Getting Started with the GR-ROSE

株式会社コア

本説明書では、GR-ROSE を使った AWSIoT デモ構築手順を説明します。デモでは、

- **センサデータ(CO2、温度、湿度)のパブリッシュ**
- **フリートプロビジョニング**
- **Shadow**
- **OTA**

の動作をご覧いただくことができます。

目次

1　デモ環境

本デモを実行するには、以下のご準備が必要です。

必須

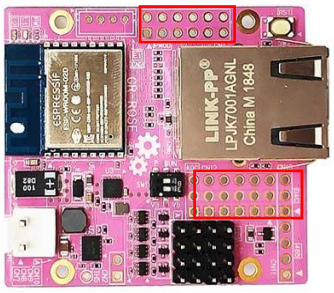
- WindowsPC (e²stduio 用)
- WSL or Git Bash（TSIP 鍵情報生成用シェルスクリプト実行用）
- Teraterm 等のターミナルソフト
- GR-ROSE
- GR-ROSE と PC 接続用 USB ケーブル(Type-A/micro-B)
- 無線 LAN ルータ
- AWS アカウント

デモの全機能を動かす場合に必要なもの

※センサをご用意いただかなくても、デモの動作を確認することができます

- センサを接続するためのジャンパワイヤ（オス-メス）
- 2.54mm ピッチのピンヘッダ

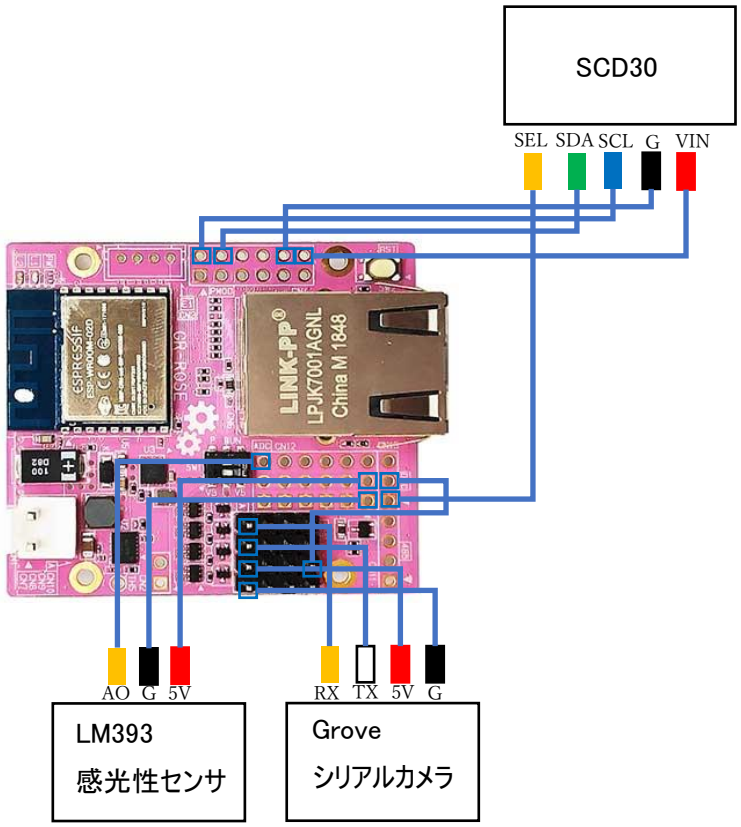
  ※GR-ROSE の以下赤枠部分に半田付けしてください。



- 以下のセンサ類

SCD 30	https://www.sensirion.com/jp/environmental-sensors/humidity-sensors/co2-sensor/
LM393 感光性センサ	https://www.amazon.co.jp/gp/product/B010GX9DHQ/ref=ppx_yo_dt_b_asin_title_o03_s00?ie=UTF8&psc=1
Grove シリアルカメラ	https://akizukidenshi.com/catalog/g/gM-09161/

センサ接続図

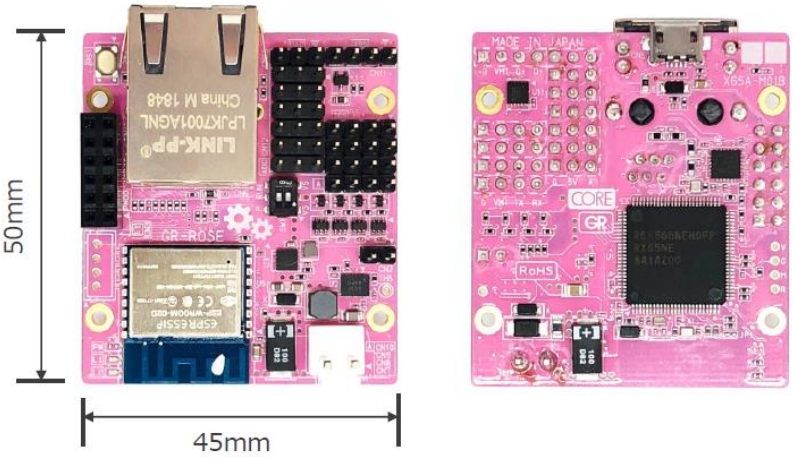


こちらに GR-ROSE のソースコードを公開していますので、ダウンロードして、以下手順どおりにデモ環境を構築してください。

なお、デモに使用するセンサ類がお手元に用意できない場合でも、GR-ROSE 単体でデモの動作確認が可能です（その場合は、ダミーデータを AWS にパブリッシュします）。

1.1　GR-ROSE について

デモのエッジデバイスとして利用している GR-ROSE について、その特長を簡単に説明します。



GR-ROSE は、ルネサスエレクトロニクス製 32bit MCU の RX65N 搭載した超小型 IoT プロトタイピングボードです。ネットワークのインターフェースとして Ethernet ,Wi-Fi に対応しているため、I2C,SPI,ADC などのインターフェースから取得したセンサ情報を AWS などのクラウドにアップロードすることができます。また、センシングだけでなく、シリアルサーボ制御用のインターフェースも持つため、ロボットの制御など、あらゆる用途にお使いいただける組み込みボードです。
回路図等、スペックに関する情報は、こちらをご参照ください。

1.2　TSIP について

本デモでは、RX65N に搭載されている TSIP を使用して、鍵の管理、フリートプロビジョニングによる鍵データの更新を行います。TSIP を使用することで、鍵データが暗号化された形で ROM に保存され、デバイスに対して物理的にアクセスされた場合でも、鍵情報の漏洩を防ぐことができます。
TSIP の詳細については、
https://www.renesas.com/jp/ja/document/apn/rx-family-tsip-trusted-secure-ip-module-firmware-integration-technology-binary-version?language=ja
を参照してください。

1.3　AWS アカウントについて

※AWS のアカウントをお持ちではない方は、AWS のドキュメントを参考にアカウントの作成を行ってください。

https://aws.amazon.com/jp/premiumsupport/knowledge-center/create-and-activate-aws-account/

AWS のコンソールを利用する IAM ユーザーの作成については、AWS のドキュメントを参考に作成を行ってください。

https://docs.aws.amazon.com/ja_jp/IAM/latest/UserGuide/id_users_create.html

AWS IoT Core 及び FreeRTOS を利用するための権限として以下の 2 つの権限が IAM ユーザーに必要となります。

- AmazonFreeRTOSFullAccess
- AWSIoTFullAccess

IAM ユーザーの作成及び、権限の付与については、FreeRTOS のドキュメントの「Setting up your AWS account and permissions」で紹介されておりますので、その手順を参考に進めてください。

https://docs.aws.amazon.com/freertos/latest/userguide/freertos-prereqs.html

注意）この資料に含まれる IoT Policy やセキュリティーの設定は開発用途向けですので、本番環境での利用では「AWS IoT Core policy examples」や、「Security best practices in AWS IoT Core」を参考にしてください。

https://docs.aws.amazon.com/iot/latest/developerguide/example-iot-policies.html

https://docs.aws.amazon.com/iot/latest/developerguide/security-best-practices.html

2　デモ環境構築手順

2.1　GR-ROSE ビルド環境

e2studio のインストール

https://www.renesas.com/jp/ja/software-tool/e-studio

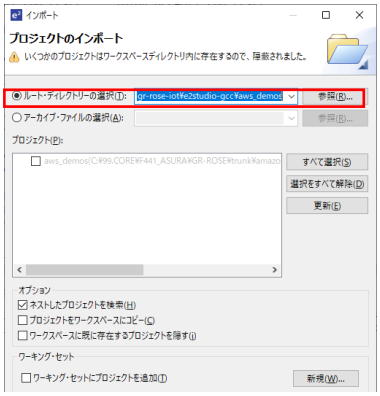
の「統合開発環境 e² studio 2021-07 Windows 用インストーラ」をダウンロードしてインストール

RX マイコンのツールチェインをインストールしてください

インストール手順詳細：

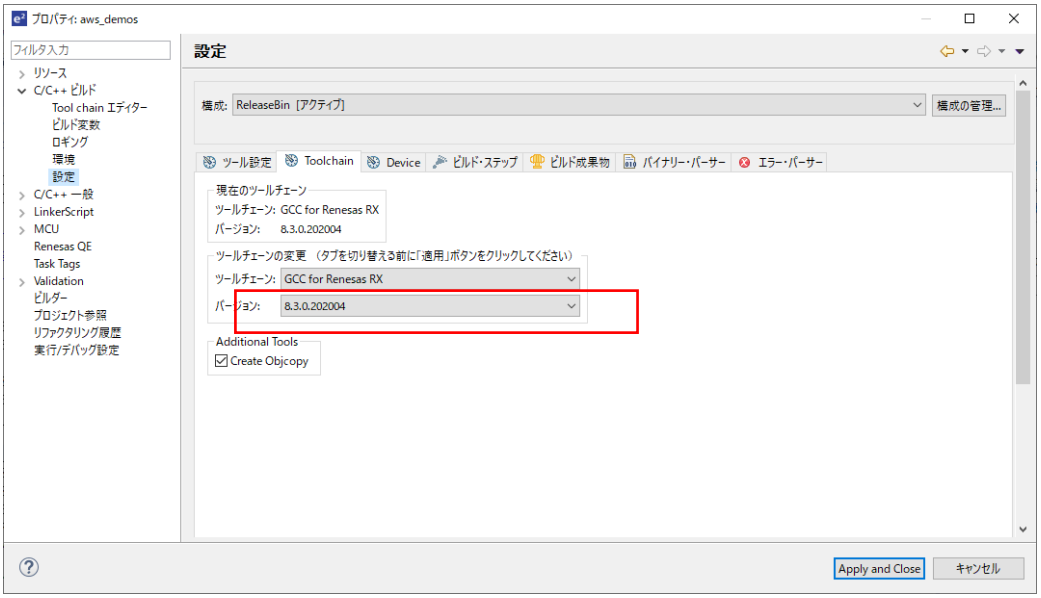
https://www.renesas.com/jp/ja/document/man/e-studio-integrated-development-environment-users-manual-getting-started-guide-rx-rl78-rh850-family?r=1175496

インストール後、ダウンロードした GR-ROSE_demo/amazon-freertos.zip を任意の場所に展開してください。上記「3.3　ワークスペースへの既存プロジェクトのインポート」を参照し、「(3) e2 studio　の [ファイル(F)] → [インポート(I)...]を選びます。」で、ルートディレクトリの選択->amazon-freertos¥projects¥renesas¥rx65n-gr-rose-iot¥e2studio-gcc¥aws_demos を選択してください。



インポート後、プロジェクト->プロジェクトのビルドで aws_demos をビルドしてください。

※インストールしたツールチェインのバージョンによってはビルド時にエラーが発生することがあります。その場合は、プロジェクト->プロパティ-> C/C++ビルド->設定->Toolchain で、バージョンをインストールしたツールチェインのバージョンに設定してください



2.2 TSIP のサンプルプロジェクトの展開

ダウンロードした GR-ROSE_demo/r20an0548jj0112-lib-rx-tsip-security.zip を任意のフォルダに展開してください。

2.3　ルート CA 証明書、クライアント証明書、鍵ペアのダウンロード、バイナリ化

AWS マネジメントコンソール上でのルート CA 証明書、クライアント証明書、鍵ペアのダウンロード保方法について説明します。まず、AWS IoT Core のページを開き、下図赤枠の証明書をクリックしてください。

※本書で使用している AWS マネジメントコンソールのキャプチャは、2021/10/5 時点のものです。

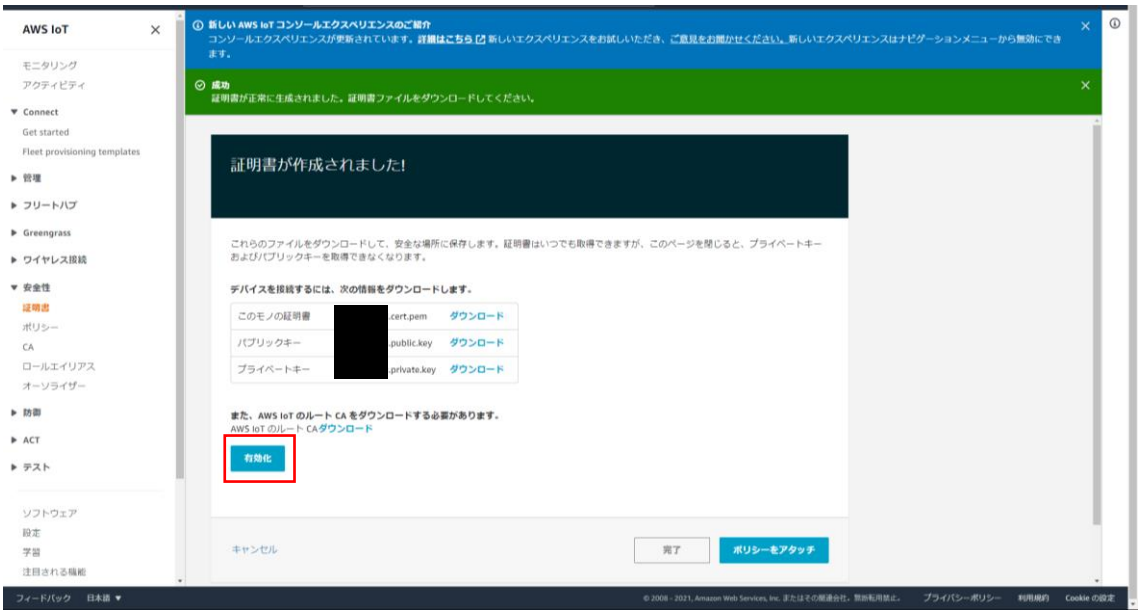
作成ボタンを押すと、以下画面が表示されます。1-Click 証明書作成(推奨)の「証明書を作成」をクリックしてください

モノの証明書（クライアント証明書）、パブリックキー、プライベートキー、AWS IoT のルート CA をダウンロードしてください

AWS IoT のルート CA は、別ページに遷移するので、ページ内の

RSA 2048 bit key: Amazon Root CA 1.

を開き、AmazonRootCA1.pem の名前で保存してください



ダウンロードしたルート CA 証明書(AmazonRootCA1.pem)は、

r20an0548jj0112-lib-rx-tsip-security

¥FITDemos¥rx_tsip_freertos_mbedtls_sample¥key_crt_sig_generator¥ca

に格納してください

ダウンロードしたクライアント証明書(xxxxx-certificate.pem.crt)、パブリックキー(xxxxx-public.pem.key)、プライベートキー(xxxxx-private.pem.key)は、

r20an0548jj0112-lib-rx-tsip-security

¥FITDemos¥rx_tsip_freertos_mbedtls_sample¥key_crt_sig_generator¥client-rsa2048

に格納してください

最後に、有効化を押し、完了を押してください。

次に、クライアント証明書を DER 形式に変換します

r20an0548jj0112-lib-rx-tsip-security

¥FITDemos¥rx_tsip_freertos_mbedtls_sample¥key_crt_sig_generator ¥key_crt_sig_generator

フォルダ内の 1_rsa2048_convertCrt.sh を WSL もしくは Git Bash で実行してください。

output フォルダに生成された

- ✓ client_rsa2048_crt_array.txt
- ✓ AmazonRootCA1_crt_array.txt
- ✓ AmazonRootCA1_sig_array.txt

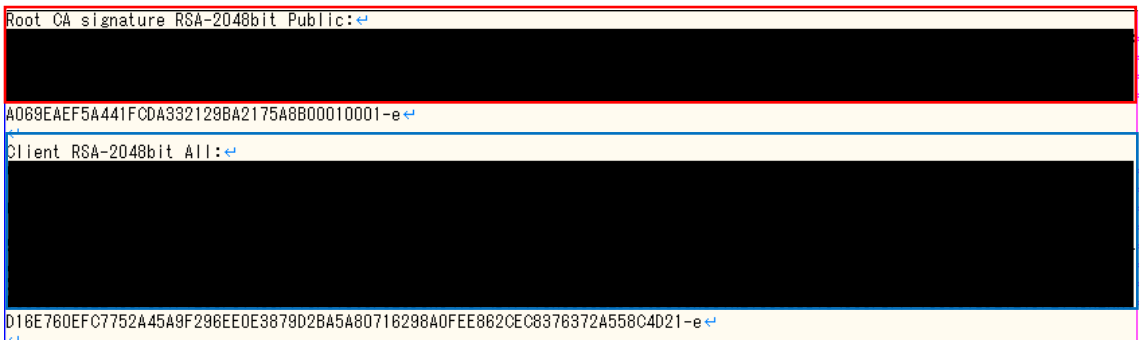
を

amazon-freertos¥vendors¥renesas¥boards¥rx65n-gr-rose-iot-gcc¥aws_demos¥application_code

に格納してください(上書きで OK)

2.4　秘密鍵の暗号化、バイナリ化

key_crt_sig_generator フォルダ内の 3_showkeyValues.sh を実行してください。コンソールに、



のように表示されます。

r20an0548jj0112-lib-rx-tsip-security¥tool¥ Renesas Secure Flash Programmer.exe を実行してください。GR-ROSE に埋め込む鍵情報の C 言語ソースファイルを作成します

① Key Wrap タブを選択し、Select MCU で TSIP(DF Memory 32KB)を選択してください

② Key Type で「RSA-2048bit Public」を選択し、コンソールの上記赤枠部分をコピーし、Key Data に貼り付けてください。ただし、末尾の-e は不要です。Register を押下してください

③ Key Type で「RSA-2048bit All」を選択し、コンソールの上記青枠部分をコピーし、Key Data に貼り付けてください。ただし、末尾の-e は不要です。Register を押下してください

④ Key Type で「AES-128bit」を選択し、KeyData に 11111111222222223333333344444444 を入力してください。。Register を押下してください

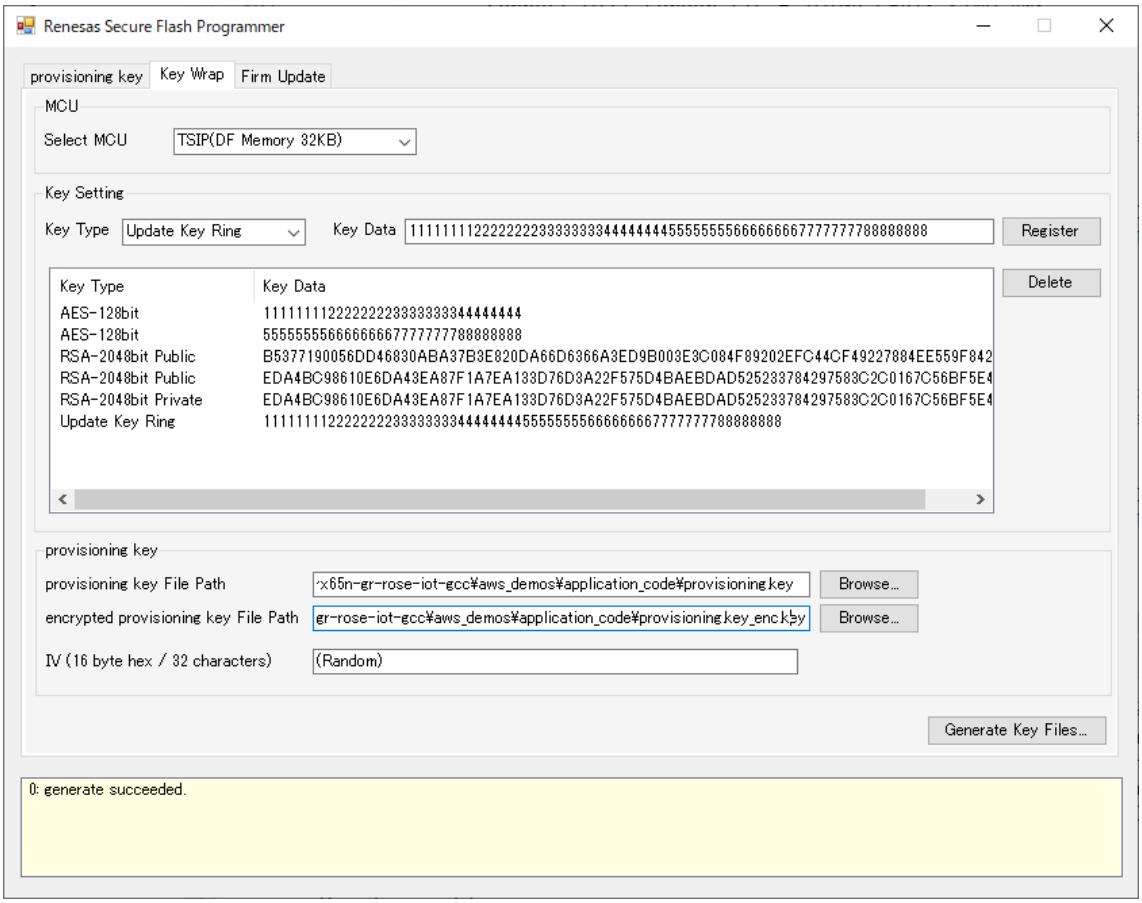
⑤ Key Type で「AES-128bit」を選択し、KeyData に 55555555666666667777777788888888 を入力してください。。Register を押下してください

⑥ Key Type で「Update Key Ring」を選択し、KeyData に 1111111122222222333333334444444455555555666666667777777788888888 を入力してください。Register を押下してください

⑦ provisioning key の provisioning key File Path に amazon-freertos¥vendors¥renesas¥boards¥rx65n-gr-rose-iot-gcc¥aws_demos¥application_code¥ provisioning.key を選択してください

⑧ provisioning key の encrypted provisioning key File Path に amazon-freertos¥vendors¥renesas¥boards¥rx65n-gr-rose-iot-gcc¥aws_demos¥application_code¥provisioning.key_enc.key を選択してください

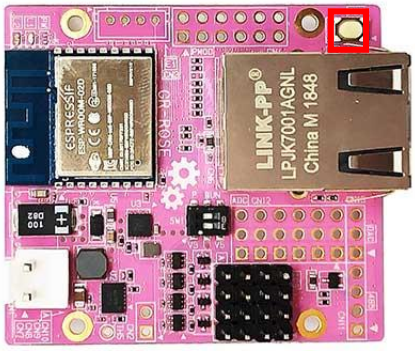
⑨ Generate Key Files を押下してください。key_data.c, key_data.h の保存先を amazon-freertos¥vendors¥renesas¥boards¥rx65n-gr-rose-iot-gcc¥aws_demos¥application_code としてください（上書き OK）。



※手順⑦、⑧で指定する provisioning.key, provisioning.key_enc.key は、本来ユーザ自身でご用意いただく必要があります。本環境では構築の手順を簡単にするため、サンプルプロジェクトに追加しておりますが、ご自身で provisionig.key、provisioning.key_enc.key を用意される場合は、r20an0548jj0112-lib-rx-tsip-security¥reference_documents¥ja¥ r01an5792jj0101-rx-tsip.pdf をご参照ください

2.5　ビルド、プログラムの書き込み

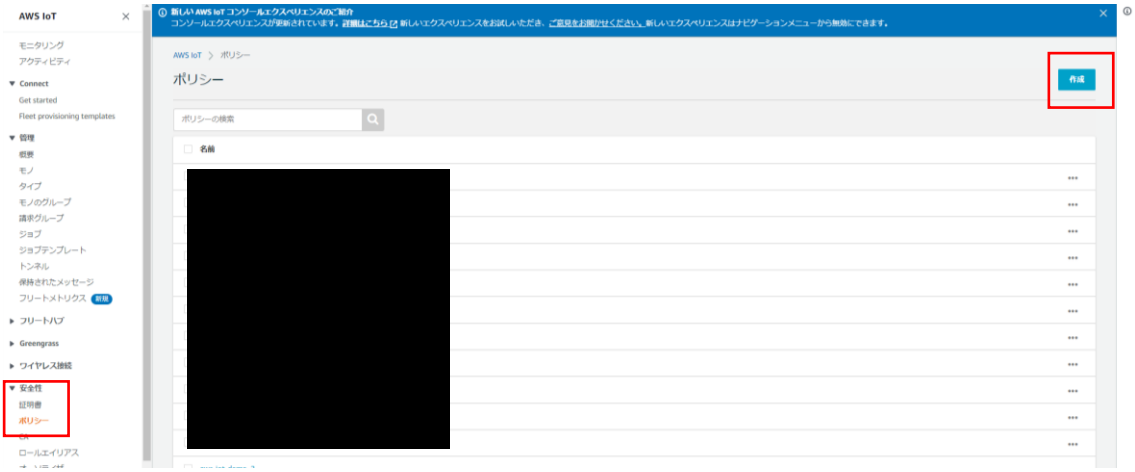
e2studio で aws_demos プロジェクトのビルドを実行してください。ビルドに成功すると、amazon-freertos¥projects¥renesas¥rx65n-gr-rose-iot¥e2studio-gcc¥aws_demos¥ReleaseBin フォルダに、aws_demos.bin が生成されます。PC と GR-ROSE を USB で接続し、GR-ROSE のリセットスイッチ（下図赤枠）を押してください。



しばらくすると LED の赤、緑が点灯し、PC 上でストレージ：GR-ROSE としてアクセスすることができます。aws_demos.bin をドラッグ＆ドロップで転送してください。プログラムの更新が始まります。LED の点滅->消灯したらアップデート完了です。

2.6　フリートプロビジョニングの準備

まず、ポリシーを作成します。AWS マネジメントコンソール上で、以下のように安全性 -> ポリシーを開き、作成を押してください。



ポリシーの名前を aws-iot-demo とし、ポリシーを以下のように入力してください。アドバンスモードをクリックすることで、ポリシードキュメントを直接入力することができます。入力後、作成を押します。

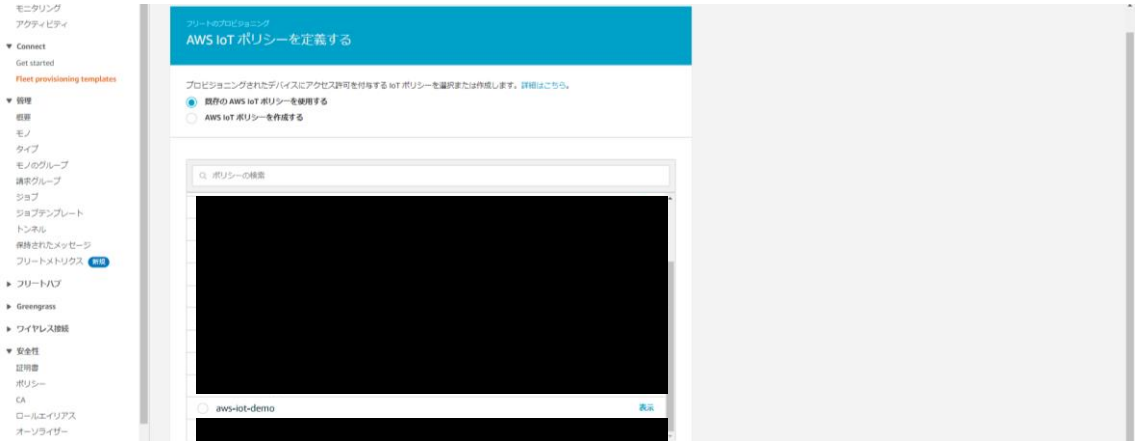
※<your-account-id>は AWS のアカウント ID に変更してください


```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Subscribe"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Publish",
        "iot:Receive"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
      ],
      "Resource": [
        "arn:aws:iot:ap-northeast-1:<your-account-id>:client/${iot:Connection.Thing.ThingName}"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:GetThingShadow",
        "iot:UpdateThingShadow",
        "iot:DeleteThingShadow"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

次に、フリートプロビジョニングのテンプレートを作成します。
AWS マネジメントコンソール上で、以下のように Connect -> Fleet provisioning templates を選択して、作成を押してください。

開始方法をクリックし、以下のように入力して、次へを押してください。

テンプレート名は、fleet-prov-template-grrose-tsip としてください。
プロビジョニングロールについては、「ロールの作成」を押し、作成してください。名前の指定はありません。



既存の AWS IoT ポリシーを使用する、を選択した状態で、先程作成した aws-iot-demo を選択してください。ここで選択したポリシーがフリートプロビジョニング後にモノにアタッチされます。
次へを押し、以下のようにモノの情報を定義します。モノの名前プレフィックスを「GR-ROSE_」とし、その他は未入力でかまいません。

テンプレートを作成、をクリックし、

2. 3. ルート CA 証明書、クライアント証明書、鍵ペアのダウンロード、バイナリ化

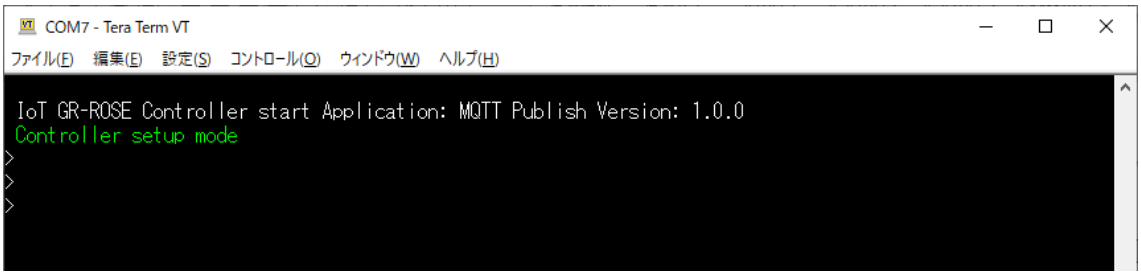
で作成したクライアント証明書を選択してください。選択後、「ポリシーをアタッチ」を押してください。

最後に、テンプレートを有効化する、を押して完了です

2.7 AWS との接続

teraterm 等、ターミナルソフトで GR-ROSE が接続されているポートを開いてください。USB-CDC 通信のため、ボーレート等の設定は不要です。

ターミナルにログが表示されたら、「ctrl+s」を入力し、setup mode(デモアプリケーションを起動せず、コンソールからの入力のみを受けつける)へ移行してください。setup mode で起動すると、コンソールに以下のようなログが表示されます。



この状態で、

>default

と入力してください。RX65N のデータフラッシュ領域がすべて初期化されます。

次に、以下を入力し、WiFi の設定を行います。

> param,set,socket,1,ssid,{接続する WifiAP の SSID}

> param,set,socket,1,password,{WifiAP のパスワード}

※SSID,パスワードは環境に合わせて入力してください

入力に成功すると、以下のようなログが表示されます。

```
>param,set,socket,1,ssid,CORE
Socket[1] Gr=4 id=0x20148000(group:4 size:164 offset:0)
  SSID=CORE id=0x20040014(group:4 size:32 offset:20)
>param,set,socket,1,password,CORE
Socket[1] Gr=4 id=0x20148000(group:4 size:164 offset:0)
  Password=CORE id=0x20040034(group:4 size:32 offset:52)
```

次に、端末のシリアル番号を入力します。以下コマンドを入力してください

> param,set,mqtt,serial_no,{シリアル番号(任意の文字列)}

ここで指定したシリアル番号と、フリートプロビジョニングテンプレートで指定した prefix =GR-ROSE_ からモノの名前およびクライアント ID がフリートプロビジョニング時に決定されます。

モノの名前

GR-ROSE_{シリアル番号}

最後に AWSIoTCore のエンドポイントを設定します。

>param,set,socket,1,url,XXXXXXX-ats.iot.ap-northeast-1.amazonaws.com

IoTCore のエンドポイントは、AWS マネジメントコンソール上の設定画面にデバイスデータエンドポイントとして表示されます。

SCD30 などのセンサが用意できない場合

センサが用意できない場合は、以下コマンドでセンサを無効化することで、ダミーデータを送信することができます。

>param,set,sensor,3,connect,false　　SCD30 の無効化

>param,set,application,camera,false　　カメラの無効化

Ctrl + d を押下後、端末が再起動し、Wifi に接続成功すると AWS に接続し、フリートプロビジョニングを実行します。

2.8　フリートプロビジョニング

2.7 で端末を再起動すると、設定された Wifi のアクセスポイントに接続し、その後 AWS と接続します。正常に接続されると、フリートプロビジョニングを行います。フリートプロビジョニングが成功すると、以下のようなログが端末のターミナルに出力されます。

```
ation 0x80c278) Wait complete with result SUCCESS.
19 13820 [Apl Main] [INFO ][MQTT][13820] (MQTT connection 0x80b738) SUBSCRIBE op
eration scheduled.
20 13820 [Apl Main] [INFO ][MQTT][13820] (MQTT connection 0x80b738, SUBSCRIBE op
eration 0x80c278) Waiting for operation completion.
21 13875 [Apl Main] [INFO ][MQTT][13875] (MQTT connection 0x80b738, SUBSCRIBE op
eration 0x80c278) Wait complete with result SUCCESS.
22 13909 [Apl Main] [INFO ][MQTT][13909] (MQTT connection 0x80b738) SUBSCRIBE op
eration scheduled.
23 13909 [Apl Main] [INFO ][MQTT][13909] (MQTT connection 0x80b738, SUBSCRIBE op
eration 0x80c278) Waiting for operation completion.
24 13965 [Apl Main] [INFO ][MQTT][13965] (MQTT connection 0x80b738, SUBSCRIBE op
eration 0x80c278) Wait complete with result SUCCESS.
25 14110 [Apl Main] [INFO ][MQTT][14110] (MQTT connection 0x80b738) MQTT PUBLISH
 operation queued.
26 14110 [Apl Main] [INFO ][MQTT][14110] (MQTT connection 0x80b738, PUBLISH oper
ation 0x80c278) Waiting for operation completion.
27 14168 [Apl Main] [INFO ][MQTT][14168] (MQTT connection 0x80b738, PUBLISH oper
ation 0x80c278) Wait complete with result SUCCESS.
[FLEETPROV]FleetProvisioning completed
```

その後、再起動を行い、フリートプロビジョニングで AWS から取得したクライアント証明書、秘密鍵をもとに、AWS との接続を行います。接続後、センサデータのパブリッシュを開始します。

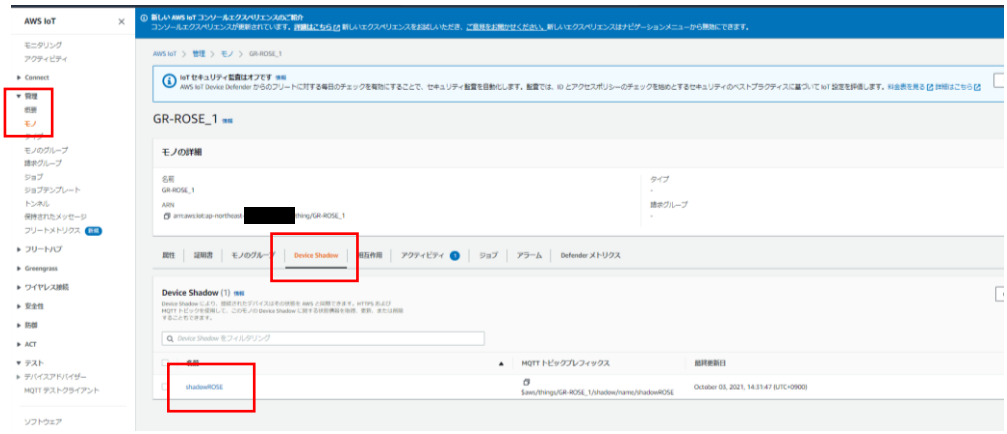
2.9　センサデータのパブリッシュ

フリートプロビジョニングで AWS から通知されたトピック名＝data/sensor/GR-ROSE_{シリアル番号}にセンサデータをパブリッシュします。

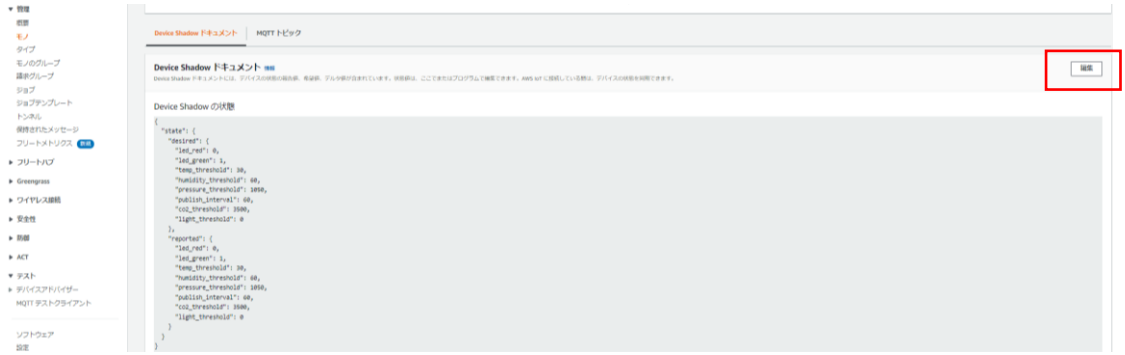
AWS マネジメントコンソール上、IoTCore->テスト->MQTT テストクライアントのトピックをサブスクライブする、で「トピックあのフィルター」に以下のようにトピックを入力し、サブスクライブを押してください。センサデータがパブリッシュされると、図のように json 形式でセンサデータが表示されます。

2.10 Shadow

本デモでは、Shadow の動作も確認いただけます。AWS マネジメントコンソール上、下図のように管理->モノ->Device Shadow-> shadowROSE を選択してください。



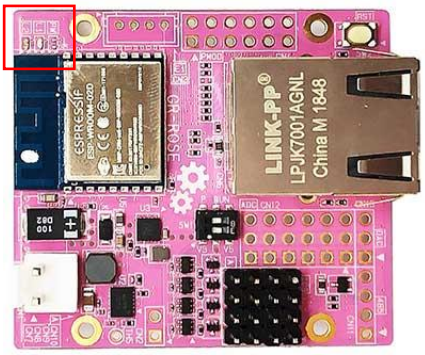
本デモでは、Shadow の動作も確認いただけます。AWS マネジメントコンソール上、下図のように管理->モノ->Device Shadow-> shadowROSE を選択してください。編集を押し、



Device Shadow の状態

```
{
  "state": {
    "desired": {
      "led_red": 0,
      "led_green": 1,
      "temp_threshold": 30,
      "humidity_threshold": 60,
      "pressure_threshold": 1050,
      "publish_interval": 60,
      "co2_threshold": 3500,
      "light_threshold": 0
    },
    "reported": {
      "led_red": 0,
      "led_green": 1,
      "temp_threshold": 30,
      "humidity_threshold": 60,
      "pressure_threshold": 1050,
      "publish_interval": 60,
      "co2_threshold": 3500,
      "light_threshold": 0
    }
  }
}
```

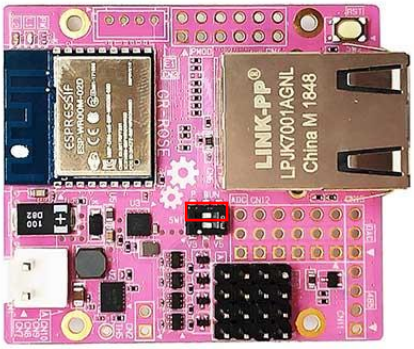
“state”: { “desired”: { の “led_red”、もしくは”led_green”を 0 もしくは 1 に変更してください。
0 の場合は LED が消灯し、1 の場合は LED が点灯します。更新を押すと shadow の変更が端末に通知され、設定した LED 状態が GR-ROSE に反映されます。



2.11 OTA

GR-ROSE は OTA にも対応しています。IoTCore のコンソール画面で OTA 更新ジョブを作成することで、GR-ROSE のファームウェアをアップデートすることができます。

OTA を実行するためには、GR-ROSE の bootloader を更新する必要があります。まず、
https://www.renesas.com/jp/ja/software-tool/renesas-flash-programmer-programming-gui
より、RenesasFlashProgrammer をダウンロード、インストールしてください。※本デモ作成時は、V3.06 を使用しました。
インストール後、RenesasFlashProgrammer を起動してください。ファイル->プロジェクトを開く、
amazon-freertos¥vendors¥renesas¥bootloader¥gr-rose¥gr-rose.rpj
を選択してください。その後、プログラムファイルの参照から、
amazon-freertos¥vendors¥renesas¥bootloader¥rose_usbfw.mot
を選択してください。
下図赤枠のようにディップスイッチを設定し、PC と GR-ROSE を USB で接続してください。



接続設定タブから、COM port ->ツール詳細で、GR-ROSE のポートを選択してください
※RX USB Boot(CDC)と表示されます
操作タブでスタートを押すと、Bootloader が更新されます。アプリケーションプログラムがすべて消去されてしまうので、再度 aws_demos.bin を書き込み、2.7 にしたがって、WiFi の設定、シリアル番号をしてください。

※本節で説明する OTA 機能は、本デモ用に作成した USB ストレージとしてのプログラムアップデート機能と併用可能なブートローダを使用しているため、ここで紹介されているブートローダに求められる要件を満たしていません。

2.11.1 Amazon S3 バケットの作成

a. AWS マネジメントコンソールから Amazon S3 アクセスしてください。
b. 「バケット -> 「バケットの作成」を選択します。
c. バケット名(gr-rose-aws-demo-XXX)を入力、バケットのバージョニングを「有効」に設定し、「バケットを作成」を選択します。(XXX は任意の文字)
※以降の説明では、バケット名＝gr-rose-aws-demo としています

2.11.2 OTA Update サービスロールの作成

a. AWS マネジメントコンソールで、AWS IAM にアクセスします。
b. 「ロール」 -> 「ロールの作成」を選択します。
c. 「信頼されたエンティティの種類を選択」で「AWS サービス」を選択します。
d. 「サービスを選択してユースケースを表示」で「IoT」を選択し、「ユースケースの選択」で「IoT」を選択し、「次のステップ」を選択します。
e. そのまま「次のステップ」を選択します。
f. そのまま「次のステップ」を選択します。
g. ロール名（ここでは test_ota_role）を入力し、「ロールの作成」を選択します。

2.11.3 OTA Update サービスロールに OTA Update アクセス許可を付与

a. 「ロール」 ＞ 検索ボックスに作成したロール名（test_ota_role）を入力 ＞ 作成したロールを選択 ＞ 「ポリシーをアタッチします」を選択します。
b. “amazonfreertos”でフィルタをかけ、「AmazonFreeRTSOTAUpdate」を選択し、「ポリシーのアタッチ」を選択します。

2.11.4 OTA Update サービスロールに AWS IAM アクセス許可を付与

a. 「ロール」 ＞ 検索ボックスに作成したロール名（test_ota_role）を入力 ＞ 作成したロールを選択 ＞ 「インラインポリシーの追加」を選択します。

b. 「JSON」タブに切り替え、以下を入力し、「ポリシーの確認」を選択します。

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
            "Effect": "Allow",
            "Action": [
                  "iam:GetRole",
                  "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::<your_account_id>:role/test_ota_role"
      }
    ]
}
```

※<your_account_id>は AWS アカウント ID（12 桁）で、コンソール右上のアカウントタブ（下図赤部分）を参照してください。

c. ポリシー名(ここでは test_ota_iam_policy)を入力し、「ポリシーの作成」を選択します。

2.11.5 OTA Update サービスロールに Amazon S3 アクセス許可を付与

a. 2.11.4.と同様の方法で以下のポリシーを作成します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObjectVersion",
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::gr-rose-aws-demo/*"
            ]
        }
    ]
}
```

b. ポリシー名(ここでは test_ota_s3_policy)を入力し、「ポリシーの作成」を選択します。

2.11.6 OTA ユーザーポリシーの作成

a. 「ユーザー」 ＞ ユーザー選択 ＞ アクセス権限 ＞「アクセス権限の追加」を選択します。

b. 「既存のポリシーを直接アタッチ」 ＞「ポリシーの作成」を選択します。

c. 「JSON」タブに切り替え、以下を入力し、「ポリシーの確認」を選択します。

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:CreateBucket",
                "s3:PutBucketVersioning",
                "s3:GetBucketLocation",
                "s3:GetObjectVersion",
                "acm:ImportCertificate",
                "acm:ListCertificates", "iot:*",
                "iam:ListRoles",
                "freertos:ListHardwarePlatforms",
                "freertos:DescribeHardwarePlatform"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::gr-rose-aws-demo /*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::<your-account-id>:role/test_ota_role"
        }
    ]
}
```

※<your-account-id>は AWS アカウント ID（12 桁）を入力します。

d. ポリシー名を test-ota-policy と入力し、「ポリシーの作成」を選択します。

e. アクセス権限の追加->作成したポリシー(test-ota-policy)を検索・選択し、「次のステップ」 > 「アクセス権限の追加」を選択します。

2.11.7コード署名証明書の作成

a. OpenSSL をインストールしてください。https://www.openssl.org/

b. AWS CLI をインストールしてください。
https://awscli.amazonaws.com/AWSCLIV2.msi (Windows)

c. AWS CLI の設定をします。

i. AWS IAM > 「ユーザー」 > ユーザー選択 > 「認証情報」タグ > 「アクセスキーの作成」を選択します。

ii. CSV ファイルをダウンロードしてください。 ※CSV ファイルは厳重に保管する。

iii. コマンドプロンプトを開き、”aws configure”を入力します。

```
C:¥                >aws configure
AWS Access Key ID [                    ]: ①
AWS Secret Access Key [                    ]: ②
Default region name [ap-northeast-1]: ③
Default output format [text]: ④
```

① CSV ファイルにある”Access key ID”を入力します。

② CSV ファイルにある”Secret access key”を入力します。

③ リージョンを入力します(東京の場合：ap-northeast-1)。

④ “JSON”を入力します。

d. 作業用ディレクトリで、以下の内容の”cert_config.txt”ファイルを作成します。

```
[ req ]
prompt                  = no
distinguished_name = my_dn

[ my_dn ]
commonName = test_signer@amazon.com

[ my_exts ]
keyUsage              = digitalSignature
extendedKeyUsage = codeSigning
```

※test_signer@amazon.com は自身のメールアドレスに変更します。

e. 以下のコマンドを用いてコード署名プライベートキー、コード署名証明書を作成します。

[コード署名プライベートキー]

```
openssl genpkey -algorithm EC -pkeyopt ec_paramgen_curve:P-256 -pkeyopt
ec_param_enc:named_curve -outform PEM -out ecdsasigner.key
```

[コード署名証明書]

```
openssl req -new -x509 -config cert_config.txt -extensions my_exts -nodes -days 365 -key
ecdsasigner.key -out ecdsasigner.crt
```

f. コード署名証明書、プライベートキー、および証明書チェーンを AWS Certificate Manager にインポートします。

```
aws acm import-certificate --certificate fileb://ecdsasigner.crt --private-key fileb://ecdsasigner.key
```

インポートに成功すると、CertificateArn が表示されるのでメモします

g. 作成した証明書(ecdsasigner.crt)をデバイスのソースコードに記述します。
amazon-freertos¥vendors¥renesas¥rose_sdk¥iotgr_sdk¥include¥iotgr_param_config.h

```
#define IOT_DEMO_signingcredentialSIGNING_CERTIFICATE_PEM    ¥
"-----BEGIN CERTIFICATE-----¥n"¥
・・・
"-----END CERTIFICATE-----¥n"
```

2.11.8 Code Signing for AWS IoT に IAM ユーザーアカウントのアクセス許可を付与

a. AWS マネジメントコンソールで、AWS IAM にアクセスします。
b. 「ポリシー」 > 「ポリシーの作成」 > 「JSON」タブで以下を入力 > 「ポリシーの確認」

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "signer:*"
            ],
            "Resource": "*"
        }
    ]
}
```

c. ポリシー名(code_signing_policy)を入力し、「ポリシーの作成」を選択します。
d. 「ユーザー」 > ユーザー選択 > 「アクセス権限の追加」を押します。
e. 「既存のポリシーを直接アタッチ」 > 作成したポリシー(code_signing_policy)を選択 > 「次のステップ」を選択します。
f. 「アクセス権限の追加」を選択します。

2.11.9 OTA 更新ジョブの作成

a. AWS マネジメントコンソールで、AWS IoT コンソールにアクセスします。
b. 「管理」 > 「ジョブ」 > 「ジョブを作成」> 「FreeRTOS OTA 更新ジョブを作成」

を選択します。

c. ジョブ名を入力し、次へを押します。

d. 更新するデバイス(GR-ROSE_XX)を選択して「次へ」を選択します。

e. ファームウェアイメージ転送プロトコルを MQTT とします。

f. コード署名プロファイルを作成します。

g. プロファイル名(profile_ota_grrose)を入力します。

h. デバイスハードウェアプラットフォーム（ESP32-DevKitC）を選択します。

i. 「既存の証明書の選択」を選択し、先程インポートした証明書を選択します。
※先程メモした CertificateArn を選択してください

j. 「デバイスのコード署名証明書のパス名」に certs/ecdsasigner.crt を入力します。

k. 「作成」を押します。

l. 更新用ファームウェアを準備します
amazon-freertos¥demos¥include¥ aws_application_version.h の以下赤枠部分を”3”に変更し、ビルドを実行してください。

```
 * FreeRTOS V202012.00

#ifndef _AWS_APPLICATION_VERSION_H_
#define _AWS_APPLICATION_VERSION_H_

#include "iot_appversion32.h"
extern const AppVersion32_t xAppFirmwareVersion;

#define APP_VERSION_MAJOR    0
#define APP_VERSION_MINOR    9
#define APP_VERSION_BUILD    2

#endif
```

m. 2.11.1.で作成したバケットに更新用ファームウェアをアップロードします。
「新しいファイルをアップロードします」を選択した状態で、k で作成した aws_demos.bin を選択してください。

n. 「デバイス上のファイルパス名」に、certs/ecdsasigner.crt 入力します。

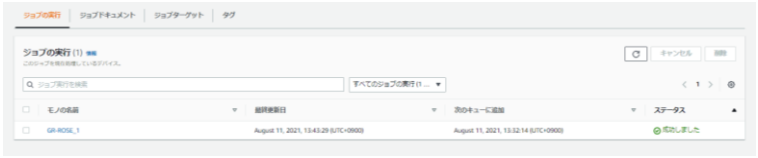
o. IAM ロールで test_ota_role を選択し「次へ」を選択します。

p. ジョブの作成を押します

2.11.10 OTA Update の実行

OTA ジョブ作成後、OTA のステータスは、ジョブ画面で参照することができます。

2.11.9.で入力したジョブ名をクリックすると、ジョブのステータスを確認することができます



以上