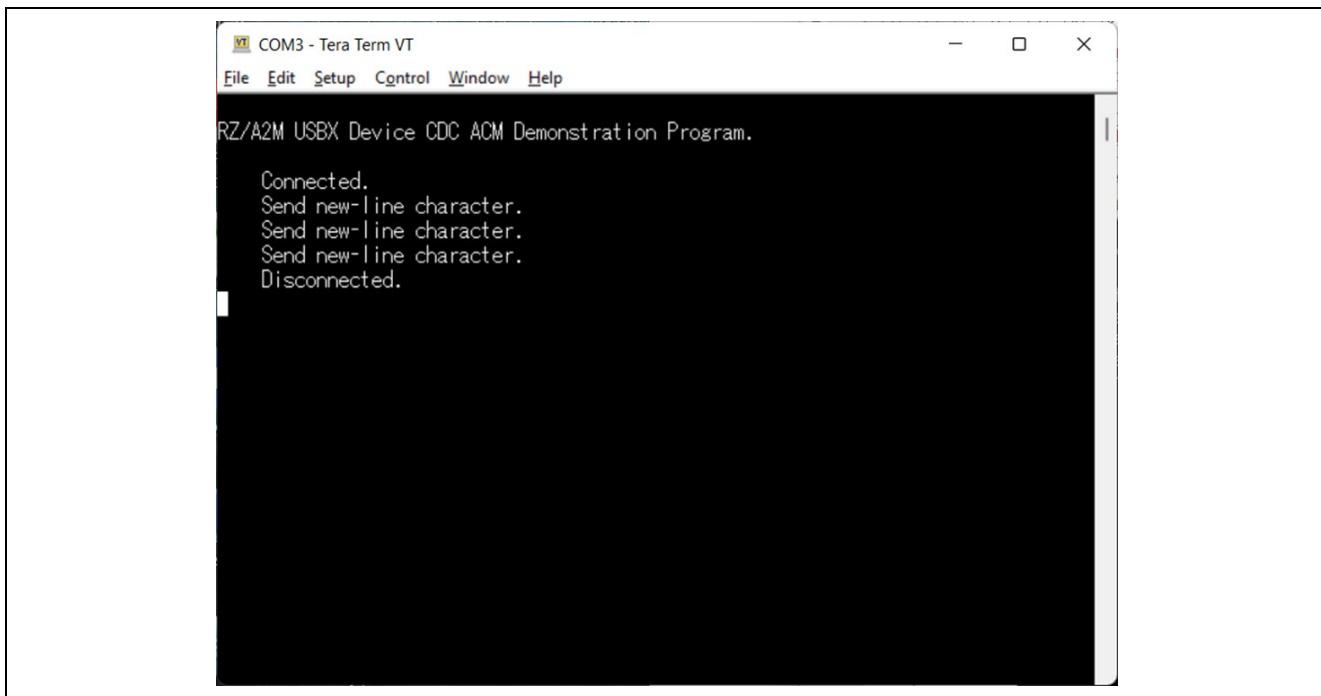


図 6-17 "demo_usbx_cdc_acm"のコンソールメッセージ

6.2.10 demo_usbx_host_hid

demo_usbx_host_hid は USBX host の HID 機能のサンプルプロジェクトです。USB ポートは GR-MANGO USB コネクタ Type-A ポート(CN5)の下段を使用します。

6.2.10.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- USBX

6.2.10.2 ターゲットボードとの接続

ターゲットボードと PC およびマウスを、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

2. ボードの CN5 コネクタの下段に USB マウスを接続します。

6.2.10.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト "demo_usbx_host_hid"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.10.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. コンソールに、以下のようなマウスの位置情報の表示が行われることを確認してください。

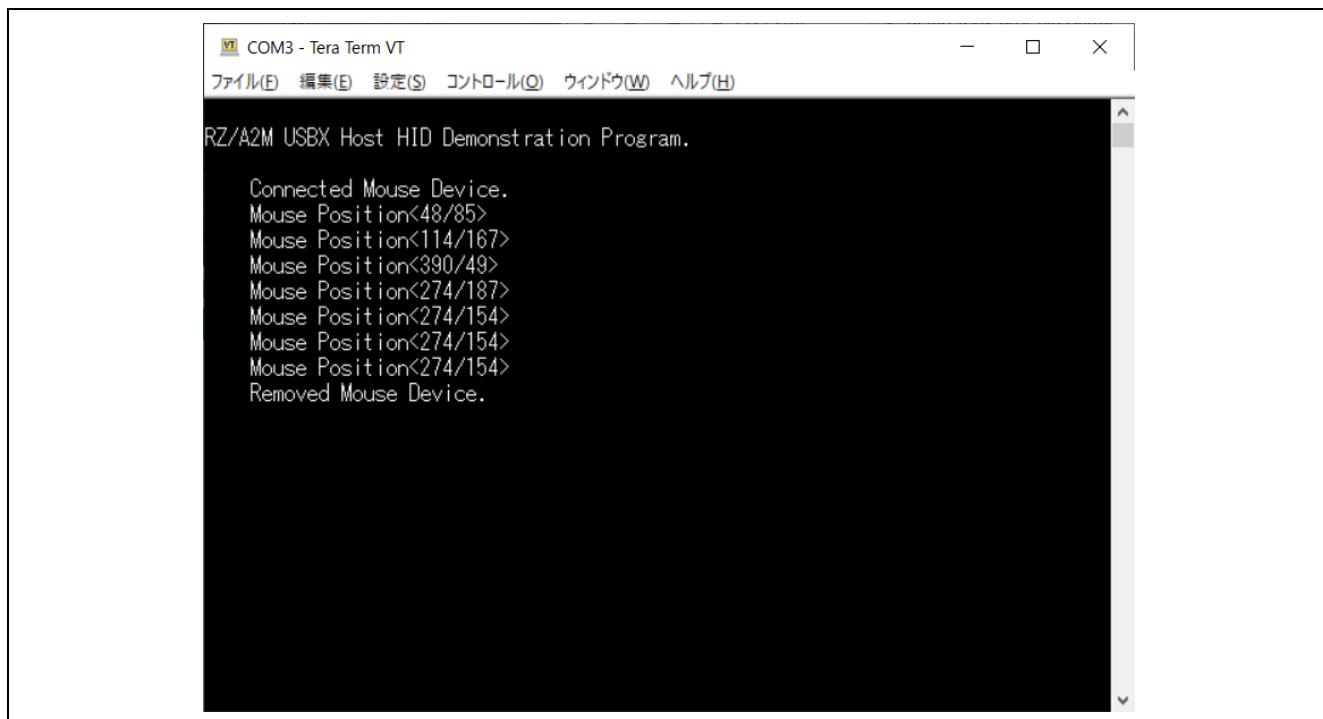


図 6-18 "demo_usbx_hid"のマウス位置情報の表示

6.2.11 demo_usbx_host_mass

demo_usbx_host_mass は USBX host の Mass Storage Class のサンプルプロジェクトです。USB ポートは GR-MANGO USB コネクタ Type-A ポート(CN5)の下段を使用します。

6.2.11.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- USBX

6.2.11.2 ターゲットボードとの接続

ターゲットボードと PC および USB メモリを、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

2. ボードの CN5 コネクタの下段に USB メモリを接続します。

6.2.11.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト "demo_usbx_host_mass"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.11.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. コンソールに、以下のような USB メモリにあるファイルの情報の表示が行われることを確認してください。

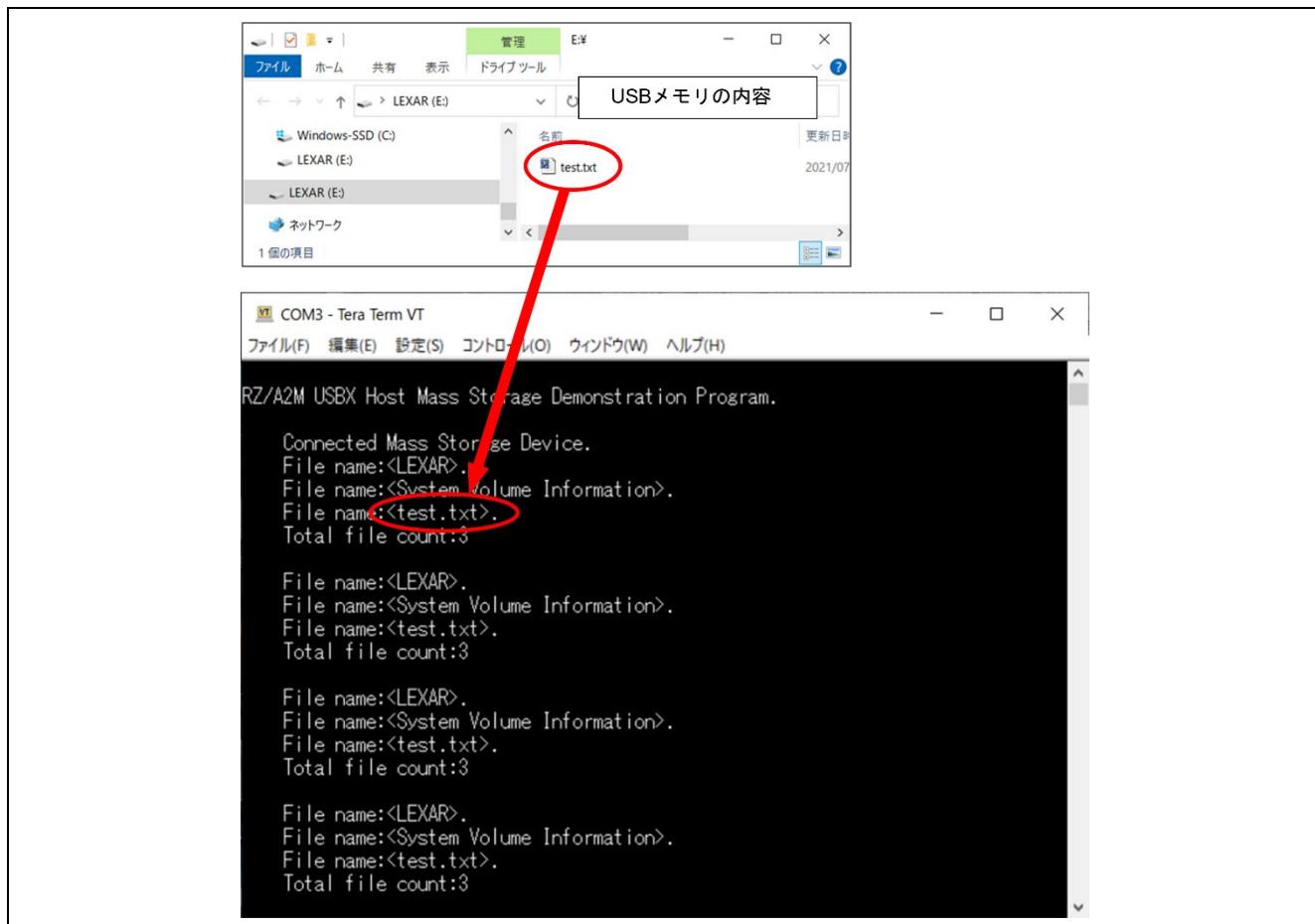


図 6-19 "demo_usbx_host_mass"の USB メモリの内容表示

6.2.12 demo_usbx_host_uvc

demo_usbx_host_uvc は USBX host の USB Video Class(UVC)のサンプルプロジェクトです。USB ポートは GR-MANGO USB コネクタ Type-A ポート(CN5)の上段を使用します。カメラ画像は PC 上のブラウザで確認できます。

Note : ボードの IP アドレスは 192.168.2.120 に設定しています。動作させるネットワーク環境が設定されている IP アドレスに合わない場合は、nx_ip_create()で設定している IP アドレスを変更してください。

6.2.12.1 Azure RTOS コンポーネント

本サンプルプロジェクトでは、以下の Azure RTOS コンポーネントを使用します。

- ThreadX
- USBX
- NetXDuo

6.2.12.2 ターゲットボードとの接続

ターゲットボードと PC、Ethernet ケーブルおよび USB カメラを、以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. PC の Ethernet コネクタとボードの CN8 コネクタを Ethernet ケーブルで接続します。
3. ボードの CN5 コネクタの上段に USB カメラを接続します。



図 6-20 USB カメラの接続

6.2.12.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、ビルド対象のサンプルプロジェクト "demo_usbx_host_uvc"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.12.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。

Note: コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。

6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. PC で motion JPEG に対応したブラウザ(e.g. Google Chrome)を起動し、ボードに設定されている IP アドレスにアクセスします。以下のようなカメラからの画像が、ブラウザに表示されることを確認してください。

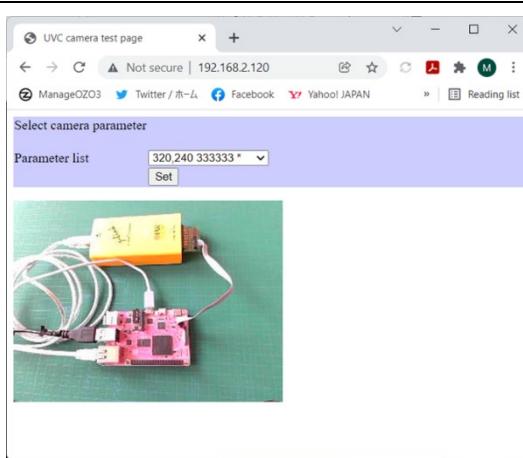


図 6-21 "demo_usbx_host_uvc"のカメラ画像の表示

6.2.13 demo_cam_and_disp

demo_cam_and_disp は、MIPI カメラの映像をディスプレイ表示するサンプルプロジェクトです。ビデオディスプレイコントローラ(VDC)に搭載される画像調整機能を CUI で制御する機能を含んでおり、表 6-1 に示す画像調整を行うことができます。

表 6-1 サンプルプロジェクトがサポートする画像調整

周辺機能	補正	内容
VDC	ブрайトネス(注 1)	輝度成分の変更により、明るさを調整する
	コントラスト(注 1)	各色成分の変更により、コントラストを調整する
	シャープネス	エッジ部分に対し、オーバーシュート、アンダーシュートを調整することで、輪郭強調を行う
	ガンマ補正	プリセットされているガンマ補正をかける(プリセットは 4 種類)
	ディザ処理	ランダムパターンによる、ディザ処理を行う
	回転・水平鏡像(注 2)	画像を 180 度回転、または左右反転させる

【注 1】 MIPI や CEU を用いたカメラ入力の場合は補正タイミングに制限あり。

【注 2】 MIPI や CEU を用いたカメラ入力の場合は使用不可。

6.2.13.1 周辺機能

本サンプルプロジェクトでは、以下の周辺機能を使用します。

- VDC6
- MIPI・VIN

6.2.13.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタおよび MIPI カメラを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。
3. ボードの CN13 コネクタに MIPI カメラを接続します。



図 6-22 MIPI カメラの接続

6.2.13.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト "demo_cam_and_disp"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.13.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。

Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。

6. PC 上のターミナルソフト(e.g. Teraterm)を起動し、ボード接続時に認識した COM ポートに接続します。
7. PC とボードを接続している USB ケーブルを抜き、再度接続します。
8. 以下のようなカメラからの画像が、HDMI モニタに表示されることを確認してください。



図 6-23 "demo_cam_and_disp"のカメラ画像の表示

9. ターミナルソフトに、以下のような表示が行われることを確認してください。

```

COM7 - Tera Term VT
-----
RZ/A2M sample for GCC Ver. 3.1
Copyright (C) 2020 Renesas Electronics Corporation. All rights reserved.
Build Info Date Jul 20 2022 at 09:54:49
threadx Version 3.8
Camera and Display Application

-----
Display Setting Top Menu
-----
Please select the item you want to set.
0 : Brightness
1 : Contrast
2 : Sharpness
3 : Gamma Correction
4 : Dither process
5 : Horizontal Mirroring and Rotation
-----
R : Show each value of the customizable registers.
D : All Default Settings.
>>■

```

図 6-24 "demo_cam_and_disp"のコンソールメッセージ

10. ターミナルソフト上で以下のコマンドを入力することにより、表示される画像が調整されることを確認してください。

表 6-2 コマンド一覧

コマンド	動作
数字	各メニューにおける操作 ・調整内容の選択 ・補正位置の選択(画像調整する H/W ブロックの選択) ・プリセット選択 ・カスタム値入力
C, c	カスタム設定選択 (各種調整アイテムでプリセット以外を設定したい場合に選択)
D, d	画像調整をデフォルトに設定 (H/W マニュアルに記載された各レジスタのデフォルト値)
B, b	一つ前のメニューへ戻る
R, r	現在の画像調整値を出力
T, t	Top メニューへ遷移
Enter	入力内容を確定
Delete / Back space	入力した文字を 1 文字削除

6.2.14 demo_drp_basic_sample

demo_drp_basic_sample は、RZ/A2M に搭載された動的再構成プロセッサ(DRP)の基本動作を確認するサンプルプロジェクトです。MIPI カメラからの入力画像を DRP Library を使ってグレースケール画像へ変換し、ディスプレイ表示します。

6.2.14.1 周辺機能

本サンプルプロジェクトでは、以下の周辺機能を使用します。

- DRP

6.2.14.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタおよび MIPI カメラを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation および demo_cam_and_disp を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。

3. ボードの CN13 コネクタに MIPI カメラを接続します。

6.2.14.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、"でビルド対象のサンプルプロジェクト" "demo_drp_basic_sample"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.14.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタに、カメラからの画像が表示されることを確認してください。

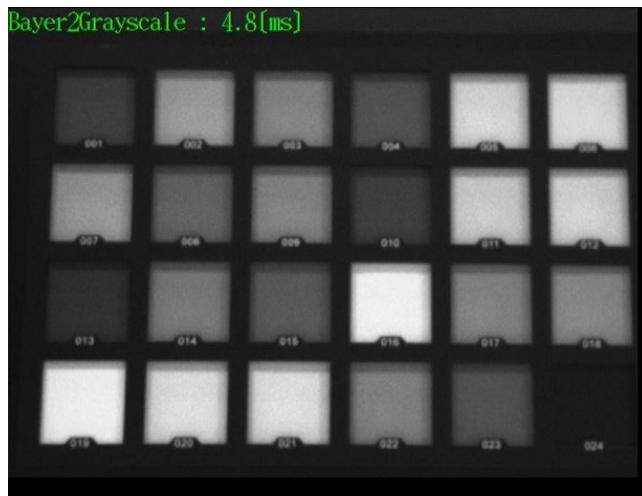


図 6-25 "demo_drp_basic_sample"のカメラ画像の表示

6.2.15 demo_drp_dynamic_sample1

demo_drp_dynamic_sample1 は、動的再構成プロセッサ(DRP)のダイナミックローディング機能を確認するサンプルプロジェクトです。MIPI カメラからの入力画像に対し DRP Library を動的に切り替えながら Canny 法でエッジを検出し、結果をディスプレイ表示します。

6.2.15.1 周辺機能

本サンプルプロジェクトでは、以下の周辺機能を使用します。

- DRP

6.2.15.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタおよび MIPI カメラを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation および demo_cam_and_disp を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。

3. ボードの CN13 コネクタに MIPI カメラを接続します。

6.2.15.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、"ビルド対象のサンプルプロジェクト" "demo_drp_dynamic_sample1"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.15.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタにカメラからの画像が表示され、さらに画像内にある物体のエッジが検出されていることを確認してください。

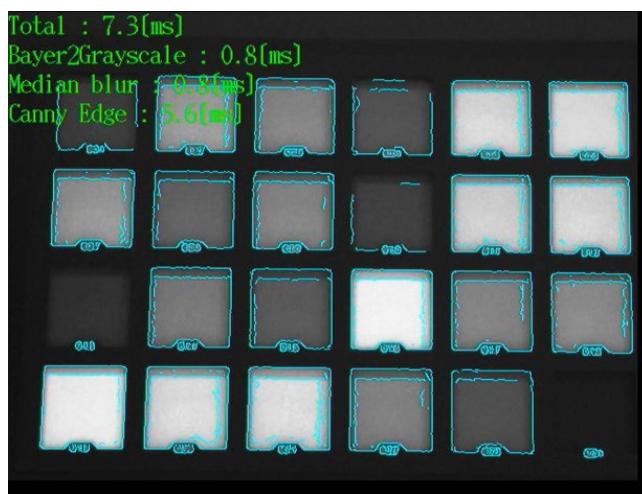


図 6-26 "demo_drp_dynamic_sample1" のカメラ画像の表示

6.2.16 demo_drp_dynamic_sample2

demo_drp_dynamic_sample2 は、動的再構成プロセッサ(DRP)のダイナミックローディング機能を確認するサンプルプロジェクトです。MIPI カメラからの入力画像に対し DRP Library を動的に切り替えながら Harris コーナー検出を行い、結果をディスプレイ表示します。

6.2.16.1 周辺機能

本サンプルプロジェクトでは、以下の周辺機能を使用します。

- DRP

6.2.16.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタおよび MIPI カメラを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation および demo_cam_and_disp を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。
3. ボードの CN13 コネクタに MIPI カメラを接続します。

6.2.16.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、"でビルド対象のサンプルプロジェクト" "demo_drp_dynamic_sample2"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.16.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタにカメラからの画像が表示され、さらに画像内にある物体のコーナーが検出されていることを確認してください。



図 6-27 "demo_drp_dynamic_sample2"のカメラ画像の表示

6.2.17 demo_drp_dynamic_sample3

demo_drp_dynamic_sample3 は、動的再構成プロセッサ(DRP)のダイナミックローディング機能を確認するサンプルプロジェクトです。MIPI カメラからの入力画像に対し、DRP Library を動的に切り替えながら Canny 法でエッジ検出した後、FindContours により輪郭情報を抽出。結果をディスプレイ表示します。

6.2.17.1 周辺機能

本サンプルプロジェクトでは、以下の周辺機能を使用します。

- DRP

6.2.17.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタおよび MIPI カメラを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation および demo_cam_and_disp を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。
3. ボードの CN13 コネクタに MIPI カメラを接続します。

6.2.17.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、"でビルド対象のサンプルプロジェクト" "demo_drp_dynamic_sample3"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.17.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタにカメラからの画像が表示され、さらに画像内にある物体の輪郭情報が抽出されていることを確認してください。また、ボードの SW2 により自動露出補正(AE)の ON/OFF が制御できます。

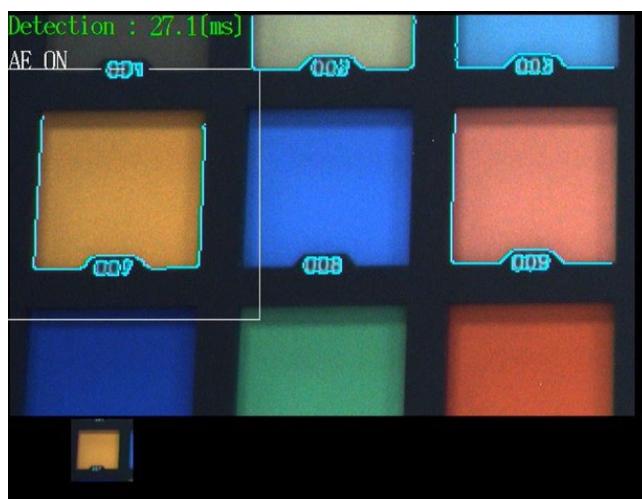


図 6-28 "demo_drp_dynamic_sample3" のカメラ画像の表示

6.2.18 demo_drp_parallel_sample

demo_drp_parallel_sample は、動的再構成プロセッサ(DRP)の並列動作機能を確認するサンプルプロジェクトです。MIPI カメラからの入力画像を DRP Library の並列動作機能を使って高速にグレースケール画像へ変換して、ディスプレイ表示します。

6.2.18.1 周辺機能

本サンプルプロジェクトでは、以下の周辺機能を使用します。

- DRP

6.2.18.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタおよび MIPI カメラを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation および demo_cam_and_disp を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。
3. ボードの CN13 コネクタに MIPI カメラを接続します。

6.2.18.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、"ビルド対象のサンプルプロジェクト" "demo_drp_parallel_sample"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.18.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. カメラからの画像が HDMI モニタに表示されることを確認してください。

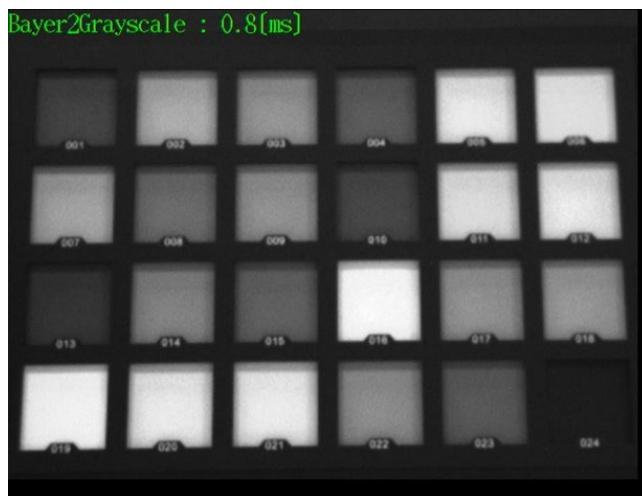


図 6-29 "demo_drp_parallel_sample"のカメラ画像の表示

6.2.19 demo_drp_simple_isp_sample1

demo_drp_simple_isp_sample1 は、動的再構成プロセッサ(DRP)のイメージシグナルプロセッサ(ISP)機能を確認するサンプルプロジェクトです。MIPI カメラからの入力画像に対し、DRP Library を使用して色の補正やノイズ除去を行い、色再現性の高い画像をディスプレイ出力します。

6.2.19.1 周辺機能

本サンプルプロジェクトでは、以下の周辺機能を使用します。

- DRP

6.2.19.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタおよび MIPI カメラを、以下のように接続します。

Note : 接続の状態は demo_guix_home_automation および demo_cam_and_disp を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。
Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。
2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。
3. ボードの CN13 コネクタに MIPI カメラを接続します。

6.2.19.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、ビルド対象のサンプルプロジェクト "demo_drp_simple_isp_sample1"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.19.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタに、カメラからの画像がカラーで表示されることを確認してください。また、ボードの SW2 と SW3 によって、それぞれ色補正のモードとノイズ除去の ON/OFF が制御できます。



図 6-30 "demo_drp_simple_isp_sample1" のカメラ画像の表示

6.2.20 demo_jpeg_codec_sample

demo_jpeg_codec_sample は、RZ/A2M の JPEG コーデックユニット(JCU)の機能を確認するサンプルプロジェクトです。JPEG 圧縮画像のデコードおよび JPEG 圧縮画像へのエンコードを行います。

6.2.20.1 周辺機能

本サンプルプロジェクトでは、以下の周辺機能を使用します。

- JPEG コーデックユニット(JCU)

6.2.20.2 ターゲットボードとの接続

ターゲットボードと PC、HDMI モニタを以下のように接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。本サンプルプロジェクトではコンソールを使用しません。

2. ボードの CN9 コネクタと HDMI モニタを HDMI ケーブルで接続し、HDMI モニタの電源も接続します。

6.2.20.3 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト "demo_jpeg_codec_sample"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

6.2.20.4 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. HDMI モニタ上で、以下の画像が表示されることを確認してください。



図 6-31 "demo_jpeg_codec_sample"の表示画面

7. Azure RTOS IoT Embedded SDK サンプル

このパッケージには、Azure RTOS Embedded SDKに基づいたサンプルプロジェクトが含まれており、実行することでGR-MANGOをAzure IoTに接続することができます。なお、GR-MANGOは”[Azure Certified Device](#)”および”[IoT Plug and Play](#)”の認定デバイスです。

7.1 sample_azure_iot_embedded_sdk

sample_azure_iot_embedded_sdkは、Azure RTOSのAzure IoT ミドルウェアを用いて Azure IoT Hub に接続するサンプルプロジェクトです。

7.1.1 Azure リソースの用意

以下の手順で Azure のリソースを用意してください。

- 以下のドキュメントを参照し、Azure IoT Hub を作成してください。また、作成した IoT Hub にデバイス(GR-MANGO)を登録してください。

[Azure Portal を使用して IoT Hub を作成する | Microsoft Learn](#)

Note1: IoT Hub を作成する際、”レベル”は、”Free”もしくは”Standard”を選択してください。



図 7-1 ”レベル”で”Free”もしくは”Standard”を選択

Note2: IoT Hub にデバイスを登録する際、"認証の種類"は、"対称キー"を選択してください。



図 7-2 "認証の種類"で"対称キー"を選択

2. 1.で登録したデバイスに対応する以下の値を記録してください。これらの値は、GR-MANGO を IoT Hub に接続するために必要です。
 - IoT Hub Hostname
 - Device ID
 - Primary Key
3. 以下のドキュメントを参照し、Azure IoT Explore をインストールしてください。また、IoT Hub の iothubowner 接続文字列を取得し、IoT Explore に登録してください。

[Azure IoT エクスプローラーをインストールして使用する | Microsoft Learn](#)

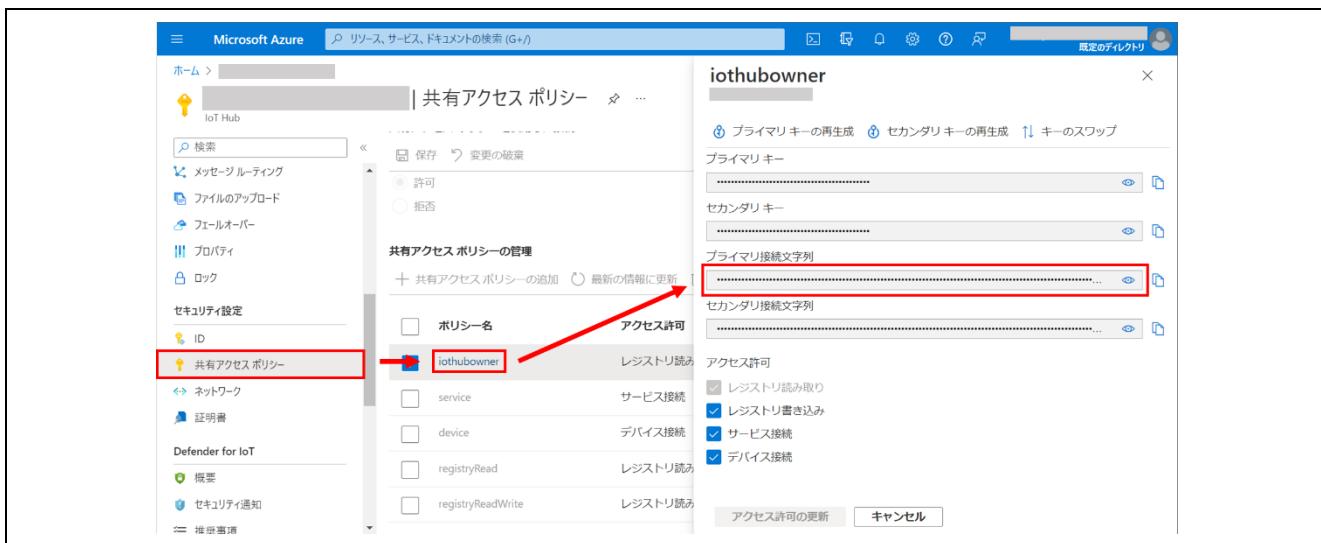


図 7-3 iothubowner 接続文字列

7.1.2 ターゲットボードとの接続

ターゲットボードと PC を以下のように接続します。さらに、ターゲットボードをインターネット接続します。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

2. ボードの CN8 コネクタにイーサネットケーブルを接続し、インターネット接続します。

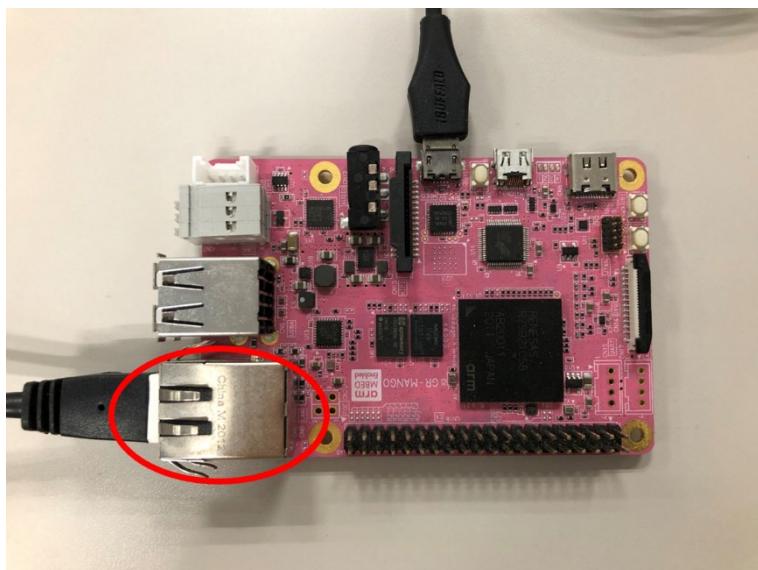


図 7-4 イーサネットの接続

7.1.3 マクロの設定

サンプルプロジェクトの config ファイルに、Azure IoT に接続するための情報を入力します。

1. サンプルプロジェクト sample_azure_iot_embedded_sdk に含まれる config ファイル "sample_config.h" を開きます。
2. 以下の表に沿って、マクロに値を設定します。

表 7-1 マクロの設定

Constant name	Value
HOST_NAME	{Your IoT Hub Hostname value}
DEVICE_ID	{Your Device ID value}
DEVICE_SYMMETRIC_KEY	{Your Primary Key value}

```

@sample_config.h @@
57  /* Required when DPS is not used. */
58  /* These values can be picked from device connection string which is of format : HostName=<host1>;DeviceId=<device1>;SharedAcc
59
60 #ifndef HOST_NAME
61 #define HOST_NAME           "{Your IoT Hub Hostname value}"
62 #endif // HOST_NAME
63
64 #ifndef DEVICE_ID
65 #define DEVICE_ID            "{Your Device ID value}"
66 #endif // DEVICE_ID
67
68 #else /* !ENABLE_DPS_SAMPLE */
69
70 #ifndef ENDPOINT
71 #define ENDPOINT              "global.azure-devices-provisioning.net"
72 #endif // ENDPOINT
73
74 #ifndef ID_SCOPE
75 #define ID_SCOPE               ""
76 #endif // ID_SCOPE
77
78 #ifndef REGISTRATION_ID
79 #define REGISTRATION_ID        ""
80 #endif // REGISTRATION_ID
81
82 #endif /* ENABLE_DPS_SAMPLE */
83
84
85 #ifndef DEVICE_SYMMETRIC_KEY
86 #define DEVICE_SYMMETRIC_KEY   "{Your Primary Key value}"
87 #endif // DEVICE_SYMMETRIC_KEY
88
89 #endif // DEVICE_SYMMETRIC_KEY
90
91

```

図 7-5 マクロの設定

7.1.4 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト"sample_azure_iot_embedded_sdk"を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが"0 errors"(正常終了)を表示して終了することを確認します。

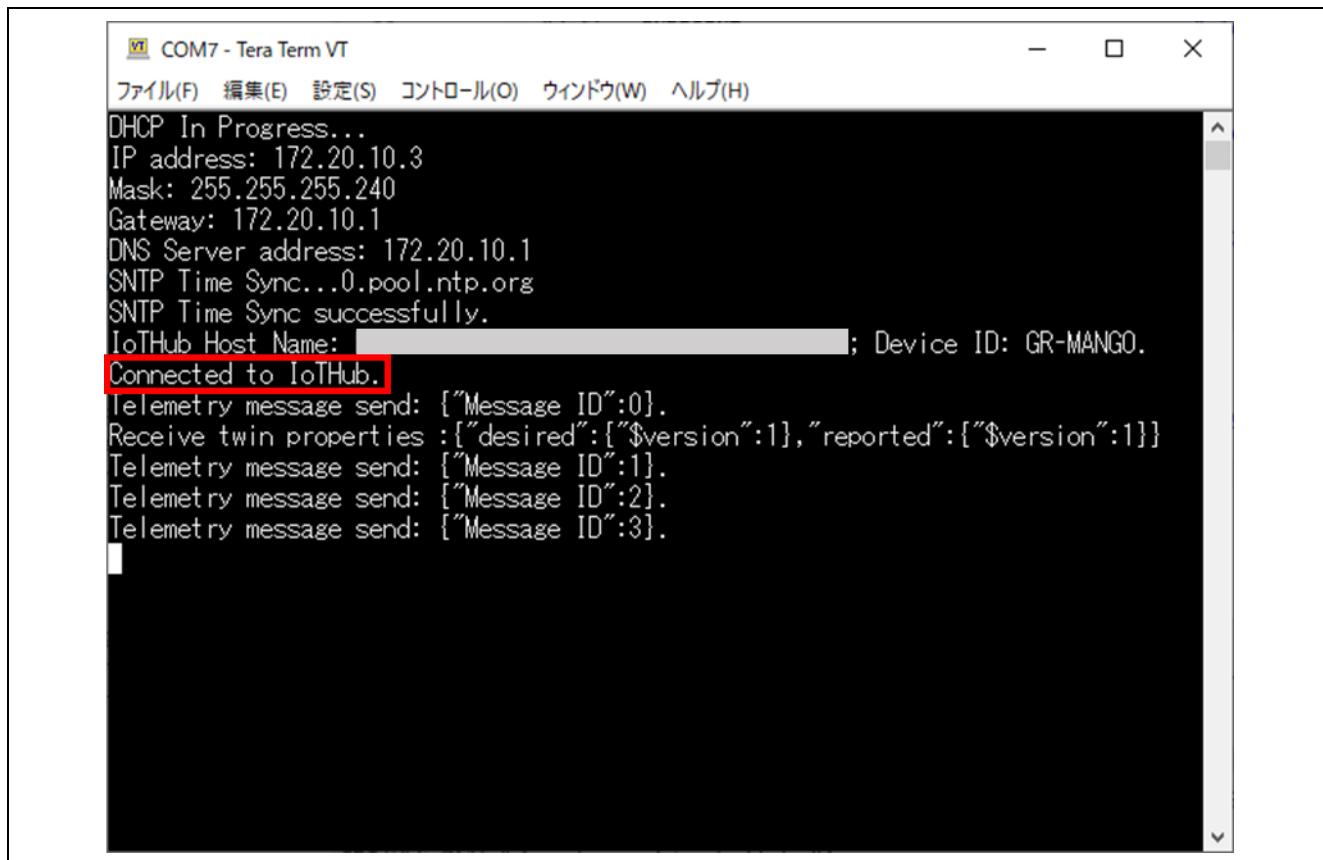
7.1.5 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。

Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。

6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. ターミナルソフトに、以下のような表示が行われることを確認してください。



The screenshot shows a terminal window titled "COM7 - Tera Term VT". The window displays the following text:

```
DHCP In Progress...
IP address: 172.20.10.3
Mask: 255.255.255.240
Gateway: 172.20.10.1
DNS Server address: 172.20.10.1
SNTP Time Sync...0.pool.ntp.org
SNTP Time Sync successfully.
IoTHub Host Name: [REDACTED]; Device ID: GR-MANGO.
Connected to IoTHub.
Telemetry message send: {"Message ID":0}.
Receive twin properties :[{"desired":{"$version":1}, "reported":{"$version":1}}
Telemetry message send: {"Message ID":1}.
Telemetry message send: {"Message ID":2}.
Telemetry message send: {"Message ID":3}.
```

図 7-6 "sample_azure_iot_embedded_sdk"からのメッセージ

8. Azure IoT Explorer を用いて、以下の操作を行うことができます。

- “Telemetry”タブで、デバイスからクラウドに送信されたテレメトリーを確認できます。

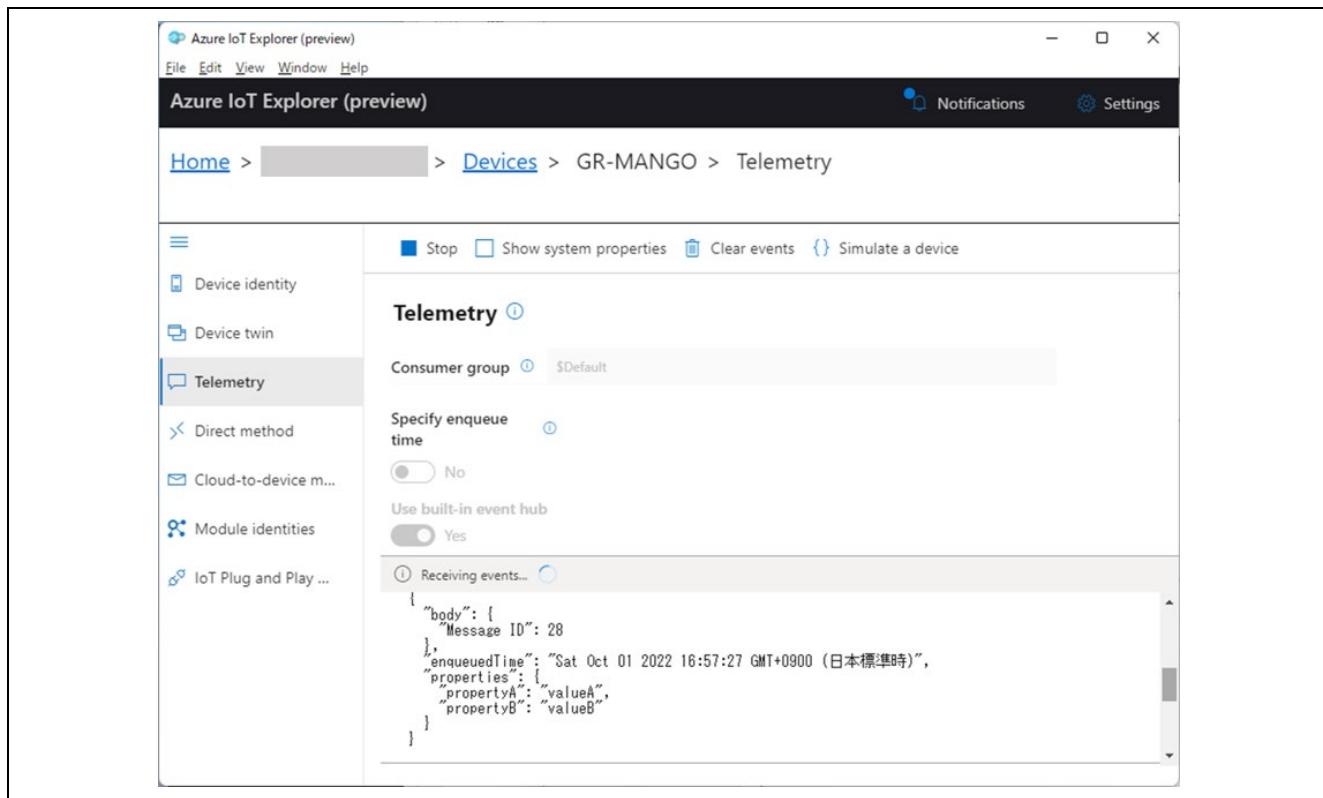


図 7-7 テレメトリーの確認

- “Device Twin”タブで、デバイスツインの“desired”部分を書き換えることでデバイスのプロパティを更新できます。このサンプルでは、以下のプロパティを更新できます。

```
"weather": {  
    "temperature": "25"  
},
```

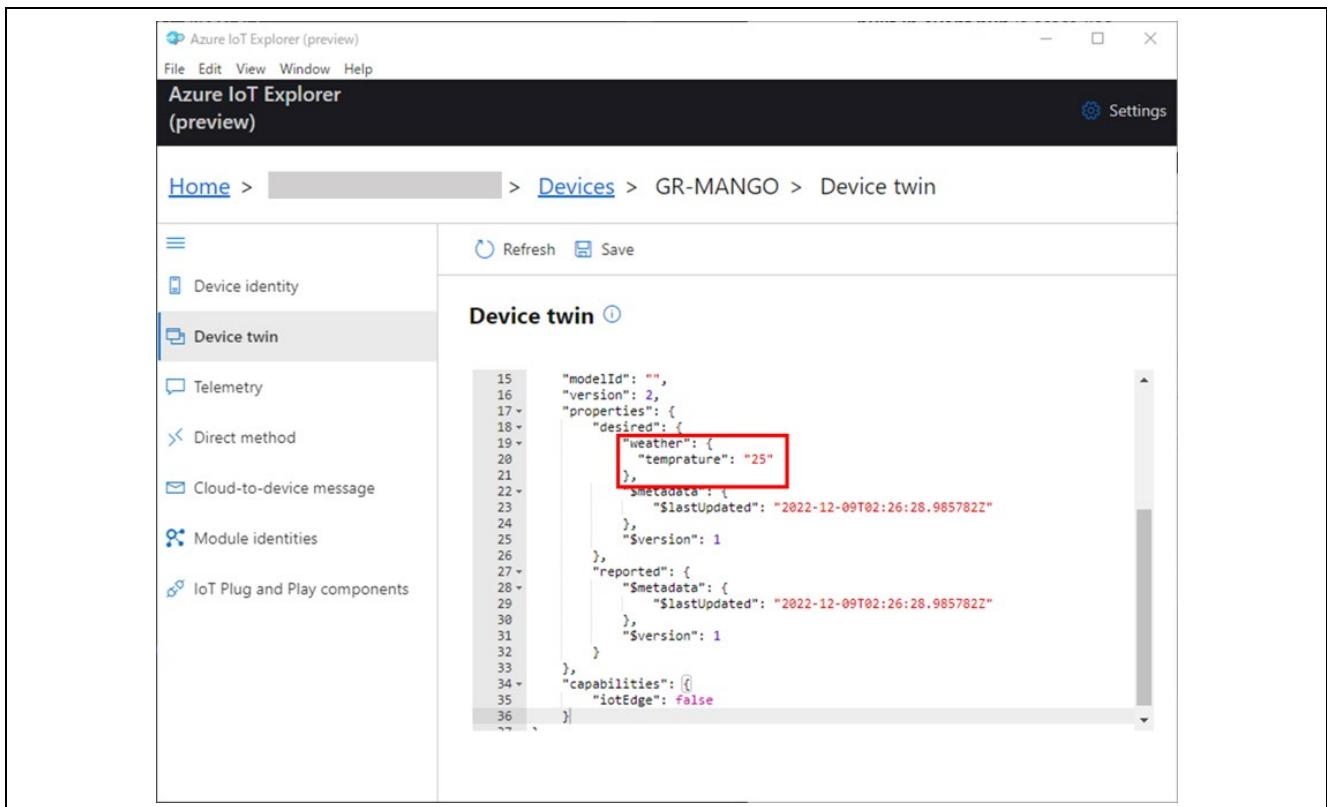


図 7-8 デバイスツインの更新

- “Direct Method”タブで、ダイレクトメソッドを呼び出すことができます。このサンプルでは以下のメソッドに対応しています。

Method name:

reboot

Payload:

{"timeout": 500}

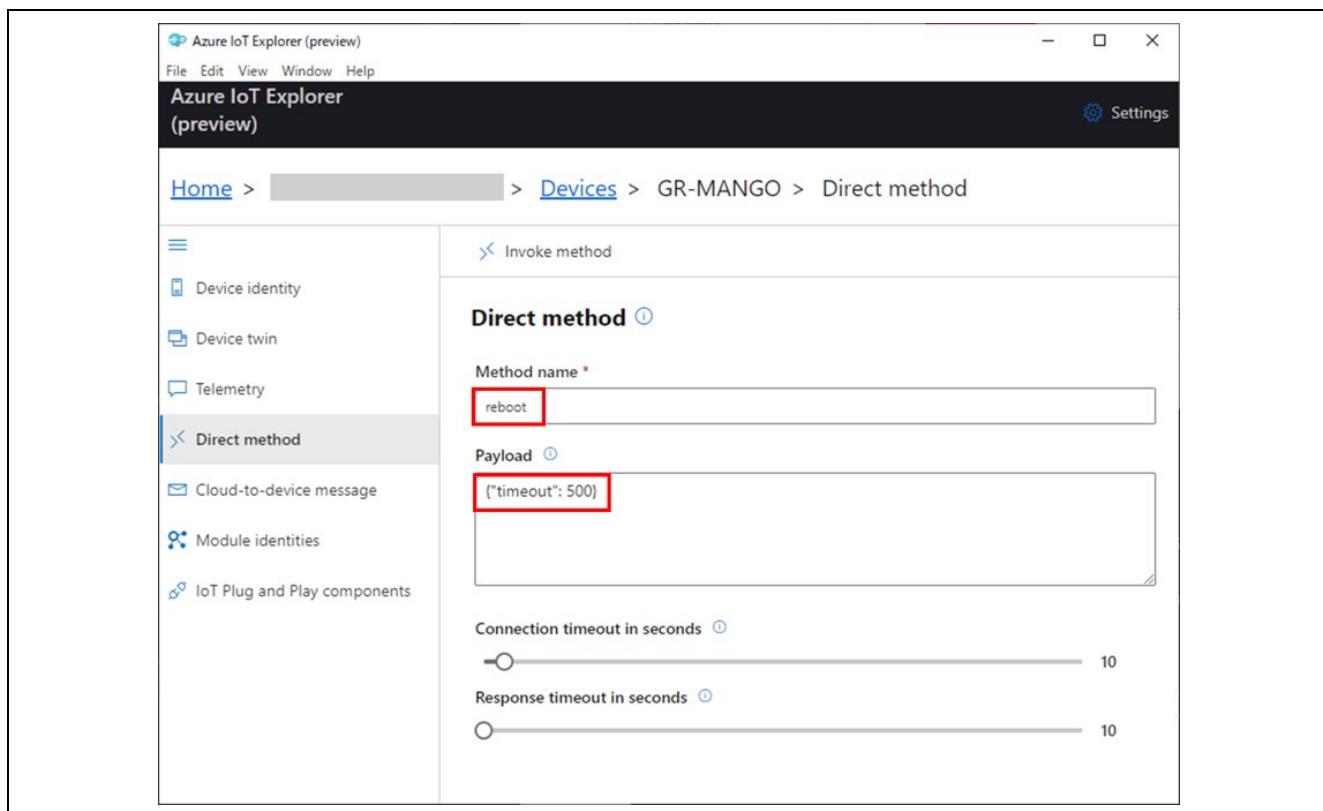


図 7-9 ダイレクトメソッドの呼び出し

- “Cloud to device message”タブで、クラウドからデバイスへメッセージを送信できます。

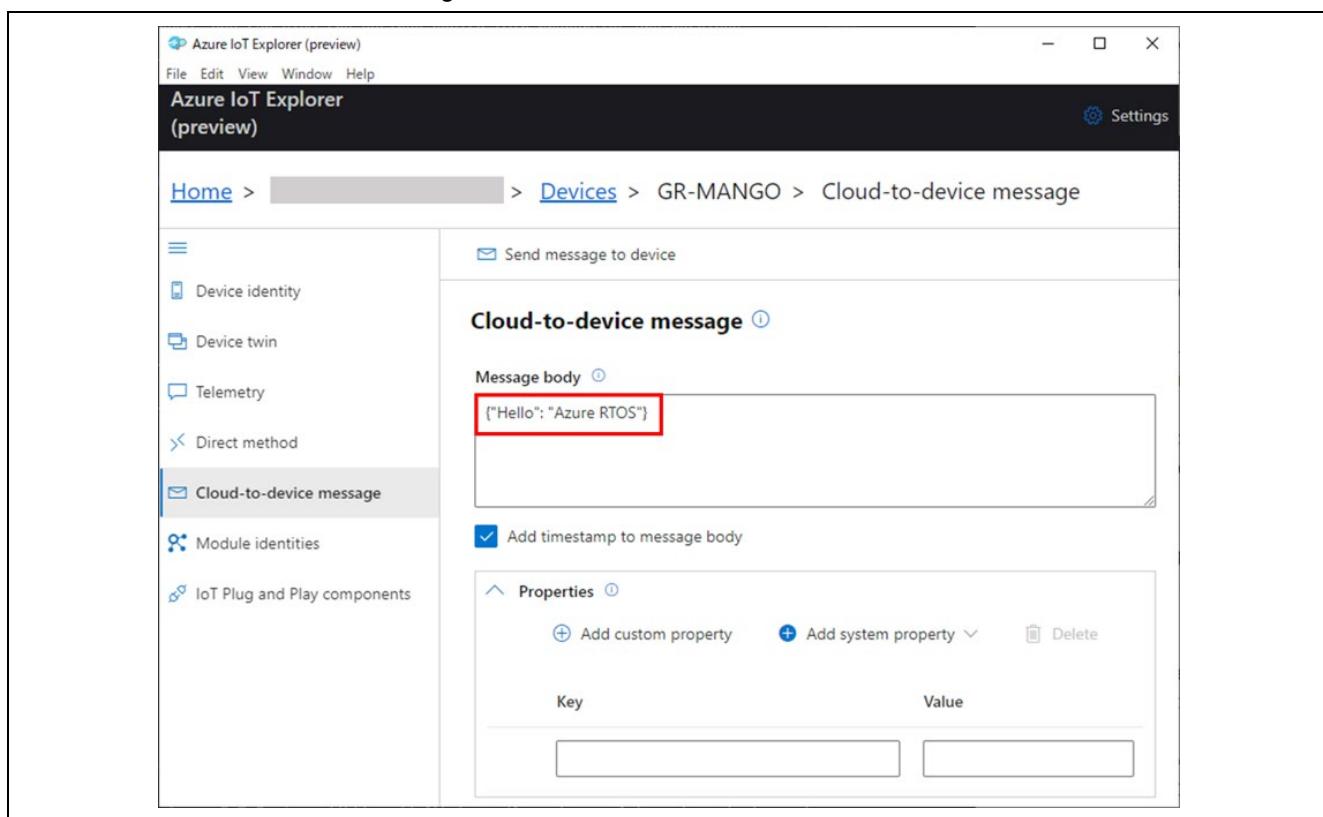


図 7-10 Cloud to device メッセージの送信

7.2 sample_azure_iot_embedded_sdk_pnp

sample_azure_iot_embedded_sdk_pnp は、Azure RTOS の Azure IoT ミドルウェアを用いて Azure IoT Hub に接続するサンプルプロジェクトです。本サンプルプロジェクトは、IoT Plug and Play の機能を含んでいます。IoT Plug and Play については、以下のドキュメントを参照してください。

[Introduction to IoT Plug and Play | Microsoft Learn](#)

7.2.1 Azure リソースの用意

以下の手順で Azure のリソースを用意してください。

1. 以下のドキュメントを参照し、IoT Central アプリケーションを作成してください。さらに、作成したアプリケーションにデバイス(GR-MANGO)を登録してください。

[Quickstart - Connect a device to an Azure IoT Central application | Microsoft Learn](#)

2. IoT Central アプリケーションにデバイスを登録した際に生成される以下の値を記録してください。これらの値は、GR-MANGO を IoT Central に接続するために用います。

- ID Scope
- Device ID
- Primary Key

7.2.2 ターゲットボードとの接続

ターゲットボードと PC を以下のように接続します。さらに、ターゲットボードをインターネット接続します。

Note : 接続の状態は sample_azure_iot_embedded_sdk を参考にしてください。

1. PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

2. ボードの CN8 コネクタにイーサネットケーブルを接続し、インターネット接続します。

7.2.3 config ファイルの修正

サンプルプロジェクトの config ファイルに、IoT Central に接続するための情報を入力します。

1. サンプルプロジェクト“sample_azure_iot_embedded_sdk_pnp”に含まれる config ファイル“sample_config.h”を開きます。
2. 以下の表に沿って、マクロに値を定義します。

表 7-2 マクロ定義

Constant name	Value
ID_SCOPE	{Your ID Scope value}
REGISTRATION_ID	{Your Device ID value}
DEVICE_SYMMETRIC_KEY	{Your Primary Key value}

```

@ sample_config.h
 63  /* Required when DPS is used. */
64  #ifndef ENDPOINT
65  #define ENDPOINT "global.azure-devices-provisioning.net"
66  #endif /* ENDPOINT */
67
68  #ifndef ID_SCOPE
69  #define ID_SCOPE "{Your idScope value}"
70  #endif /* ID_SCOPE */
71
72  #ifndef REGISTRATION_ID
73  #define REGISTRATION_ID "{Your registrationId value}"
74  #endif /* REGISTRATION_ID */
75
76  #endif /* ENABLE_DPS_SAMPLE */
77
78  /* Optional SYMMETRIC KEY. */
79  #ifndef DEVICE_SYMMETRIC_KEY
80  #define DEVICE_SYMMETRIC_KEY "{Your primaryKey value}"
81  #endif /* DEVICE_SYMMETRIC_KEY */

```

図 7-11 マクロ定義

7.2.4 サンプルプロジェクトのビルド

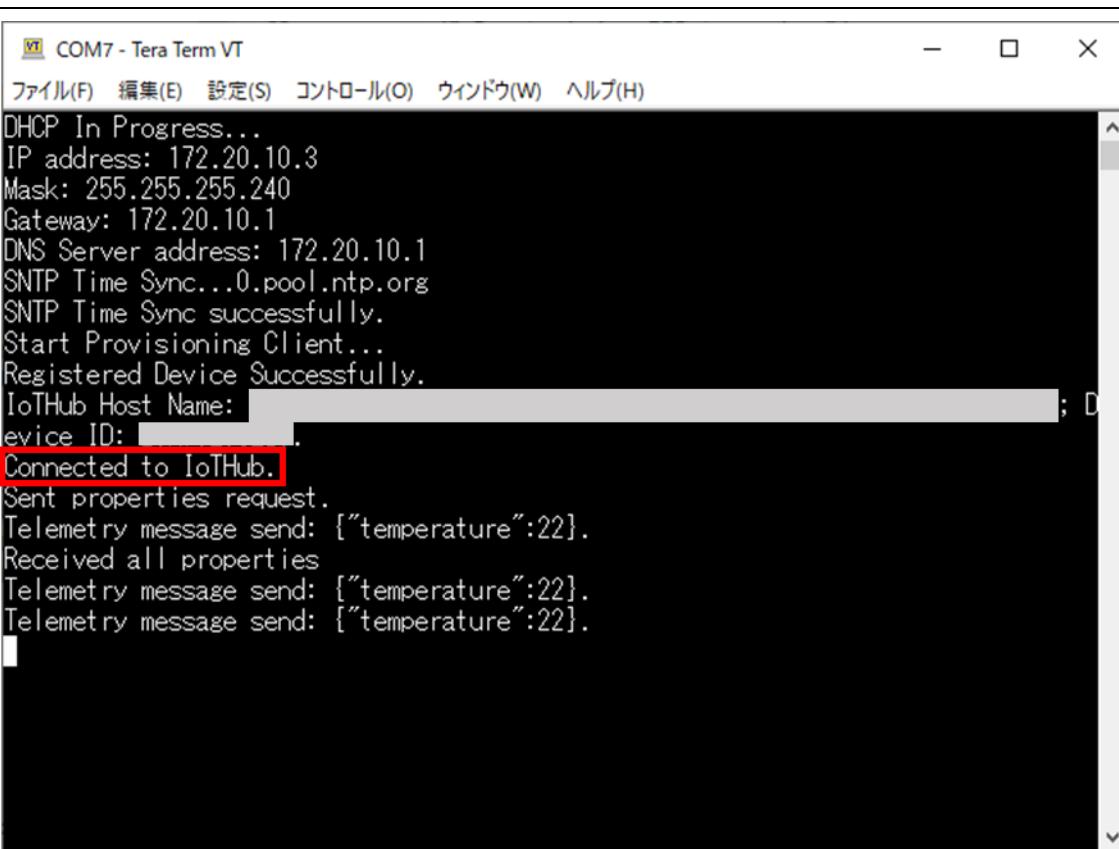
以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト“sample_azure_iot_embedded_sdk_pnp”を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが”0 errors”(正常終了)を表示して終了することを確認します。

7.2.5 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、"MBED.HTM"と"DETAILS.TXT"の二つのファイルだけがあることを確認します。
Note : コピーが失敗すると、ドライブに"FAIL.TXT"ファイルが用意されます。詳しくは Mbed の資料を確認してください。
6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. ターミナルソフトに、以下のような表示が行われることを確認してください。



The screenshot shows a terminal window titled "COM7 - Tera Term VT". The menu bar includes "ファイル(F)", "編集(E)", "設定(S)", "コントロール(O)", "ウインドウ(W)", and "ヘルプ(H)". The terminal output is as follows:

```
DHCP In Progress...
IP address: 172.20.10.3
Mask: 255.255.255.240
Gateway: 172.20.10.1
DNS Server address: 172.20.10.1
SNTP Time Sync...0.pool.ntp.org
SNTP Time Sync successfully.
Start Provisioning Client...
Registered Device Successfully.
IoTHub Host Name: [REDACTED]; D
evice ID: [REDACTED].
Connected to IoTHub.
Sent properties request.
Telemetry message send: [{"temperature":22}].
Received all properties
Telemetry message send: [{"temperature":22}].
Telemetry message send: [{"temperature":22}].
```

図 7-12 "sample_azure_iot_embedded_sdk_pnp"からのメッセージ

8. IoT Central から、以下の操作を行うことができます。

- “Overview”タブで、受信データのグラフが確認できます。本サンプルプロジェクトでは、デバイスからダミーデータとして{“temperature”: 22} が送信されます。

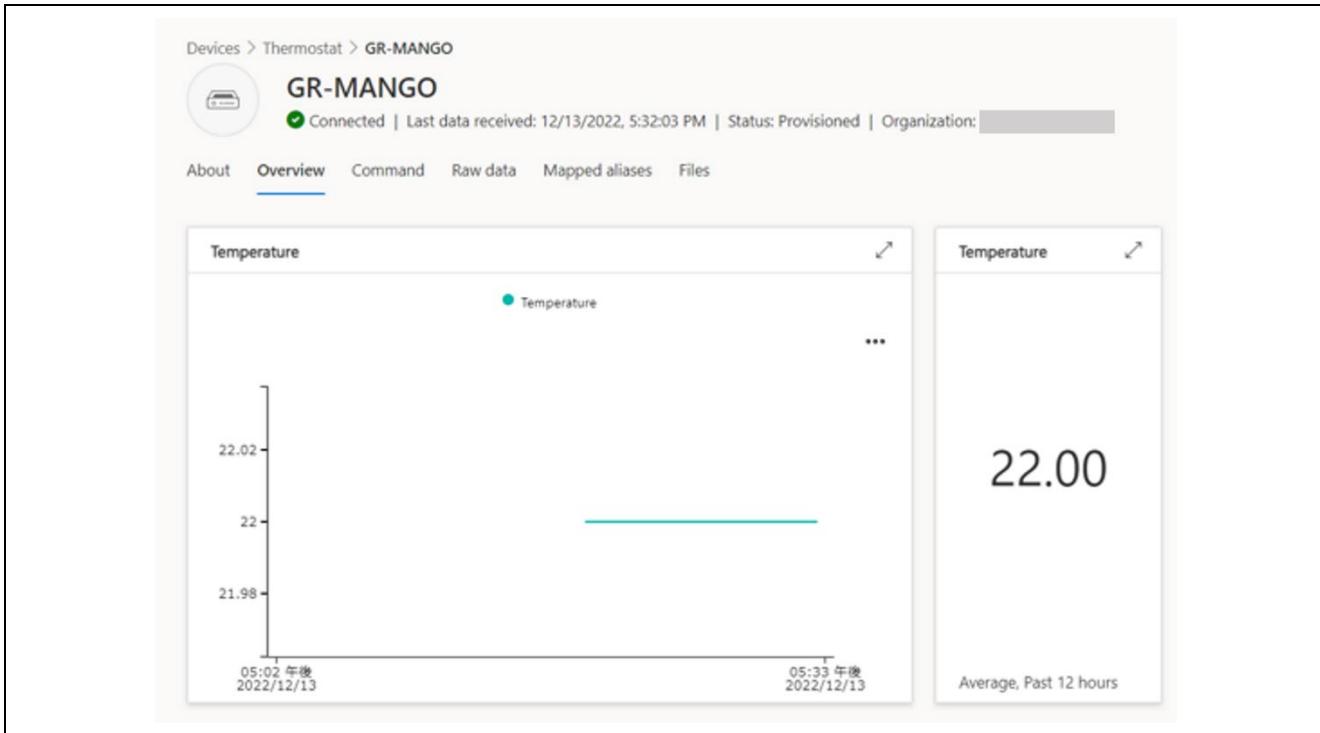


図 7-13 受信データのグラフ

- “Command”タブで、デバイスに IoT Plug and Play コマンドを送信できます。

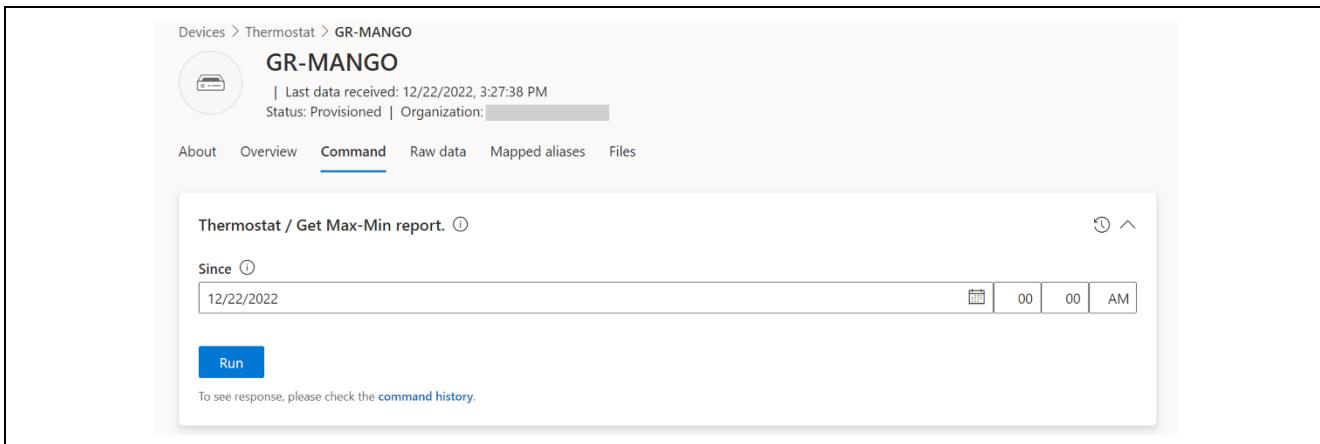


図 7-14 IoT Plug and Play コマンドの送信

- “Raw data”タブで、“Telemetry”や“Command response”等の生データを確認できます。

The screenshot shows the Azure portal interface for a GR-MANGO device. The top navigation bar includes 'Devices > Thermostat > GR-MANGO'. Below the navigation is a summary card for 'GR-MANGO' showing it is connected, last data received on 12/22/2022 at 3:29:08 PM, and its status is Provisioned. Below the summary are tabs: About, Overview, Command, Raw data (which is selected), Mapped aliases, and Files. The main content area is titled 'Raw data' and displays a table of log entries. The columns are: Timestamp, Message type, Event creation time, Temperature, and Max temperature since last reb. The table contains the following data:

Timestamp	Message type	Event creation time	Temperature	Max temperature since last reb.
12/22/2022, 3:28:57 PM	Command response	12/22/2022, 3:28:57 PM		
12/22/2022, 3:28:57 PM	Command request	12/22/2022, 3:28:56 PM		
12/22/2022, 3:28:53 PM	Telemetry		22	
12/22/2022, 3:28:48 PM	Telemetry		22	

A specific log entry for the command response is expanded, showing its raw JSON content:

```
1  {
2   "eventcreationtime": "2022-12-22T06:28:57.499Z",
3   "getAvgHinReport": {
4     "maxTemp": 22,
5     "minTemp": 22,
6     "avgTemp": 22,
7     "startTime": "2022-12-21T15:00:00Z",
8     "endTime": "2023-01-10T10:00:00Z"
9   },
10  "_eventtype": "Command response",
11  "timestamp": "2022-12-22T06:28:57.052Z"
```

図 7-15 生データの確認

7.3 sample_pnp_temperature_controller

sample_pnp_temperature_controller は、Azure RTOS の Azure IoT ミドルウェアを用いて Azure IoT Hub に接続するサンプルプロジェクトです。本サンプルプロジェクトは、複数の IoT Plug and Play コンポーネントを含んでいます。IoT Plug and Play については、以下のドキュメントを参照してください。

[Introduction to IoT Plug and Play | Microsoft Learn](#)

7.3.1 Azure リソースの用意

以下の手順で Azure のリソースを用意してください。

- 以下のドキュメントを参照し、IoT Central アプリケーションを作成してください。さらに、作成したアプリケーションにデバイス(GR-MANGO)を登録してください。

[Quickstart - Connect a device to an Azure IoT Central application | Microsoft Learn](#)

- IoT Central アプリケーションにデバイスを登録した際に生成される以下の値を記録してください。これらの値は、GR-MANGO を IoT Central に接続するために用います。
 - ID Scope
 - Device ID
 - Primary Key

7.3.2 ターゲットボードとの接続

ターゲットボードと PC を以下のように接続します。さらに、ターゲットボードをインターネット接続します。

Note : 接続の状態は sample_azure_iot_embedded_sdk を参考にしてください。

- PC とボードの CN1 コネクタを USB ケーブルで接続し、ボードの電源を供給します。

Note : CN1 と USB で接続することにより、USB メモリ(Mass Storage Class)と UART(CDC ACM)として PC に認識されます。UART はコンソールとして使用します。

- ボードの CN8 コネクタにイーサネットケーブルを接続し、インターネット接続します。

7.3.3 config ファイルの修正

サンプルプロジェクトの config ファイルに、IoT Central に接続するための情報を入力します。

- サンプルプロジェクト“sample_pnp_temperature_controller”に含まれる config ファイル “sample_config.h”を開きます。
- マクロに値を定義します。

Note : sample_azure_iot_embedded_sdk_pnp を参考にしてください。

7.3.4 サンプルプロジェクトのビルド

以下の手順でサンプルプロジェクトをビルドし、実行ファイルを作成してください。

1. e2 studio の[プロジェクトエクスプローラ]上で、でビルド対象のサンプルプロジェクト”sample_pnp_temperature_controller”を選択します。
2. 選択したプロジェクトが、[HardwareDebug]ビルド構成になっていることを確認します。
3. ビルド対象のコンポーネントプロジェクトの上でマウスを右クリックします。表示されたメニューから[プロジェクトのビルド]を選択し、プロジェクトのビルドを実行します。
4. プロジェクトのビルドが”0 errors”(正常終了)を表示して終了することを確認します。

7.3.5 サンプルプロジェクトの実行

以下の手順で実行ファイルをボードに書き込み、プログラムを実行します。

1. 実行ファイルが保存されているフォルダをエクスプローラで開きます。
2. 別のエクスプローラで、ボードを PC に接続したときに認識したドライブを開きます。
3. 実行ファイルが保存されているフォルダにある.bin 拡張子のファイルを、ボードを PC に接続したときに認識したドライブにコピーします。
4. コピーが終了すると、PC が認識したドライブが閉じられ、再度ドライブとして認識します。
5. 再度認識したドライブに、”MBED.HTM”と”DETAILS.TXT”的二つのファイルだけがあることを確認します。

Note : コピーが失敗すると、ドライブに”FAIL.TXT”ファイルが用意されます。詳しくは Mbed の資料を確認してください。

6. PC とボードを接続している USB ケーブルを抜き、再度接続します。
7. ターミナルソフトに、以下のような表示が行われることを確認してください。

```
COM7 - Tera Term VT
DHCP In Progress...
IP address: 172.20.10.3
Mask: 255.255.255.240
Gateway: 172.20.10.1
DNS Server address: 172.20.10.1
SNTP Time Sync...0.pool.ntp.org
SNTP Time Sync successfully.
Start Provisioning Client...
Registered Device Successfully.
IoTHub Host Name: [REDACTED]; D
evice ID: [REDACTED]
Connected to IoTHub.
Sent properties request.
Temp Controller Telemetry message send: {"workingSet":1193}.
Thermostat thermostat1 Telemetry message send: {"temperature":22}.
Thermostat thermostat2 Telemetry message send: {"temperature":22}.
Received all properties
Temp Controller Telemetry message send: {"workingSet":1081}.
Thermostat thermostat1 Telemetry message send: {"temperature":22}.
Thermostat thermostat2 Telemetry message send: {"temperature":22}.
Temp Controller Telemetry message send: {"workingSet":1011}.
Thermostat thermostat1 Telemetry message send: {"temperature":22}.
Thermostat thermostat2 Telemetry message send: {"temperature":22}.
```

図 7-16 "sample_pnp_temperature_controller"からのメッセージ

8. IoT Central から、さまざまな操作を行うことができます。

Note : sample_azure_iot_embedded_sdk_pnp を参考にしてください。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Dec. 14.21	—	初版
1.01	Oct. 4.22	—	複数のサンプルプロジェクトを追加
1.02	Jan. 23.23	—	“Azure RTOS IoT Embedded SDK サンプル”を追加

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策

CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。

2. 電源投入時の処置

電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. 電源オフ時における入力信号

当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。

4. 未使用端子の処理

未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。

5. クロックについて

リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

6. 入力端子の印加波形

入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、 V_{IL} (Max.) から V_{IH} (Min.) までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、 V_{IL} (Max.) から V_{IH} (Min.) までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。

7. リザーブアドレス（予約領域）のアクセス禁止

リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

8. 製品間の相違について

型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行なうにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

標準水準：コンピュータ、OA機器、通信機器、計測機器、AV機器、家電、工作機械、パーソナル機器、産業用ロボット等
高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等
当社製品は、データシート等により高信頼性、Harsh environment向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
7. あらゆる半導体製品は、外部攻撃からの安全性を100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因したまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または默示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注2. 本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。

(Rev.5.0-1 2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24 (豊洲フォレシア)

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。