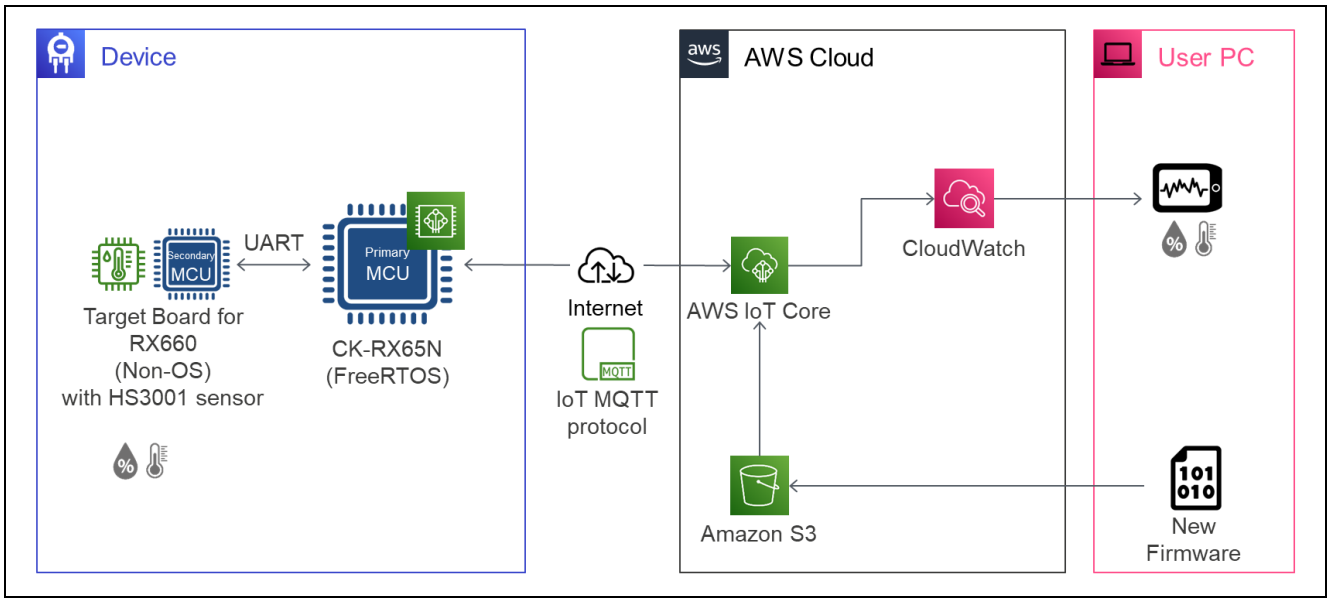
RX65N グループ

FreeRTOS を用いた Amazon Web Services によるセカンダリデバイスの OTA アップデートサンプルコード

要旨

本アプリケーションノートでは、Amazon Web Services™（以下、AWS と略します）との通信を行うプライマリ MCU の RX65N を用いて、センサデータの測定機能を持ったセカンダリ MCU の RX マイコンに対して、AWS のサービスを利用した OTA アップデート（以下、セカンダリ OTA アップデートと略します）を行うデモについて説明します。



動作確認デバイス

プライマリ MCU：RX65N
セカンダリ MCU：RX660
温湿度センサ：HS3001 高性能 相対湿度・温度センサ

動作確認ボード

プライマリ MCU: CK-RX65N (RTK5CK65N0S04000BE)
セカンダリ MCU ：Target Board for RX660 (RTK5RX6600C00000BJ)
温湿度センサ　　：温湿度センサ Pmod™ ボード (US082-HS3001EVZ)

関連文書

本アプリケーションノートは、以下の文書を参照し、説明しています。文書更新の場合に章構成等が変わる場合があります。参照の際に注意してください。

RX ファミリ ファームウェアアップデートモジュール Firmware Integration Technology (R01AN6850)

RX ファミリ AWS/Azure を利用したファームウェア更新ソフトの開発ガイド QE for OTA (R20AN0712)

RX ファミリ RX65N における Amazon Web Services を利用した FreeRTOS OTA の実現方法 (v202210.01-LTS-rx-1.1.0 以降対応版)

RX65N Group CK-RX65N v1 User's Manual (R20UT5100)

RX660 グループ Target Board for RX660 ユーザーズマニュアル (R20UT5068)

（最新版をルネサス エレクトロニクスホームページから入手してください。）

テクニカルアップデート／テクニカルニュース

（最新の情報をルネサス エレクトロニクスホームページから入手してください。）

目次

1. 概要

デモでは、セカンダリ OTA アップデートによって稼働センサを追加し、ブラウザ上の AWS 画面でセンサデータを可視化することで取得するセンサデータの追加を確認できます。

IoT 機器にはセキュリティの脆弱性の適宜修正や、お客様要望に応じた機能のアップデートが求められます。従来、提供しているプライマリ MCU の OTA アップデートだけでなく、セカンダリ OTA アップデートを実現することで、セカンダリ MCU の脆弱性への対応やフレキシブルなサービスのアップデートが可能な製品開発を実現できます。

2. 動作確認条件

本アプリケーションノートのサンプルプログラムは以下に示す条件でデモの動作を確認しています。

表 2-1 デモ動作確認条件 (RX65N)

項目	内容
使用マイコン	RX65N (R5F565NEHDFB)
使用ボード	CK-RX65N (RTK5CK65N0S04000BE)
動作電圧	3.3V
RTOS	FreeRTOS v202210.01-LTS-1.1.3
統合開発環境（IDE）	e² studio 2024-01 (24.1.0) QE for OTA v2.00
C コンパイラ	ルネサス製 RX ファミリ用 C/C++コンパイラ CC-RX v3.06.00
フラッシュ書き込みツール	Renesas Flash Programmer V3.14.00

表 2-2 デモ動作確認条件 (RX660)

項目	内容
使用マイコン	RX660 (R5F56609BDFP)
使用ボード	Target Board for RX660 (RTK5RX6600C00000BJ)
動作電圧	3.3V
統合開発環境（IDE）	e² studio 2024-01 (24.1.0)
C コンパイラ	ルネサス製 RX ファミリ用 C/C++コンパイラ CC-RX v3.06.00 GCC for Renesas RX 8.3.0.202202
フラッシュ書き込みツール	Renesas Flash Programmer V3.14.00
MOT ファイル変換ツール	Renesas Image generator (RX660 プロジェクトに付属)
USB-UART コンバータ	Pmod USBUART™

表 2-3 デモ動作確認条件 (センサ)

項目	内容
温湿度センサボード	US082-HS3001EVZ Board

表 2-4 デモ動作確認条件 (その他)

項目	バージョン
Python	3.10.4

QE for OTA は、https://www.renesas.com/qe-ota/ から入手できます。

Python は、https://www.python.org/ から入手できます。

3.　ハードウェア説明

本システムは、AWS との通信制御機能を持つ RX65N マイコン(プライマリ MCU)と、HS3001 センサが接続された RX660 マイコン(セカンダリ MCU)で構成されています。2 つのマイコンは UART による相互通信を行います。

図 3-1 にシステム構成を示します。

プライマリ MCU として、RX65N を搭載した CK-RX65N を使用します。

セカンダリ MCU として、RX660 を搭載した Target Board for RX660 (以下、TB-RX660 と略します)を使用します。

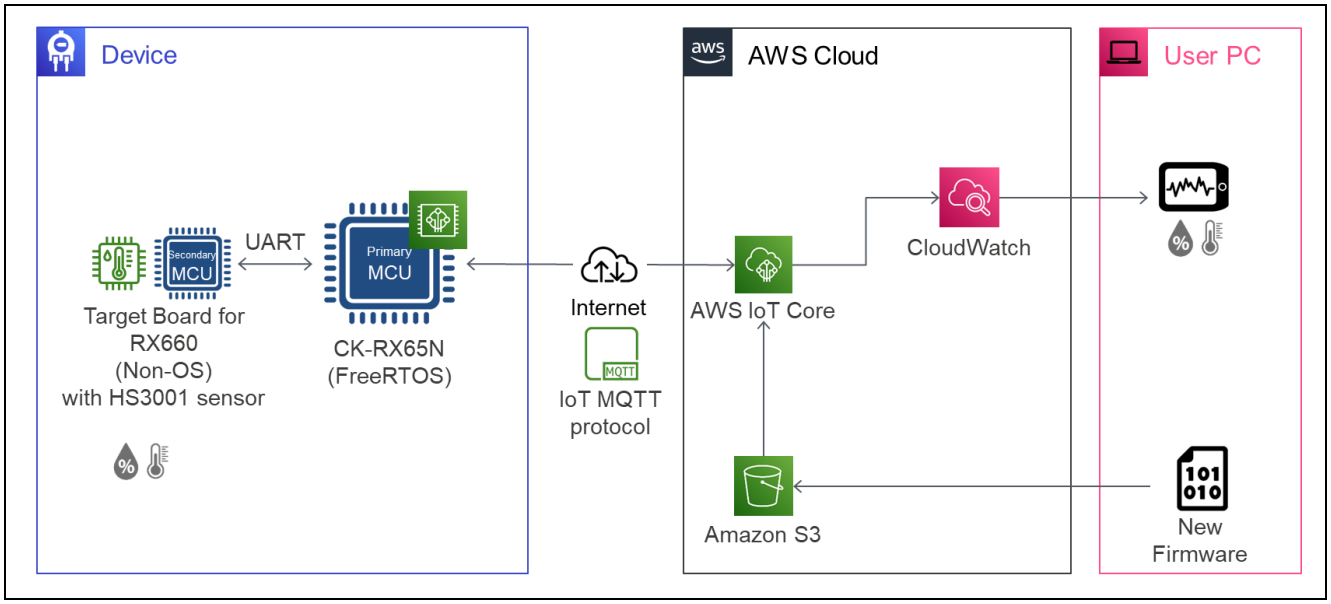


図 3-1　本デモのシステム構成

4.　ソフトウェア説明

AWS 認定済プログラムを使用した RX65N のファームウェアには、FreeRTOS™ with IoT Library が実装されています。そのため、AWS が提供しているマネージドサービスである AWS IoT Core および AWS IoT Device Management を利用して、OTA によるファームウェアアップデートや MQTT 通信によるクラウドへのデータのアップロードが実行可能です。

セカンダリ OTA アップデートの制御には、プライマリ MCU 側の RX65N マイコンでは AWS IoT Over-the-air Update Library を利用し、AWS から受信したセカンダリ MCU 用の更新ファームウェアをセカンダリ MCU に転送し、ファームウェア更新を実現しています。

セカンダリ MCU 側の RX マイコンのファームウェアアップデートの制御は「RX ファミリ ファームウェアアップデート モジュール Firmware Integration Technology アプリケーションノート Rev.2.01」を使用します。

4.1　ファームウェアアップデート方式

本サンプルプログラムのセカンダリ MCU のファームウェア更新の仕組みは、ファームウェアアップデートモジュールが提供している方式のうち、「リニアモードの半面更新方式」を使用しています。この方式の詳細は、「RX ファミリ ファームウェアアップデート モジュール Firmware Integration Technology アプリケーションノート Rev.2.01」の 1.3 章「各ファームウェアアップデート方式について」の「リニアモードの半面更新方式」をご参照ください。

以下に本サンプルプログラムのメモリマップを示します。

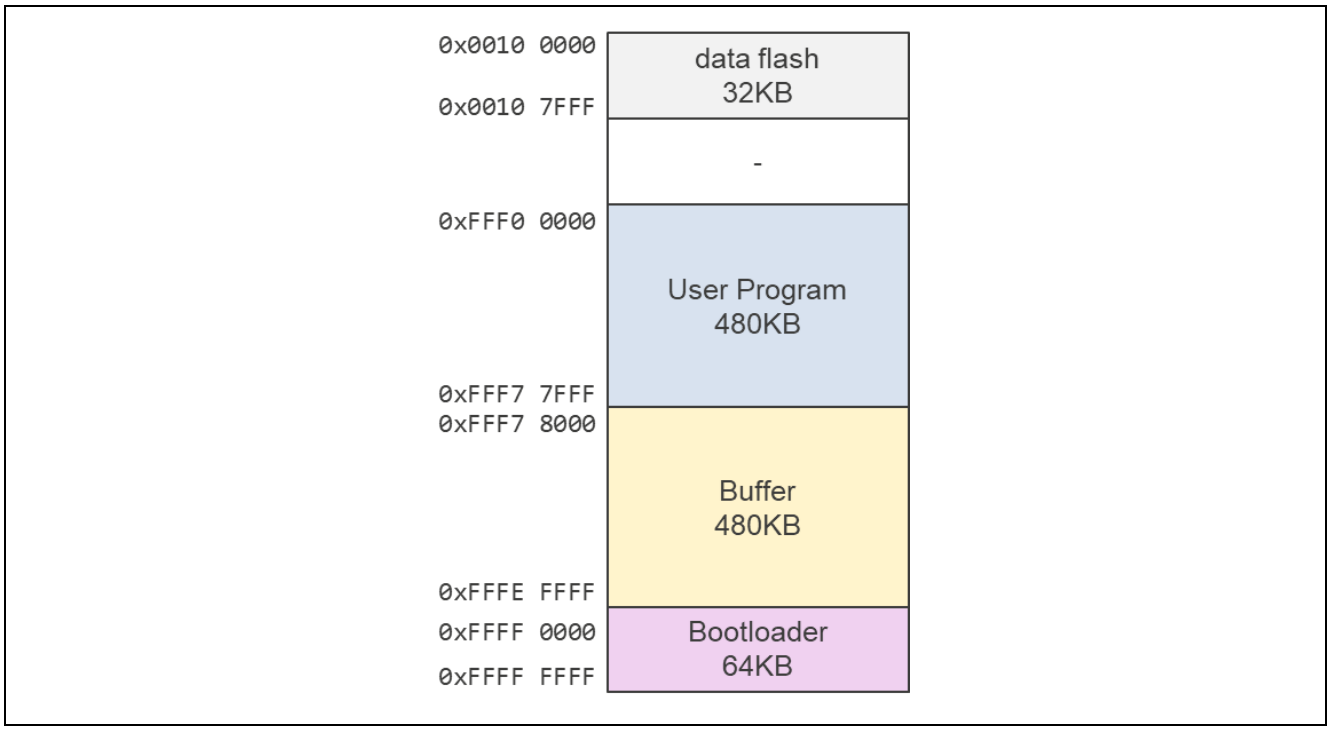


図 4-1　TB-RX660 サンプルプログラムのメモリマップ

以下の図 4-2 にセカンダリ OTA アップデートの動作概要を示します。また、図 4-3 にアップデートの実行中の各フェーズでの ROM の状態を示します。なお、図 4-3 の赤枠はその時実行されているプログラムを表しています。

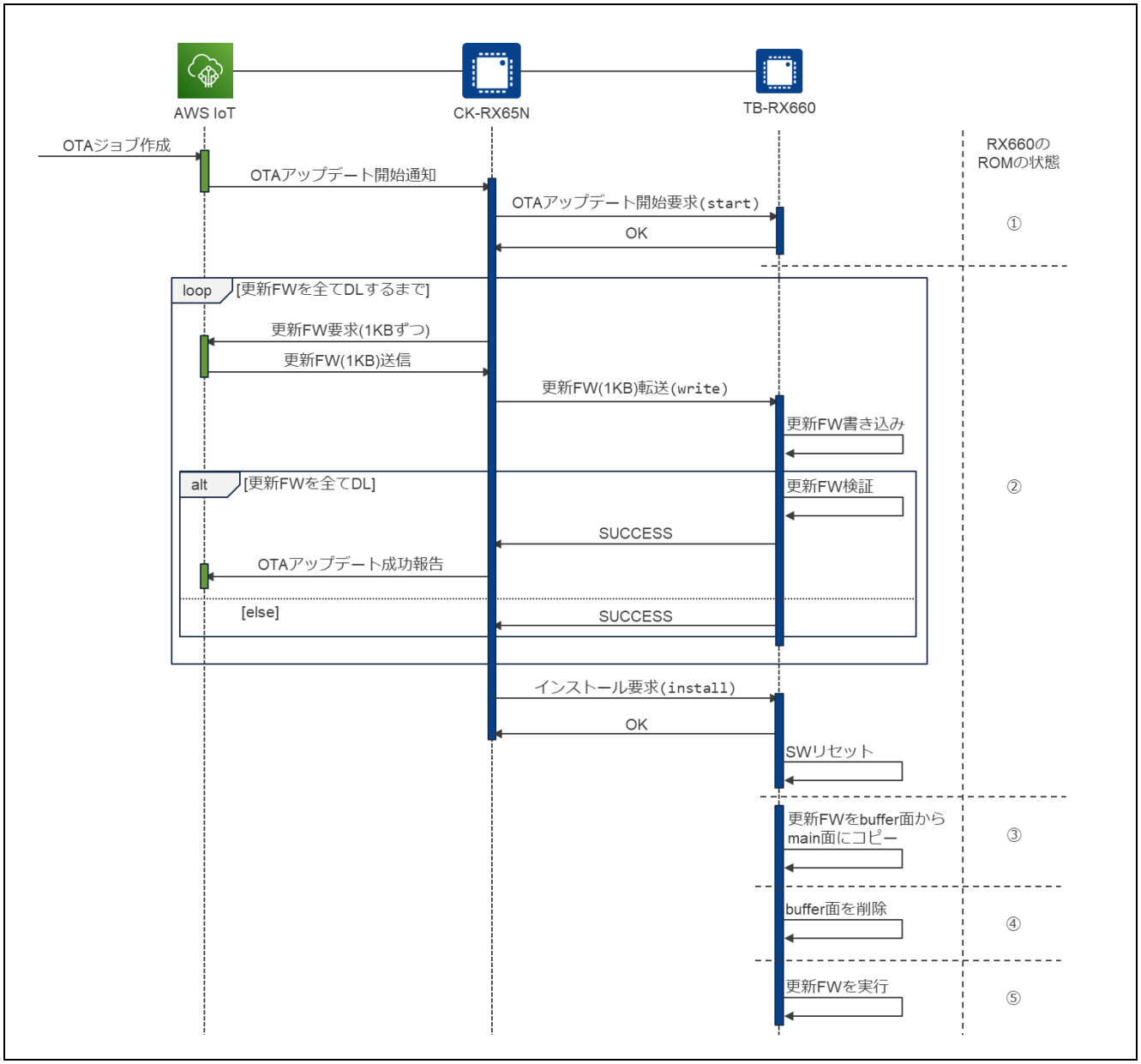


図 4-2　セカンダリ OTA アップデートの動作概要

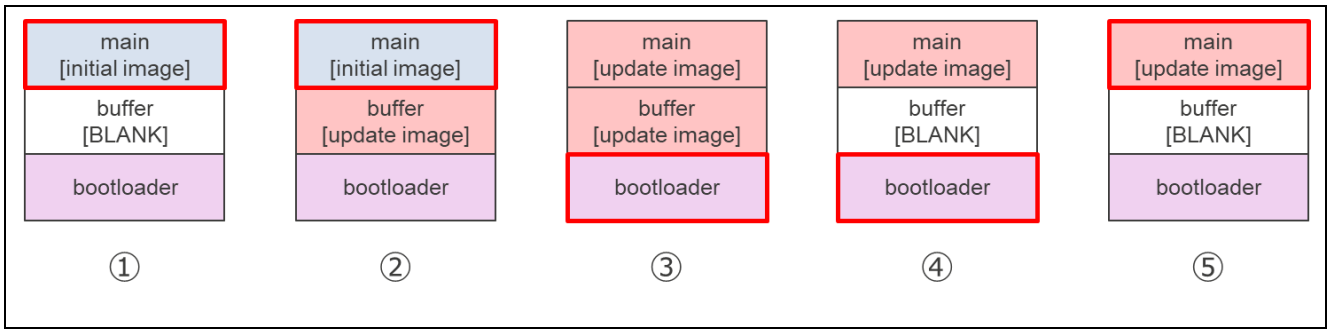


図 4-3　アップデート実行中のセカンダリ MCU の ROM の状態

4.2　マイコン間 UART 通信

プライマリ MCU とセカンダリ MCU 間の UART 通信は、プライマリ MCU を Master、セカンダリ MCU を Slave として、Master からコマンドを送信し、受信した Slave がそれに応じた処理を実施し、結果をレスポンスとして返します。

4.2.1　UART 通信設定

UART 通信設定を表 4-1 に示します。

表 4-1　MCU 間 UART コマンド／レスポンス通信ソフトウェアの UART 通信設定

項目	内容
Data Length	8-bit
Parity	None
Stop Bits	1-bit
Flow Control	None
Bitrate	1Mbps or under（動作確認速度）

4.2.2　データフォーマット

Master⇔Slave 間で行うコマンド／レスポンスの Packet 通信の仕様を説明します。

4.2.2.1　Packet 全体のデータフォーマット

Packet 全体のデータフォーマットを図 4-4 に示します。Header と Payload で構成されています。

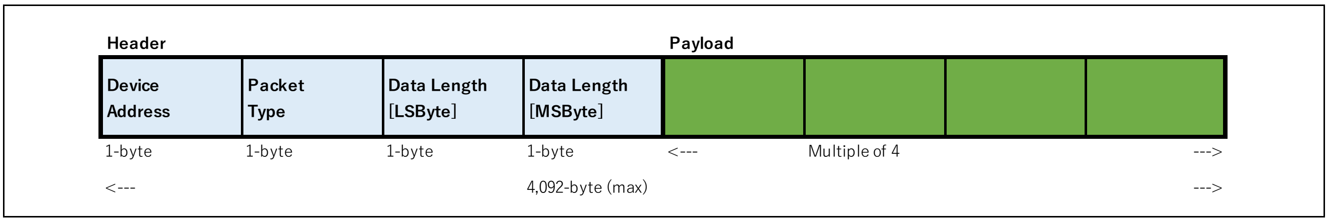


図 4-4　Packet 全体のデータフォーマット

Packet の Header 仕様を表 4-2 に示します。

表 4-2　Packet の Header 仕様

項目	内容
Device Address	デバイスアドレスです。 各デバイスは、Device Address を参照し、自デバイスかどうかを判定します。 Slave Device は自アドレス宛と判断した場合、Packet の詳細解析を開始します。 • 0x00: Device Address for MCU without Slave function • 0x01 – 0xFE: Slave Device Address • 0xFF: Reserved
Packet Type	Packet 属性です。 • b7-b6: 00b fixed • b5: 0: Request / 1: Response • b4: [Request] 0 fixed [Response] 0: ACK / 1: NACK【注 1】 • b3 - b0: Command Type 0000b: Common command 0001b: Reserved 0010b: FWUP command 0011b: Sensor command 0100b – 1111b: Reserved
Data Length	Payload のデータサイズです。

注 1：ACK/NACK は、Packet のデータフォーマットの正常／異常を示します。

4.2.2.2　Payload のデータフォーマット

Payload のデータフォーマットを図 4-5 に示します。Command Header と Command Data で構成されています。

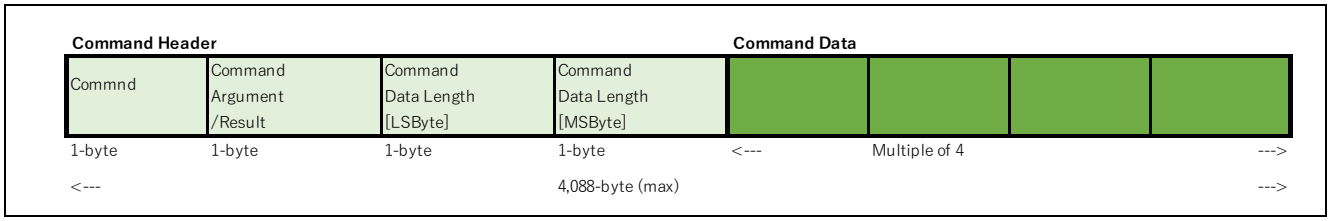


図 4-5　Payload のデータフォーマット

Command Header 仕様を表 4-3 に示します。

表 4-3　Command Header 仕様

項目	内容
Command	コマンドです。詳細は「4.2.3 コマンド仕様」を参照してください。
Command Argument / Result	Request 時、コマンド用の引数です。 • b7-b4: 00b fixed • b3: 0b Reserved • b2: 0b Reserved • b1-b0: Argument 00: Slave0 01: Master0 10: Master1 11: Master2
	Response 時、コマンド実行結果です。 • b7-b4: 1111: コマンド実行失敗 1110: 無効な Command Header（未サポートコマンド） … 0001: コマンド実行正常終了 1 0000: コマンド実行正常終了 • b3: 0b Reserved: Request 時と同じ • b2: 0b Reserved: Request 時と同じ • b1-b0: Argument: Request 時と同じ
Command Data Length	Command Data のデータサイズです。4 の倍数値を設定してください。

4.2.3　コマンド仕様

4.2.3.1　FWUP コマンド

ファームウェア(FW)更新時に使用するコマンドです。表 4-4 にコマンド一覧を示します。

表 4-4　FWUP コマンドリスト

Command Type	Command	内容
FWUP	START：FW 更新開始コマンド	FW 更新開始
	WRITE：FW 書き込みコマンド	更新 FW の書き込み
	INSTALL：FW インストールコマンド	更新 FW のインストールと実行
	CANCEL：FW キャンセルコマンド	FW 更新の中止

(1)　START：FW 更新開始コマンド

FW 更新開始を要求します。Target ID に任意の値を設定できます。ユーザ側での識別コード等に利用可能です。

Slave は結果を Result に設定し、応答を返します。

ファームウェアアップデートモジュールを用いて更新 FW を書き込み可能な状態にします。

表 4-5　FWUP Command Type: FWUP_START Command Data Format

COMMAND	Packet	Device Address	Packet Type	Data Length		Command	Command Argument	Command Data Length		Command Data			
				LSB	MSB		/Result	LSB	MSB	0	1	2	3
FWUP / START	Request	0xXX	0x02	0x08	0x00	0x02	0x00	0x04	0x00	Target ID [Little Endian]			
	ACK Resp.		0x22	0x04			【注 1】	0x00					

注 1：Response 時の Result 値は、以下を示します。
0x00: Successful Operation
0xFF: Failed Operation

(2)　WRITE：FW 書き込みコマンド

バイナリデータの先頭を Offset 0 とする Offset Address とその FW データを送信し、FW 書き込みを要求します。

Slave は書き込み処理を実行します。また、最終ブロックの場合、さらに署名検証処理を実行します。Slave は結果を Result に設定し、応答を返します。

表 4-6　FWUP Command Type: FWUP_WRITE Command Data Format

COMMAND	Packet	Device Address	Packet Type	Data Length		Command	Command Argument	Command Data Length		Command Data			
				LSB	MSB		/Result	LSB	MSB	0	1	2	3
FWUP / WRITE	Request	0xXX	0x02	8 + (Length of Write Data)		0x04	0x00	4 + (Length of Write Data)		Offset Address [Little Endian] : 4-byte Write Data [Big Endian] : N bytes 【注 2】			
	ACK Resp.		0x22	0x04	0x00		【注 1】	0x00	0x00				

注 1：Response 時の Result 値は、以下を示します。
0x00: Successful Operation
0xFF: Failed Operation
注 2：送信したバイナリデータの順で書き込まれます。

(3)　INSTALL：FW インストールコマンド

書き込まれた更新 FW のインストールと実行を要求します。

Slave は結果を Result に設定し、応答を返した後、更新 FW を実行します。

表 4-7　FWUP Command Type: FWUP_INSTALL Command Data Format

COMMAND	Packet	Device Address	Packet Type	Data Length		Command	Command Argument /Result	Command Data Length	
				LSB	MSB			LSB	MSB
FWUP / INSTALL	Request	0xXX	0x02	0x04	0x00	0x06	0x00	0x00	0x00
	ACK Resp.		0x22				【注 1】		

注 1：Response 時の Result 値は、以下を示します。

0x00: Successful Operation

0xFF: Failed Operation

(4)　CANCEL：FW キャンセルコマンド

FW 更新中止を要求します。

Slave は更新を中断し、書き込んだ更新 FW の消去処理を実行します。Slave は結果を Result に設定し、応答を返します。

表 4-8　FWUP Command Type: FWUP_CANCEL Command Data Format

COMMAND	Packet	Device Address	Packet Type	Data Length		Command	Command Argument /Result	Command Data Length	
				LSB	MSB			LSB	MSB
FWUP / CANCEL	Request	0xXX	0x02	0x04	0x00	0xF0	0x00	0x00	0x00
	ACK Resp.		0x22				【注 1】		

注 1：Response 時の Result 値は、以下を示します。

0x00: Successful Operation

0xFF: Failed Operation

4.2.3.2　SENSOR コマンド

センサ制御用に利用可能なコマンドです。表 4-9 にコマンド一覧を示します。

表 4-9　SENSOR コマンドリスト

Command Type	Command	内容
SENSOR	GET_HS300X：HS300x 測定データ取得コマンド	HS300x 測定データ取得

(1)　GET_HS300X：HS300x 測定データ取得コマンド

Slave に接続された HS300x センサの測定データ送信を要求します。Slave は結果を Result に設定し、応答を返します。

Master は、Result を参照し、データの有効性を確認してください。Stale Data の場合、旧データ状態を示します。

表 4-10　Sensor Command Type: GET_HS300X Command Data Format

COMMAND	Packet	Device Address	Packet Type	Data Length		Command	Command Argument	Command Data Length		Command Data			
				LSB	MSB		/Result	LSB	MSB	0	1	2	3
SENSOR / GET_HS300X	Request	0xXX	0x03	0x04	0x00	0x01	0x00	0x00	0x00				
	ACK Resp.		0x23	0x0C			【注 1】	0x08		Humidity [Little Endian] : 4-byte [float type] Temperature [Little Endian] : 4-byte [float type]			

注 1：Response 時の Result 値は、以下を示します。

0x00: Valid Data
0x01: Stale Data
0xFX: Failed Operation

4.3　フォルダ/ファイル構成

図 4-6 にフォルダ/ファイル構成を示します。

```
r01an6220xx0200-rx-2nd-ota-apl
├─Demo
│   ├─ccrx
│   │   ├─afr-v202210.01-LTS-rx-1.1.3
│   │   ├─ck_rx65n_2ndota_demo
│   │   ├─ck_rx65n_demo_bootloader
│   │   ├─rx660_tb_2ndota_demo
│   │   └─rx660_tb_demo_bootloader
│   └─gcc
│       ├─rx660_tb_2ndota_demo
│       └─rx660_tb_demo_bootloader
└─r01an6220jj0200-rx-2nd-ota-apl.pdf
```

図 4-6　フォルダ／ファイル構成

ck_rx65n_2ndota_demo フォルダと ck_rx65n_demo_bootloader フォルダには CK-RX65N 用のプロジェクトファイルが格納されています。

rx660_tb_2ndota_demo フォルダと rx660_tb_demo_bootloader フォルダには、TB-RX660 用のプロジェクトファイルが格納されています。

CK-RX65N 用のプロジェクトは CC-RX コンパイラのみ、TB-RX660 用のプロジェクトは CC-RX, GCC コンパイラに対応しています。

4.4　コードサイズ

本アプリケーションのサンプルコードに含まれるプロジェクトの ROM, RAM サイズを下表に示します。下表の値は以下の条件で確認しています。

CC-RX
　Compiler
　　最適化レベル (-optimize): Level 2: Performs whole module optimization
　　最適化タイプ (-speed/-size): Optimizes with emphasis on code size
　Linker
　　最適化タイプ (-nooptimize/-optimize): All
　Library Generator
　　最適化レベル (-optimize): Level 2: Performs whole module optimization
　　最適化タイプ (-speed/-size): Optimizes with emphasis on code size

表 4-11　コードサイズ (CC-RX)

プロジェクト	ROM	RAM
ck_rx65n_demo_bootloader	33 KB	9 KB
ck_rx65n_2ndota_demo	368 KB	345 KB
rx660_tb_demo_bootloader	27 KB	13 KB
rx660_tb_2ndota_demo	49 KB	22 KB

GCC
　最適化レベル: Optimize for debug (-Og)

表 4-12　コードサイズ (GCC)

プロジェクト	ROM	RAM
rx660_tb_demo_bootloader	60 KB	17 KB
rx660_tb_2ndota_demo	90 KB	41 KB

5.　デモの動作説明

(1)　デモの初期状態では、TB-RX660 は接続されている HS3001 センサを使って湿度データのみ取得します。

(2)　セカンダリ OTA アップデートの仕組みを用いて、AWS から CK-RX65N 経由で TB-RX660 の更新ファームウェアをダウンロードし、ファームウェアの更新を行います。

(3)　ファームウェア更新後は、TB-RX660 は HS3001 センサから湿度データに加えて温度データも取得します。

一連の流れで、取得しているセンサデータの種類とその値は、両マイコンからの PC へのログ出力と AWS 上のダッシュボードから確認できます。

6.　デモのセットアップ

本アプリケーションノートのデモを実行するために必要なセットアップについて説明します。

CK-RX65N と TB-RX660 の配線や HS3001 センサの接続方法等のハードウェアのセットアップ、それぞれのマイコンボード用の初期ファームウェアの作成と書き込み等のソフトウェアのセットアップ、そして OTA アップデートの実行や AWS 上でのセンサデータの可視化のための AWS クラウド側の準備が必要です。

6.1　ハードウェアのセットアップ

最初に、本デモを構成するハードウェア全体の構成を示します。実際にセットアップ後の画像は図 6-12 を参照してください。以降でそれぞれのボードのセットアップ方法について詳しく説明します。

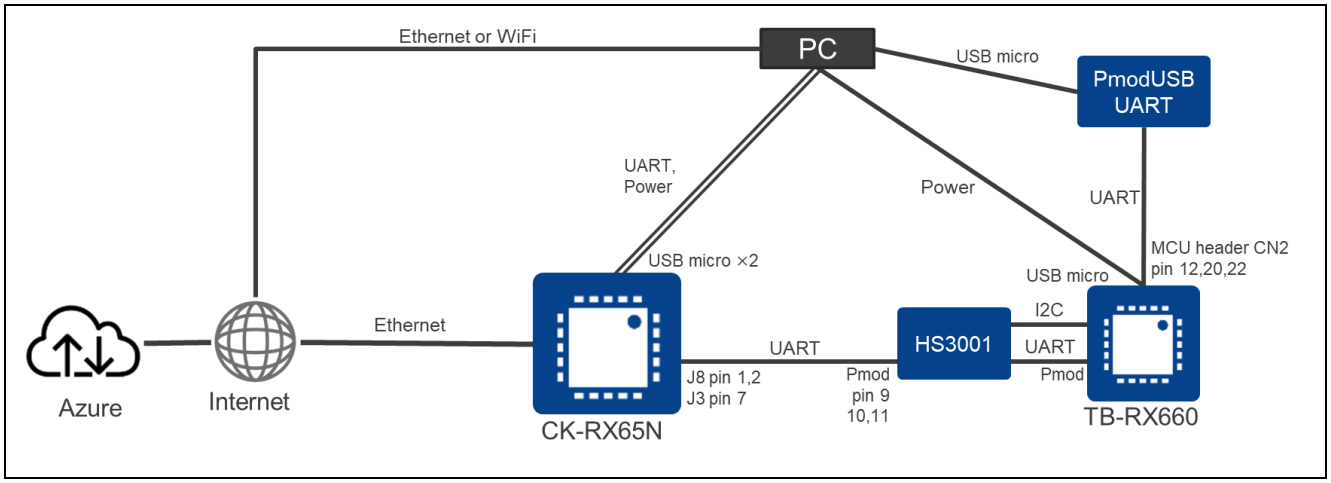


図 6-1　本デモのハードウェア全体構成

6.1.1　CK-RX65N のセットアップ方法

CK-RX65N のセットアップ方法を示します。

(1)　TB-RX660 との UART 通信用ケーブル接続

TB-RX660 と UART 通信を行うための TXD, RXD, GND は CK-RX65N の J8, J3 コネクタの以下の端子に割り当てています。6.1.2(3)に示す TB-RX660 側の端子と以下の表のように UART 信号の対応を取って接続してください。

表 6-1　マイコン間 UART 接続方法（CK-RX65N ⇔ HS3001 センサボード ⇔ TB-RX660）

CK-RX65N	（注 1）	HS3001 センサボード Pmod I/F		TB-RX660 Pmod I/F
J8 Pin 1: D0/RX (RXD7)	⇔	Pin 9: TXD（注 1）	⇔	Pin 9: TXD9
J8 Pin 2: D1/TX (TXD7)	⇔	Pin 10: RXD（注 1）	⇔	Pin 10: RXD9
J3 Pin 7: GND	⇔	Pin 11: GND	⇔	Pin 11: GND

注 1：HS3001 センサボードの両端子は直結処理のため、TB-RX660 の Pmod I/F の入出力信号を扱うことができます。

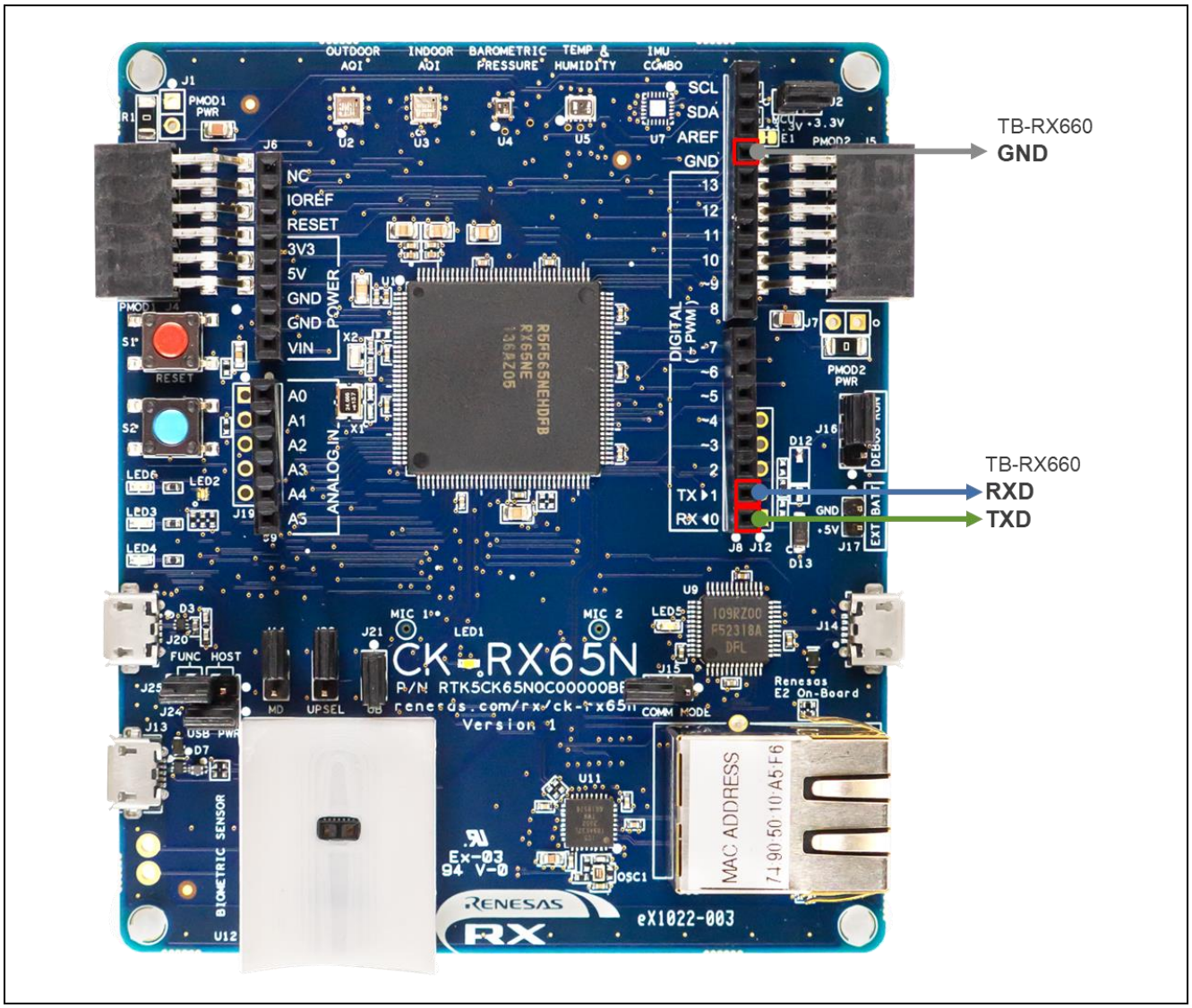


図 6-2　CK-RX65N のマイコン間 UART 通信に使用する端子位置

(2)　PC へのログ出力用ケーブル接続

PC と CK-RX65N の USB シリアルコネクタ(micro USB Type-B)を USB ケーブルで接続します。

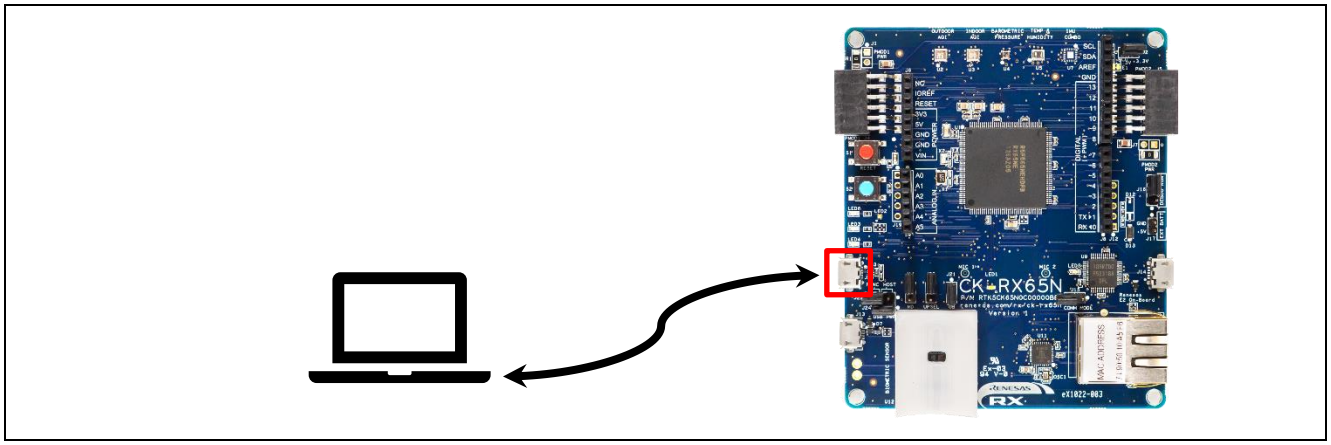


図 6-3　PC へのログ出力用の USB 接続

(3)　電源供給・デバッガとの接続

PC と CK-RX65N の E2OB Debugger コネクタ (micro USB Type-B)を USB ケーブルで接続します。

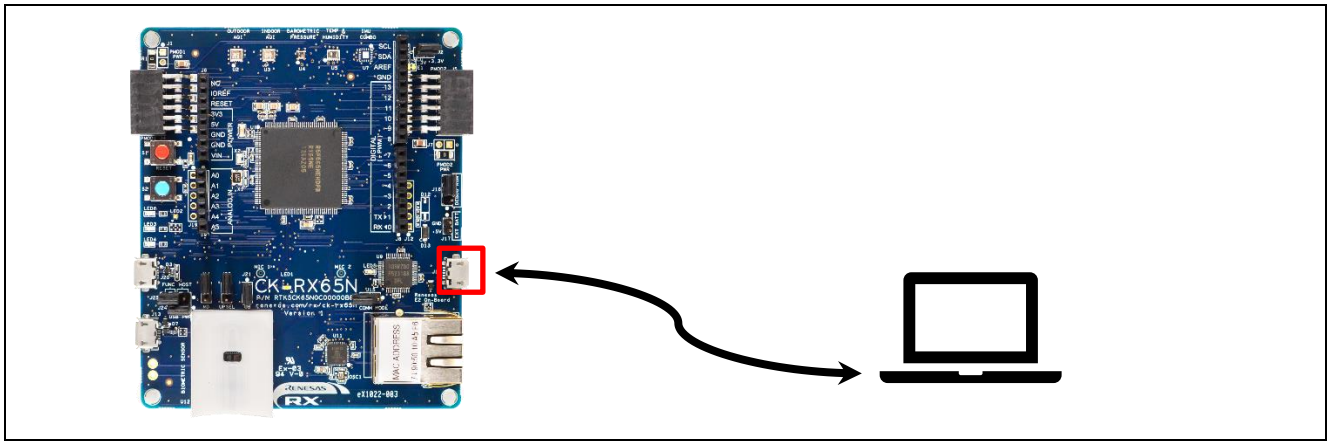


図 6-4 電源供給・エミュレータ用の USB 接続

(4)　インターネット接続用の LAN ケーブル接続

CK-RX65N のイーサネットコネクタにインターネットに接続されている LAN ケーブルを接続します。

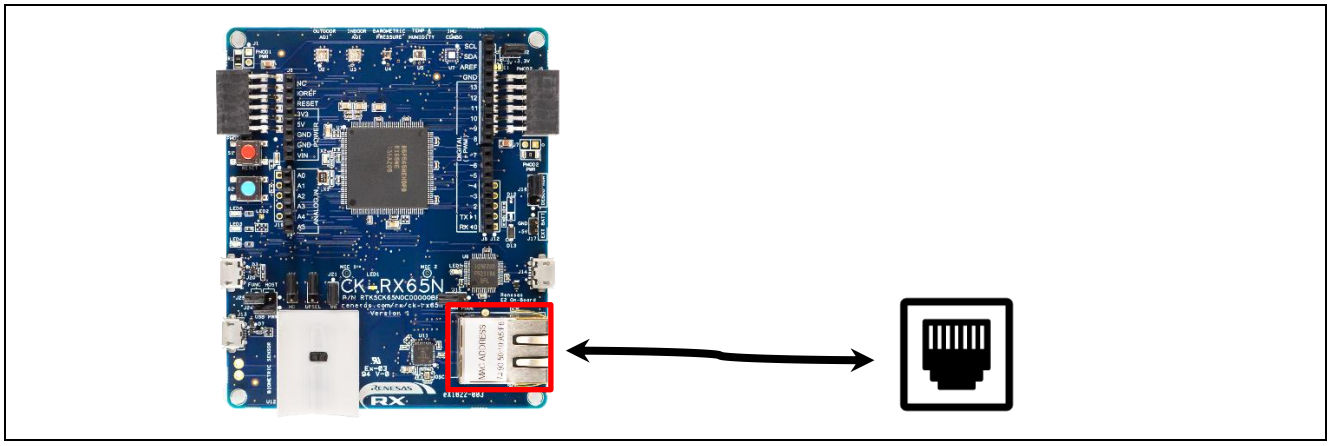


図 6-5 イーサネットによる有線インターネット接続

(5)　ジャンパ J16 を DEBUG 側に短絡する

CK-RX65N をデバッグモードにするためにジャンパ J16 を DEBUG 側(pin 1-2)に短絡します。

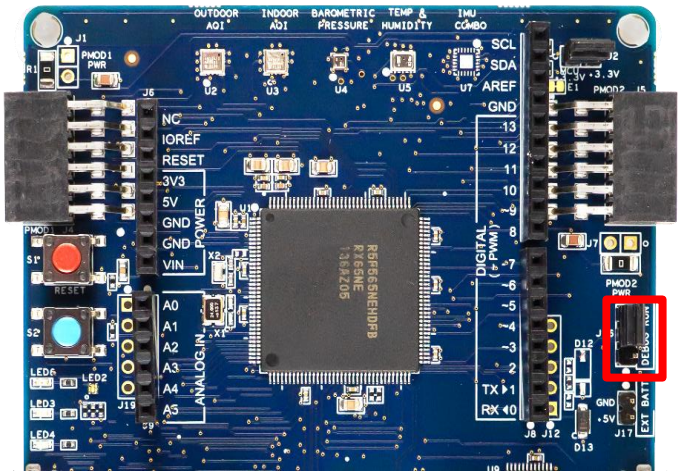


図 6-6 ジャンパ J16 の位置

6.1.2　TB-RX660 のセットアップ方法

TB-RX660 のセットアップ方法を示します。

(1)　事前準備

出荷状態のボードは、スルーホールにヘッダが取り付けられていないため、以下の事前準備をしてください。

- TB-RX660 ユーザーズマニュアルの「5.13 エミュレータリセットヘッダ」を参照し、ヘッダピンを付けてください。
- TB-RX660 ユーザーズマニュアルの「5.14 電源選択ヘッダ」を参照し、3.3V 電圧供給ができるようしてください。本デモでは RX660 を 3.3V で動作させます。
- TB-RX660 に CN2 コネクタを付けてください。

(2)　HS3001 ボードの接続

TB-RX660 の Pmod コネクタに HS3001 ボードを接続します。

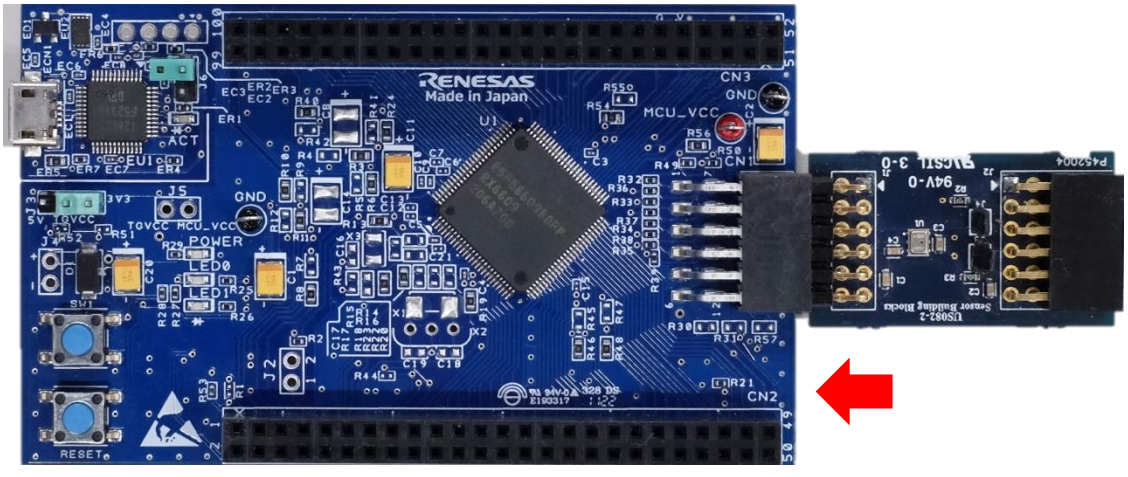


図 6-7　TB-RX660 の Pmod コネクタとセンサボードの接続

(3)　CK-RX65N との UART 通信用ケーブル接続

CK-RX65N と UART 通信を行うための TXD, RXD, GND は TB-RX660 の Pmod コネクタの以下の端子に割り当てています。6.1.1(1)に示す CK-RX65N の端子と表 6-1 のように UART 信号の対応を取って接続してください。

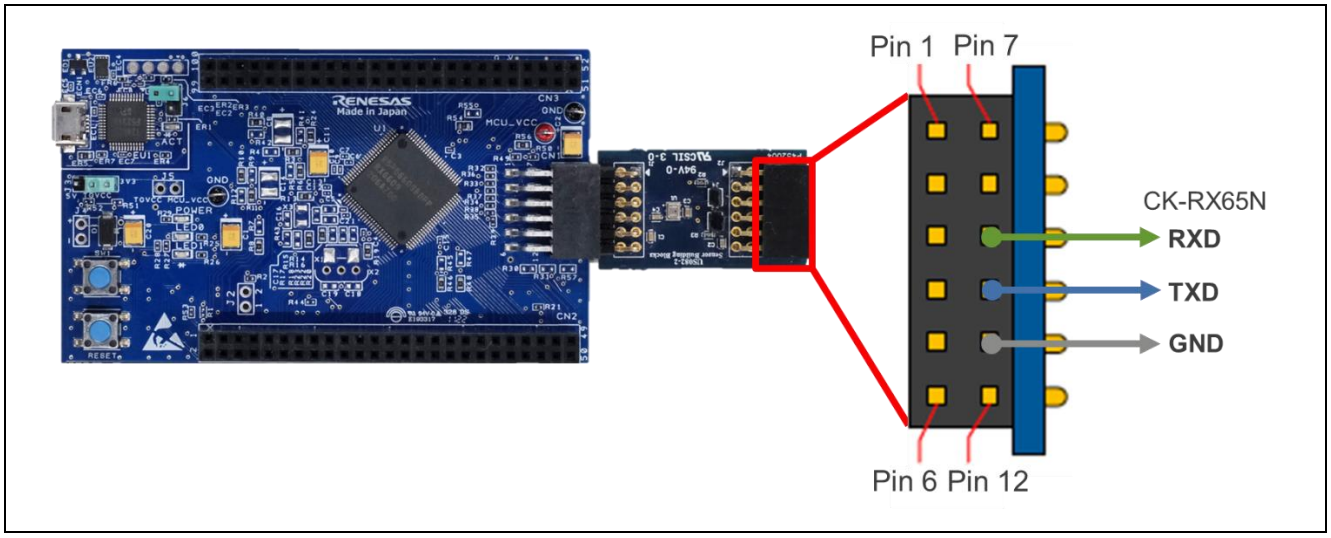


図 6-8　TB-RX660 のマイコン間 UART 通信に使用する端子位置

(4)　PC へのログ出力用シリアル通信のケーブル接続

PC へシリアル接続でログ出力するための TXD, RXD, GND は、TB-RX660 の MCU ヘッダ CN2 の以下の端子に割り当てています。Pmod USBUART コンバータの端子と以下の表および図のように UART 信号の対応を取って接続してください。

Pmod USBUART のジャンパ JP1 は VCC-LCL 側を短絡してください。これにより、micro USB 側からの電源により UART I/F 電圧を 3.3V にします。

さらに、PC と Pmod USBUART コンバータの micro USB Type-B コネクタを USB ケーブルで接続します。

表 6-2　TB-RX660 と Pmod USBUART コンバータの接続方法

TB-RX660 MCU Header CN2		Pmod USBUART コンバータ Pmod I/F
Pin 12: GND	⇔	Pin 5: GND
Pin 20: RXD1 (MCU P30)	⇔	Pin 3: TXD
Pin 22: TXD2 (MCU P26)	⇔	Pin 2: RXD

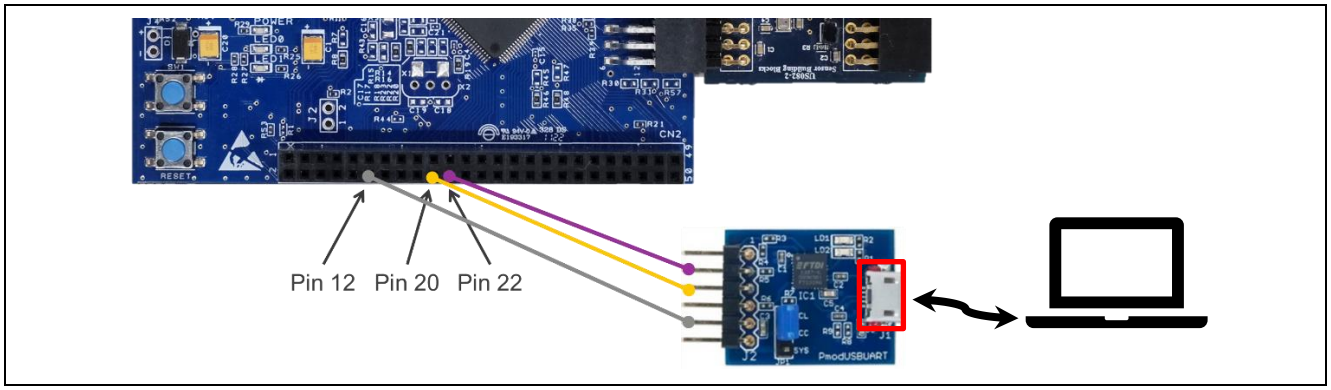


図 6-9　PC へのログ出力用シリアル通信のケーブルと TB-RX660 の接続方法

(5)　電源供給用ケーブル接続

PC と TB-RX660 の micro USB Type-B コネクタを USB ケーブルで接続します。

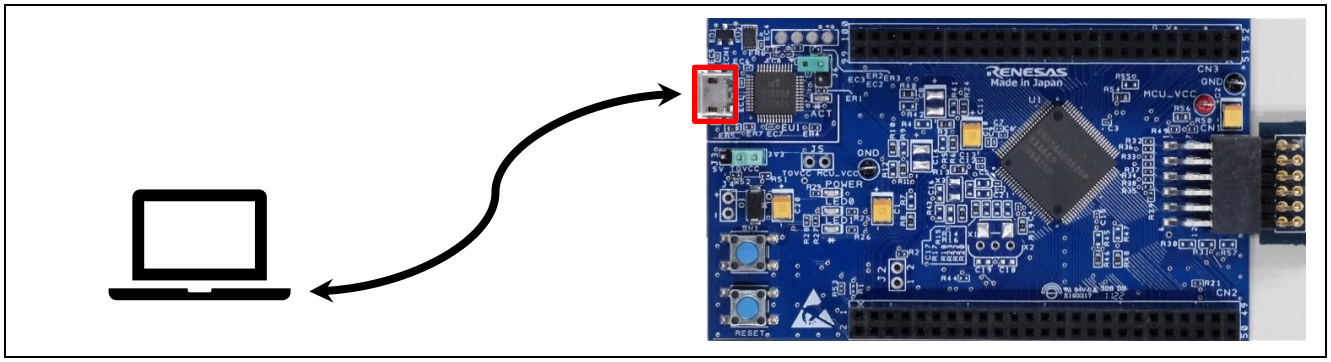


図 6-10　電源供給・エミュレータ用の USB 接続

(6)　エミュレータリセットヘッダ(J6)を開放する

TB-RX660 のエミュレータリセットヘッダ(J6)を開放します。

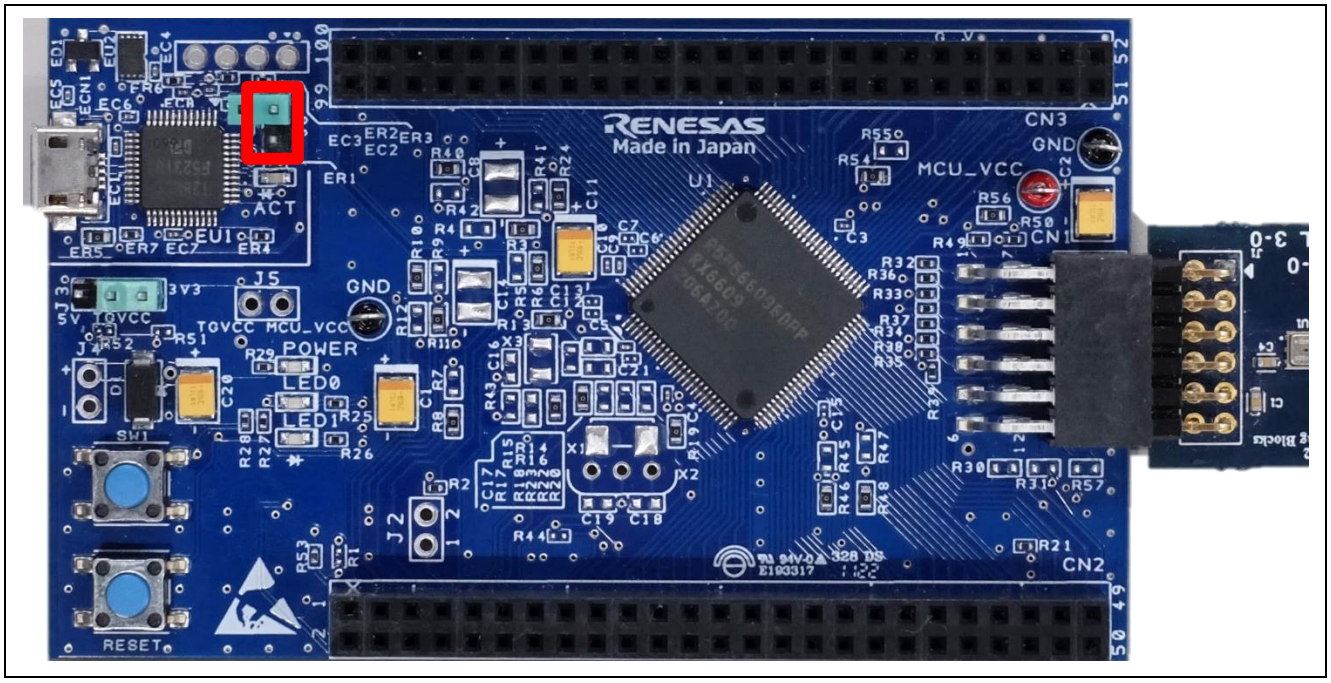


図 6-11　エミュレータリセットヘッダ(J6)の位置

以上でデモを実施するためのハードウェアのセットアップは完了です。図 6-12 にデモ構成の全体画像を示します。

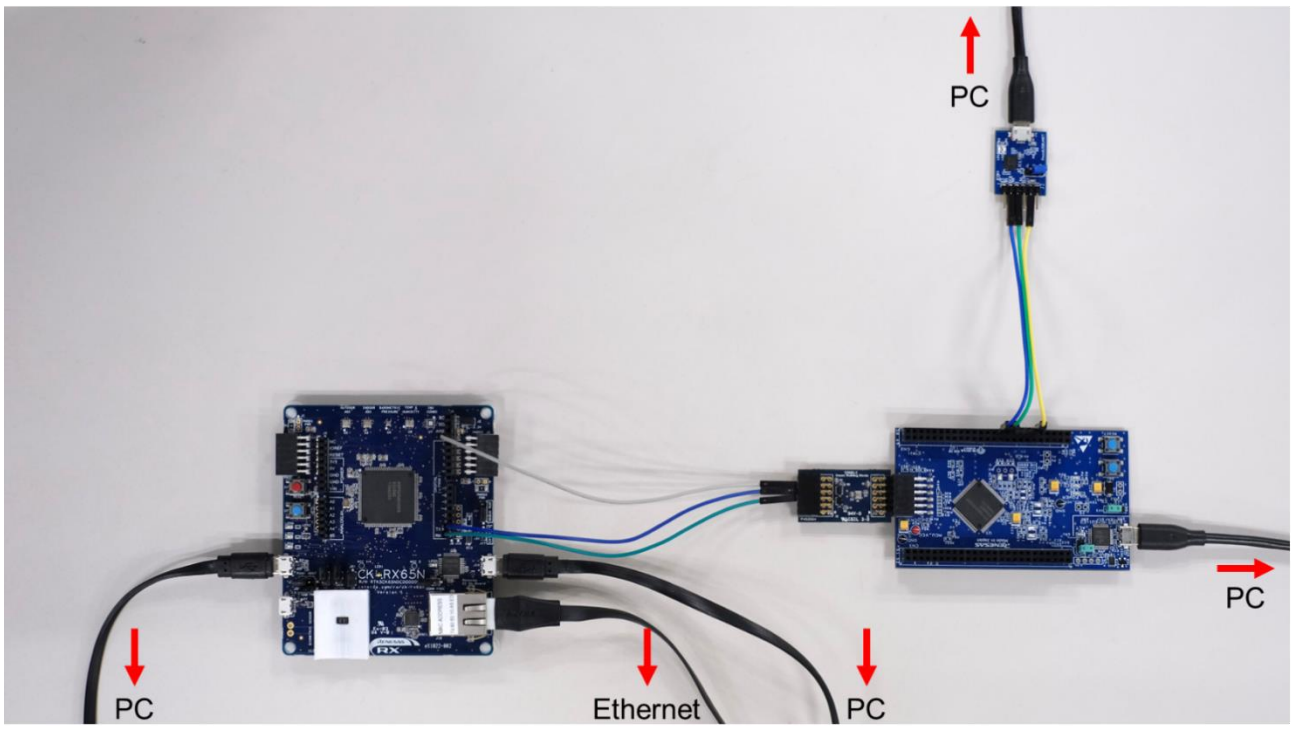


図 6-12　デモ構成の全体画像

6.2　ソフトウェアのセットアップ

6.2.1　事前準備

動作確認済みのそれぞれのソフトウェアのバージョンは表 2-4 をご参照ください。

(1)　QE for OTA のインストール

e² studio のメニューバーから、[Renesas Views] → [Renesas QE]を開き、QE for OTA がインストールされているか確認します。OTA Main (QE), OTA Manage IoT Device (QE)があればインストール済みです。

無い場合は「RX ファミリ AWS/Azure を利用したファームウェア更新ソフトの開発ガイド QE for OTA」の「2.1 QE for OTA のインストール」章を参照し、QE for OTA をインストールしてください。

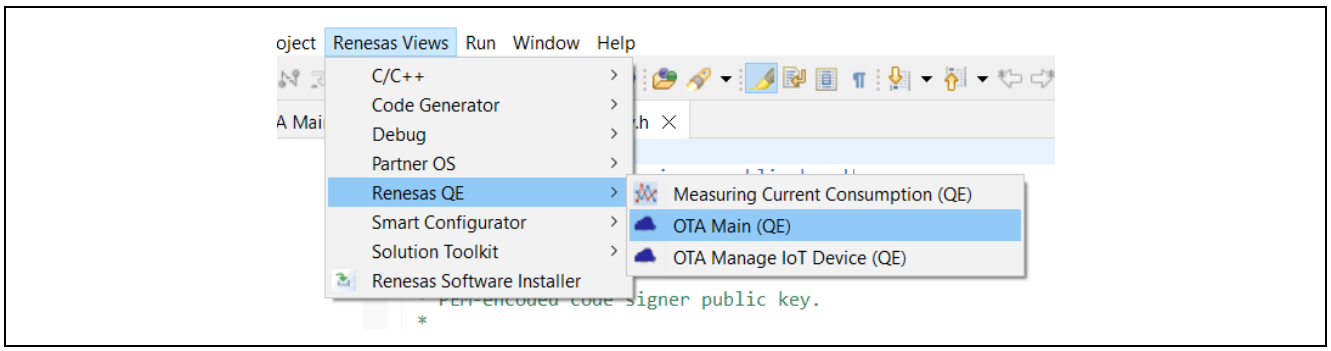


図 6-13　QE for OTA がインストールされているか確認

(2)　Python 実行環境のインストール

Python は、https://www.python.org/ から入手できます。

また、Python の pycryptodome ライブラリを使用します。Python をインストール後、以下の pip コマンドを実行し、インストールしてください。

```
> pip install pycryptodome
```

(3)　Renesas Flash Programmer のインストール

Renesas Flash Programmer は、Renesas Flash Programmer (Programming GUI) | Renesas から入手できます。

6.2.2　ターミナルソフトの設定

シリアル通信を使ったログ出力を利用時にターミナルソフト（例：Tera Term 等）が必要です。以下にシリアルポートの設定を示します。

表 6-3　シリアルポート設定

項目	設定
ボーレート	115,200 bps
データ	8-bit
パリティ	なし
ストップ	1-bit
フロー制御	なし

6.2.3　CK-RX65N 用の初期ファームウェアの作成と実行

QE for OTA を使って CK-RX65N 用の初期ファームウェアを作成し、e2 studio でデバッグ実行します。以下に手順を示します。

(1)　プロジェクトのインポート

CK-RX65N 用のブートローダである **ck_rx65n_demo_bootloader** プロジェクトと、ユーザプログラムである **ck_rx65n_2ndota_demo** プロジェクトを e2 studio にインポートします。

インポート時は、オプションの「Copy projects into workspace」のチェックを外してください。

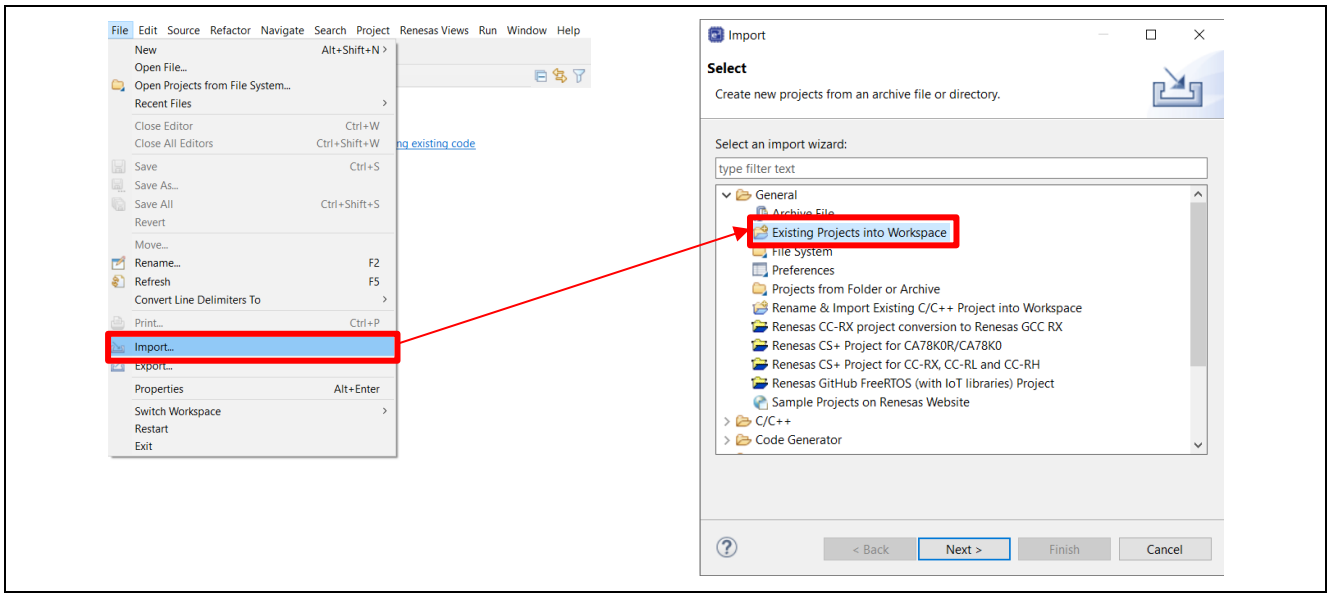


図 6-14　プロジェクトのインポート①

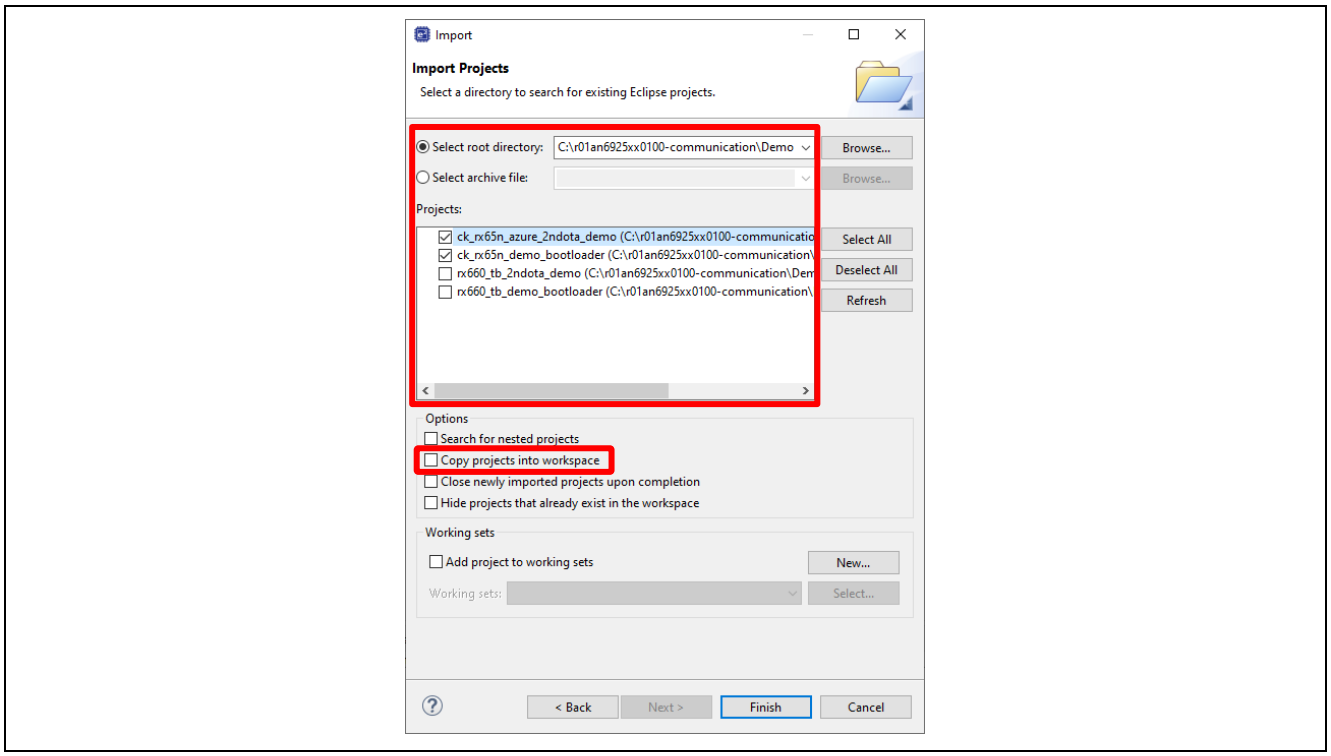


図 6-15　プロジェクトのインポート②

(2)　QE for OTA 画面を開く

e2 studio のメニューバーから、[Renesas Views] → [Renesas QE] → [OTA Main (QE)]を選択します。

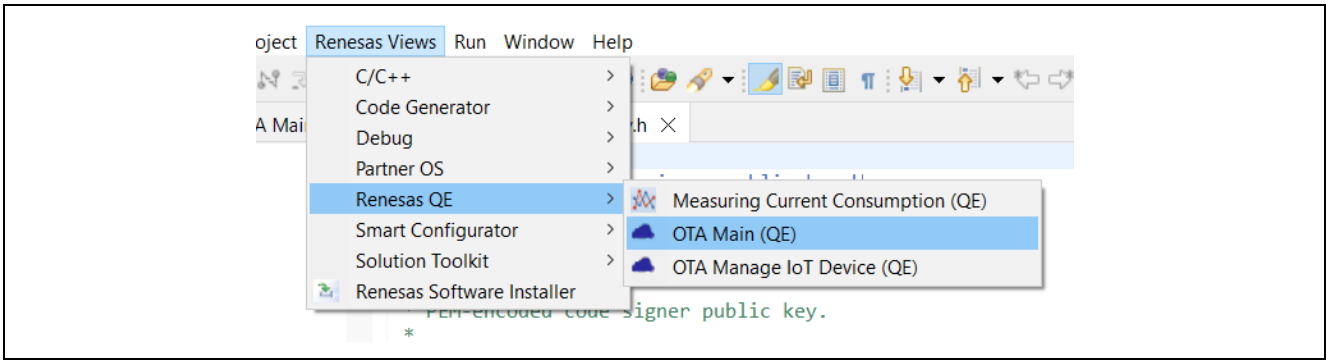


図 6-16 QE for OTA 画面を開く

(3)　[QE for OTA] 1. Cloud Settings – Sign-in to Cloud

以降は QE for OTA の GUI 画面に表示された手順通りに進めます。

まず、Cloud に「AWS」を選択しサインインします。ログイン時に選択したリージョンに AWS のリソースが生成されます。

図 6-17 QE for OTA で AWS にサインイン

(4)　[QE for OTA] 2. Prepare projects – Select projects

先ほど e2 studio にインポートした ck_rx65n_demo_bootloader プロジェクトと ck_rx65n_2ndota_demo プロジェクトを選択します。

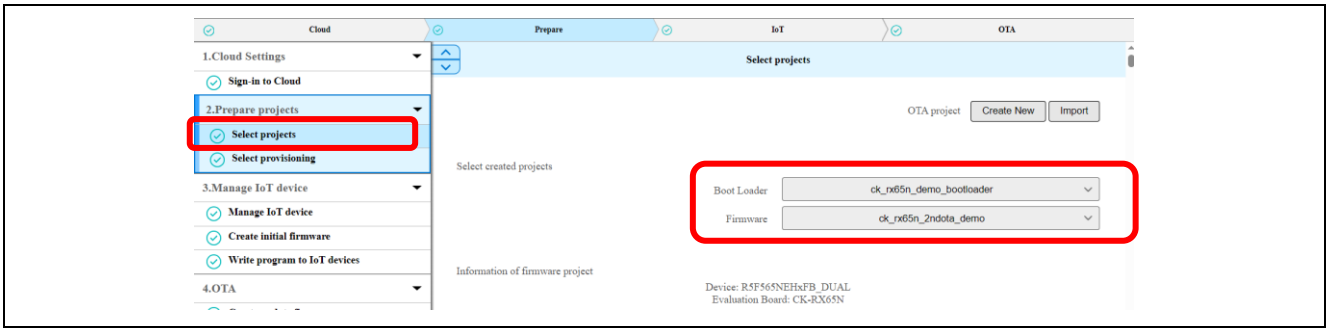


図 6-18 プロジェクトの選択

(5)　[QE for OTA] 2. Prepare projects – Select provisioning

プロビジョニング方法として、「Source code includes credentials (asymmetric keys)」を選択します。

図 6-19 プロビジョニング方法の選択

(6)　[QE for OTA] 3. Manage IoT device – Manage IoT device

QE for OTA の手順に従い IoT デバイスを作成します。

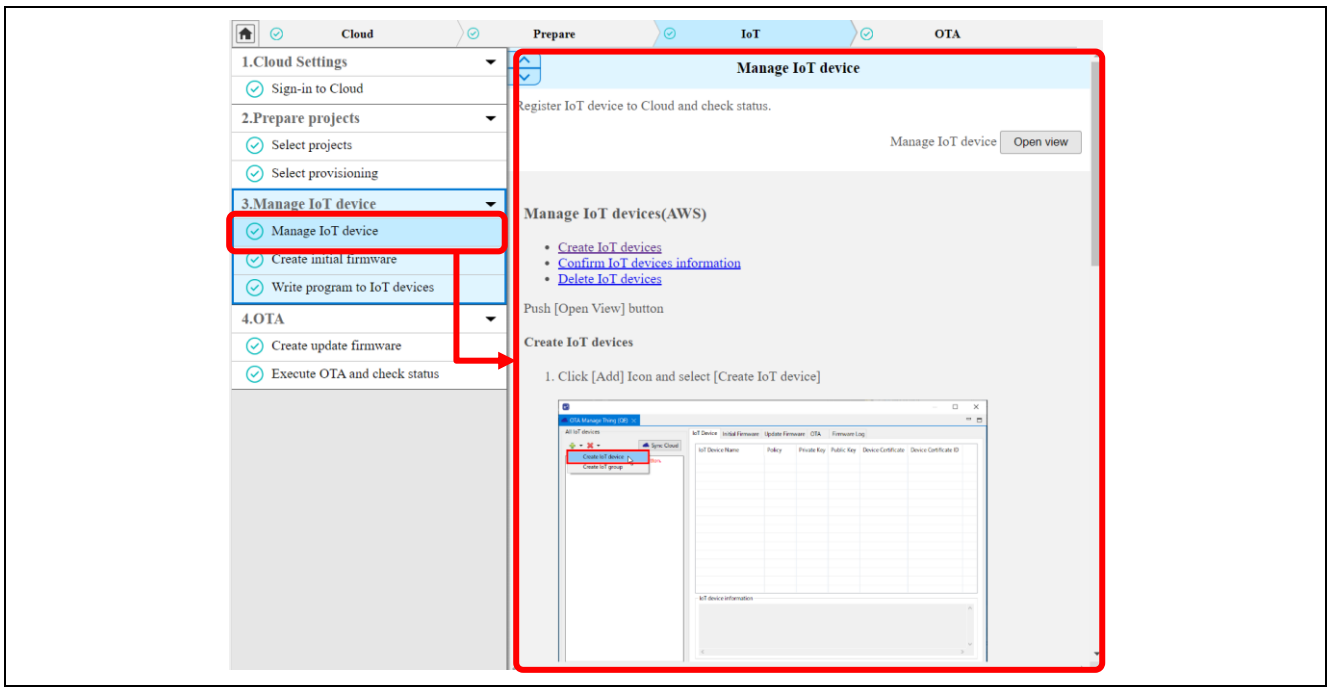


図 6-20 IoT デバイスの作成

(7)　[QE for OTA] 3. Manage IoT device – Create initial firmware

QE for OTA の手順に従い初期ファームウェアを作成します。

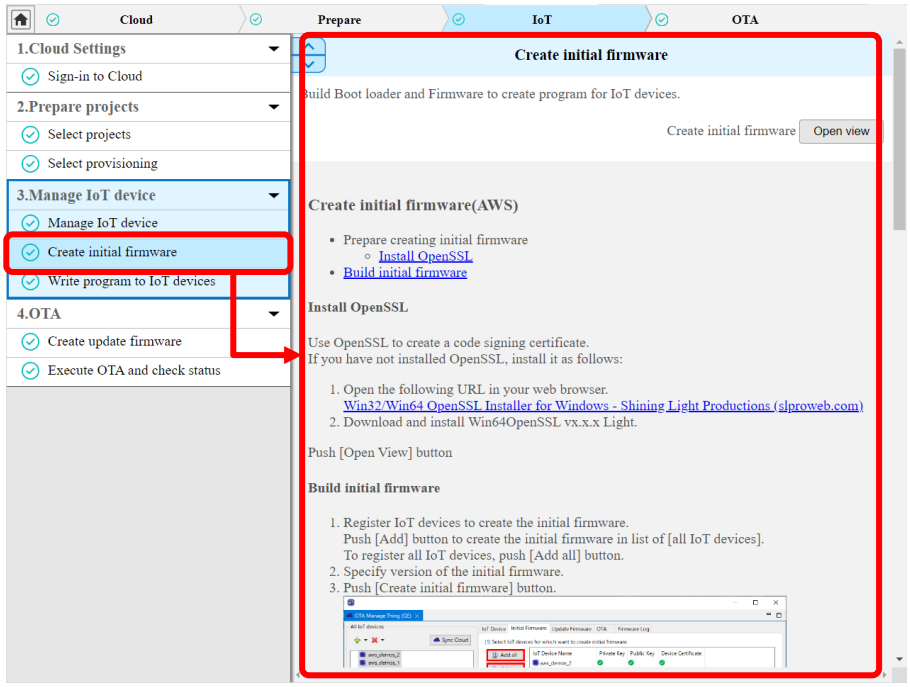


図 6-21 初期ファームウェアの作成

(8)　[QE for OTA] 3. Manage IoT device – Write program to IoT devices

QE for OTA の手順に従い CK-RX65N に初期ファームウェアを書き込みます。

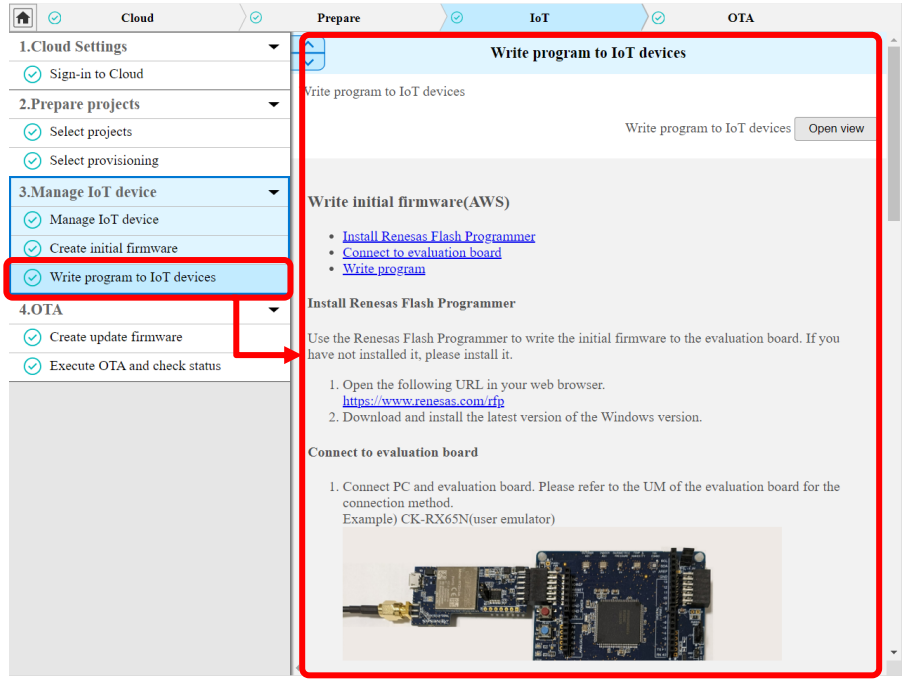


図 6-22 初期ファームウェアの書き込み

(9)　動作確認

図 6-6 のジャンパ J16 を RUN 側(pin 2-3)に短絡します。

ターミナルソフトを起動し、図 6-23 のようにログが出力されていれば CK-RX65N の準備は完了です。

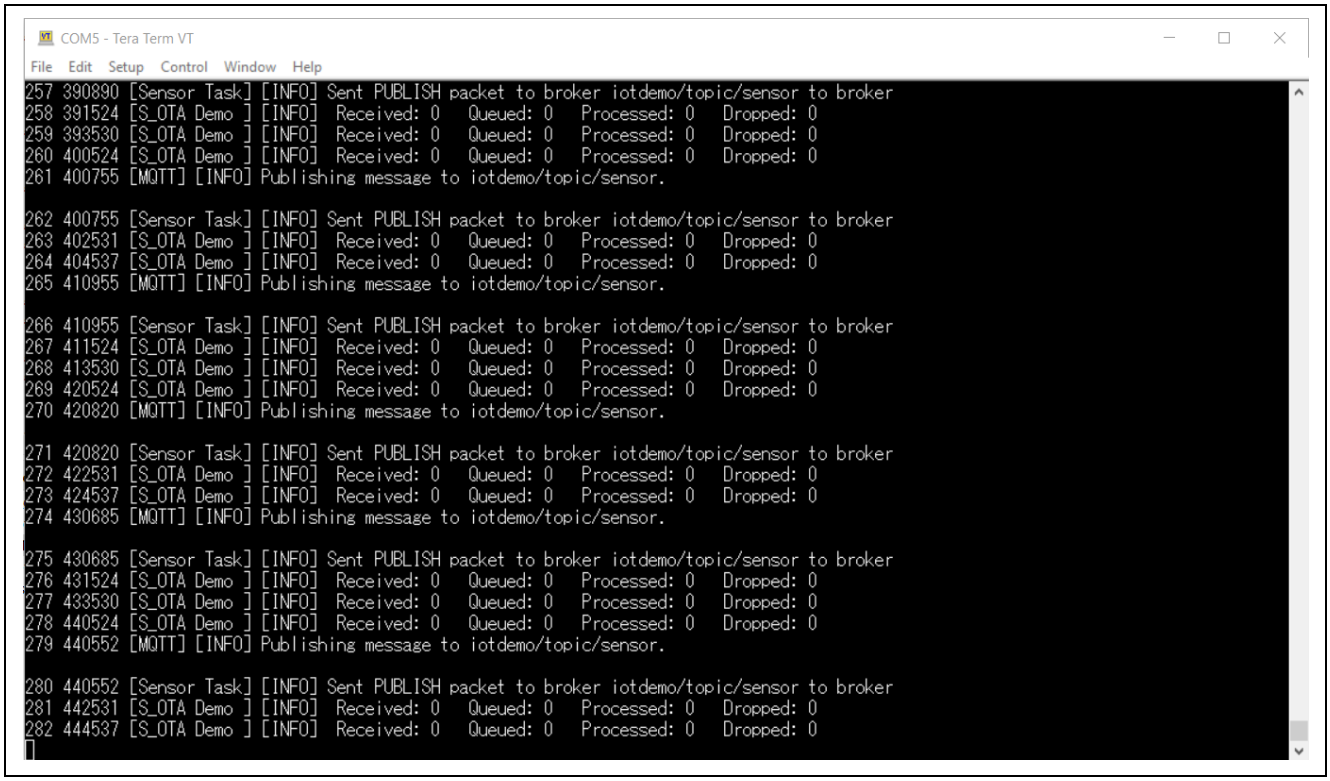


図 6-23 CK-RX65N のログ画面

6.2.4　TB-RX660 用の初期ファームウェアの作成と実行

TB-RX660 用の初期ファームウェアを作成し、Renesas Flash Programmer でマイコンに書き込みます。以下に手順を示します。

(1)　プロジェクトのインポート

先ほどの CK-RX65N 用のプロジェクトと同様に、本アプリケーションノートでサンプルコードとして提供している、TB-RX660 用のブートローダである **rx660_tb_demo_bootloader プロジェクト**と、ユーザプログラムである **rx660_tb_2ndota_demo プロジェクト**を e² studio にインポートします。

(2)　プロジェクトのビルド

rx660_tb_demo_bootloader プロジェクトと rx660_tb_2ndota_demo プロジェクトをビルドし、MOT ファイルを作成します。MOT ファイルはプロジェクトフォルダ直下の HardwareDebug フォルダに作成されます。

(3)　初期ファームウェアの作成

作成された rx660_tb_demo_bootloader と rx660_tb_2ndota_demo の MOT ファイルを結合して TB-RX660 用の初期ファームウェアを作成します。MOT ファイルの結合には Renesas Image Generator を使用します。Renesas Image Generator は「RX ファミリ ファームウェアアップデート モジュール Firmware Integration Technology アプリケーションノート Rev.2.01」に付属するツールです。詳細については、上記リンク先のアプリケーションノートの「Renesas Image Generator」章をご参照ください。

rx660_tb_2ndota_demo\RenesasImageGenerator フォルダで以下のコマンドを実行し、初期ファームウェア initial_firm.mot を作成します。RenesasImageGenerator フォルダの howtouse.txt ファイルにコマンド文があります。

```
> python .\image-gen.py -iup ..\HardwareDebug\rx660_tb_2ndota_demo.mot -
ibp ..\..\rx660_tb_demo_bootloader\HardwareDebug\rx660_tb_demo_bootloader.mot -
o .\initial_firm -key .\keys\secp256r1.privatekey -
ip .\RX660_Linear_Half_ImageGenerator_PRM.csv -vt ecdsa
```

(4)　初期ファームウェアの書き込み

Renesas Flash Programmer を用いて、上で作成した初期ファームウェア initial_firm.mot を TB-RX660 に書き込みます。

rx660_tb_2ndota_demo\rfp フォルダ内の Renesas Flash Programmer プロジェクト rx660_program.rpj を開き、Program File に先ほど作成した RX660 用の初期ファームウェア initial_firm.mot を指定して、Start をクリックしてください。

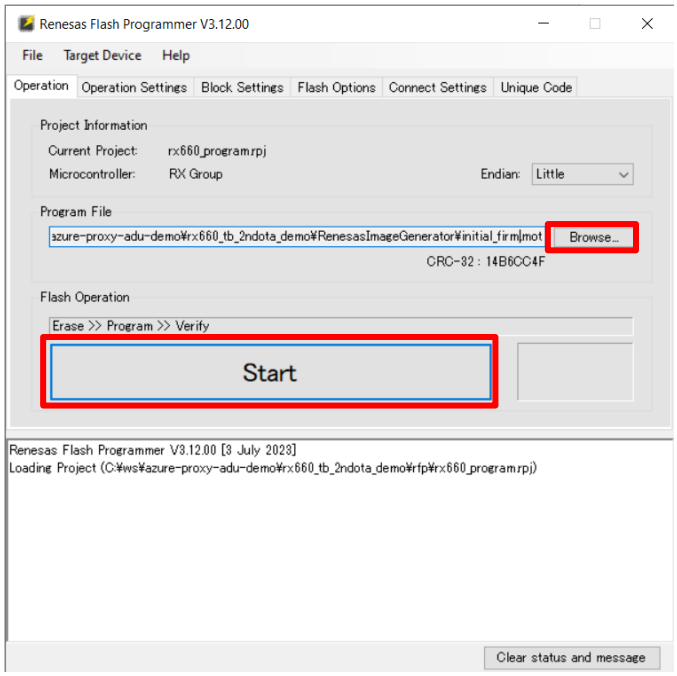


図 6-24　Renesas Flash Programmer での初期ファームウェアの書き込み

(5)　動作確認

TB-RX660 のエミュレータリセットヘッダ(J6)を短絡します。

ターミナルソフトを起動し、図 6-25 のように湿度の値がログ出力されていれば TB-RX660 の準備は完了です。

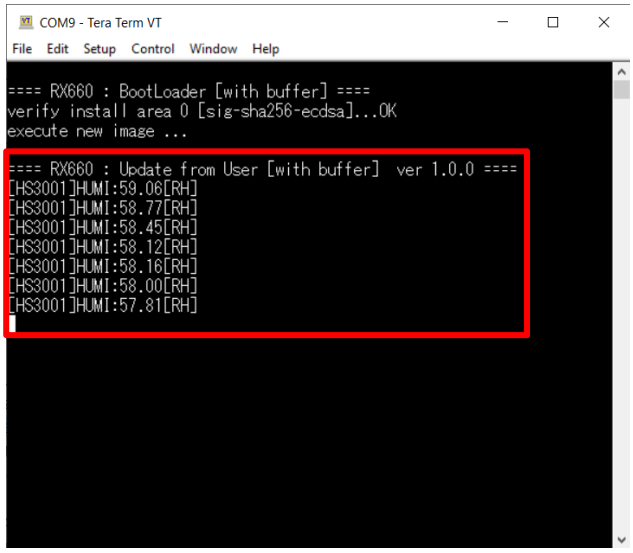


図 6-25　TB-RX660 のログ画面

6.3　AWS クラウドの準備

AWS マネジメントコンソールにログインします。

AWS マネジメントコンソール | AWS (amazon.com)

マネジメントコンソール画面の右上に表示されているリージョンを確認し、QE for OTA ログイン時の設定と同じリージョンを選択します。

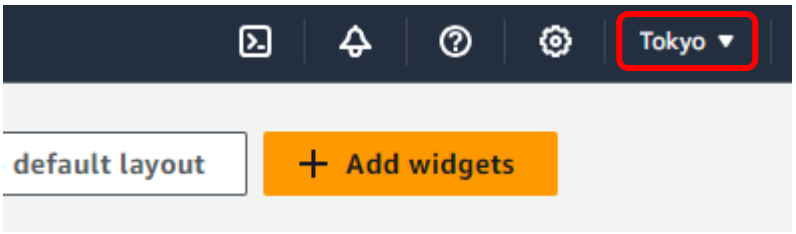


図 6-26 リージョンの確認

6.3.1　OTA アップデートのための設定

「RX ファミリ RX65N における Amazon Web Services を利用した FreeRTOS OTA の実現方法 (v202210.01-LTS-rx-1.1.0 以降対応版)」を参照し、設定を行います。

(1)　上記アプリケーションノートの「3.4 Amazon S3 バケットの作成」の手順で Amazon S3 バケットを作成します。ここで設定したバケット名はデモ実行時に使用します。

(2)　上記アプリケーションノートの「3.5 IAM ユーザーに OTA の実行権限を割り当てる」の手順でサービスロールを作成します。ここで設定したサービスロール名はデモ実行時に使用します。

(3)　上記アプリケーションノートの「5.2 ファームウェアの更新 (5)~(9)」の手順でコード署名証明書を登録します。ここで、登録するコード署名証明書は 6.2.3(7)章で CK-RX65N 用の初期ファームウェアを QE for OTA で作成したときに作成した証明書を使用します。

証明書は ck_rx65n_demo_bootloader/QE-OTA/codesigning に作成されています。証明書は secp256r1.crt, 秘密鍵は secp256r1.privateKey, 証明書チェーンは ca.crt を指定します。

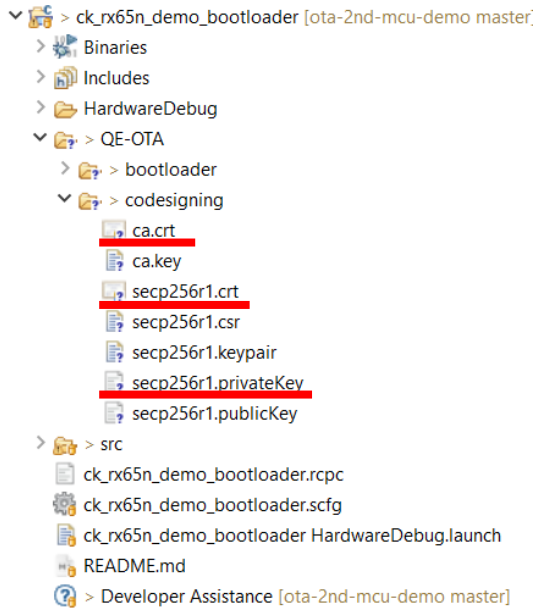


図 6-27 コード署名証明書の作成場所

6.3.2　センサデータ可視化のための設定

受信したセンサデータをグラフ形式で可視化するために、Amazon CloudWatch および AWS IoT Core で以下の手順で設定を行います。

Note　グラフ形式での可視化は必要なく、データが AWS に届いていることをブラウザ上で確認できればいいという場合は 6.3.2 章の作業は丸ごと省略可能です。
この場合、図 6-28 のように AWS IoT の「MQTT テストクライアント」で「iotdemo/topic/sensor」をサブスクライブすることでセンサデータが期待通り受信できていることをテキスト形式で確認できます。

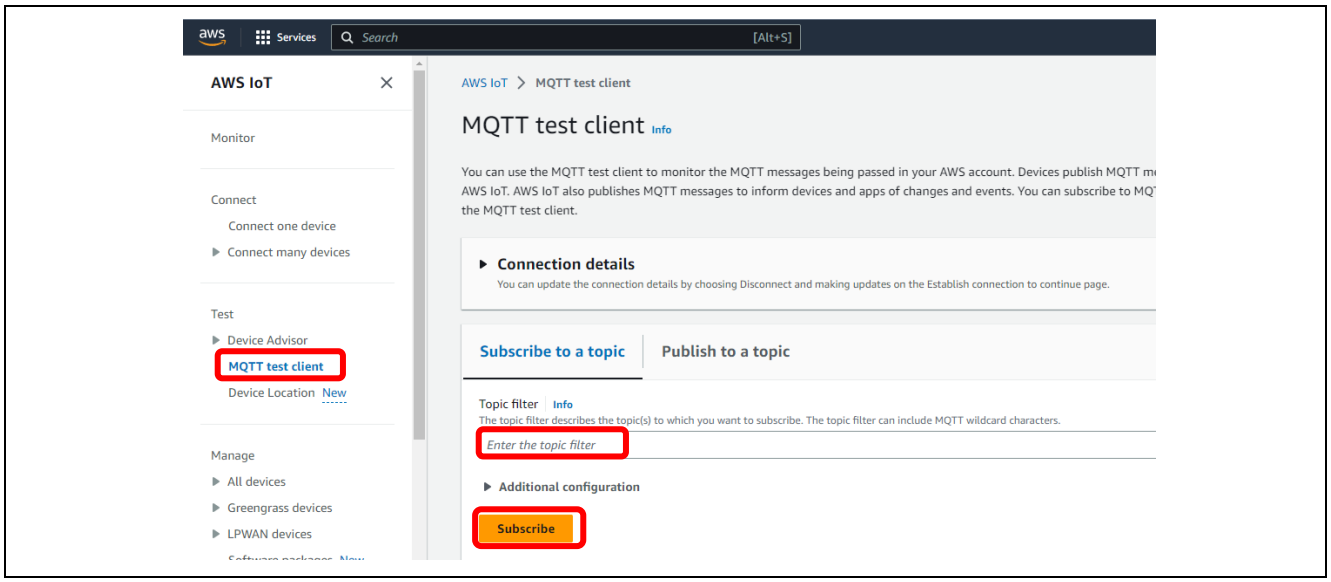
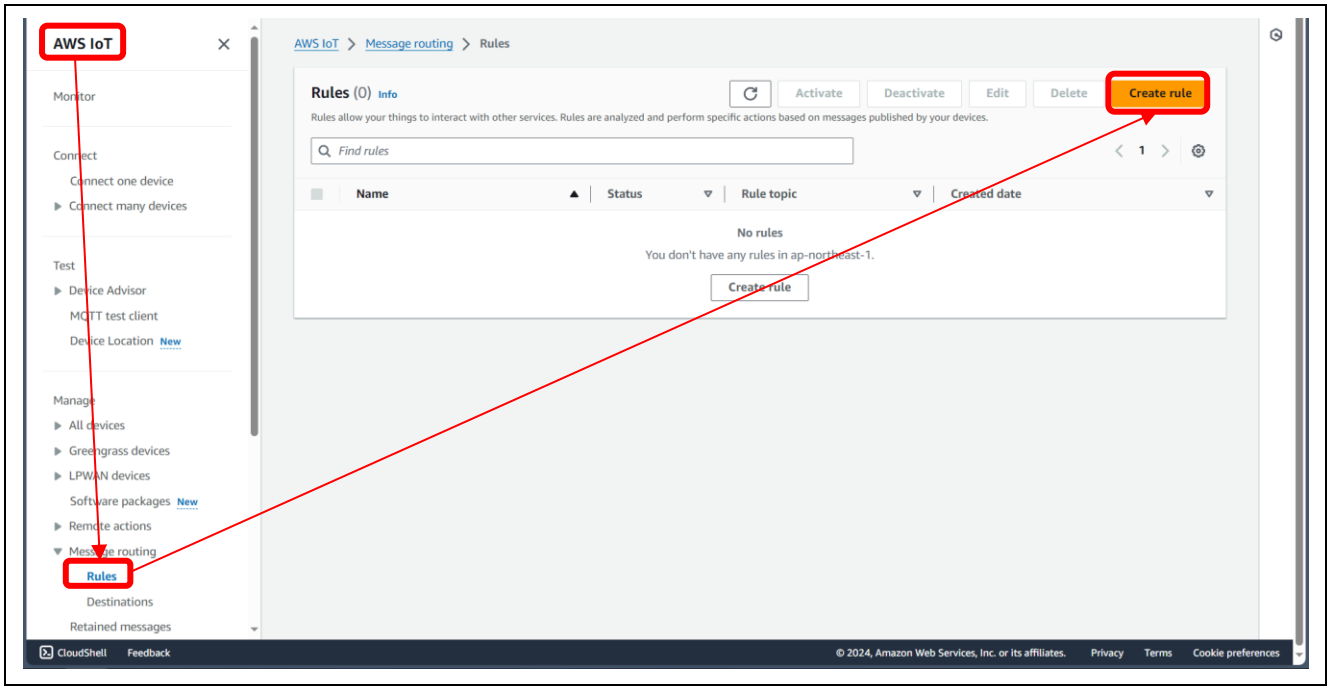


図 6-28 MQTT テストクライアントでのデータ受信確認

6.3.2.1　Amazon CloudWatch の設定

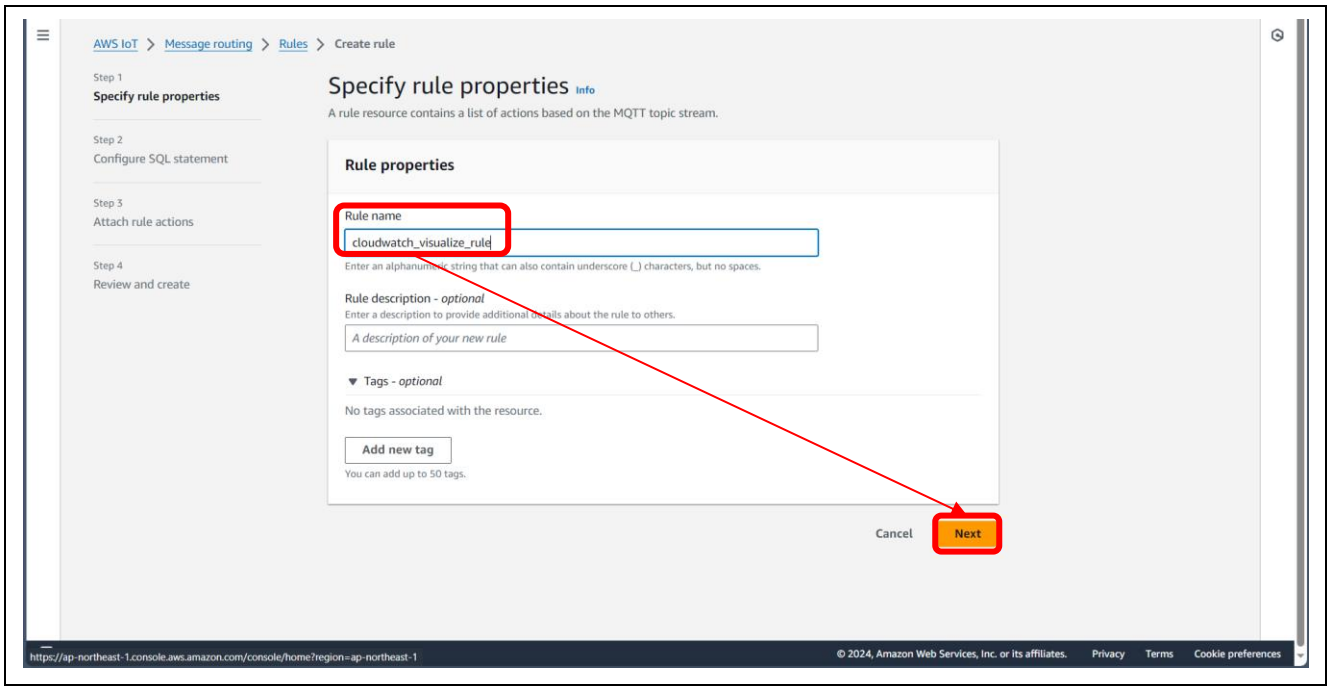
(1)　AWS IoT でルールを作成

[AWS IoT] ⇒ [ルール] ⇒ [ルールを作成]をクリックします。



(2)　ルールのプロパティを指定

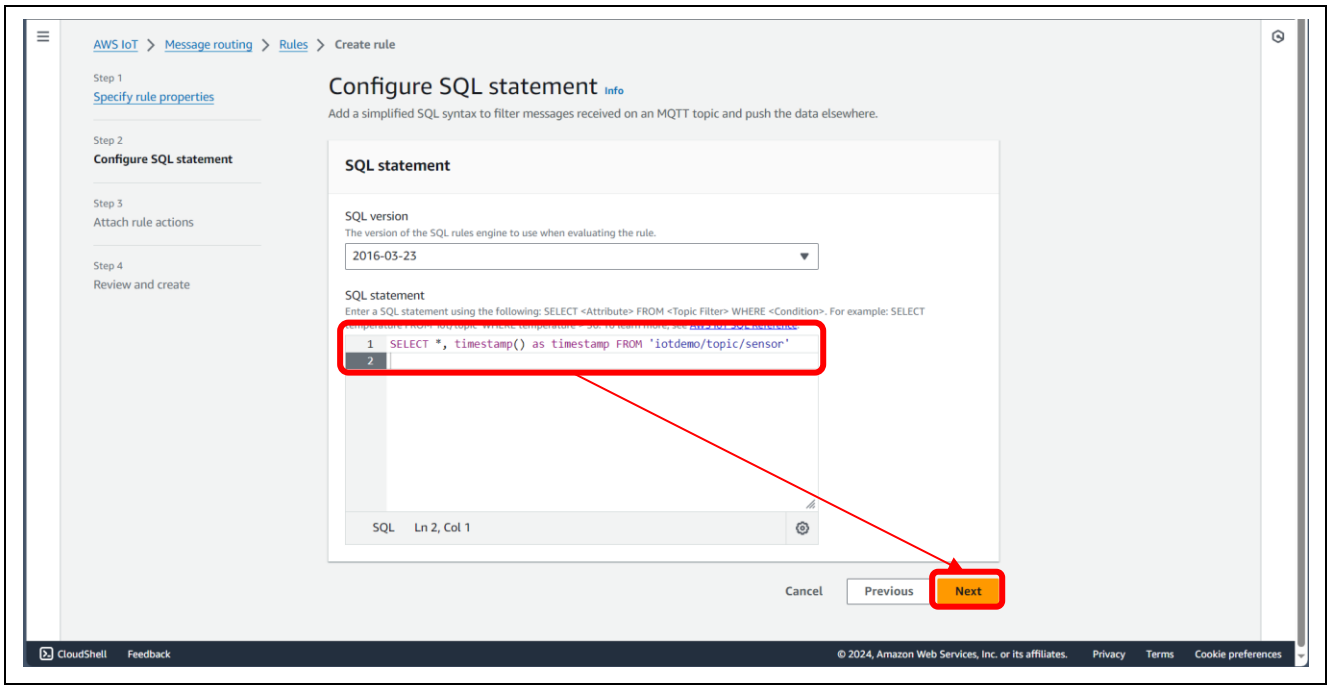
ルール名を入力し、[次へ]をクリックします。



(3)　SQL ステートメントを設定

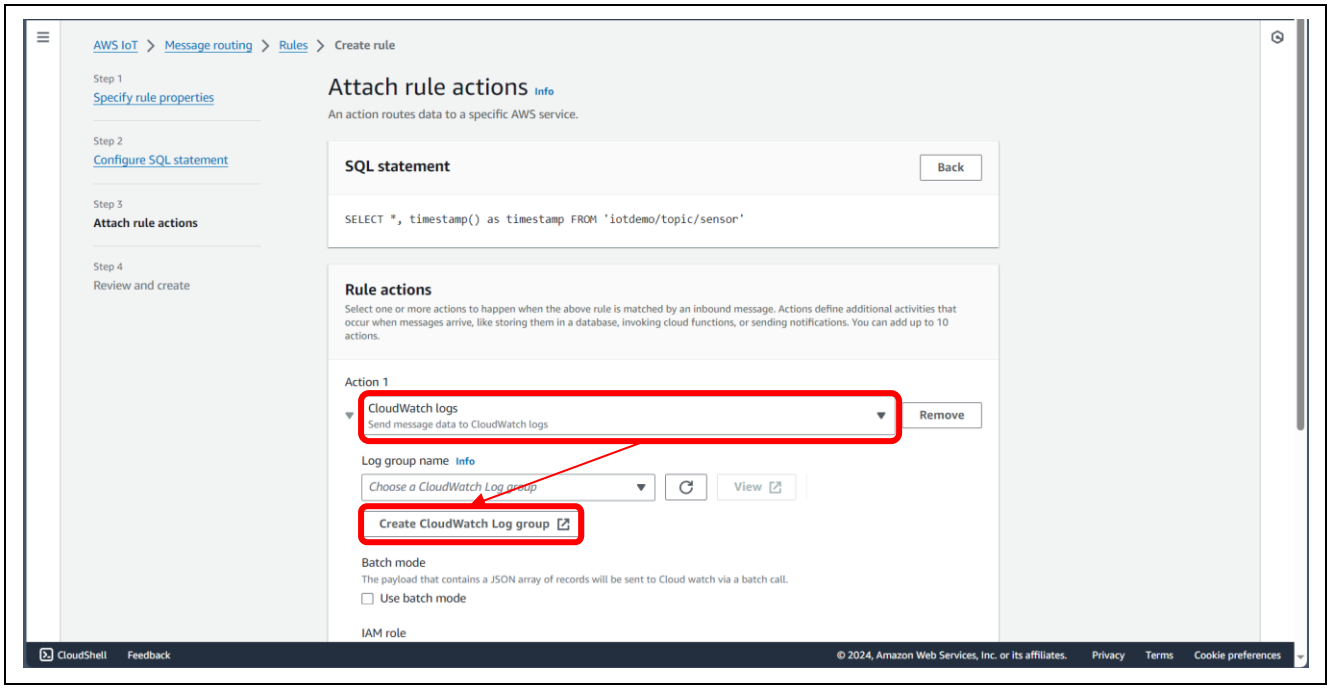
SQL ステートメントを入力します。テキストエディタに以下のように入力します。最終行に改行が必要な点に注意してください。

```
SELECT *, timestamp() as timestamp FROM 'iotdemo/topic/sensor'
(改行が必要)
```



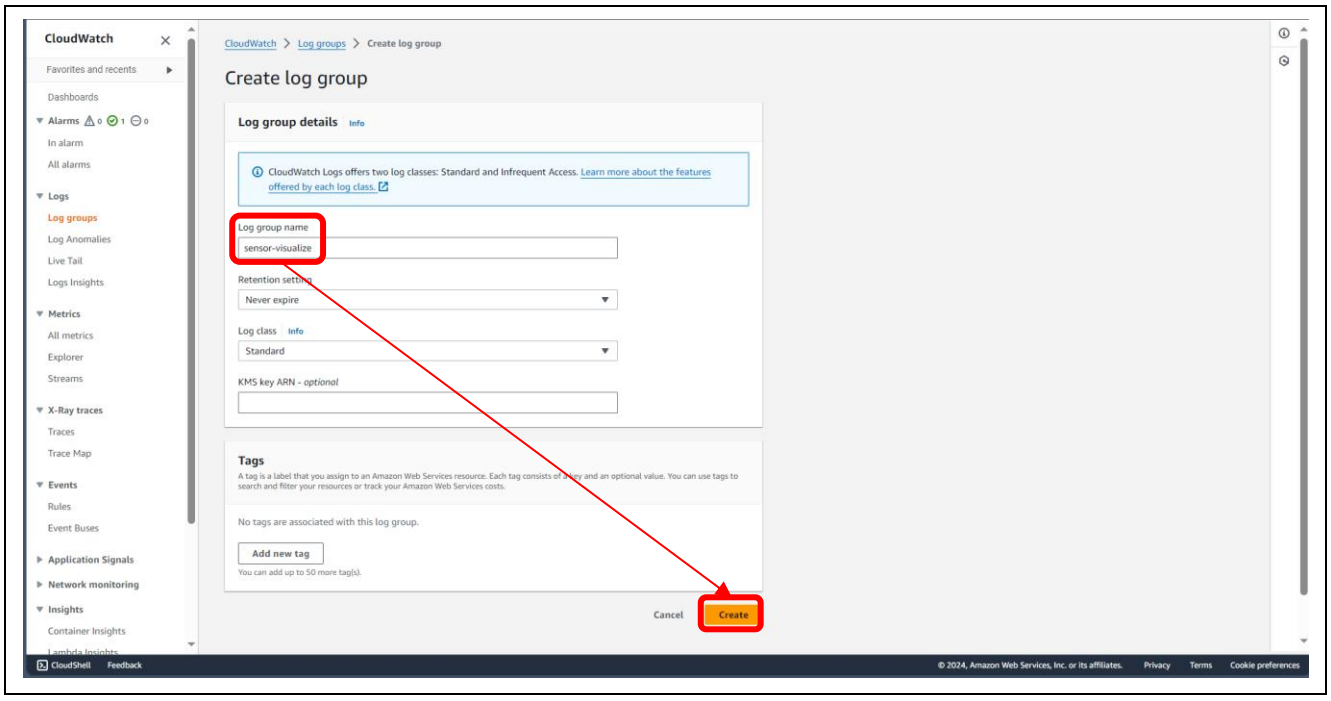
(4)　「ルールアクションをアタッチ」ステップでルールアクションを選択

アクション 1 に「CloudWatch logs」を選択し、[CloudWatch Log グループを作成]をクリックします。



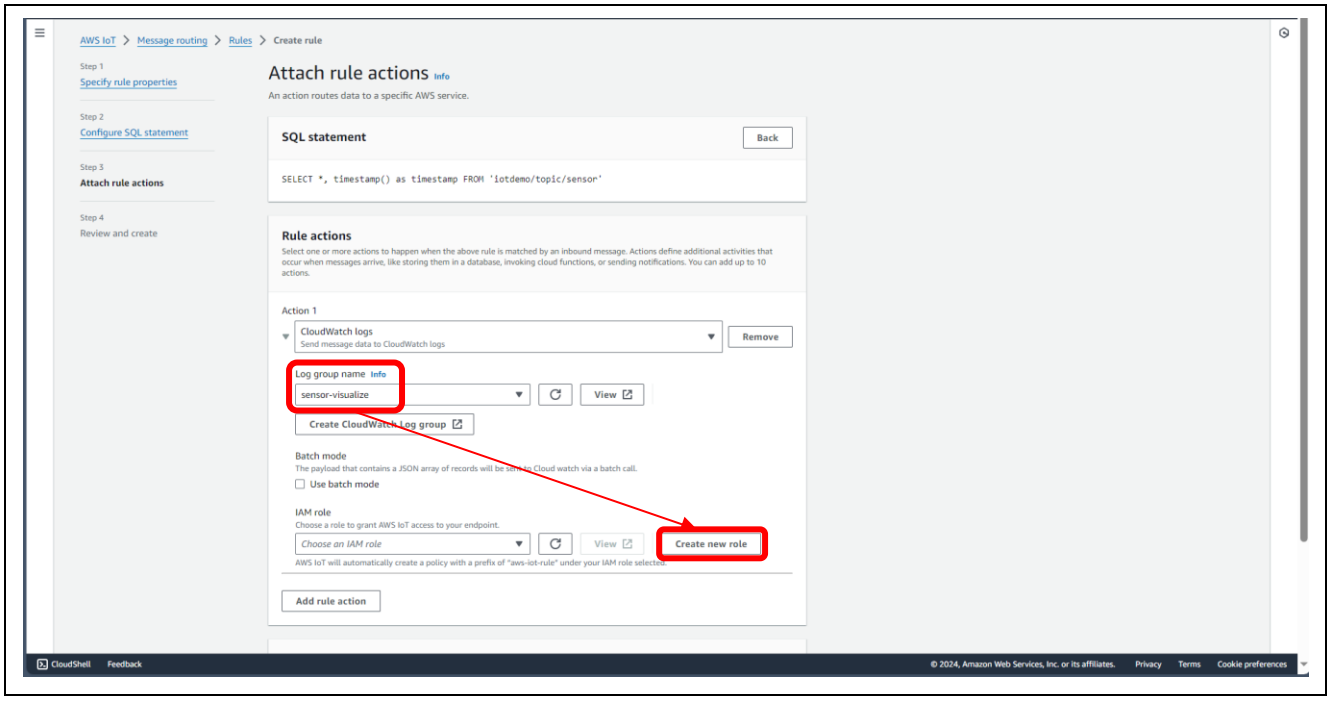
(5)　ロググループを作成

ロググループ名を入力し、[作成]をクリックします。



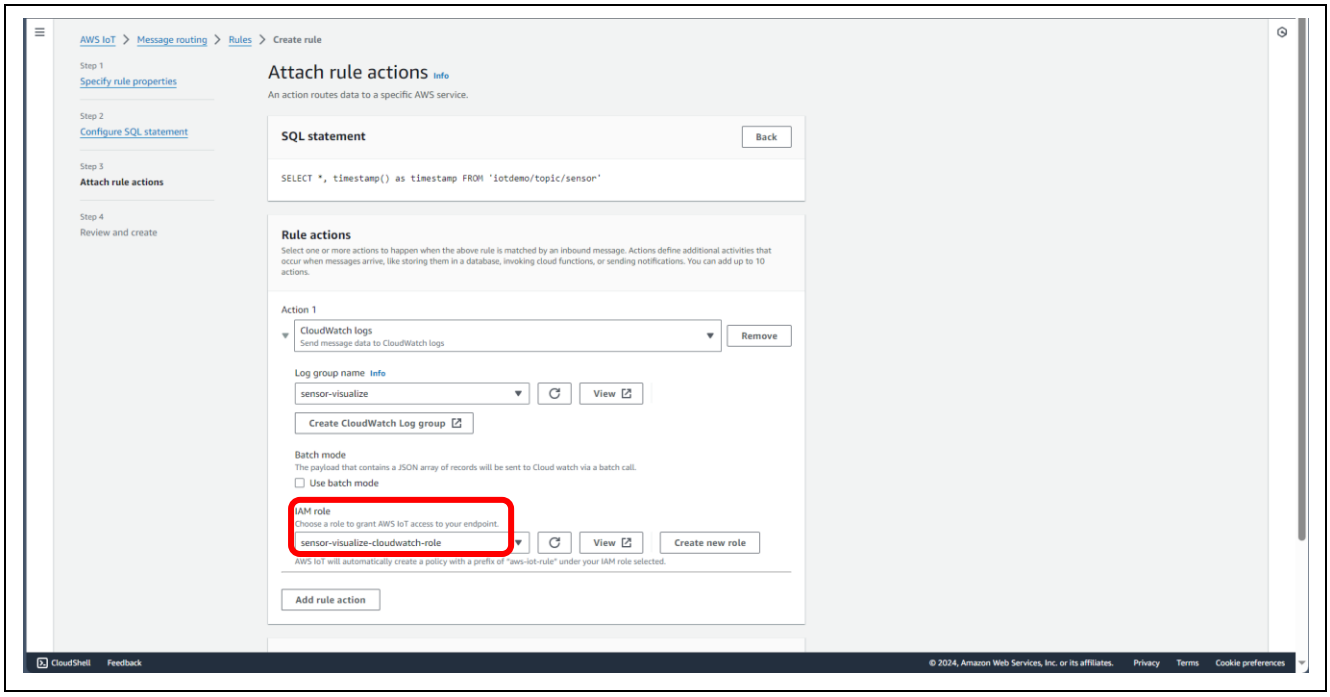
(6)　新しいロールを作成

ログのグループ名に、作成したロググループを選択し、[新しいロールを作成]をクリックします。



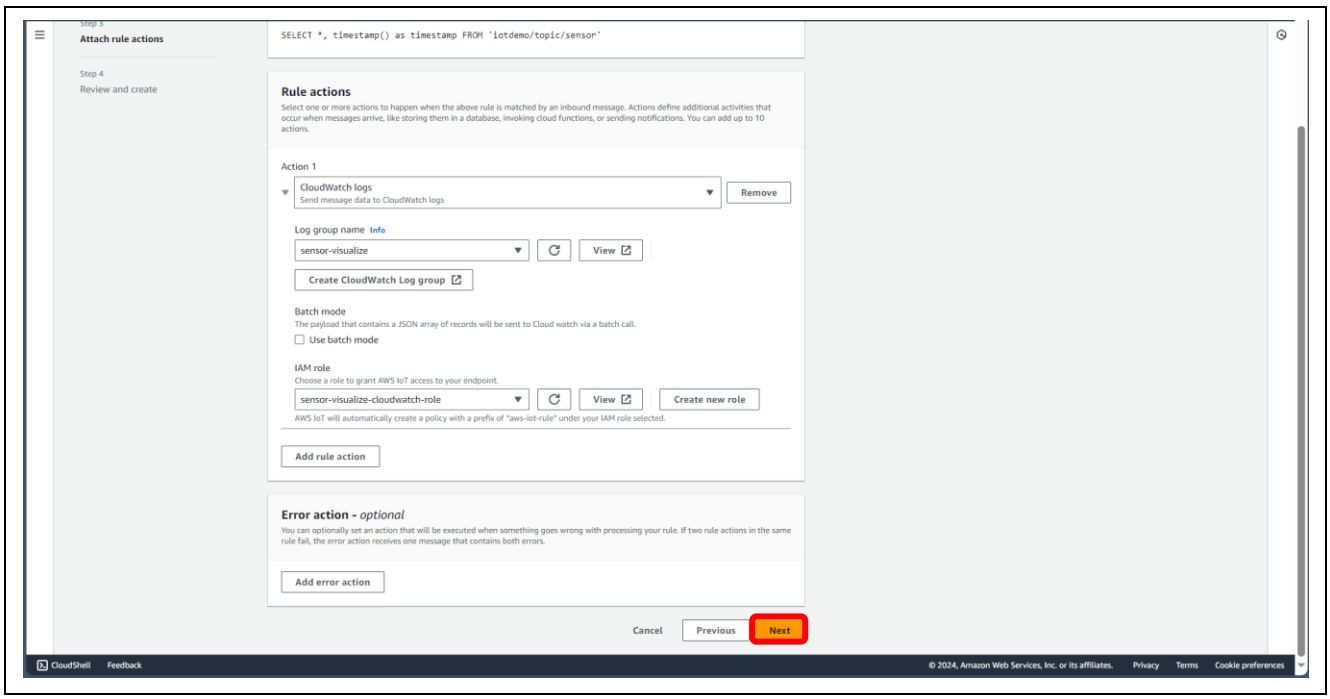
(7)　作成した IAM ロールを選択

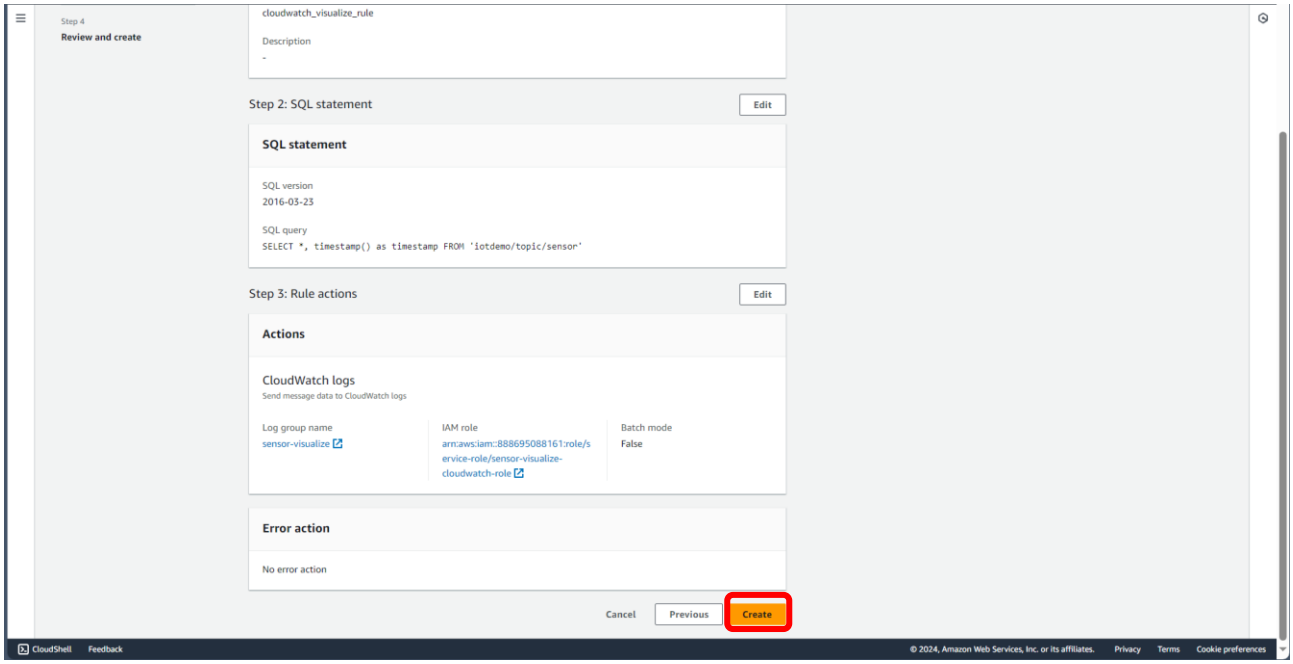
IAM ロールに、作成したロールを選択します。

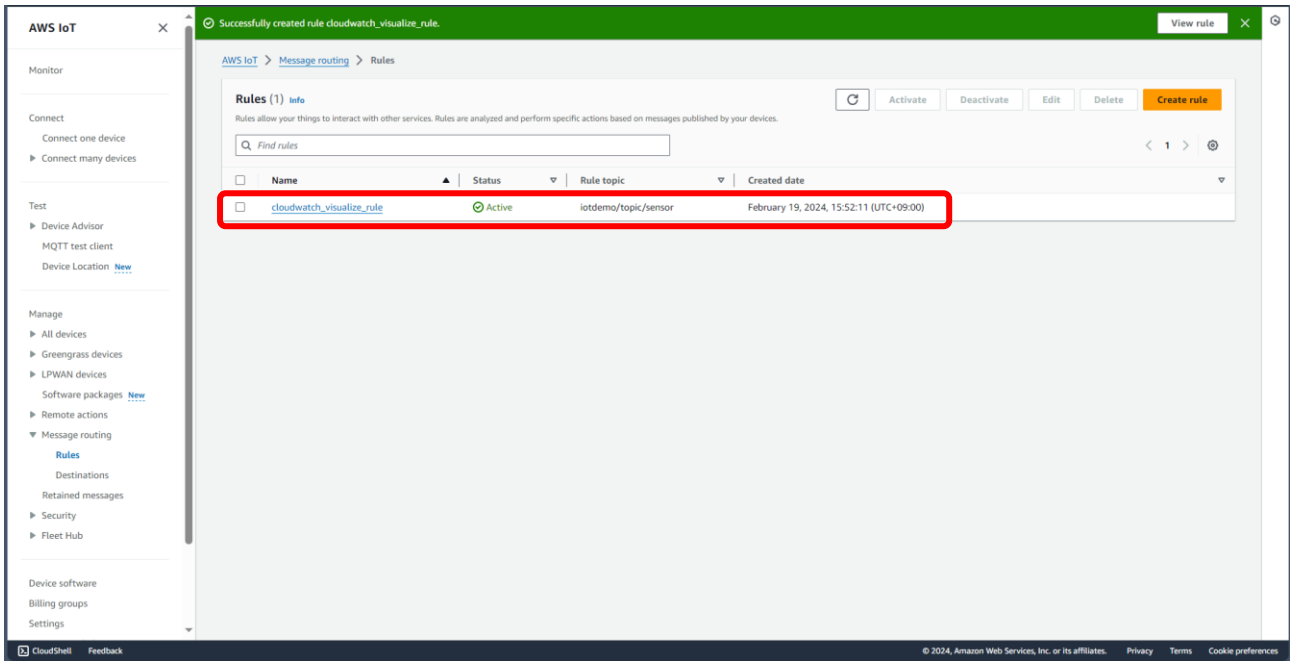


(8)　ルール作成に成功したことを確認

[次へ] ⇒ [作成]と進み、ルールの一覧に、作成したルールが表示されていることを確認します。







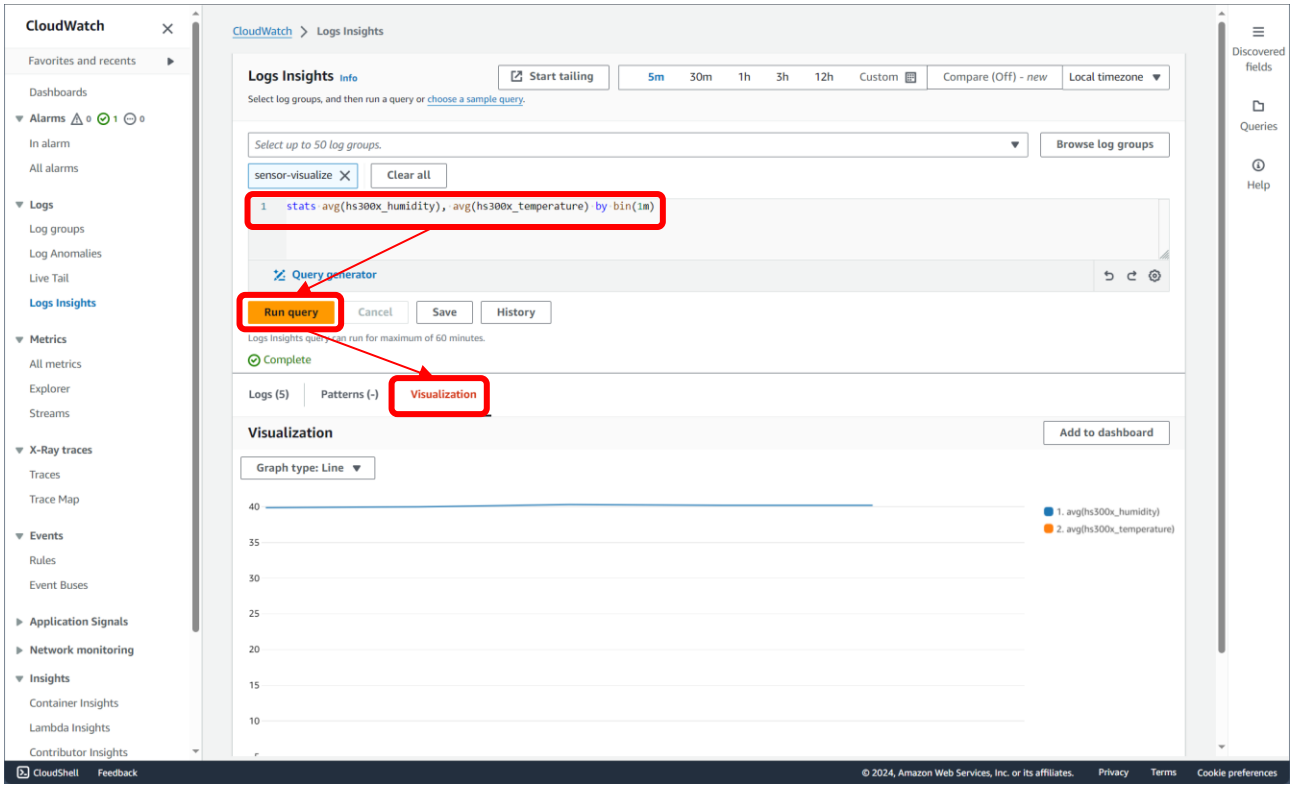
(9)　CloudWatch でグラフ表示を確認

CloudWatch ページを表示し、左メニューの「ログのインサイト」をクリックします。

ロググループに、先ほど 6.3.2.1(5)で作成したグループを選択し、以下のクエリを入力し、[クエリの実行]をクリックします。

```
stats avg(hs300x_humidity), avg(hs300x_temperature) by bin(1m)
```

可視化タブにグラフが表示されます。



7.　デモの実行手順

デモを実行する手順について説明します。

7.1　初期状態の動作確認

6 章のデモのセットアップが完了している状態で、TB-RX660 のリセットスイッチ(RESET)を押下しハードウェアリセットを実行します。同様に CK-RX65N もリセットスイッチ(S1)を押下しハードウェアリセットを実行します。

ターミナルソフトで各マイコンからのログを確認します。

図 7-1 に CK-RX65N のログ画面を示します。HS3001 センサの湿度データが出力されていることを確認します。また、その下には MQTT 通信でセンサデータを AWS へ送信しているログが表示されます。

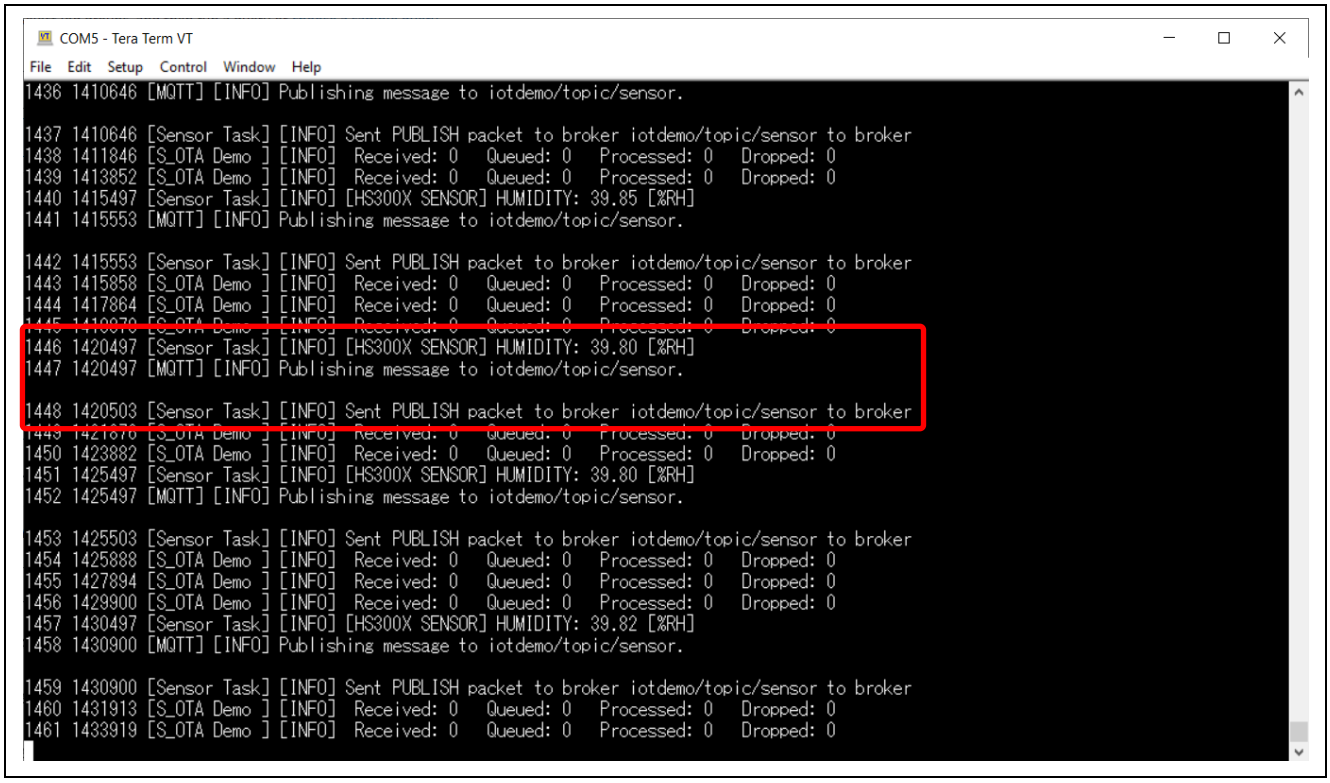


図 7-1 CK-RX65N のログ画面

次に、図 7-2 に TB-RX660 のログ画面を示します。HS3001 センサの湿度データのみ表示されていることを確認します。

また、初期状態では TB-RX660 の LED0 が点滅します。

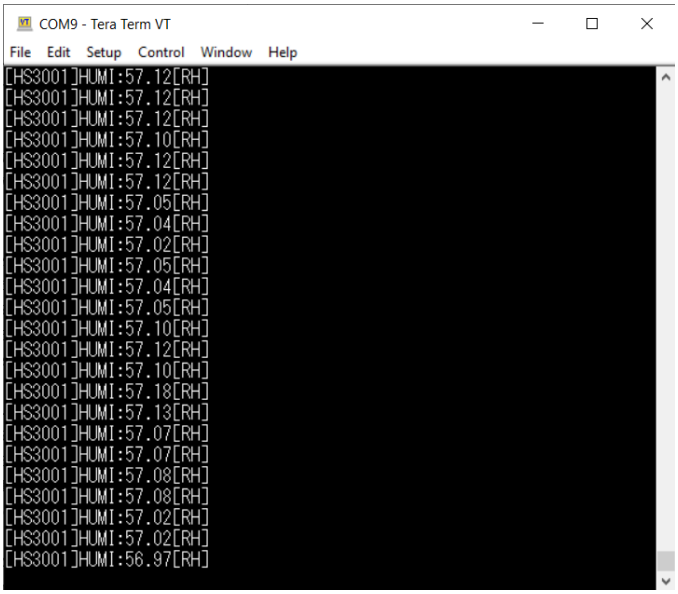


図 7-2　TB-RX660 のログ画面

最後に、図 7-3 に Amazon CloudWatch の画面を示します。HS3001 センサから取得した湿度データがグラフ化されていることを確認します。

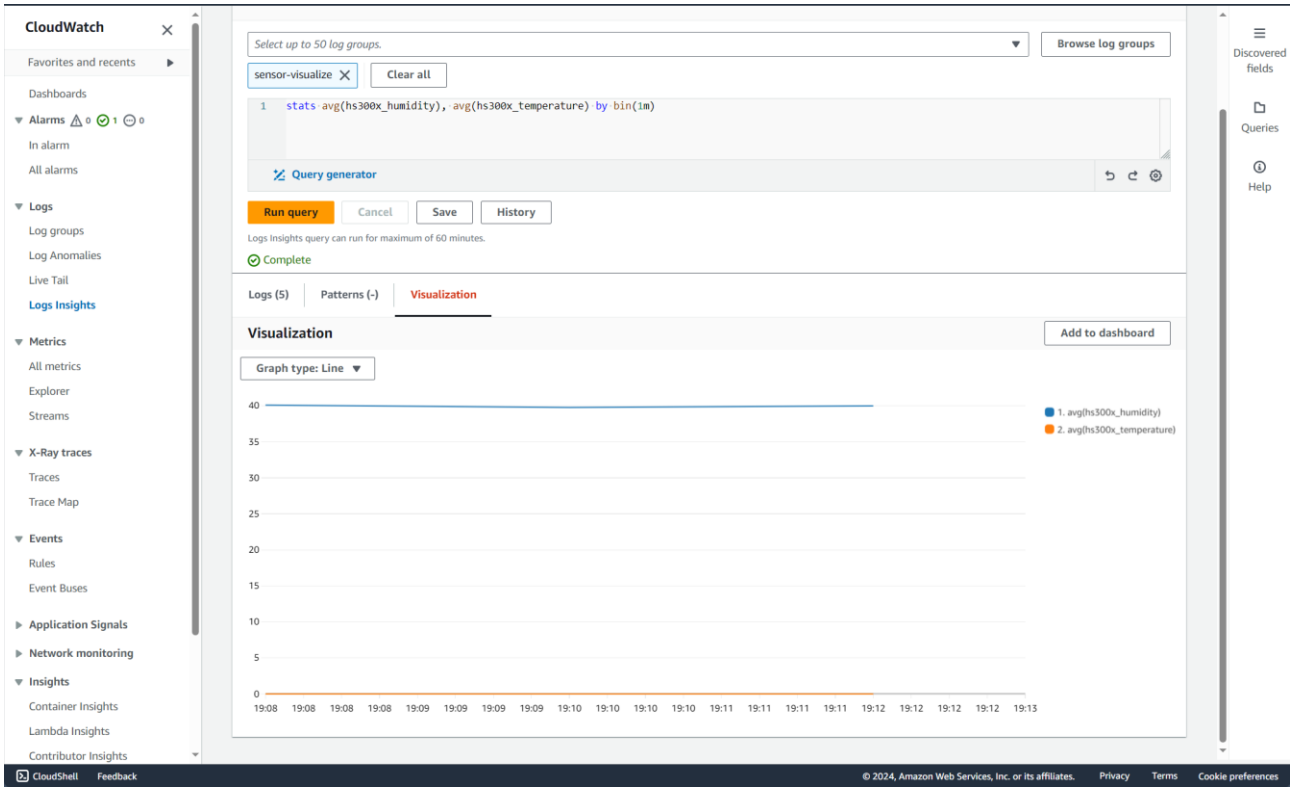


図 7-3 セカンダリ OTA アップデート前の Amazon CloudWatch のグラフ表示

この状態がセカンダリ OTA アップデート実行前の初期状態です。

7.2　TB-RX660 の OTA アップデートの実行

7.2.1　更新ファームウェアの作成

(1)　rx660_tb_2ndota_demo プロジェクトのソースコードの変更

rx660_tb_2ndota_demo/src/rx660_tb_2ndota_demo.c の MEASURE_TEMPERATURE のマクロを(1)にセットします。

また、その下の DEMO_VER_MAJOR，DEMO_VER_MINOR，DEMO_VER_BUILD を変更することで、実行時にログ表示されるバージョン値を変更できます。

```
#define MEASURE_HUMIDITY            (1)
#define MEASURE_TEMPERATURE         (1)

#define DEMO_VER_MAJOR              (2)
#define DEMO_VER_MINOR              (0)
#define DEMO_VER_BUILD              (0)
```

(2)　更新ファームウェア(MOT ファイル形式)の作成

rx660_tb_2ndota_demo プロジェクトをビルドし、MOT ファイルを作成します。

(3)　更新ファームウェア(RSU ファイル形式)の作成

作成された rx660_tb_2ndota_demo の MOT ファイルを、Renesas Image Generator を使用して RSU 形式の更新ファームウェアに変換します。

rx660_tb_2ndota_demo\RenesasImageGenerator フォルダで以下のコマンドを実行し、RSU 形式の更新ファームウェア update_firm.rsu を作成します。

```
> python .\image-gen.py -iup ..\HardwareDebug\rx660_tb_2ndota_demo.mot -o .\update_firm
-key .\keys\secp256r1.privatekey -ip .\RX660_Linear_Half_ImageGenerator_PRM.csv -vt
ecdsa
```

7.2.2　AWS で OTA ジョブの作成

(1)　AWS マネジメントコンソールにサインインし、左上の「**Service**」→「**IoT**」→「**IoT Core**」を選択します。

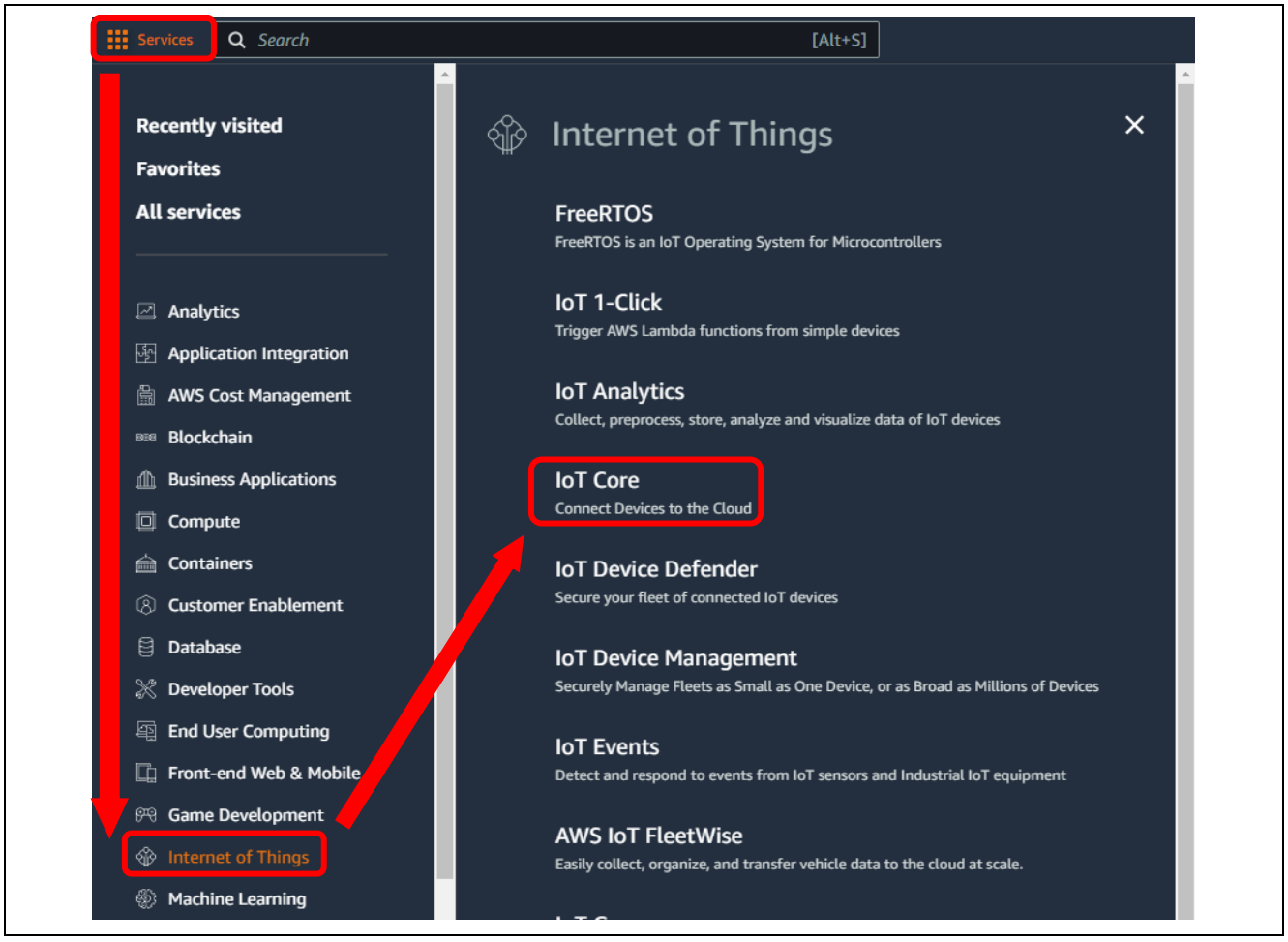


図 7-4 AWS のサービス画面

(2)　AWS IoT Core の左のメニューから **Remote action → Jobs** を選択し、**[Create job]**をクリックします。

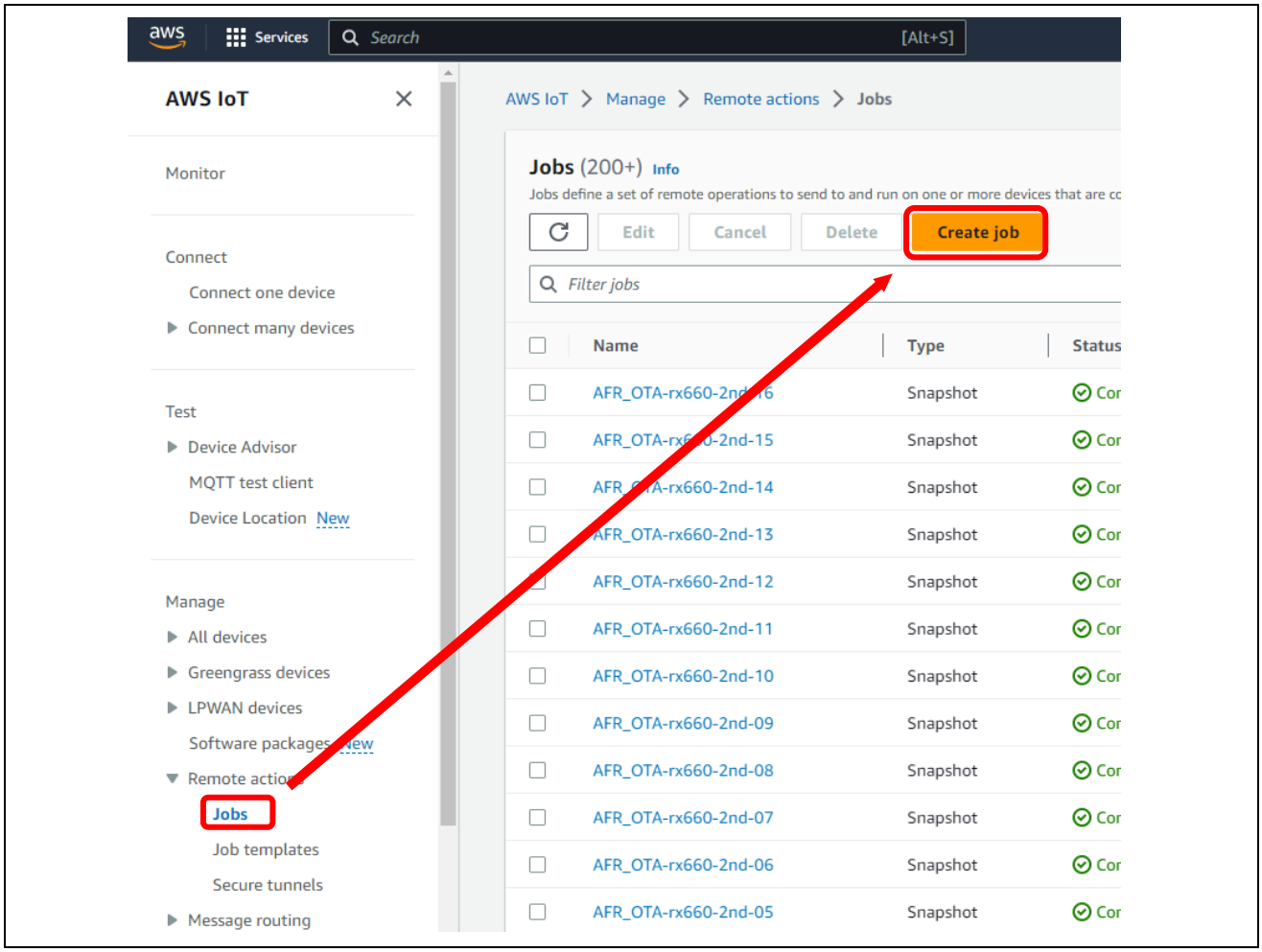


図 7-5 AWS IoT Core 画面

(3)　「Create job」画面で「**Create FreeRTOS OTA update job**」を選択して**[Next]**をクリックします。

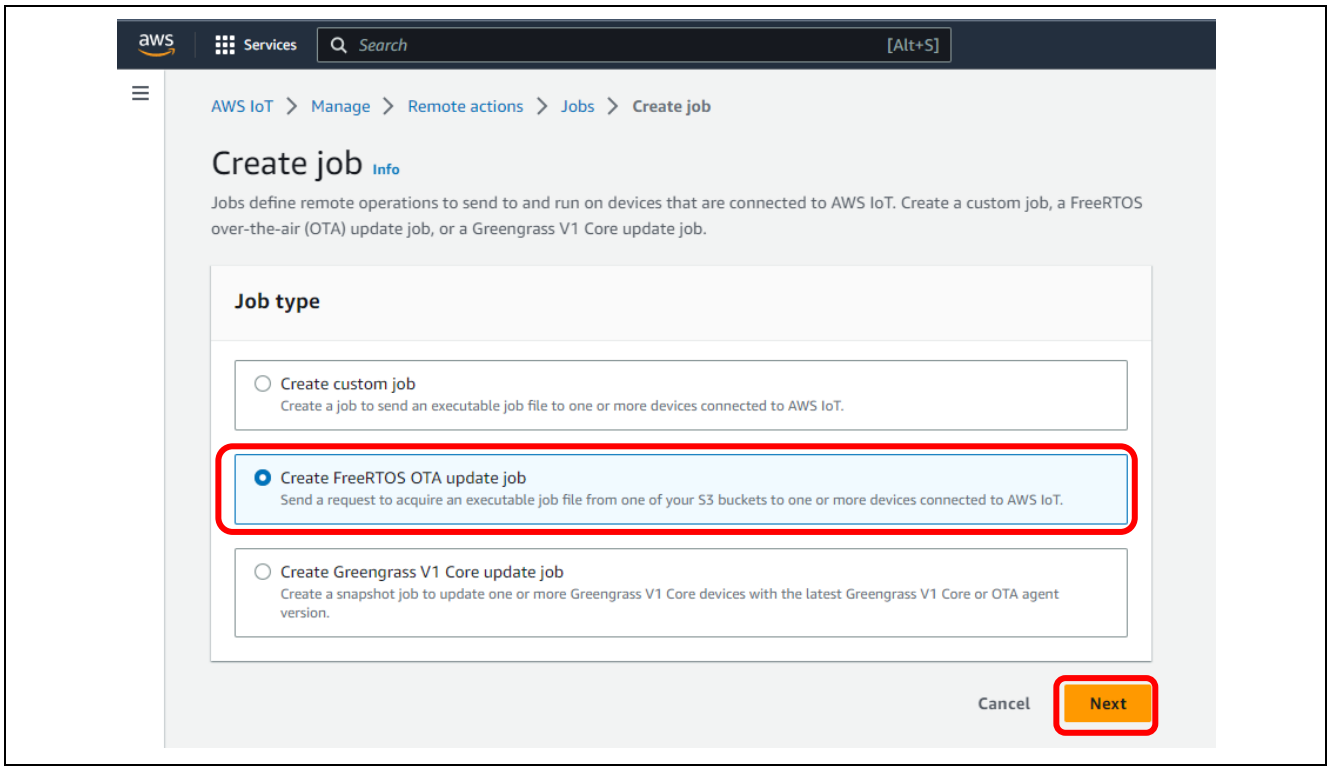


図 7-6 ジョブ作成画面

(4)　「OTA job properties」画面で「**Job name**」を入力し、**[Next]**をクリックします。

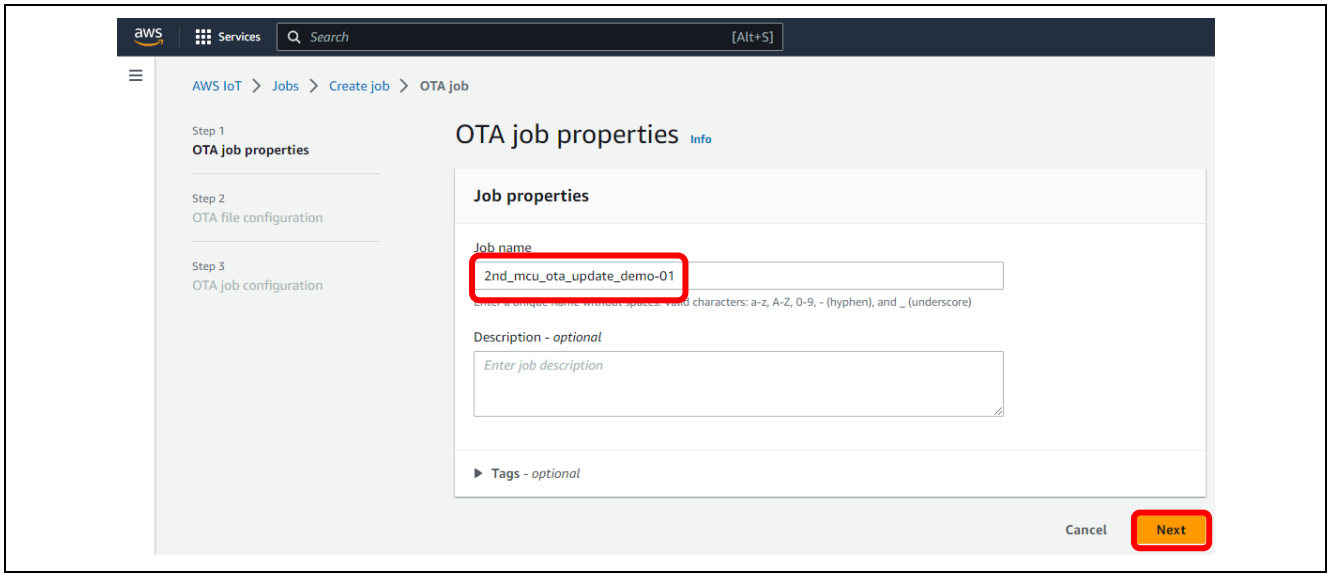


図 7-7 OTA ジョブのプロパティ入力画面

(5)　「OTA ファイル設定」画面で各種項目を入力します。

① **「Device to update」**では 6.2.3(6)章で QE for OTA で設定した **IoT device name** を選択します。

② **「Select the protocol for file transfer」**では **MQTT** を選択します。

③ **「Sign and choose your file」**では**「Sign a new file for me.」**を選択します。

④ **「Code signing profile」**では 6.3.1(3)章で作成した **Code signing profile** を選択します。
Note　ここで指定したコード署名証明書のプロファイルはセカンダリ MCU のファームウェアのコード署名検証には使用されないため、任意のプロファイルを指定できます。コード署名は Renesas Image Generator で更新用 RSU ファイルを作成した際にファイル内に記述されます。

⑤ **「File」**では**「Upload a new file.」**を選択します。

⑥ **「File to upload」**では「**Choose file**」をクリックし、7.2.1 章で作成した **TB-RX660 の更新用のファームウェア(RSU 形式)**を選択します。

⑦ **「S3 URL」**では「**Browse S3**」をクリックし、6.3.1(1)章で設定した **Amazon S3 バケット**を選択します。

⑧ **「Path name of file on device」**には任意の文字列を入力します。

⑨ **「File type」**には、「**1**」を入力します。

⑩ **「Role」**には 6.3.1(2)章で設定した **OTA 更新用サービスロール**を選択します。

以上を入力後、 **[Next]**をクリックします。

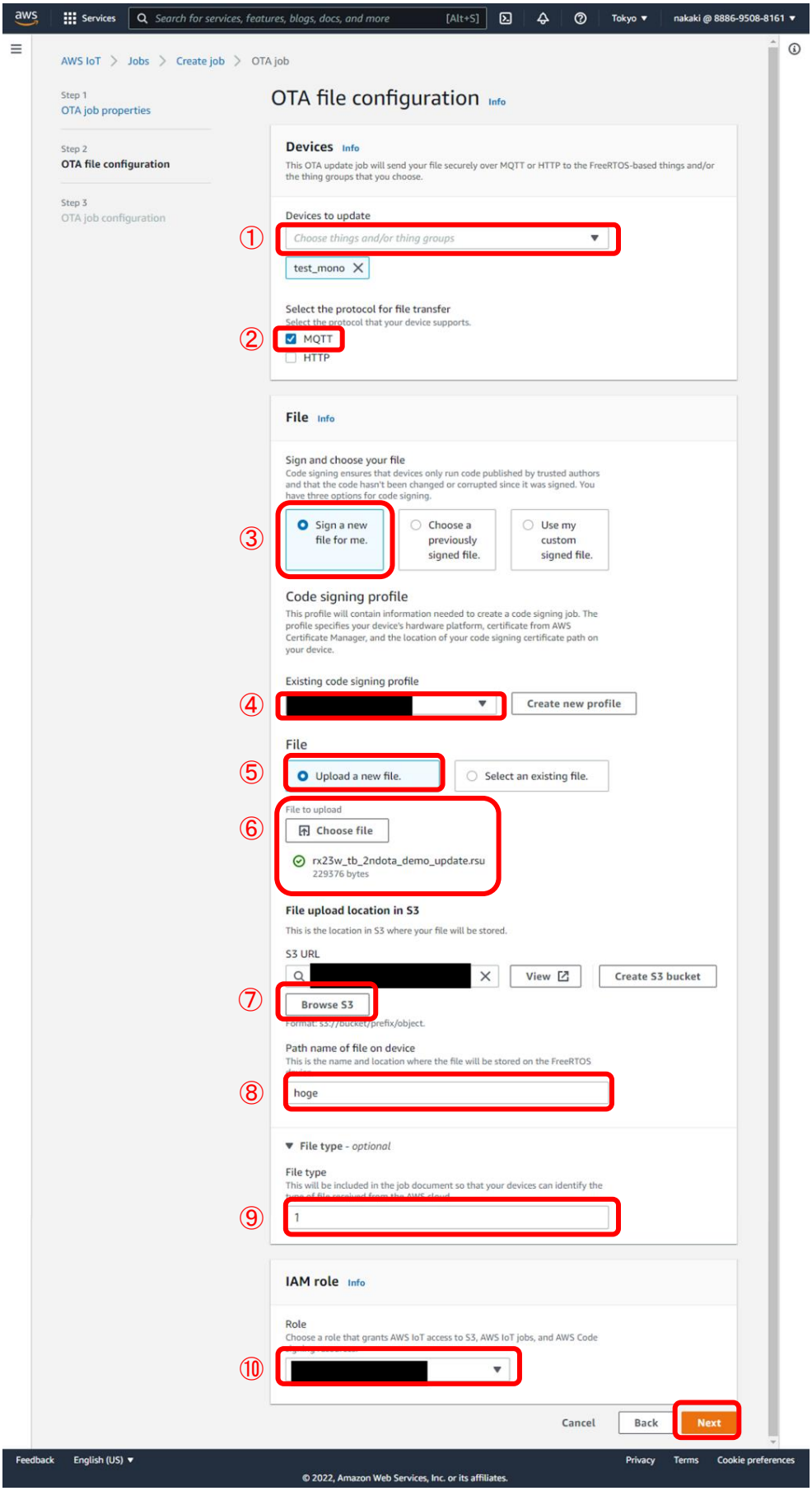


図 7-8 OTA ファイル設定画面

(6)　「OTA job configuration」画面では変更不要のため、そのまま**[Create job]**をクリックします。

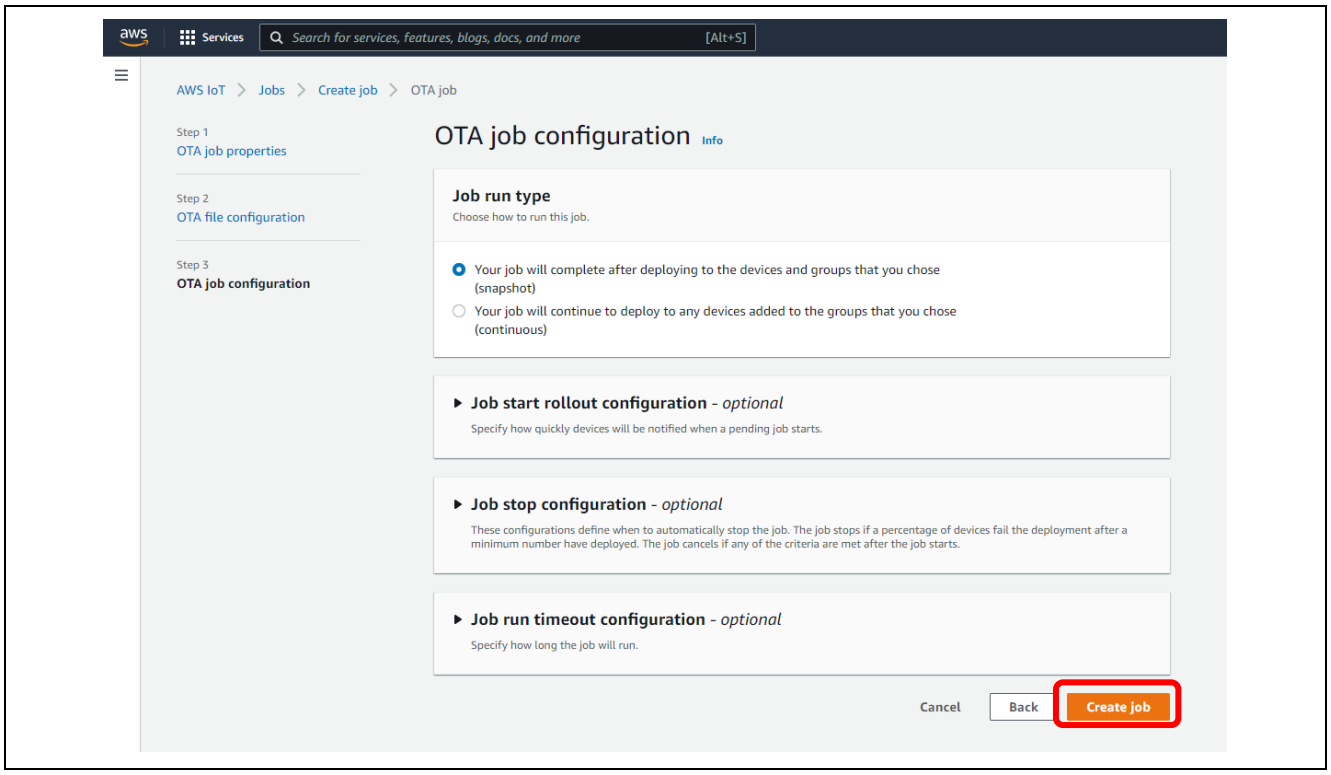


図 7-9 OTA ジョブ設定画面

以上の手順でセカンダリ OTA アップデート用の OTA ジョブが作成され、指定したデバイスに向けて OTA ジョブが配信されます。

7.2.3　セカンダリ OTA アップデート実行中の動作確認

ジョブの作成から数秒で OTA アップデートが始まります。CK-RX65N と TB-RX660 の両方から、セカンダリ OTA アップデートの進捗がログ出力されます。

7.3　OTA アップデート後の動作確認

図 7-10 にアップデート後の CK-RX65N のログ画面を示します。HS3001 センサで取得した湿度データに加えて、新たに温度データが表示されていることを確認できます。

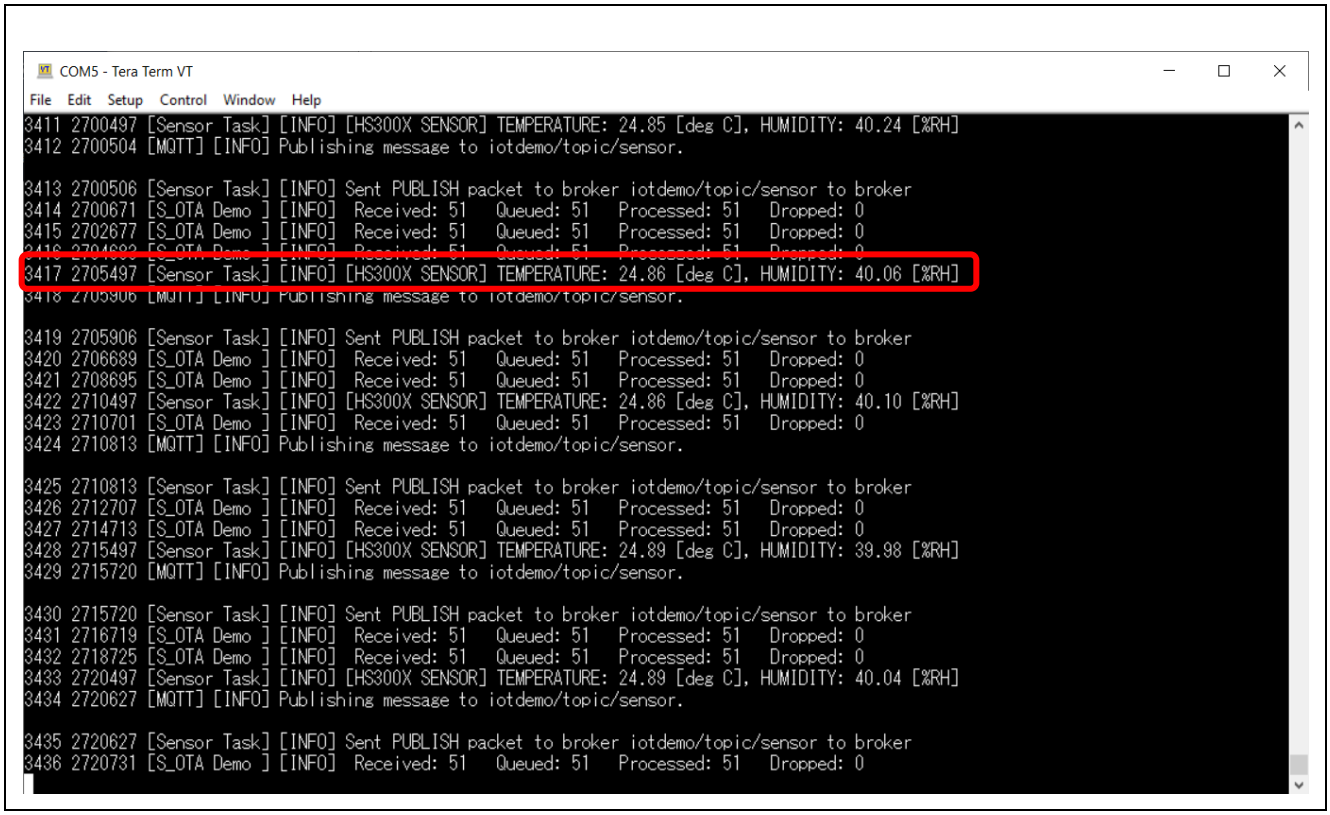


図 7-10 ファームウェア更新後の CK-RX65N のログ画面

次に、図 7-11 に TB-RX660 のログ画面を示します。TB-RX660 ファームウェア更新に成功すると、HS3001 センサで湿度と温度の測定データを取得します。

また、初期状態で点滅していた LED0 に加えて LED1 も点滅します。

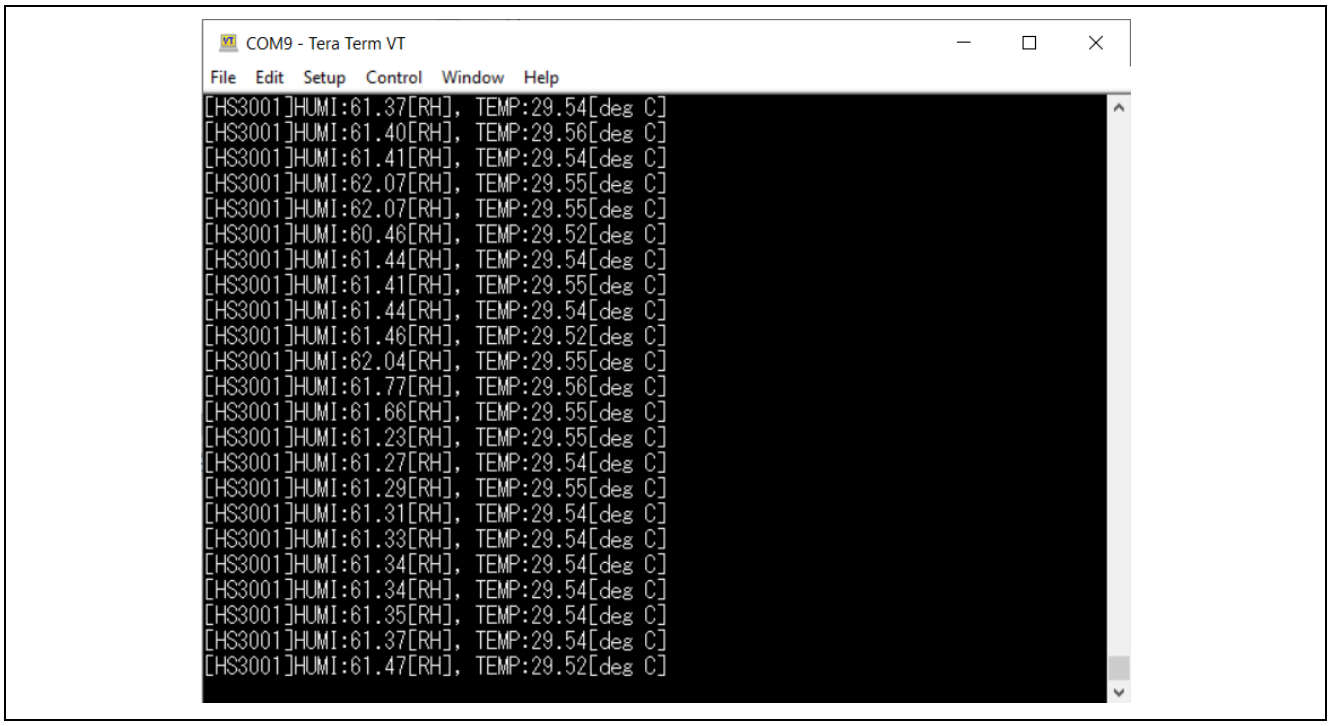


図 7-11　ファームウェア更新後の TB-RX660 のログ画面

最後に、図 7-12 に Amazon CloudWatch の画面を示します。HS3001 センサから取得した湿度と温度の測定データがグラフ化されていることを確認します。

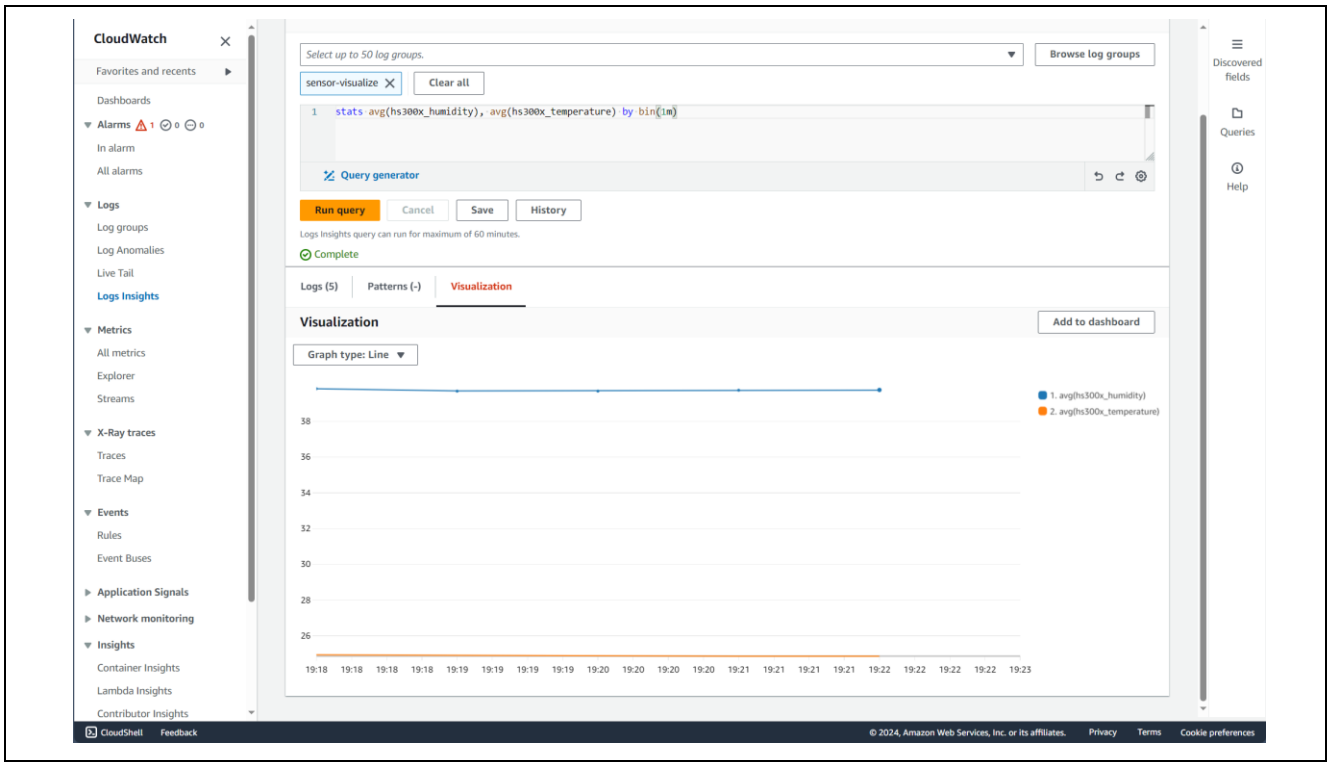


図 7-12　セカンダリ OTA アップデート後の Amazon CloudWatch のグラフ表示

デモ動作は以上となります。

8.　注意事項

8.1　使用するオープンソースソフトウェアのライセンス情報

以下のオープンソースソフトウェアを使用しています。

- TinyCrypt Cryptographic Library
  - URL　https://01.org/tinycrypt/
  - ライセンス　https://github.com/intel/tinycrypt/blob/master/LICENSE

- FreeRTOS
  - URL　https://www.freertos.org/
  - ライセンス　FreeRTOS open source licensing, FreeRTOS license description, FreeRTOS license terms and OpenRTOS commercial licensing options.

8.2　デモを実施する AWS のリージョンとユーザ権限について

デモ実施の際の AWS の設定について、使用リージョンとユーザ権限についての注意事項を以下に示します。

【使用リージョンについて】

このデモは AWS の ap-northeast-1（アジアパシフィック（東京））リージョンで実施しています。
他のリージョンでこのデモを実施する場合は、デモで使用しているサービスがそのリージョンで提供されているか事前にご確認ください。

【ユーザ権限について】

このデモは AWS Identity and Access Management (IAM)で AdministratorAccess の権限が付与されたユーザで実行しています。そのため、各種サービスを使用する際の IAM での必要権限の付与に関して未記載です。

8.3　AWS の利用料金について

AWS の利用状況によっては、デモで作成・使用したクラウドリソースによって料金が発生する場合があります。意図しない課金を防ぐために、デモ実施後は作成したクラウド上のリソースを削除することをお勧めします。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2021/12/29	—	新規作成
1.01	2022/01/24	3	商標記載を目次ページに移動
		5	1.2.1 フォルダ／ファイル構成　を更新
		8	1.3 動作確認環境 IDE 欄を e2 studio 2021-07 に修正 表 1-3 デモ動作確認条件（センサボード）　I/F 変換ボード欄を見直し
		8	1.4 コードサイズ　タイトルの誤記を修正
		9	1.5 1.5　デモ実施上 AWS のリージョンとユーザ権限についてを追加
		10 - 12	2. ファームウェア書き込み　文章を見直し
		13	3.1 デモ概要　表 3-1 デモ内容概要　記載内容を見直し
		15	3.2.2 TB-RX23W ボードの構成　(1) RSK+RX65N-2MB ボードとの UART 通信用の端子を接続　文章を見直し
		18	3.3.1 RSK+RX65N-2MB ボード用の初期ファームウェアの作成と書き込み　表 3-2 変更が必要なファイルの格納場所　記載内容を見直し 表下のフォルダ記載箇所の情報を詳細に追記
		18	3.3.1 RSK+RX65N-2MB ボード用の初期ファームウェアの作成と書き込み　(4) 手順 4： 初期ファームウェアの結合　タイトルの誤記を修正
		20	3.3.2 TB-RX23W ボード用の初期ファームウェアの作成と書き込み　(2) 手順 2： 公開鍵情報入力によるプロジェクトの変更　タイトルに「公開鍵情報入力による」を追記
		20	3.3.2 TB-RX23W ボード用の初期ファームウェアの作成と書き込み　(4) 手順 4： 初期ファームウェアの結合　タイトルの誤記を修正
		21	3.3.4 AWS 側の設定　「AWS CLI を利用するための設定」を「3.Python スクリプトを実行するための設定」　に変更
		22 - 45	3.3.4 AWS 側の設定　(2) センサデータの可視化のための設定説明を見直し
		46	3.3.4 AWS 側の設定　「(3) AWS CLI を利用するための設定」を「(3) Python スクリプトを実行するための設定」　に変更
		50	3.4.2 TB-RX23W ボードのファームウェアの 2nd OTA アップデート実行　(2) 手順 2：Renesas Secure Flash Programmer でアップデート用のファームウェアを作成　内容を見直し
		50	3.4.2 TB-RX23W ボードのファームウェアの 2nd OTA アップデート実行　(3) 手順 3：Python スクリプトを実行するために、PC 上でコマンドプロンプトを起動　タイトルを更新
		50	3.4.2 TB-RX23W ボードのファームウェアの 2nd OTA アップデート実行　(4) 手順 4：コマンドプロンプト上で script フォルダをカレントフォルダに設定　タイトルを更新
		51	3.4.2 TB-RX23W ボードのファームウェアの 2nd OTA アップデート実行　(5) 手順 5：2nd OTA アップデートを実行　表 3-7 コマンドに記述する入力値の参照先　USER_REGION 欄を更新

		51 - 52	3.4.2 TB-RX23W ボードのファームウェアの 2nd OTA アップデート実行　(6) 手順 6：2nd OTA アップデート完了を確認 文章を見直し 注意　を追加
		53	3.4.3　HS3001 センサと ZMOD4410 センサのデータのクラウドへのアップロードとデータ可視化　文章を見直し
1.10	2022/03/31	—	AWS IoT Over-the-air Update Library v3.0.0 に対応
		5 - 7	RX65N 用プロジェクト変更につき、パッケージ及びフォルダ構成を修正
		5 - 7	各プロジェクトのフォルダ構成に.settings フォルダを追加
		9	動作確認環境の IDE を e2studio 2022-01 に更新、Toolchain を CC-RX V3.04.00 に更新、RX65N 用プロジェクトの FreeRTOS を Version 2021.07 に更新
		9	コードサイズを更新
		19 - 20	RX65N 用プロジェクト変更に伴い、初期ファームウェアの作成方法を更新
		47	RX65N 用プロジェクト変更に伴い、出力ログのスクリーンショットを更新
		51 – 55	OTA アップデートの実行方法を更新
		56	RX65N 用プロジェクト変更に伴い、出力ログのスクリーンショットを更新
2.00	2024/03/31	—	使用ボードを CK-RX65N と TB-RX660 に変更。
		—	RX65N 用のプロジェクトの FreeRTOS パッケージを更新
		—	RX660 用のプロジェクトの FWUP FIT のバージョンを更新
		—	使用ボードとプロジェクトの変更に伴い、アプリケーションノートを全面的に改訂

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、明示または黙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。