Renesas Synergy™ プラットフォーム

Synergy MQTT/TLS Azure クラウド接続ソリューション

R11AN0337JU0102
Rev.1.02
2019.05.07

本資料は英語版を翻訳した参考資料です。内容に相違がある場合には英語版を優先します。資料によっては英語版のバージョンが更新され、内容が変わっている場合があります。日本語版は、参考用としてご使用のうえ、最新および正式な内容については英語版のドキュメントを参照ください。

要旨（Introduction）

このアプリケーションノート（application note）は、IoT (モノのインターネット) Cloud 接続ソリューション（connectivity solution）について一般的に説明するとともに、IoT クラウドプロバイダ（IoT Cloud provider）である Microsoft Azure について紹介します。この中では、Synergy MQTT/TLS モジュール（module）、その機能（features）、および動作フローシーケンス（operational flow sequence）として初期化/データフロー（Initialization/Data flow）について説明します。パッケージ（package）に付属しているサンプルアプリケーション（application example）は、Microsoft Azure を使用します。本アプリケーションノートは、初めて Microsoft Azure を使用するユーザに対して、AWS IoT Core プラットフォームを設定して、サンプルアプリケーションデモを実行する方法を段階的に詳しく紹介します。

このアプリケーションノートによってユーザは、Synergy MQTT/TLS モジュールを利用した製品設計が効率的に行えるようになります。このアプリケーションノートをマスターすれば、ユーザは開発中の製品に MQTT/TLS モジュールを追加して、ターゲットアプリケーション向けの正しい設定が行え、付属のサンプルアプリケーションコードを参照してコードが作成できるようになります。API のさらに詳細な説明と、このモジュールのより高度な使用方法に関する他のアプリケーションプロジェクトの参考資料が『Synergy ソフトウェアパッケージ (SSP) ユーザーズマニュアル』に掲載されていますので (第 6 章を参照)、より複雑な設計を実施する場合に活用いただけます。

現在、Synergy MQTT/TLS 接続ソリューションは、Microsoft Azure IoT を使用して PK-S5D9 キットあるいは AE-CLOUD1 および AE-CLOUD2 キット上で実装およびテストされています。他の Synergy キットや他の IoT クラウドプロバイダは、今後のリリースで対応する予定です。

必須リソース（Required Resources）

MQTT/TLS サンプルアプリケーションをビルドして実行するには、以下のリソースが必要です。

開発ツールとソフトウェア

- e² studio ISDE v6.2.1 またはそれ以降、もしくは IAR Embedded Workbench® for Renesas Synergy™ v8.23.x またはそれ以降
  https://www.renesas.com/jp/ja/products/synergy/software/tools.html
- Synergy Software Package (SSP) 1.5.3 またはそれ以降
  https://www.renesas.com/jp/ja/products/synergy/software/ssp.html
- Synergy Standalone Configurator (SSC) 6_2_1 またはそれ以降
- https://www.renesas.com/jp/ja/products/synergy/software/tools/renesas-ssc.html
- Renesas Synergy™ USB CDC ドライバ
  https://www.renesas.com/jp/ja/products/synergy/software/add-ons/usb-cdc-drivers.html

ハードウェア

- Renesas Synergy™ PK-S5D9 キット
- AE-CLOUD1 キット（Wi-Fi ボードが付属）
- AE-CLOUD2 キット (ピラーボード（Pillar board）、Wi-Fi ボード、BG96 セルラーシールド（Cellular shield）が付属) (https://www.renesas.com/jp/ja/products/synergy/hardware/kits/ae-cloud2.html)
  Renesas Synergy™ サンプルアプリケーションキット (AE-wifi1)
  (注：AE-wifi1 はサポートを終了しています)
- Windows® 7 または 10、Tera Term コンソールまたは類似のアプリケーション、インストール済みの Web ブラウザ (Google Chrome、Internet Explorer、Microsoft Edge、Mozilla Firefox、Safari) が動作している PC
- Micro USB ケーブル
- イーサネットケーブル

前提条件と対象ユーザ（Prerequisites and Intended Audience）

このアプリケーションノートは、ユーザが Renesas e² studio ISDE と Synergy ソフトウェアパッケージ (SSP) の使用経験があることを前提としています。ユーザに使用経験のない場合は、このアプリケーションノートの手順を実行する前に、『SSP ユーザーズマニュアル』の手順に従い「Blinky」プロジェクトをビルドして実行してください。それにより、e² studio あるいは IAR と SSP の使用に慣れ、ボードへのデバッグ接続が適切に機能していることを確認できるようになります。さらに、このアプリケーションノートは、MQTT/TLS とその通信プロトコルに関する知識があることも前提としています。

対象ユーザは、Renesas Synergy™ S5 または S7 MCU シリーズと MQTT/TLS モジュールを使用し、アプリケーションを開発することを希望しているユーザです。

目次

1. クラウド接続の要旨（Introduction to Cloud Connectivity）

1.1 概要（Overview）

IoT (モノのインターネット) は、センサ (sensor) やスマートフォン (smart-phone) などの日常的に利用される機器を World Wide Web に接続するために使用されている広範囲な各種のテクノロジーで形成されています。IoT デバイスは、インテリジェントな方法で相互にリンクし、モノ (機械、デバイス) と人の間、およびモノとモノの間 (機械相互間、M2M) で通信を行う新しい手法を実現します。

これらのデバイスまたはモノは、インターネットに接続します。これらデバイスがセンサを使用して周囲の環境から収集した情報を提供すると同時に他のシステムはこの情報にアクセスでき、さらにアクチュエータを使用して他に働きかけることができます。このプロセスで、IoT デバイスは大量のデータを生成し、クラウドコンピューティングは生成されたデータを伝達するための経路を提供することで、データを伝送することができます。

1.2 主要コンポーネント（Major Components）

IoT クラウド接続ソリューションは、以下の主要コンポーネントで形成されています。

1. デバイスまたはセンサ（Devices or Sensors）
2. ゲートウェイ（Gateway）
3. IoT クラウドサービス（IoT Cloud services）
4. エンドユーザ向けのアプリケーション/システム（End user application/system）

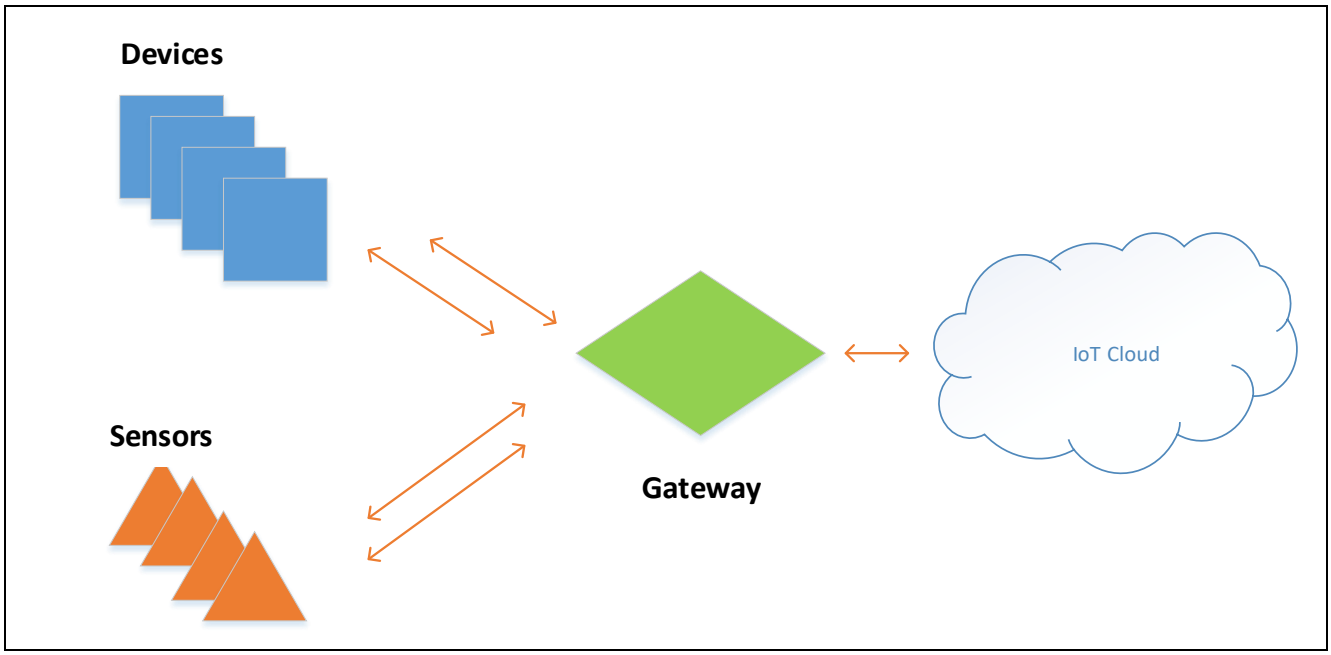

図 1　IoT クラウド接続のアーキテクチャ

デバイスまたはセンサ（Devices or Sensors）

デバイスは、ハードウェアとソフトウェアで形成されており、外部と直接通信します。各デバイスはネットワークに接続し、デバイスどうし、または中心となるアプリケーションと通信を行います。デバイスは直接的または間接的にインターネットと接続できます。

ゲートウェイ（Gateway）

ゲートウェイ（gateway）によって、インターネットに直接接続されていないデバイスもクラウドサービスを利用することができます。各デバイスからのデータは、クラウドプラットフォームに送信され、そこで他のデバイスから到着したデータや、他の業務処理データとともに処理され、組み合わされます。ほとんど一般的な通信ゲートウェイ (communication gateway) は、Wi-Fi、イーサネット、セルラーなどの、複数の通信テクノロジーをサポートします。

IoT クラウド（IoT Cloud）

多くの IoT デバイスは大量のデータを生成します。これらデバイスの管理、情報処理とその活用のためには、効率的、スケーラブルかつ低コストな方法が必要です。データ、特にビッグデータ（big data）の保存、処理、分析を行う場合、クラウドを上回る手段を見つけるのは困難です。

1.3　クラウドプロバイダの概要（Cloud Provider Overview）

1.3.1　Microsoft Azure IoT ソリューション（Microsoft Azure IoT Solution）

Microsoft のエンドツーエンド (End-To-End：完結型) IoT プラットフォームによる完全な IoT の提供によって、多くのエンタープライズ (企業) が IoT ソリューションを迅速かつ効率的に構築、実現することができます。Azure IoT ソリューションは、Azure IoT Suite と Azure IoT Hub で構成されており、クラウドの全ての能力を活用することができます。エンタープライズ向けのデータと開発者によって、大規模な IoT サービス、リッチデータとその分析、それらの深いインテグレーション（統合）を実現できます。IoT Hub を使用したカスタムソリューションの構築や、Azure IoT Suite (Azure IoT Hub 含む)が持つ事前設定済みソリューションを利用して迅速な開発を行うことができます。

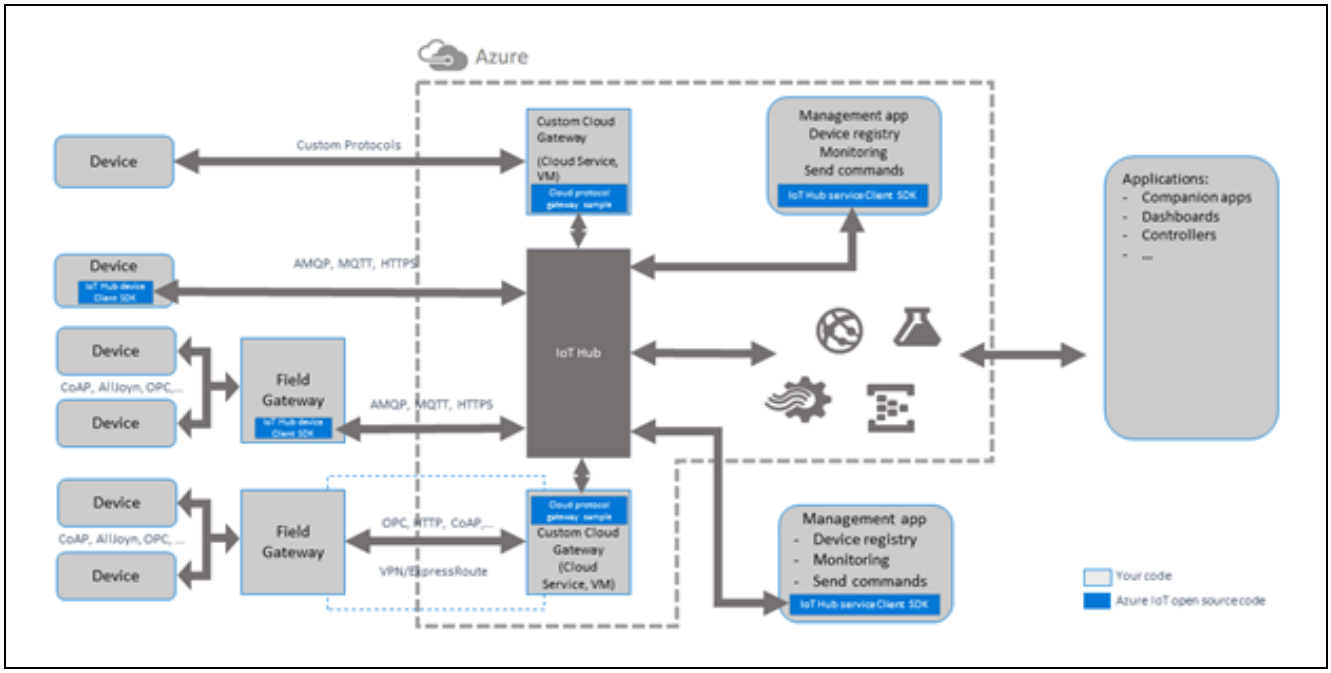

図 2　Microsoft Azure IoT クラウドによるソリューション

IoT ハブ（IoT Hub）：IoT Hub は、数百万台の IoT デバイス （毎月数十億件のメッセージ送受信を行う）との接続、プロビジョニング、管理を実行するための容易で安全な方法を提供します。IoT Hub は、デバイスと Cloud 上のソリューションの間に位置するブリッジ（bridge）であり、ソリューションがデータの保存や分析、データに基づく処理をリアルタイムで実行できるようにします。IoT Hub は IoT の分野で既に広く使用されている MQTT、HTTPS、AMQPS のようなオープンプロトコル（open protocol）を利用して、デバイスからクラウド、クラウドからデバイスへのセキュアで信頼性の高い双方向通信を実現します。

フィールドゲートウェイ（Field Gateway）：Azure IoT プロトコルゲートウェイ（protocol gateway）は、大規模で双方向のデバイス - IoT Hub 間通信を行う目的で設計されたプロトコル対応フレームワークです。プロトコルゲートウェイは、特定のプロトコルを使用するデバイス接続を受け入れるパススルーコンポーネント（pass-through component）です。プロトコルゲートウェイは、AMQP 1.0 を使用してトラフィック（traffic）を IoT Hub にブリッジ接続します。

Azure IoT プロトコルゲートウェイは、必要に応じて MQTT プロトコルの動作をカスタマイズできる MQTT プロトコルアダプタ（protocol adapter）を搭載しています。IoT Hub は MQTT v3.1.1 プロトコルをサポートしています。このため、MQTT プロトコルアダプタの使用を考慮する必要があるのは、プロトコルのカスタマイズが必要な場合や、機能を追加するための特定の要件が必要な場合のみです。

カスタムクラウドゲートウェイ（Custom Cloud Gateway）: カスタムクラウドゲートウェイ（custom cloud gateway）を使用すると、クラウドゲートウェイの通信エンドポイント（communication endpoint）に到達する前に、プロトコルの適応やいくつかの形式のカスタム処理（custom processing）を実施できます。このようなカスタム処理には、デバイス（もしくはフィールドゲートウェイ）によって要求される個別のプロトコル実装も含まれます。この個別のプロトコル実装は、処理のためにメッセージをクラウドゲートウェイに転送し、クラウドゲートウェイからデバイスにコマンドやコントロールメッセージを返送する際に必要です。

1.3.1.1　主な機能（Key Features）

(1)　デバイスのセキュリティ

IoT デバイスの展開（deploy）と管理を行う際に、セキュリティは極めて重要です。このため、IoT Hub は以下のセキュリティ機能を提供しています。

- デバイスと Azure IoT Hub の間、またはゲートウェイと Azure IoT Hub の間の通信パスは、業界標準の TLS (Transport Layer Security、トランスポートレイヤセキュリティ) と、X.509 規格で認証された Azure IoT Hub によってセキュリティを確保しています。
- 求めていない着信接続（unsolicited inbound connection）からデバイスを保護するために、Azure IoT Hub 側からデバイスへ接続を開くことはありません。常にデバイス側から接続を開始（initiate）します。
- Azure IoT Hub はデバイスへのメッセージを、耐久性を持って保存して、デバイスからの接続を待ちます。このため、これらのコマンドは Azure IoT Hub に 2 日間保存されます。消費電力と接続性のため、デバイスは間欠的に Azure IoT Hub と接続して、コマンドを受け取ります。さらに Azure IoT Hub は各デバイスのキュー（per-device queue）を保持します。

(2)　デバイスごとの鍵認証（key authentication）

以下の図に、セキュリティトークン（security token）を使用する IoT Hub の認証を示します。

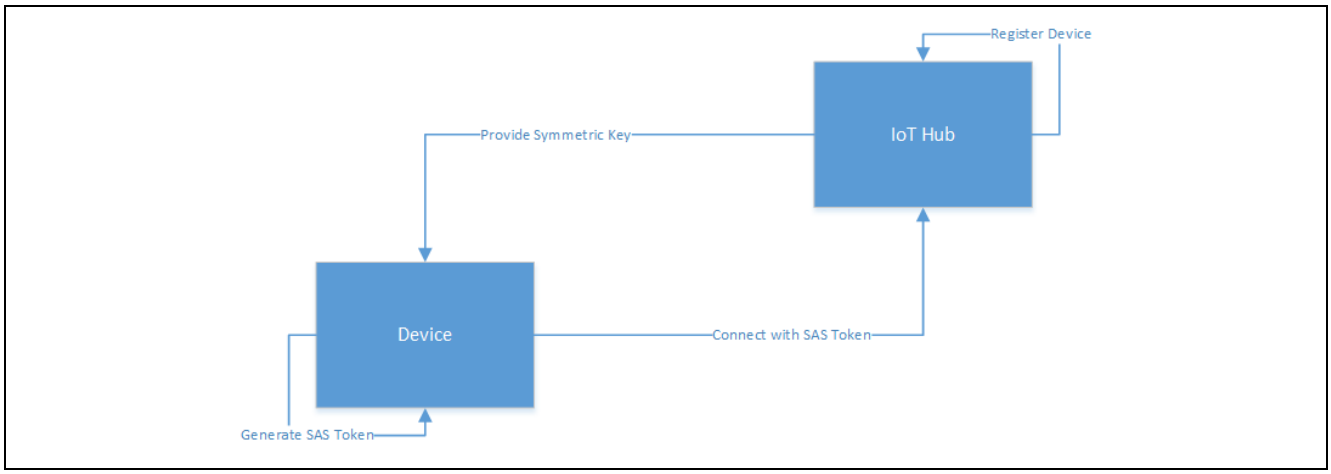

図 3　セキュリティトークンを使用する認証

- デバイスは、デバイスエンドポイント、デバイス ID、および) プライマリキー（primary key : デバイスを IoT Hub に追加したときに生成された情報の一部）を使用して、共有アクセスシグネチャ（shared access signature）(SAS) トークンを準備します。
- IoT Hub に接続するときに、デバイスは MQTT CONNECT メッセージ（message）内でこの SAS トークンをパスワード（password）として提示します。ユーザ名の内容は、デバイスエンドポイント、デバイス名、付加的な Azure 定義文字列（additional Azure defined string）を組み合わせたものです。
- IoT Hub が SAS トークンを検証し、デバイスを登録すると、接続が確立されます。
- IoT ハブはデータ暗号化用の対称鍵（symmetric key）を提供します。
- SAS トークンが期限切れの場合、接続は閉じられます。

1.4　MQTT プロトコルの概要（MQTT Protocol Overview）

MQTT は、「Message Queuing Telemetry Transport」(メッセージキューイング遠隔測定トランスポート) の略称です。MQTT は、クライアントサーバ発行サブスクライブ（publish-subscribe）によるメッセージングトランスポートプロトコル（messaging transport protocol）です。きわめて軽量、オープンでシンプルなメッセージングプロトコルであり、低い転送レート（low-bandwidth）、大きな遅延時間（high-latency）、または信頼性の低いネットワーク（unreliable networks）のような使用上の制約の大きいデバイスに対応できるように設計されています。これらの特性を活用して、制約の大きい環境（必要なコードフットプリント（code footprint）が小規模、ネットワーク帯域幅（network bandwidth）が限定された M2M (machine to Machine：機械相互間)、IoT 用途での通信など）での活用に最適です。

MQTT クライアントは、ブローカー（broker）経由で他のクライアントに情報を発行することができます。あるクライアントが特定のトピックに関心がある場合、そのクライアントはブローカー経由でそのトピックにサブスクライブする (申し込む) ことができます。ブローカーはクライアントの認証と承認を担当し、特定のトピックにサブスクライブしたクライアントに対して、発行されたメッセージを配信します。この発行（publisher）/サブスクライバモデルで、複数のクライアントが同じトピックに属するデータを発行することもできます。クライアントがその同じトピックにサブスクライブした場合、そのトピックに関して発行されたメッセージを受け取ります。

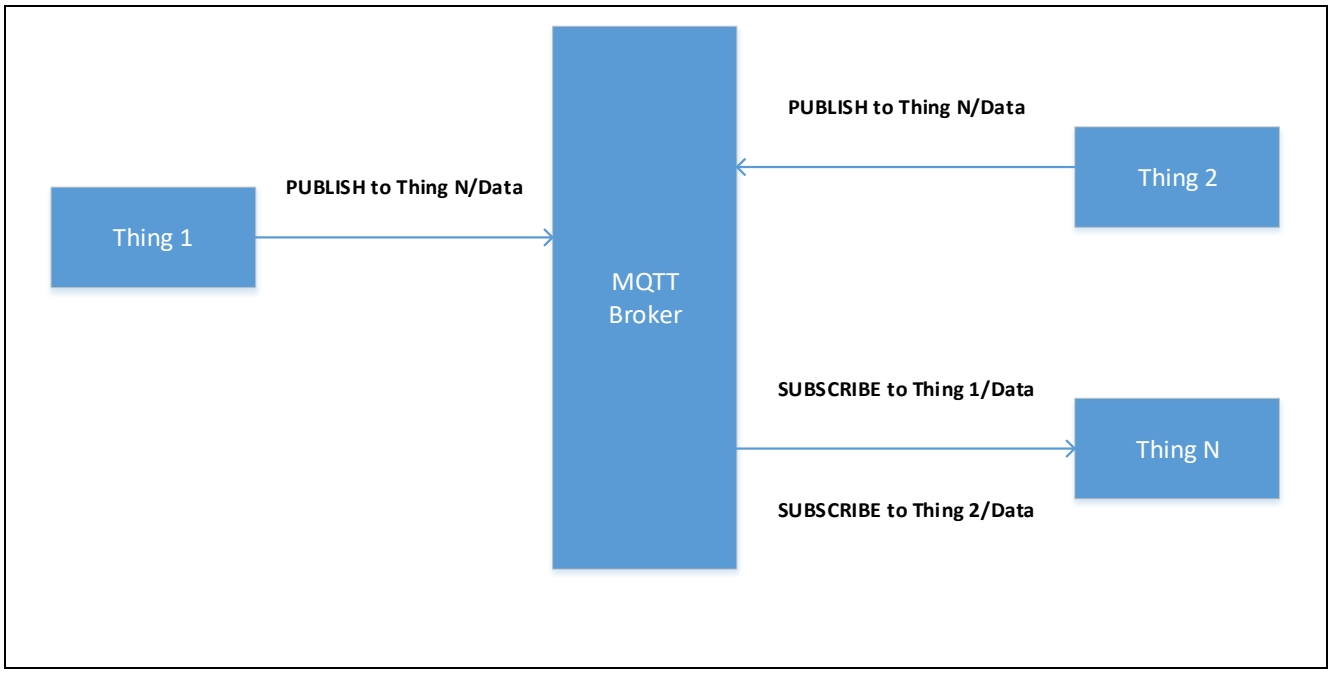

図 4　MQTT クライアントの発行/サブスクライブモデル

このモデルでは、発行者（publisher）とサブスクライバの間に直接の接続はありません。発行/サブスクライブシステムの問題に対応するために、MQTT は一般的に、QoS (サービス品質) の複数のレベルを使用します。MQTT には、以下の 3 つの QoS レベルがあります。

- 最大 1 回 (0)（At most once (0)）
- 最小 1 回 (1)（At least once (1)）
- 正確に 1 回 (2)（Exactly once (2)）

最大 1 回 (0)（At most once (0)）

メッセージが受信側によって受信確認 (Ack) されることはなく、送信側によって保存や再配信されることもありません。

最小 1 回 (1)（At least once (1)）

メッセージを受信側に最小 1 回配信することが保証されます。ただし、このメッセージは複数回の配信が可能です。送信側は、受信側からの PUBACK コマンド形式の受信確認 (Ack) を受け取るまで、メッセージを保存します。

正確に 1 回 (2)（Exactly once (2)）

メッセージを通信先に正確に 1 回のみ配信することを保証します。これは最も安全で、最も低速な QoS レベルです。送信側と受信側の間で伝送と返信による 2 つのフローを通じて、保証がなされます。

1.5　TLS プロトコルの概要（MQTT Protocol Overview）

トランスポートレイヤセキュリティ (TLS) プロトコルとその前身であるセキュアソケットレイヤ (SSL) は、コンピュータネットワーク経由でセキュアな通信を実現する暗号化プロトコルです。

TLS/SSL プロトコルは、2 つの通信アプリケーションの間でプライバシーと信頼性を確保します。以下の基本的なプロパティがあります。

暗号化（Encryption）：2 つの通信アプリケーションの間で交換されるメッセージは暗号化され、接続時のプライバシーを保証します。データの暗号化に AES (Advanced Encryption Standard、高度暗号化規格) を使用します。

認証（Authentication）：証明書を使用して通信先の識別情報を検証するメカニズムです。

完全性（Integrity）：メッセージの改ざん（tampering）や改変（forgery）が実施されたときにそのことを検出するメカニズムで、接続の信頼性を保証します。SHA (Secure Hash Algorithm、セキュアハッシュアルゴリズム) のような MAC(Message Authentication Code、メッセージ認証コード) を使用し、メッセージの完全性を保証します。

TLS/SSL は TCP を使用して、HTTP や MQTT のようなアプリケーションレイヤプロトコル（application layer protocol）に対して通信のセキュリティを確保します。

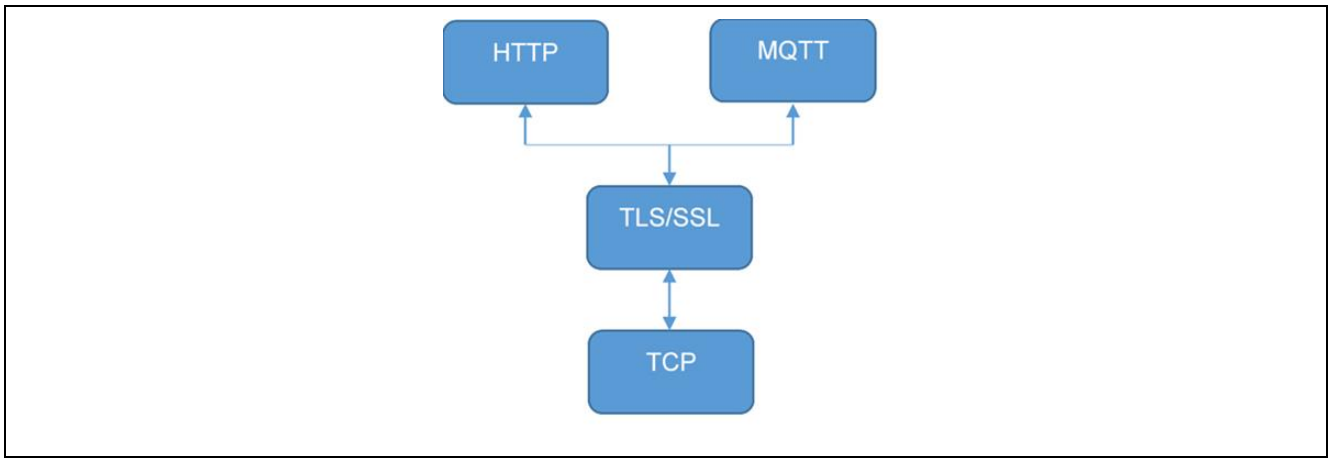

図 5　SSL/TLS の階層

1.5.1　デバイス証明書と鍵（Device Certificates and Keys）

この章では、デバイス証明書（device certificate）、公開鍵と秘密鍵、およびそれらを生成する方法について説明します。

1.5.1.1　デバイス証明書（Device Certificates）

IoT デバイスの展開（deploy）と管理を行う際、セキュリティは重要な懸念事項（critical concern）になります。通常、IoT デバイスはクラウドとの通信を実行できるようになる前に、識別情報（identity）を必要とします。デジタル証明書は、TLS でリモートホスト（remote host）を認証するための最も一般的な方法です。デジタル証明書とは、デバイスに関する識別情報を提供するための特定の書式の文書です。

TLS は通常、X.509 と呼ばれる形式を使用します。これは ITU-T (国際電気通信連合、電気通信標準化部門) が策定した規格です。ただし、TLS 通信を行う複数のホストが合意する場合は、他の形式の証明書を使用することもできます。X.509 は、証明書に関する具体的な形式とさまざまなエンコーディング方式を定義しており、これらを使用してデジタル文書を作成することができます。TLS で使用する大部分の X.509 証明書は、別の通信標準規格（telecomunication standard ）である ASN.1 の派生版を使用します。ASN.1 にはさまざまなデジタルエンコーディングが使用されていますが、TLS 証明書で最も一般的なエンコーディングは DER (Distinguished Encoding Rules) 規格です。DER は ASN.1 BER (Basic Encoding Rules) を簡略化したサブセットであり、あいまいさを排除し、解析を容易にすることを目的として策定されています。

DER 形式のバイナリ証明書が実際の TLS プロトコルで使用されていますが、これらは複数の種類のエンコーディングを使用して生成および保存でき、.pem、.crt、.p12 のようなファイル拡張子を割り当てています。最も一般的な代替の証明書エンコーディングは、PEM です。PEM (Privacy-Enhanced Mail、プライバシー強化メールに由来) 形式は、DER エンコーディングに対応する、Base64 エンコーディングバージョンです。

開発するアプリケーションによっては、ユーザ自前の証明書を生成することもできます。通常、そのような証明書は、メーカーや政府機関から提供されたもの、または商用の認証局から購入した証明書です。

1.5.1.2　デバイスへの証明書のロード（Loading Certificates onto your Device）

NetX™ Secure アプリケーションでデジタル証明書を使用するには、最初に証明書をバイナリ DER 形式に変換（convert）し、オプションで関連する秘密鍵をバイナリ形式に変換します。通常、PKCS#1 形式で、DER エンコーディングされた RSA 鍵を使用します。変換後、証明書と秘密鍵をデバイスにロードする方法は以下のオプションから選択できます。フラッシュベースのファイルシステムを使用するか、データから C アレイ（C array）を生成します (Linux® の「xxd」のようなツールで、「-i」オプションを指定)。そしてコンパイルにより、証明書と鍵を定数データとしてアプリケーションに組み込みます。

証明書をデバイスにロードした後、TLS API を使用して証明書を TLS セッションに関連付けることができます。

1.5.1.3　自己署名証明書の生成（Generating Self-Signed Certificates）

テストの目的で、自己署名証明書（self-signed certificate）を生成する方法を選択することもできます。このような証明書を生成するコマンドは、以下のとおりです。

```
openssl req -x509 -newkey rsa:2048 -keyout private.key -out cert.pem -days 365 -
nodes -subj "/C=US/ST=Oregon/L=Portland/O=Company
Name/OU=Org/CN=www.example.com"
```

このコマンドで、自己署名証明書である www.example.com が生成されます。証明書ファイルは cert.pem、秘密鍵ファイルは private.key です。「www.example.com」を「localhost」に置き換えることで、ローカルホストに対応する証明書も生成できます。この場合、インストールスクリプトの最初の引数として「localhost」を指定します。

1.5.2　デバイスのセキュリティに関する推奨事項（Device Security Recommendations）

セキュリティに関する以下の推奨事項は、Cloud IoT Core によって強制されるものではありませんが、デバイスと接続の安全を確保するために有効です。

- 秘密鍵は機密情報として取り扱う。
- IoT クラウドと通信する場合は TLS 1.2 を使用し、ルート認証局（root certificate authorities）を使用して、サーバの証明書が有効であることを確認します。
- 各デバイスは、一意（ユニーク）な公開鍵/秘密鍵ペアを使用する必要があります。仮に複数のデバイスで単一の鍵を共有していて、それらのデバイスの一つが攻撃にさらされた場合、攻撃者は単一の鍵で設定されたすべてのデバイス対してなりすますことができるようになります。
- 公開鍵を Cloud IoT Core に登録するとき、セキュアな状態を維持します。攻撃者が公開鍵を改ざんすることに成功し、プロビジョニング事業者（provisioner）を欺いて公開鍵を入れ替え、誤った公開鍵を登録した場合、それ以降、攻撃者はデバイスの代わりに認証を実施できるようになります。
- Cloud IoT Core に対してデバイスを認証するために使用した鍵ペアは、他の目的や他のプロトコルに使用しないでください。
- 鍵をセキュアに保存するデバイスの能力によっては、鍵ペアを定期的に変更（rotate）するようにしてください。現実的には、デバイスをリセットする場合は、すべての鍵を破棄（discard）してください。
- デバイスでオペレーティングシステムを実行している場合、OS のアップデートはセキュア（secure）な方法で実施する必要があります。Android Things は、セキュアなアップデートを実施するためのサービスを提供しています。オペレーティングシステムを使用していないデバイスの場合、展開後にセキュリティの脆弱性が発見された場合、セキュアな方法でデバイスをアップデートしてください。
-

2. Synergy MQTT/TLS のクラウドソリューション（Synergy MQTT/TLS Cloud Solution）

2.1 MQTT クライアントの概要（MQTT Client Overview）

NetX Duo MQTT クライアントモジュールは、MQTT (Message Queuing Telemetry Transport、メッセージキューイング遠隔測定トランスポート) プロトコルベースのクライアントに対応する高水準の API を提供します。MQTT プロトコルは、TCP/IP の上位で動作するので、MQTT クライアントは NetX Duo IP および NetX Duo Packet プールの上位で実装されています。NetX Duo IP は自らを、イーサネット、Wi-Fi、セルラーなど、適切なリンクレイヤに接続します。

NetX Duo MQTT クライアントモジュールは、通常モードまたはセキュアモードで使用できます。通常モードでは、MQTT クライアントとブローカーの間の通信はセキュアではありません。セキュアモードでは、MQTT クライアントとブローカーの間の通信は、TLS プロトコルを使用してセキュアになります。

2.2 設計に関する検討事項（Design Considerations）

- デフォルトでは、MQTT クライアントは TLS を使用せず、MQTT クライアントとブローカーの間の通信はセキュアではありません。
- Synergy MQTT クライアントは、NetX Duo TLS セッションブロックを追加しません。NetX Duo TLS 共通ブロック（common block）のみを追加します。このブロックは、NetX secure の共通プロパティをセキュアの定義と制御をおこないます。
- TLS セッションの作成、セキュリティパラメータの設定、`nxd_mqtt_client_secure_connect ()` API によって提供される TLS セットアップコールバックの際に関連する証明書を手動でロードする作業は、ユーザ/アプリケーションコード側で対応する必要があります。

2.2.1 サポート対象の機能（Supported Features）

NetX Duo MQTT クライアントは、以下の機能をサポートしています。

- 2014 年 10 月 29 日の OASIS MQTT バージョン 3.1.1 に準拠しています。この仕様は、http://mqtt.org/ に掲載されています。
- SSP 配下で NetX Secure を使用して通信をセキュアにするかどうかの目的で、TLS を有効/無効にするオプションを提供します。
- QoS をサポートし、メッセージを発行する際に選択可能な複数のレベルを選択する機能を提供します。
- 受信したメッセージを内部でバッファに保存し、キューを維持します。
- 新しいメッセージを受信したときにコールバックを登録するメカニズムを提供します。
- ブローカーとの接続を終了したときにコールバックを登録するメカニズムを提供します。

2.2.2　動作のフローシーケンス（Operational Flow Sequence）

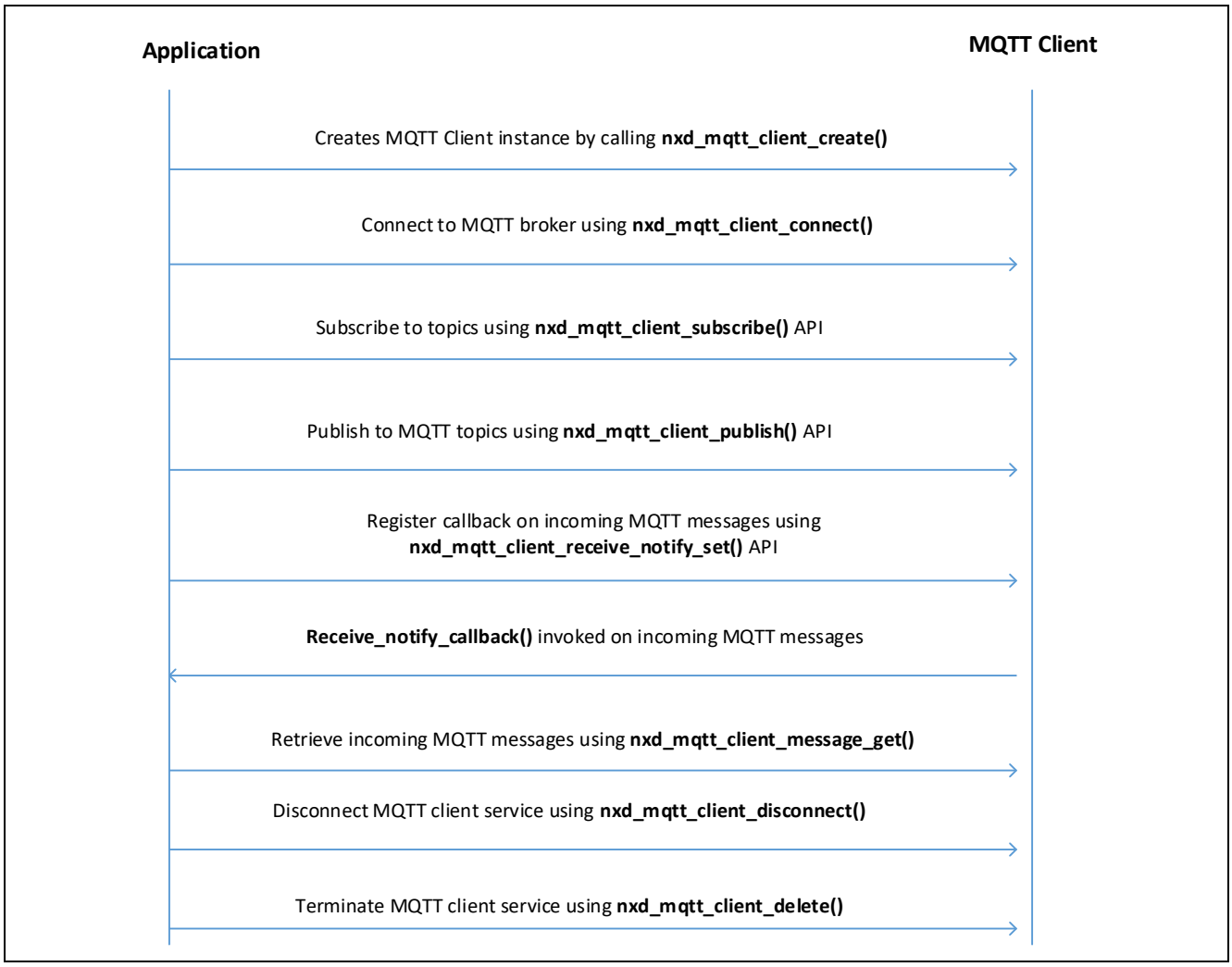

図 6　Synergy MQTT クライアントのフローシーケンス

2.3　TLS セッションの概要（TLS Session Overview）

NetX Duo TLS セッションモジュールは、TLS プロトコルベースのクライアントに対応する高水準の API を提供します。この API は SCE (Synergy Crypto Engine、Synergy 暗号化エンジン) が提供するサービスを使用し、ハードウェアアクセラレーションによる暗号化と復号化を実施します。

NetX Duo TLS セッションモジュールは、RFC 2246 (バージョン 1.0) と 5246 (バージョン 1.2) の規定に従って SSL (セキュアソケットレイヤ) とその後継である TLS プロトコルを実装する、Express Logic の NetX Secure をベースとしています。また、NetX Secure は基本的な X.509 (RFC 5280) 形式に対応するルーチンも搭載しています。NetX Secure は、プロジェクトで ThreadX RTOS を使用するアプリケーションを想定しています。

2.3.1　設計に関する検討事項（Design Considerations）

- NetX Secure TLS は、着信したサーバ証明書に対して基本パス検証（basic path certificate）のみを実行します。基本パス検証が完了した時点で、TLS はそのアプリケーションが提供する証明書検証コールバックを起動します。
- 証明書に対しる追加検証の実行は、アプリケーション側で対応する必要があります。
  追加の検証を容易にするために、NetX Secure は共通の検証動作を目的とした X.509 ルーチンを提供しています。この中には、DNS 検証機能や、CRL (Certificate Revocation List、証明書失効リスト) の確認機能があります。
- ソフトウェアベースの暗号化は、プロセッサに負荷がかかります。
  NetX Secure のソフトウェアベースの暗号化ルーチンは性能最適化済みですが、ターゲットプロセッサの能力によっては、非常に長時間の動作が発生することがあります。ハードウェアベースの暗号化機能が使用できる場合、NetX Secure の TLS 性能を最適化するためにその機能を使用してください。
- 組み込みデバイスの性質上、一部のアプリケーションは最大 TLS レコードサイズである 16 KB をサポートするためのリソースを持たない可能性があります。
  NetX Secure は、十分なリソースが使用できるデバイスで、16 KB レコードを処理できます。

2.3.2　サポート対象の機能（Supported Features）

- RFC 2246 The TLS Protocol Version 1.0 (TLS プロトコルバージョン 1.0)
- RFC 5246 The Transport Layer Security (TLS) Protocol Version 1.2 (トランスポートレイヤセキュリティ (TLS) プロトコルバージョン 1.2)
- RFC 5280 X.509 PKI Certificates (v3) (X.509 PKI 証明書 (v3))
- RFC 3268 Advanced Encryption Standard (AES) Cipher suites for Transport Layer Security (TLS) (TLS 向け高度暗号化規格 (AES) 暗号化スイート)
- RFC 3447 Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1 (公開鍵暗号化規格 (PKCS) #1: RSA 暗号化仕様バージョン 2.1)
- RFC 2104 HMAC: Keyed-Hashing for Message Authentication (HMAC: メッセージ認証用の鍵付きハッシュ)
- RFC 6234 US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF) (セキュアハッシュアルゴリズム (SHA および SHA ベースの HMAC と HKDF))
- RFC 4279 Pre-Shared Key Cipher suites for TLS (TLS 用事前共有鍵暗号化スイート)

2.3.3　動作のフローシーケンス（Operational Flow Sequence）

この章では、TLS ハンドシェイク動作シーケンス（handshake operational sequence）について説明します。

2.3.3.1　TLS ハンドシェイク

以下の図に、TLS サーバとクライアントの間の代表的な TLS ハンドシェイクを示します。

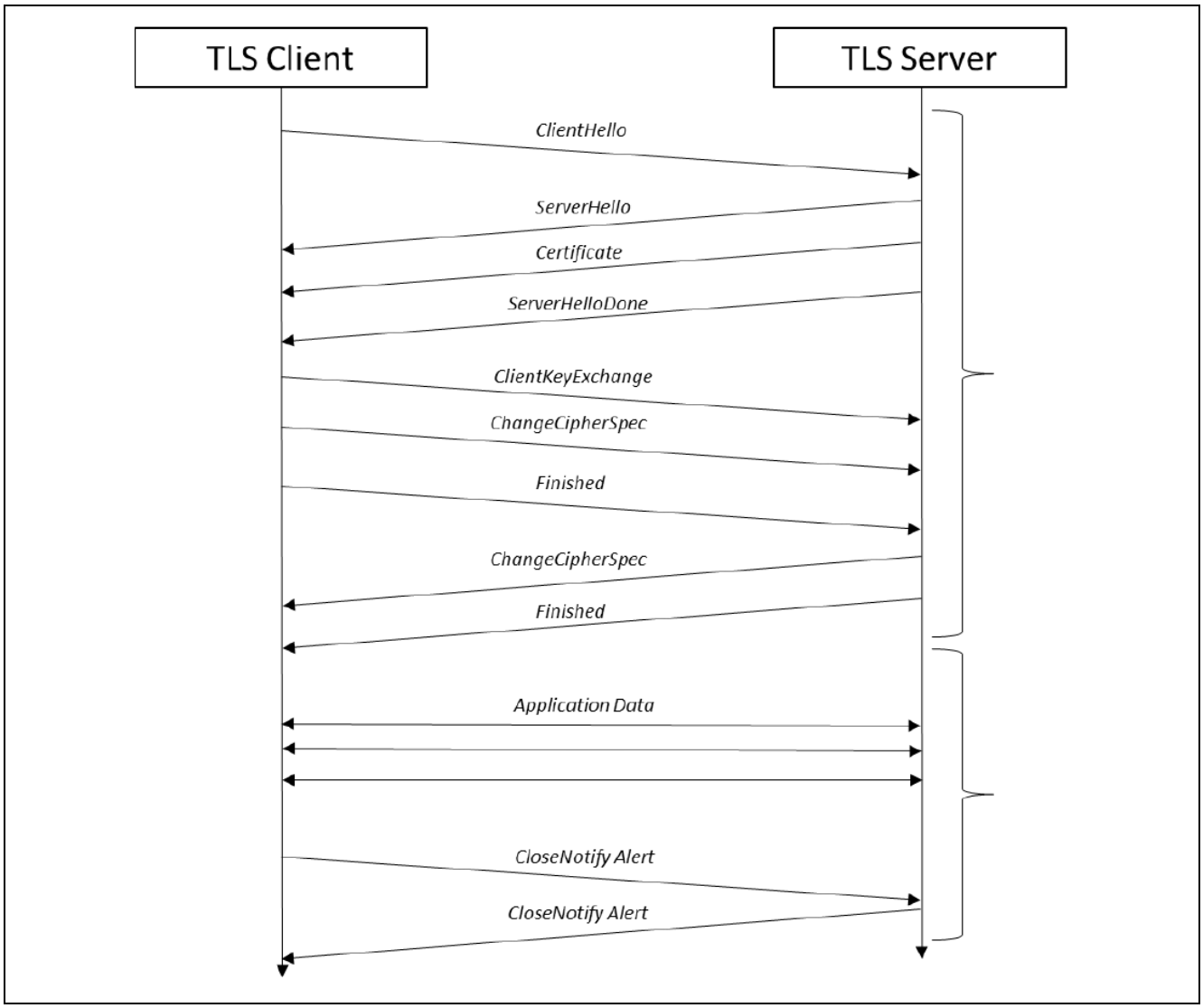

図 7　TLS ハンドシェイク

- TLS ハンドシェイクは、TLS クライアントが “TLS セッションの開始を希望していること”を示す **ClientHello** メッセージを TLS サーバに送信した時点で開始されます。
- このメッセージは、クライアントがそのセッションで使用する暗号化に関する情報や、セッション鍵を生成するために使用する情報を格納しています。
- TLS サーバは **ClientHello** に対して、クライアントから提供された暗号化オプションに基づく選択肢を示す **ServerHello** メッセージを返す形で応答します。
- その後に、クライアントがサーバの識別情報を認証できるように、サーバが自らの識別情報を提供する Certificate (証明書) メッセージが続きます。
- 最後にサーバは、これ以上サーバから送信するメッセージがないことを示す **ServerHelloDone** メッセージを送信します。

- クライアントがサーバのメッセージすべてを受信した時点で、クライアントはセッション鍵を生成するのに十分な情報を入手しました。TLS は、プリマスターシークレット (Pre-Master Secret) という、共有のランダムデータビットを生成する方法でセッション鍵の生成を行います。この鍵は固定長で、暗号化が有効になった後、必要な鍵すべてを生成するためのシード (乱数の生成源) として使用します。
- プリマスターシークレットは、一連の Hello メッセージで指定した公開鍵アルゴリズム（RSA など）と、サーバが自らの証明書で提供した公開鍵を組み合わせる形で暗号化されます。
- 暗号化されたプリマスターシークレットは、**clientKeyExchange** メッセージの一部としてサーバに送信されます。サーバは **ClientKeyExchange** メッセージを受信した時点で自らの秘密鍵を使用してプリマスターシークレットの暗号を解除し、次に TLS クライアントと並行してセッション鍵の生成に進みます。
- Hello メッセージで選択された秘密鍵アルゴリズム (AES など) を使用してセッション鍵が生成されると、これ以降のメッセージすべてを暗号化することができます。暗号化されていない最後のメッセージは、それ以降のすべてのメッセージを暗号化することを示すためにクライアントとサーバの両者が送信する **ChangeCipherSpec** です。
- 暗号化された最初のメッセージは、TLS ハンドシェイクの最後のメッセージで、クライアントとサーバの両者が送信する **Finished** です。このメッセージは、送受信したすべてのハンドシェイクメッセージのハッシュを格納しています。このハッシュを使用して、ハンドシェイクに使用した全てのメッセージにおいて改ざんや破損が発生しなかったことを確認します。
- これで、アプリケーションはデータの送受信を開始できます。どちらの側からの送信も含め、すべてのデータは Hello メッセージで選択したハッシュアルゴリズムを使用して最初にハッシュ化され、次に選択した秘密鍵アルゴリズムと生成したセッション鍵を使用して暗号化されます。
- 最後に、TLS セッションを正常に終了させることができるのは、クライアントとサーバのどちらかが終了を選択した場合のみです。セッションを正常に終了させるには、クライアントとサーバの両方が **CloseNotify** アラートを送信し、処理する必要があります。

2.3.3.2　初期化のフローシーケンス（Initialization Flow Sequence）

代表的な TLS セッション初期化のフローシーケンスを以下に示します。

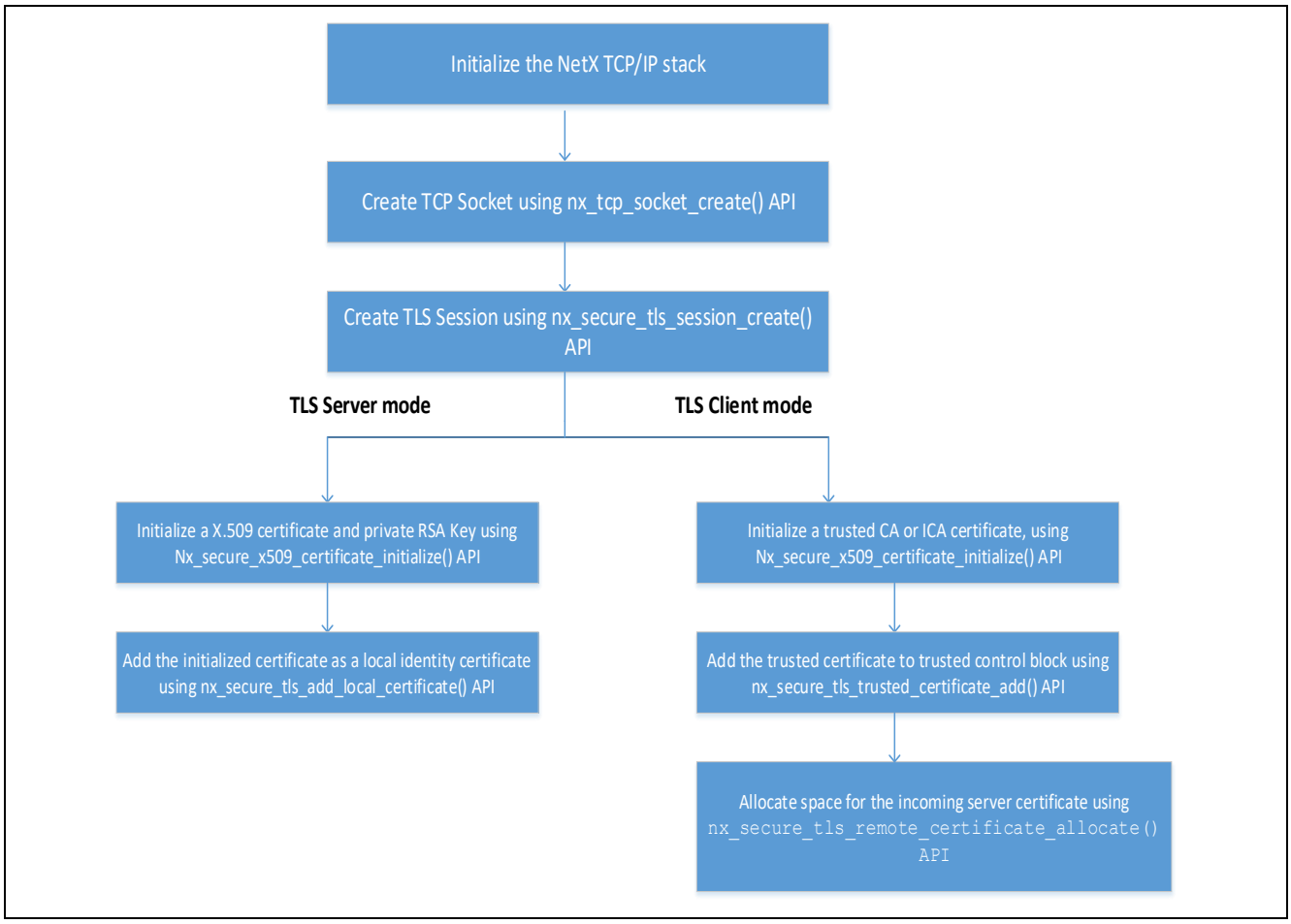

図 8　Synergy TLS セッションの初期化

2.3.3.3　データ通信のフローシーケンス（Data Communication Flow Sequence）

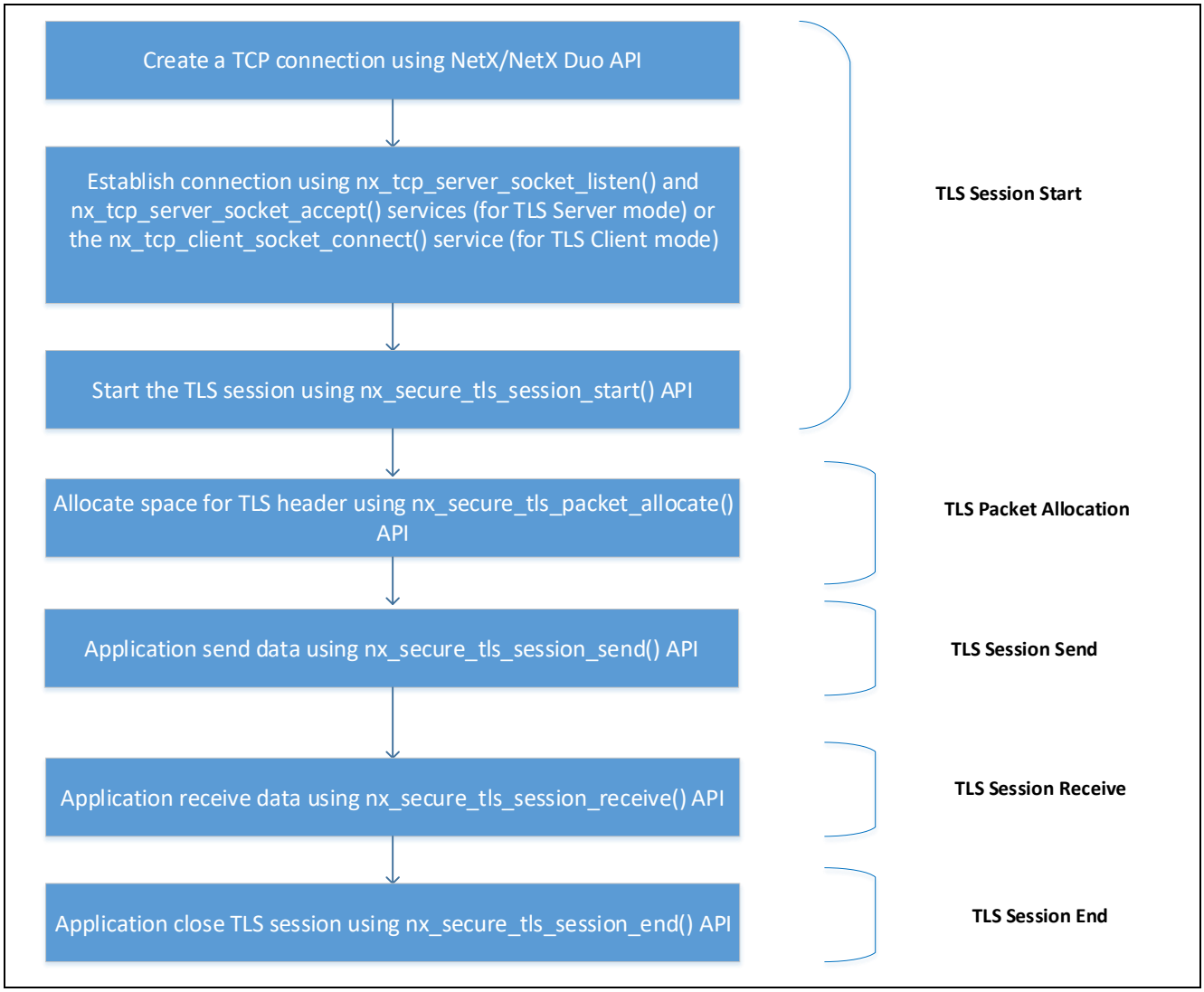

図 9　Synergy TLS セッションデータのフローシーケンス

3.　MQTT/TLS のサンプルアプリケーション（MQTT/TLS Application Example）

3.1　　アプリケーションの概要（Application Overview）

このサンプルアプリケーションプロジェクトは、オンボードの Synergy MQTT/TLS モジュールを使用した Renesas Synergy™ IoT Cloud 接続ソリューションのデモンストレーションです。デモの目的で、このアプリケーションはクラウドプロバイダとして Amazon Web Services (AWS)を使用しています。MQTT の Thing （モノ ：デバイス) と AWS IoT Core の間の主要な通信インタフェースとして、イーサネットまたは Wi-Fi または AE-CLOUD2 キットのみがサポートするセルラーネットワークを使用します。

このデモ例では、PK-S5D9 キットまたは AE-CLOUD2 キットまたは AE-CLOUD1 キットが MQTT のノード/モノ (デバイス) として動作し、AWS IoT Core に定期的に接続して自らの温度の値 (PK-S5D9 キットの場合) またはオンボードセンサの値 (AE-CLOUD2 キットまたは AE-CLOUD1 の場合) を読み出し、AWS IoT Core 宛にデータ送信を行います。また、自らの User LED state MQTT (ユーザ LED 状態 MQTT) トピックにサブスクライブします。ユーザは LED 状態への要求をリモートで発行することで、ユーザ LED の ON/OFF (点灯/消灯) を切り替えることができます。このアプリケーションは更新後の LED の状態を読み出し、ユーザ LED の ON/OFF を切り替えます。

3.2　ソフトウェアアーキテクチャの概要（Software Architecture Overview）

以下の図に、Synergy クラウド接続アプリケーションのサンプルプロジェクトに関するソフトウェアアーキテクチャを示します。

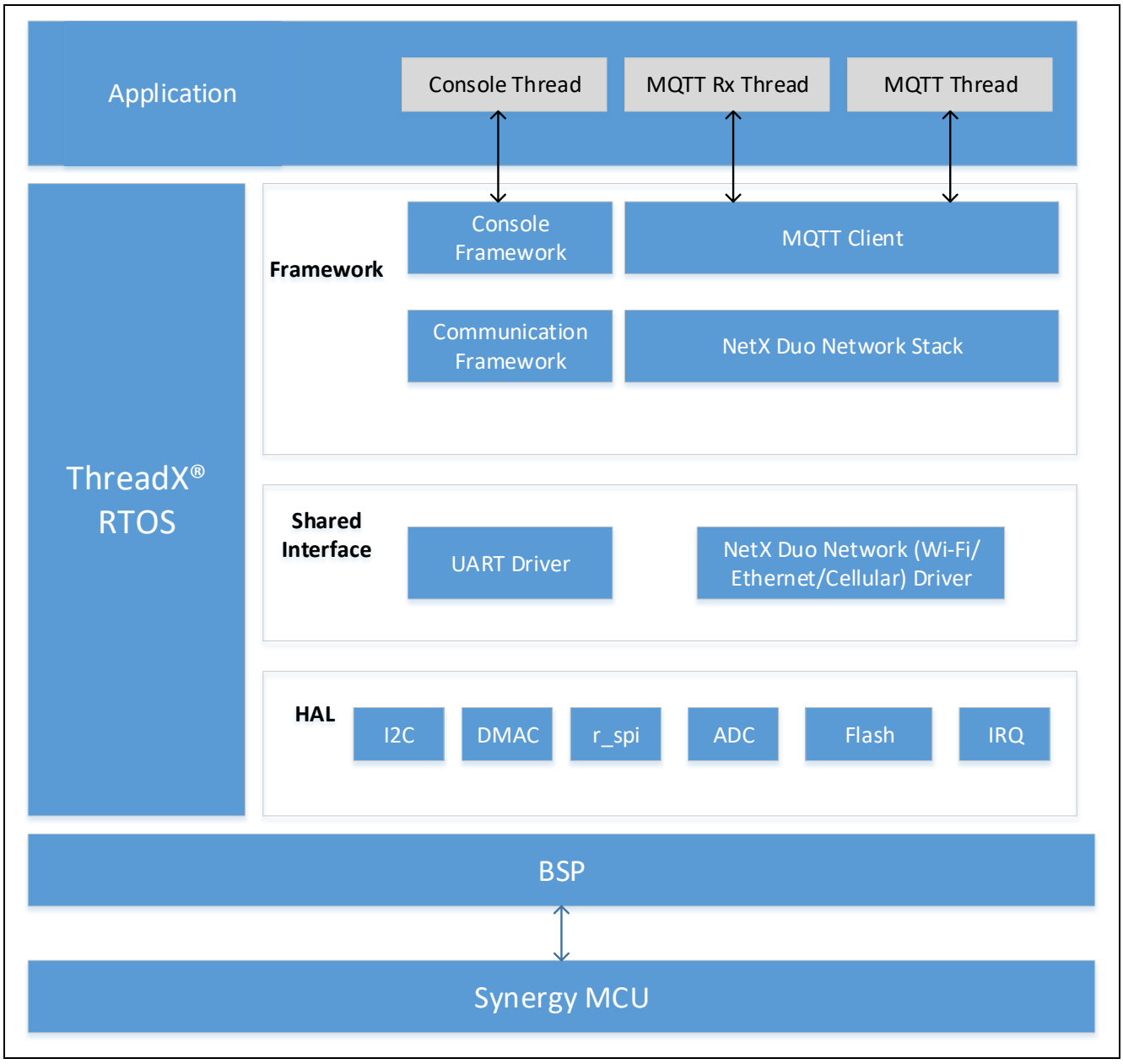

図 10　Synergy クラウド接続アプリケーションのソフトウェアアーキテクチャ

このアプリケーションの主なソフトウェアコンポーネントは以下のとおりです。

- MQTT クライアント（MQTT Client）
- NetX Duo IP スタックと、その基盤となるイーサネットや Wi-Fi 用のドライバコンポーネント
- コンソールフレームワーク（Console Framework）

このアプリケーションは、以下のアプリケーションで形成されています。

- コンソールスレッド（Console Thread）
- MQTT スレッド（MQTT Thread）
- MQTT Rx スレッド（MQTT Rx Thread）

3.2.1　コンソールスレッド（Console Thread）

このスレッドは、コマンドラインインタフェース (CLI) に関連する関数を処理します。このスレッドはコンソールフレームワークを使用し、コンソールフレームワークは通信フレームワークおよびその基盤となる USBX CDC デバイスモジュールコンポーネントを使用します。

このスレッドは、ユーザ入力を読み込んで、そのデータを内蔵データフラッシュに保存します。保存した情報は MQTT スレッドが 後に Synergy Cloud 接続デモを実行しようとするときに読み出します。

このスレッドでは、以下の CLI コマンドオプションを使用できます。

- **`cwiz`**
- **`Demo start/stop`**

`cwiz` コマンドオプション（`Cwiz command option`）

このコマンドオプションを使用して、以下の設定のいずれかを選択します。
1. イーサネットや Wi-Fi などのネットワークインタフェース、およびそれらに関連する IP モード (DHCP/Static)
2. IoT クラウドの選択 (Azure)
3. フラッシュからの既存設定のダンプ
4. メニューの終了

Demo start/stop コマンドオプション（`Demo start/stop command option`）

このコマンドオプションを使用して、Synergy Cloud 接続デモを実行/終了します。

3.2.2　MQTT スレッド（MQTT Thread）

これは主要な制御スレッドで、以下の主要な機能を処理します。
1. 通信インタフェース (イーサネット/Wi-Fi/セルラー) の初期化
2. IoT クラウドインタフェースの初期化
3. センサデータの読み出しと MQTT トピックへのデータの定期的な発行
4. 受信した MQTT メッセージのタイプに基づいてユーザ LED の状態を更新

ウェイクアップ状態時にユーザが CLI に demonstration start/stop コマンドを入力すると、このスレッドは定期的 (5 秒ごと) にユーザ入力イベントフラグ（user input event flag）の状態を確認します。CLI から demonstration start コマンドが発行された場合、このスレッドは事前設定済みのユーザ情報を内部フラッシュから読み出し、その有効性を確認します。その内容が有効な場合、このスレッドは Synergy Cloud 接続デモを開始します。demo stop コマンドが発行された場合、このスレッドは IoT クラウドインタフェースの終了処理をおこないます。

3.2.3 MQTT Rx スレッド（MQTT Rx Thread）

このスレッドは、MQTT ブローカー（broker）から着信した MQTT メッセージを処理します。新しい MQTT メッセージを受信した時点で、MQTT スレッドがユーザコールバック `receive_notify_callback()` を起動します。その後、このコールバックはセマフォ（semaphore）を設定し、MQTT Rx スレッドはこのセマフォを定期的にポーリングします。

新しい MQTT メッセージを受信した時点で、`nxd_mqtt_client_message_get()` API を使用してメッセージを読み出し、そのメッセージを解析し、受信したメッセージのタイプに基づいて処理します。

3.3　IoT クラウドの設定 (Azure)（IoT Cloud Configuration (Azure)）

3.3.1　Azure Web ポータルへのサインアップ（Azure Web Portal Signup）

Microsoft Azure は、ユーザごとに無料試用アカウントを 1 つ (12 か月間) 提供します。ここでは、次の章に進む前に、Azure Cloud サービスで 1 つのアカウントを既に作成したことを想定しています。

Azure アカウントを作成するには、Web ブラウザで以下のリンクを開きます。

https://azure.microsoft.com/en-us/account/ 必須の事項を入力し、無料ユーザアカウントを作成します。

注記：識別情報を検証するために、クレジットカード情報を追加するための画面が表示されます。アップグレードを実施しない限り、無料試用期間のうちはクレジットカードへの課金は実施されません。詳細を入力し、条項に同意した後、サインアップした無料試用アカウントが作成されます。

注記：Azure Web アカウントと IoT Hub を作成する際、このドキュメントに掲載しているスナップショットと実際の表示が多少異なっていることがあります。このプロジェクトの作業を進める際には、このような違いが生じる可能性に注意してください。

3.3.2　Azure ポータルでの IoT Hub の作成（Creating an IoT Hub on Azure Portal）

1. **[Microsoft Azure Portal]** (Microsoft Azure ポータル) で、**[All services]** (すべてのサービス) をクリックします。**[Internet of Things]** (モノのインターネット (IoT)) サブセクションで、**[IoT Hub]** (IoT ハブ) を選択します。IoT Hub にアクセスするためのスナップショットを以下の図に示します。

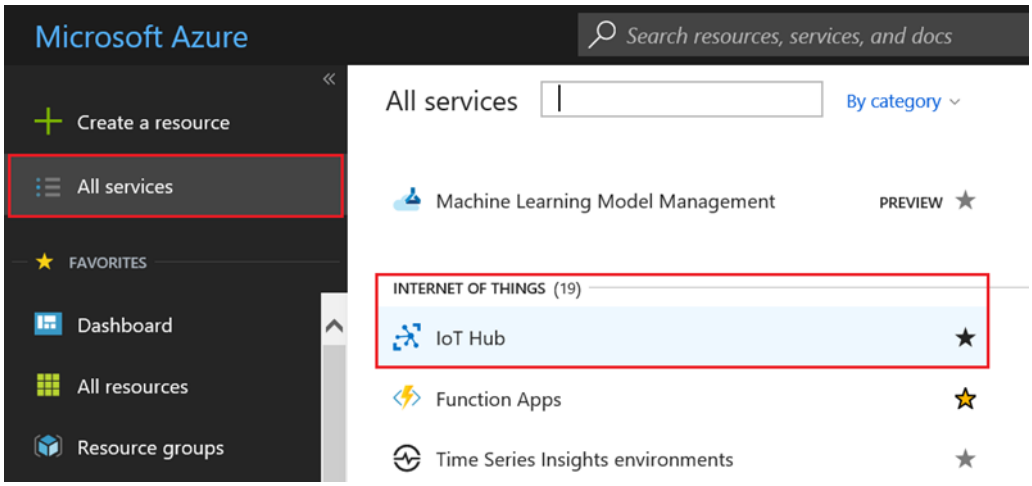

図 11　Azure の IoT Hub

2. IoT Hub を **[Add]** (追加) して作成するために、**[Desired IoT Hub name]** (希望の IoT Hub 名)、**[Region]** (地域)、**[Resource Group]** (リソースグループ)、**[Subscription]** (サブスクリプション) を入力します。ユーザが初めて使用する場合、**[Create a free account]** (無料アカウントの作成) というウィンドウが開きます。**[Start free]** (無料試用の開始) をクリックして無料アカウントを作成します。

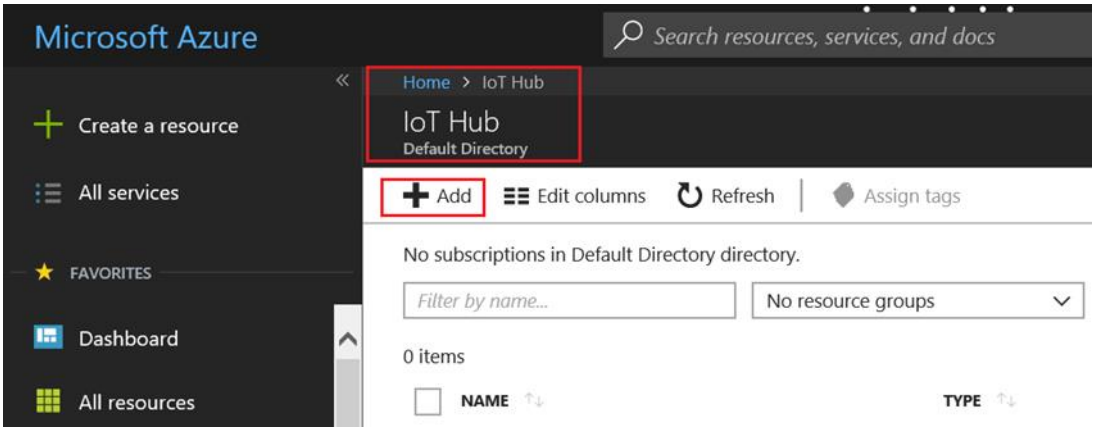

図 12　希望の名称を使用して IoT Hub を作成

3. **Choose the name of the IoT hub（IoT ハブの名前選択）**：　IoT ハブを作成するには、IoT ハブに名前を付ける必要があります。この名前は、すべての IoT ハブの間で一意の値にする必要があります。図 13 に示す画面へ入力するには以下を参考にしてください。

Choose the pricing tier（価格レベルの選択）: S1- Standard (S1- 標準)

IoT Hub Units (IoT ハブのユニット数): 1 (デフォルト)

Device to Cloud Partitions (デバイスからクラウドへの接続のパーティション数): 4 Partitions (4 個のパーティション)

Subscription (サブスクリプション): Visual Studio Enterprise

Resource Group (リソースグループ): Create new (新規作成)

Location (地域): West US (Japan (または居住地域に従って地域を選択))

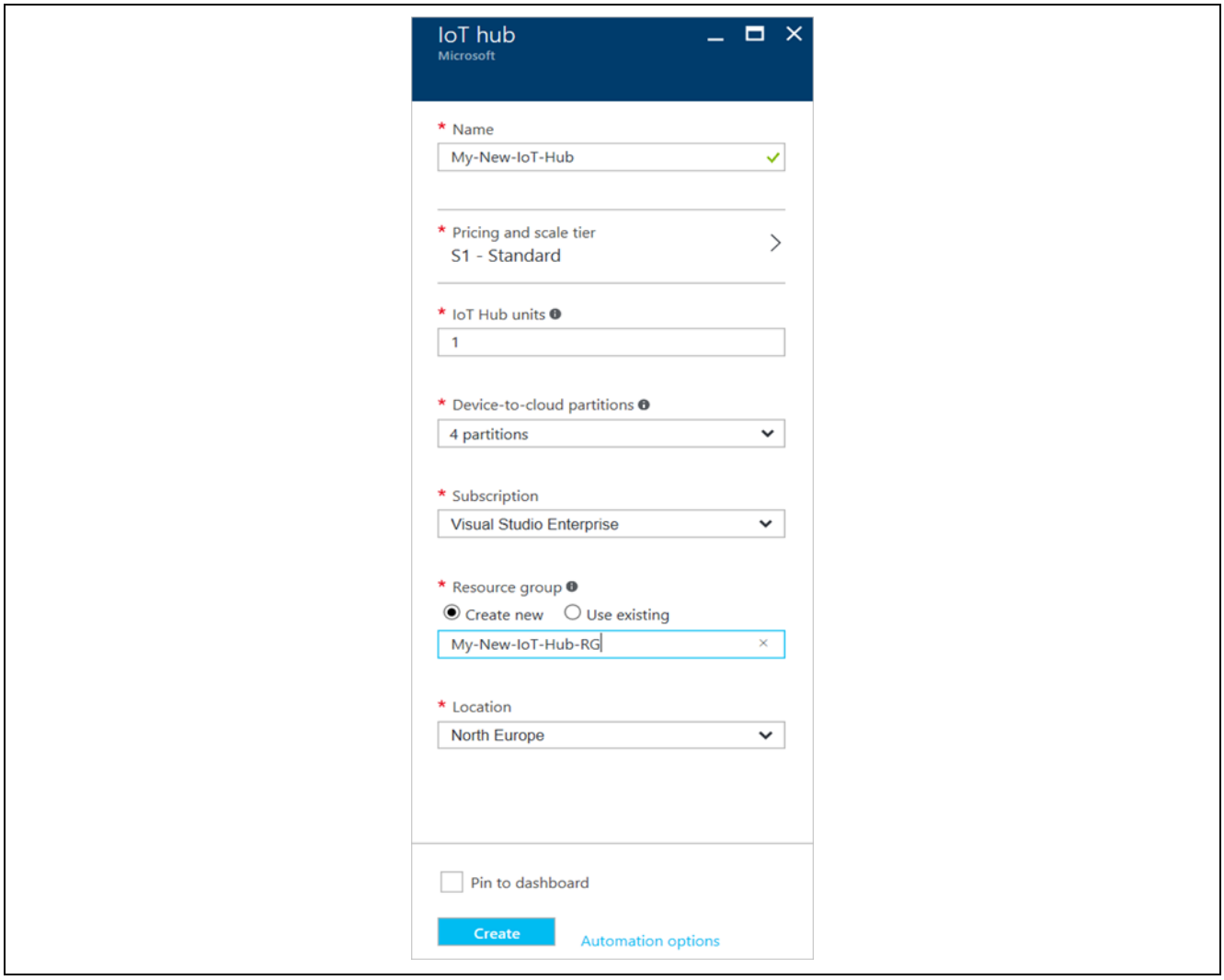

図 13　Azure IoT Hub の Subscription (サブスクリプション) 選択

注記：Azure ポータルで IoT Hub を作成するには、数秒を要します。

3.3.3　Azure IoT Hub 上でのデバイスの作成（Creating a Device on Azure IoT Hub）

Azure IoT Hub を選択して開き、以下の手順に従って IoT Hub 上でデバイスを作成します。

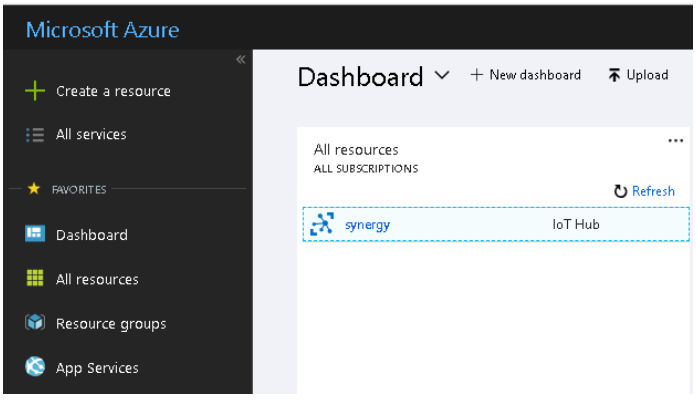

図 14　作成した IoT Hub のサンプルスナップショット

1. **[Explorers]** (エクスプローラ) の下にある **[IoT Devices]** (IoT デバイス) をクリックします。**図 16** のような新しいブレード（Brade）が表示されます。新しいデバイスを IoT Hub に追加するために、**[IoT devices]** (IoT デバイス) ブレードで、[+ **Add**] ボタンをクリックします。[Add Device] (デバイスの追加) をクリックすると、**[Device ID]** (デバイス ID) や **[Authentication Type]** (認証タイプ) のようなデバイスクレデンシャル（device credential）を入力するための新しいブレードが開きます。

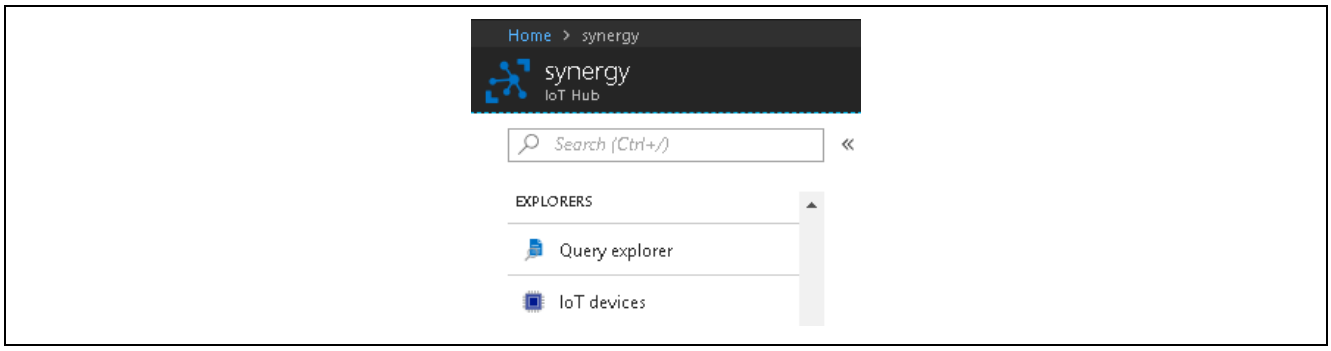

図 15　IoT Hub へのデバイスの追加

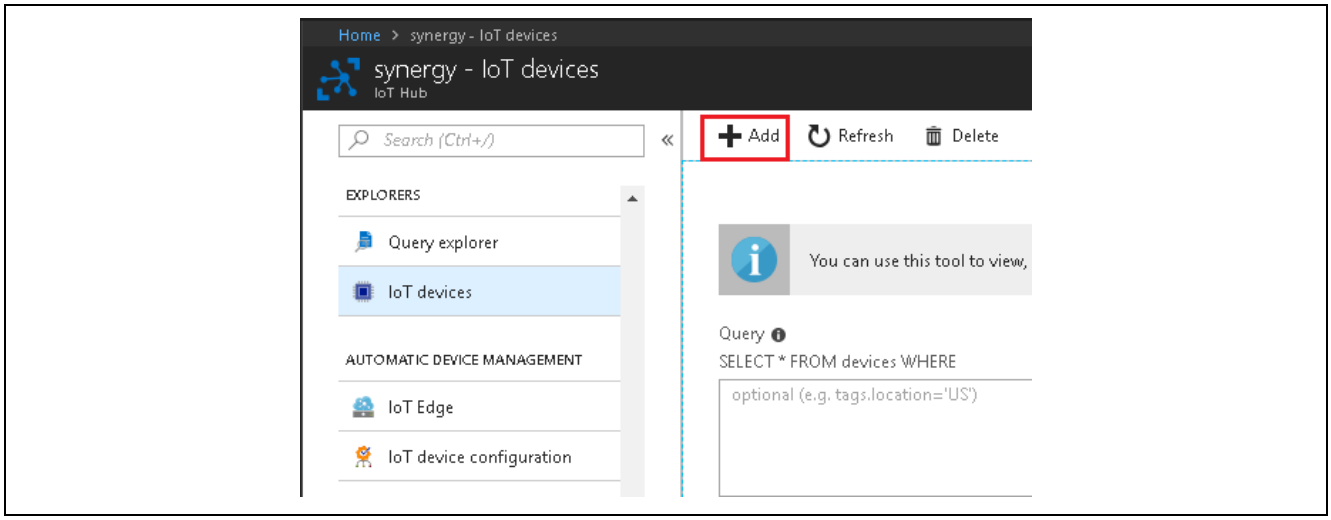

図 16　IoT Hub に追加されたデバイス

2. このアプリケーションノートの一部になっているアプリケーションプロジェクトの場合、[Authentication Type] (認証タイプ) として **[Symmetric Key]** (対称鍵) を使用します。以下の図に示すように目的の設定を入力および選択した後、[Save] (保存) ボタンをクリックします。IoT Hub 上に目的のデバイスを作成できます。IoT Hub 上で作成したデバイスのスナップショットを **図 18** に示します。

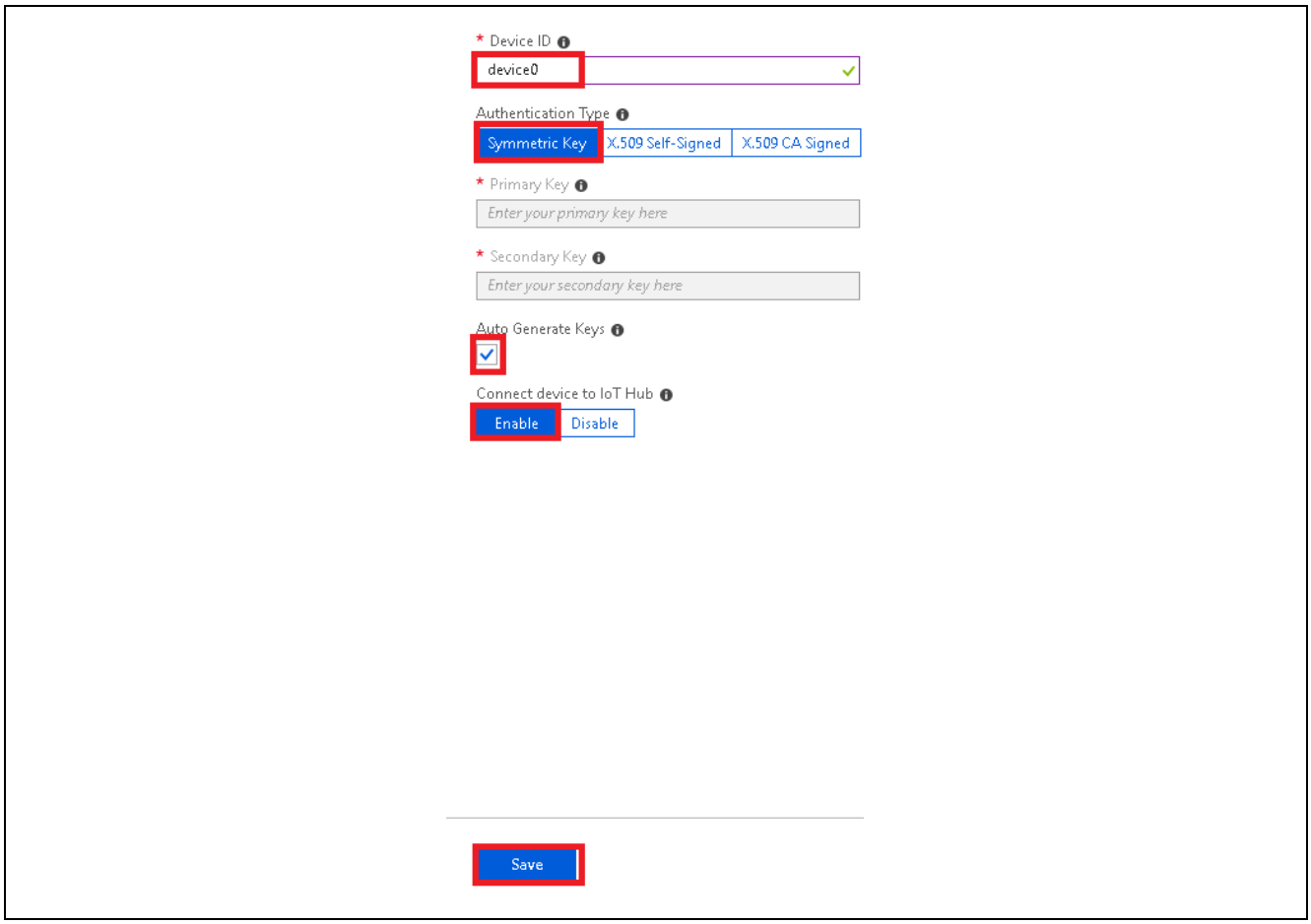

図 17　デバイスの設定

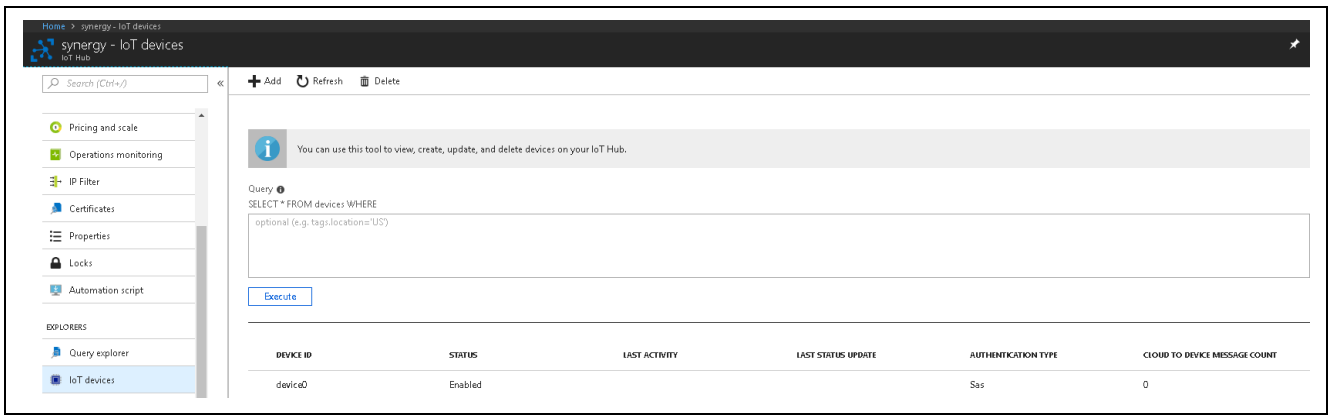

図 18　作成したデバイスのスナップショット

3. **[device]**(デバイス)、すなわち [device0] をダブルクリックして、以下のような新しいブレードを開きます。そのデバイスの詳細として、**[Device ID]** (デバイス ID)、**[Primary and Secondary symmetric shared access Key]** (プライマリとセカンダリの対称型共有アクセス鍵)、**[Primary and Secondary connection string]** (プライマリとセカンダリの接続文字列) などが表示されます。
   注記：デバイス側でアプリケーションを実行するときに、接続文字列から取得した **[Primary Key]** (プライマリ鍵) と **[Hostname info]** (ホスト名情報) が使用されます。デバイスをコマンドラインインタ

フェースを使用して設定するときに、これらの詳細が使用されます。詳細については、4.4.1.2 章で説明します。

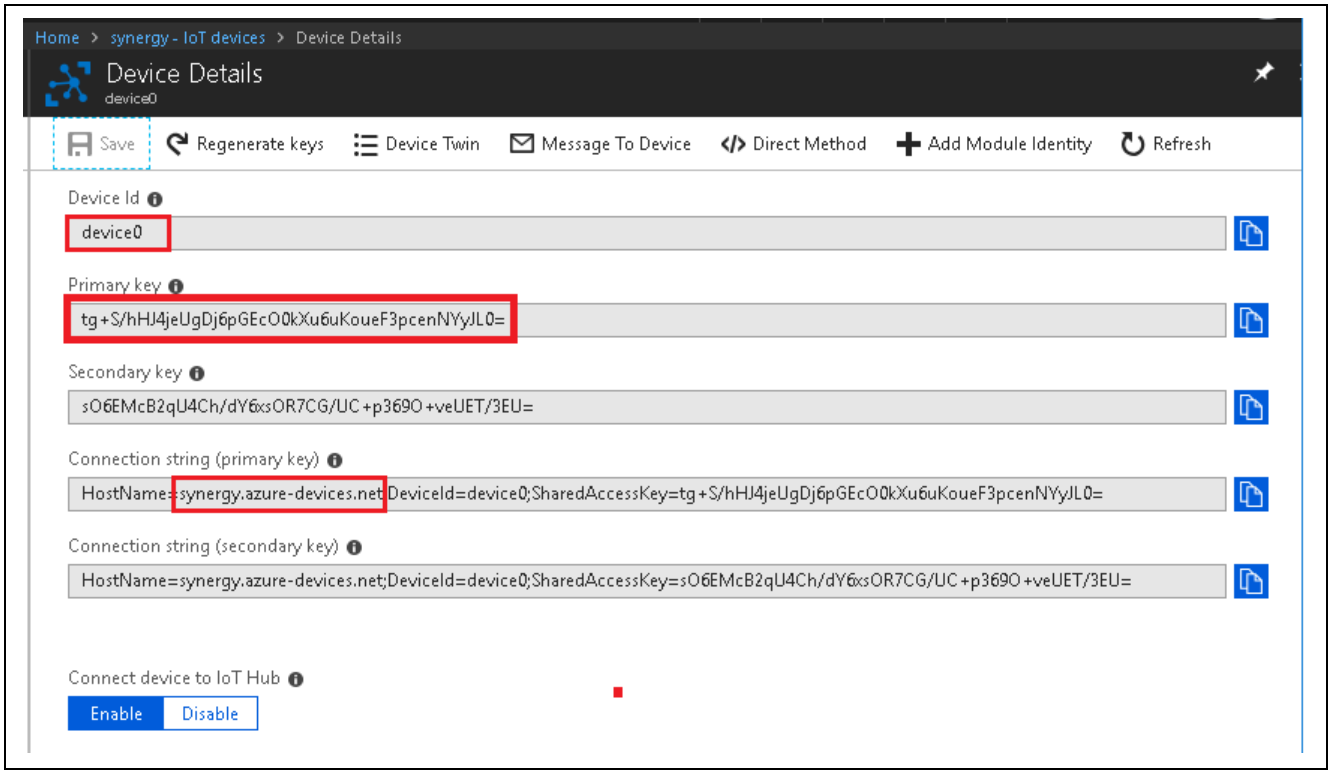

図 19　デバイスの詳細

これで、IoT Hub 上でデバイスの作成と設定が完了します。

4.　MQTT/TLS アプリケーションの実行（Running the MQTT/TLS Application）

4.1　プロジェクトのインポート、ビルド、およびロード（Importing, Building, and Loading the Project）

手順については、このパッケージに付属している『*Renesas Synergy™ Project Import Guide*』(Renesas Synergy プロジェクトインポートガイド) (r11an0023eu0121-synergy-ssp-import-guide.pdf) を参照し、プロジェクトを e² studio にインポートしてビルドおよび実行します。

4.2　AE-CLOUD2 キットまたは AE-CLOUD1 キットのボードサポートパッケージを手動で追加（Manually Adding the Board Support Package for the AE-CLOUD2 Kit or AE-CLOUD1 Kit）

4.2.1　AE-CLOUD2 キット

1. プロジェクトバンドルから、BSP ファイルである `Renesas.S5D9_PILLAR_ARDUINO_MODULE.1.5.3.pack` を見つけます。
2. e² studio ユーザの場合、以下の図に示してあるファイルを、以下の場所にコピーします。
   `C:\Renesas\Synergy\e2studio_v6.2.1_ssp_v1.5.3\internal\projectgen\arm\packs`

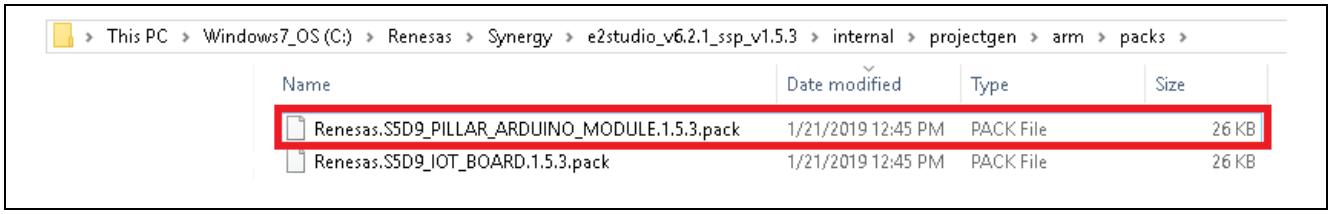

図 20　AE-CLOUD2 キットの BSP パッケージをロード

3. IAR ユーザの場合、以下の図に示してあるファイルを、以下の場所にコピーします。
C:\Renesas\Synergy\ssc_v6.2.1_ssp_v1.5.3\internal\projectgen\arm\packs.

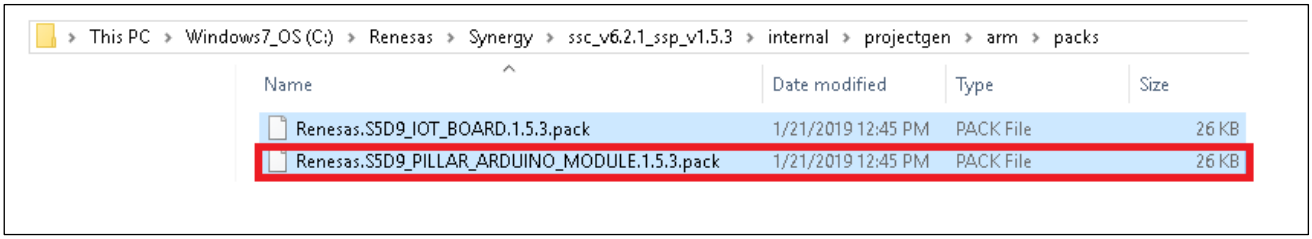

図 21　AE-CLOUD2 キットの BSP パッケージをロード

注記：e² studio または SSC を別の場所にインストールした場合、パッケージをコピーする際には対応する同じ場所を指定する必要があります。

4.2.2　AE-CLOUD1 キット

1. プロジェクトバンドルから、**Renesas.S5D9_IOT_BOARD.1.5.3.pack** を見つけます。
2. e² studio ユーザの場合、以下の図に示してあるファイルを、以下の場所にコピーします。

C:\Renesas\Synergy\e2studio_v6.2.1_ssp_v1.5.3\internal\projectgen\arm\packs.

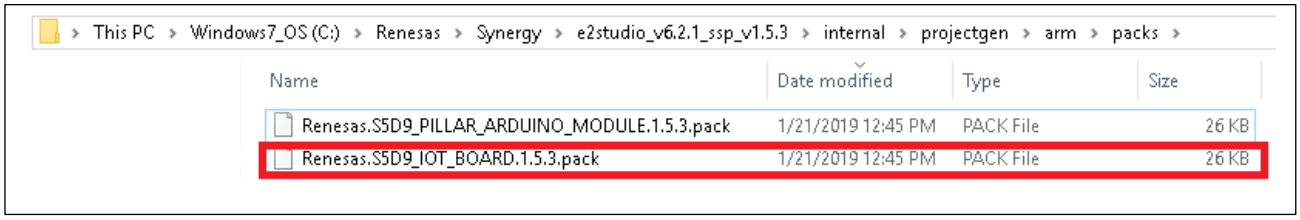

図 22　AE-CLOUD1 キットの BSP パッケージをロード

3. IAR ユーザの場合、以下の図に示してあるファイルを、以下の場所にコピーします。

C:\Renesas\Synergy\ssc_v6.2.1_ssp_v1.5.3\internal\projectgen\arm\packs.

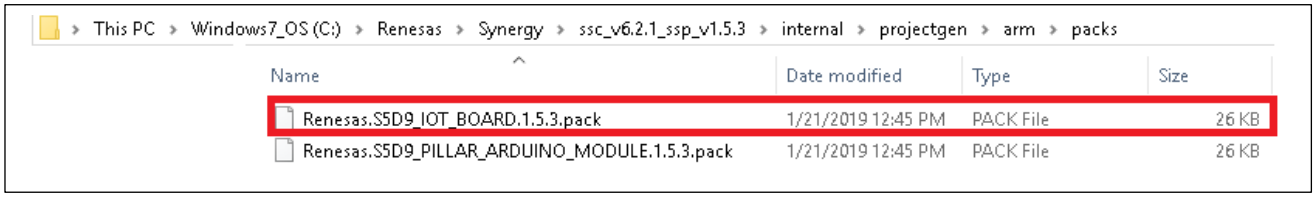

図 23　AE-CLOUD1 キットの BSP パッケージをロード

注記：e² studio または SSC を別の場所にインストールした場合、パッケージをコピーする際には対応する同じ場所を指定する必要があります。

4.3　ボードの電源投入（Powering up the Board）

電源をボードに接続するために、SEGGER J-Link® デバッガを PC に接続し、ボードを PC の USB ポートに接続して、デバッグアプリケーションを実行したうえで、以下の手順を使用します。

1. PK-S5D9 ボードおよび AE-CLOUD2 ボードの場合、付属している USB ケーブルの Micro USB コネクタを、PK-S5D9 ボードの J19 コネクタ (DEBUG_USB) または AE-CLOUD2 ボードの J6 コネクタ (DEBUG_USB) に接続します。
   USB ケーブルのもう一方のコネクタをユーザの PC の USB ポートに接続します。
   注記：このキットは、SEGGER J-Link® On-board (OB) をオンボード搭載しています。J-Link は、PK-S5D9 ボードおよび AE-CLOUD2 ボードの全てのデバッグ機能とプログラミング機能を実現します。
2. AE-CLOUD1 ボードについては、付属の 10 ピンフラットリボンケーブルを AE-CLOUD1 ボードの J2 コネクタと接続し、ケーブルのもう一方を付属の J-LinkLite の 10 ピンヘッダに接続します。USB ケーブルのもう一方のコネクタを、ユーザの PC の USB ポートに接続します。
3. PMOD ベースの GT-202 Wi-Fi モジュールを PMOD コネクタに接続します (**PK-S5D9 の場合、PMOD-A に接続します**)。
4. AE-CLOUD2 キットを使用する場合、AE-CLOUD2 Arduino Connector の BG96 セルラーシールドを接続します。次に、セルラーアンテナを LTE アンテナコネクタに取り付けてから、GPS アンテナを BG96 シールドの GNSS アンテナコネクタに取り付けます。
5. 2 番目の Micro USB ケーブルを、使用するキットに応じて以下のコネクタに接続します。
   －AE-CLOUD2 ボードもしくは AE-CLOUD1 ボードの J9 コネクタ
   －PK-S5D9 ボードの J5 コネクタ
   USB ケーブルのもう一方をユーザの PC のの USB ポートに接続します。これはシリアルコンソールを使用するために必要です。

4.4　Azure IoT Cloud への接続（Connect to Azure IoT Cloud）

以下の説明で、Synergy Cloud 接続アプリケーションプロジェクトを実行し、Azure IoT Cloud に接続する方法を示します。

注記：この段階で、3.3 章の手順を使用して、Azure IoT アカウントの作成と、Azure IoT Hub に合わせたデバイスのセットアップが完了していることを想定しています。

- 以下の説明では、クラウド接続に使用するインタフェースに応じてボードを設定する際のコマンドラインインタフェース（CLI）の使用方法を示します。
- 使用しているボードでのアプリケーションの実行中に、一度の接続につきいずれか一種類のインタフェース（イーサネット、Wi-Fi、またはセルラー）を使用して接続ができます。ユーザはアプリケーションの実行の際に使用したいインタフェースのみ設定が必要です。イーサネットを使用する場合には、Wi-Fi またはセルラーは使用できません。他のインタフェースの設定についても同様です。
- 表示されている CLI スナップショットは、セルラーを使用できない PK-S5D9 ボードおよび AE-CLOUD1 ボード等のように、適用できない場合があります。

ボード	イーサネット	Wi-Fi	セルラー
PK-S5D9	サポートされる	サポートされる	サポートされない
AE-CLOUD1	サポートされる	サポートされる	サポートされない
AE-CLOUD2	サポートされる	サポートされる	サポートされる

1. まだこれらが完了していない場合、3 章の手順を完了させ、4.3 章に進んで PK-S5D9 キットあるいは AE-CLOUD2 キットあるいは AE-ALOUD1 キットの電源を投入し、プロジェクトをロードしてください。キットの USB Device ポートをテスト PC に接続します。Windows 10 PC を使用している場合、キッ

トは USB シリアルデバイスとして自動的に検出されます。Windows 7/8 PC を使用している場合、以下のインストールガイドを参照し、Synergy USB CDC ドライバをロードしてください。
https://www.renesas.com/jp/ja/products/synergy/software/add-ons/usb-cdc-drivers.html

2. Tera Term のようなシリアルコンソールアプリケーションを開き、PK-S5D9/AE-Cloud2 キットに接続します。Tera Term のデフォルト設定は 8-N-1 (データ長 8 ビット、パリティなし、ストップビット 1 ビット) であり、ボーレート（baud rate）は 9,600 です。
3. キーボードの **Enter** キーを押します。シリアルコンソールに以下のようなプロンプトが表示されます。

図 24　コマンドラインプロンプト

注記：AE-CLOUD2 キットを使用している場合、セルラーのモデム（Cellular Modem）と GPS の初期化完了を待ちます。7 ～ 10 秒かかります。

4. キーボードの「**?**」キーを押します。以下の図 に示すように、使用可能な CLI コマンドオプションが表示されます。

図 21　ヘルプメニュー

4.4.1　設定ウィザードメニュー（Configuration Wizard Menu）

シリアルコンソール（serial console）に「**cwiz**」コマンドを入力し、Enter キーを押して設定メニューに移行します。このコマンドは、**1) ネットワークインタフェース**や **2) Azure IoT Hub Service** の設定、また **3) 内部フラッシュに保存されている以前の設定のダンプ**を行うのに使用します。

図 26　設定メニュー

4.4.1.1　ネットワークインタフェースの選択（Network Interface Selection）

ネットワークインタフェースを設定するには、設定メニューで「**1**」キーを押します。このアプリケーションプロジェクトで使用可能なネットワークインタフェースオプションの一覧が表示されます。現時点でこのアプリケーションは、イーサネット、Wi-Fi、セルラー (AE-CLOUD2 キットを使用する場合) の各ネットワーク通信インタフェースをサポートしています。

注記：　ユーザが選択できるネットワークインタフェースは、一度に 1 つのみです。たとえば、[Ethernet] (イーサネット) を選択した場合、[Wi-Fi] や [Cellular] (セルラー) は使用できません。逆も同様です。

例えば、ユーザがクラウドの接続のインタフェースとしてイーサネットネットワークインタフェースの設定を選択した場合、4.4.1.2 章（ Azure IOT Core の設定（Azure IOT Core Configuration））を直接参照します。Wi-Fi およびセルラーの場合も同様です。

図 27　ネットワークインタフェース選択メニュー

(1)　イーサネットネットワークインタフェースの設定（Ethernet Network Interface Configuration）

イーサネットネットワークの設定を選択するには、**[Network Interface Selection]** (ネットワークインタフェースの選択) メニューで、「**1**」キーを押します。

IP アドレス設定モード選択のためのサブメニューが表示されます。**[IP Address Configuration Mode]** (IP アドレス設定モード) を選択します。選択したイーサネット設定項目は内部フラッシュに保存されます。後で、通信を初期化する際にこの設定項目が使用されます。

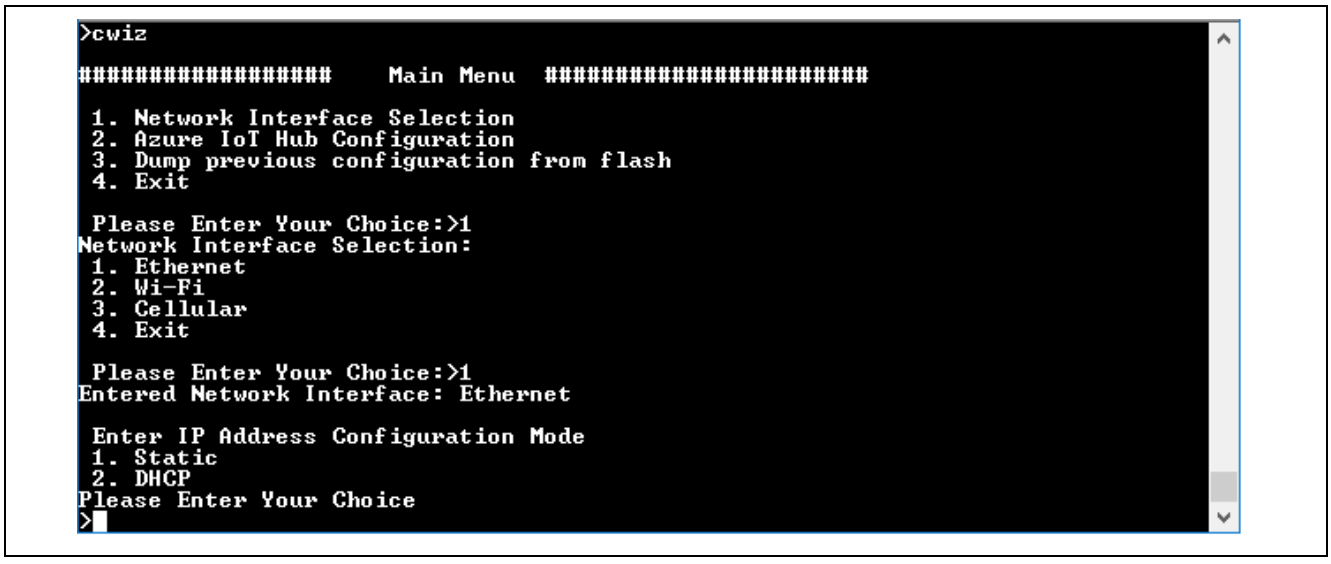

図 28　イーサネットネットワークインタフェースの設定メニュー

(2) Wi-Fi ネットワークインタフェースの設定（Wi-Fi Network Interface Configuration）

Wi-Fi ネットワークをインタフェースとして選択するには、**[Network Interface Selection]** (ネットワークインタフェースの選択) メニューで、「**2**」キーを押します。

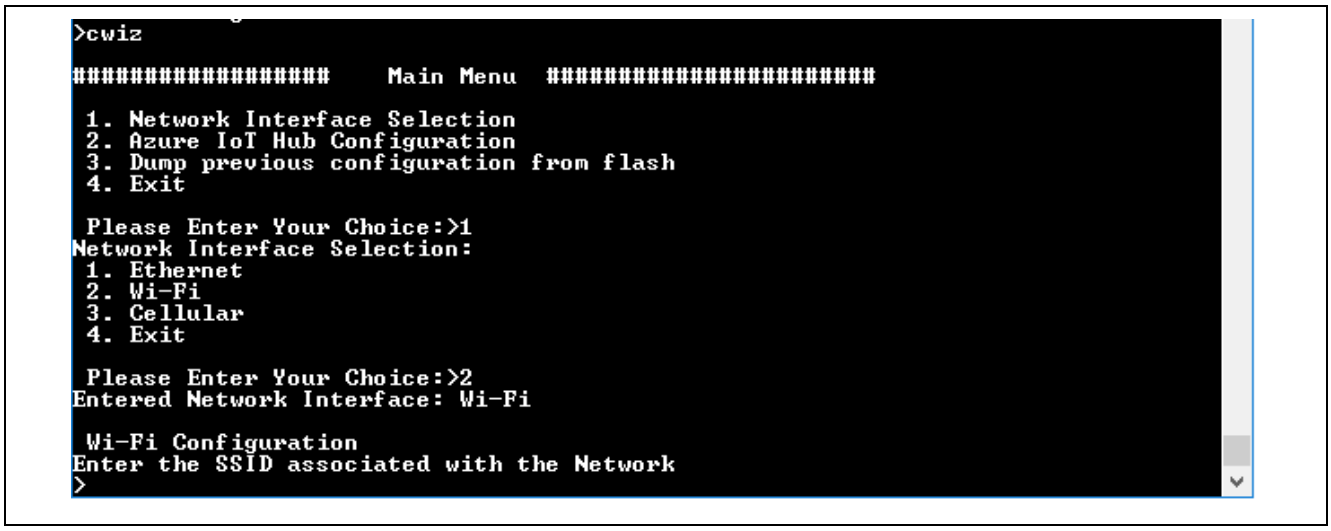

図 29　Wi-Fi ネットワークインタフェースの設定メニュー

Wi-Fi 設定項目を入力するために、**[SSID]**、**[passphrase]** (パスフレーズ)、**[Security type]** (セキュリティタイプ)、**[IP Address Configuration Mode]** (IP アドレス設定モード) などのオプションが表示されます。
選択した Wi-Fi 設定項目が内部フラッシュに保存されます。後ほど、通信を初期化する際にこれらの設定項目が使用されます。

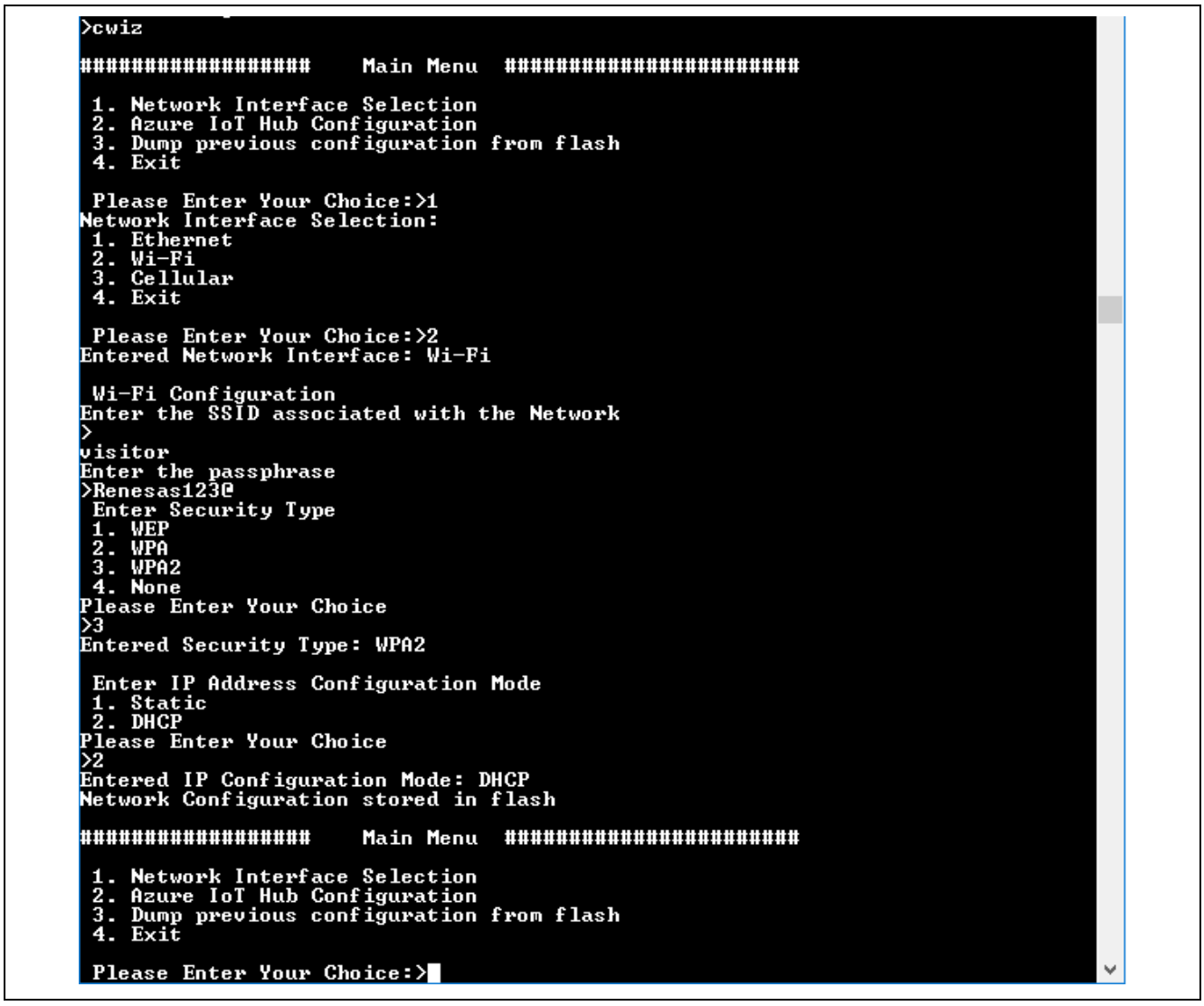

図 30　Wi-Fi 設定

(3) セルラーネットワークインタフェースの設定（Cellular Network Interface Configuration）

セルラーネットワークをインタフェースとして選択するには、**[Network Interface Selection]** (ネットワークインタフェースの選択) メニューで、「**3**」キーを押します。
以下の図のような 2 つの選択肢（オプション）が表示されます。

- オプション 1: SIM プロビジョニング（SIM provisioning）の場合、「**1**」と入力します。この場合、SIM カードを既に設定してあることを想定しています。
- オプション 2: SIM 設定（SIM configuration）の場合、「**2**」と入力します。AT シェルインタフェースを使用して新規 SIM カードを設定する場合、このオプションは最適です。

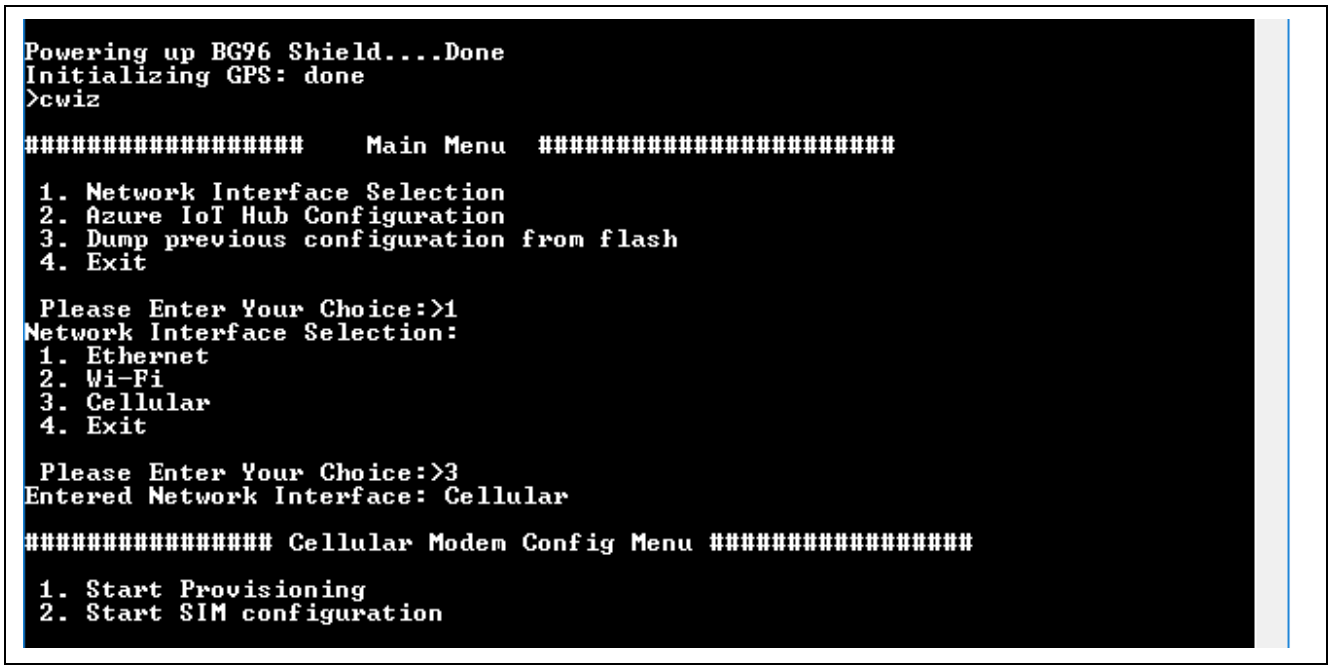

図 31　セルラーの設定

(a) プロビジョニングの開始オプション（Start Provisioning Option）

[Cellular Modem Configuration Menu] (セルラーのモデム設定メニュー) で、以下の図のようにオプション **[1]** を選択し、[Start Provisioning] (プロビジョニングの開始) サブメニューに入ります。

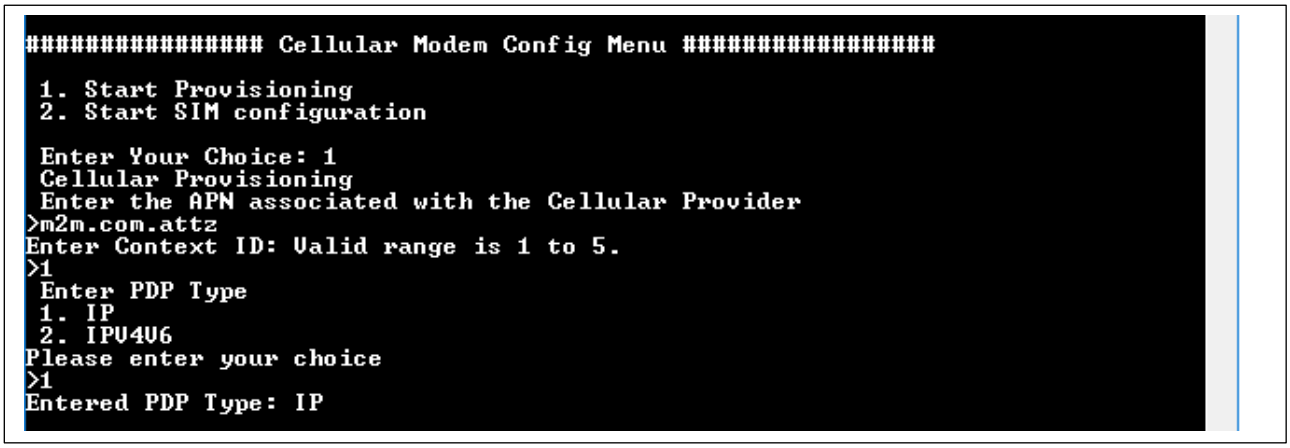

図 32　セルラーモデムのプロビジョニングメニュー

スクリーンショットは、AT&T（米国のセルラー事業者) の SIM カードで使用する [Cellular] (セルラー) の設定項目を示しています。 実際は、各国のセルラー事業者のものに合わせてください。
セルラー設定項目を入力するために、**[APN]**、**[Context ID]** (コンテキスト ID)、**[PDP type]** (PDP タイプ) などのオプションが表示されます。

注記：[APN name] (APN 名)、[Context ID] (コンテキスト ID)、[PDP Type] (PDP タイプ) は、Cellular Service (セルラーサービス) プロバイダから提供されます。既にこれらの情報を把握している場合、CLI を使用してそれらの情報を入力します。

選択したセルラー設定項目は内部フラッシュに保存されます。後で、通信インタフェースを初期化する際にこれらの設定項目が使用されます。

(b) 注記：使用する SIM カードを設定し、セルラーモデムのプロビジョニングが完了している場合には 4.4.1.1 章の(3)(b)項をスキップしてもかまいません。SIM の設定開始オプション（Start SIM Configuration Option）

[Cellular Modem Configuration Menu] (セルラーのモデム設定メニュー) で、以下の図のようにオプション **[2]** を選択し、[Start SIM Configuration] (SIM 設定の開始) サブメニューに入ります。

注記： ファームウェアがバックグラウンドでセルラーフレームワークインスタンスを開くので、[Cellular Configuration] (セルラー設定) メニューに入るには数秒を要します。

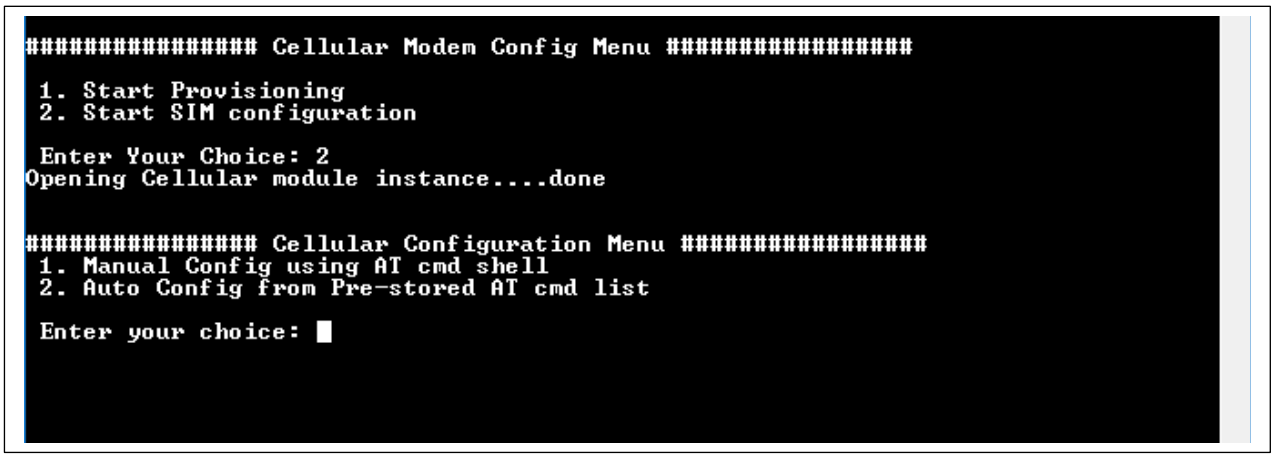

図 33　セルラーの設定メニュー

このモードでオプション **[1]** を選択すると、AT コマンドシェルモードが開きます。ここで、個別の AT コマンドを入力し、確認を行うことができます。また、セルラーサービスプロバイダが必要としている順序に基づき、複数の AT コマンドをフラッシュメモリに保存することもできます。オプション **[2]** を選択すると、事前保存した AT コマンドリストを取得します。

注記：ファームウェアがバックグラウンドでセルラーフレームワークインスタンスを開くので、[Cellular Configuration] (セルラー設定) メニューに入るには数秒を要します。

オプション **[1]** を選択して AT コマンドシェルを使用する手動設定モードに入るか、オプション **[2]** を選択して事前保存した AT コマンドリストから選択を行う自動設定モードに入ることができます。このリストは、まず手動設定を行い、その後に生成します。

AT コマンドシェルを使用した手動設定（Manual Configuration using AT Command Shell）

オプション **[1]** の場合、以下のように AT コマンドシェルに入ります。SIM カードを設定するために、さまざまな AT コマンドを試すことができます。

BG96 セルラーモデムを使用して SIM カードのプロビジョニングを実施するためのベースラインとして、Renesas が公開している以下のナレッジベースの記事を参照してください。

https://en.na4.teamsupport.com/knowledgeBase/18027787

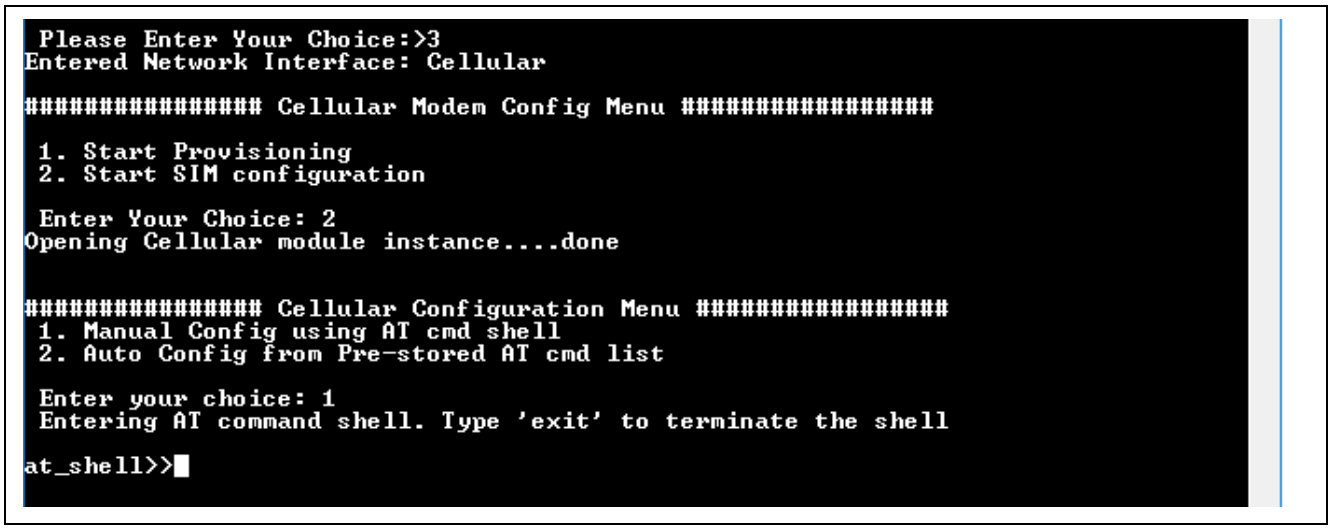

図 34　AT コマンドシェル

AT コマンドシェルを終了するには、「**exit**」または「**EXIT**」コマンドを入力します。以下の図のように、AT コマンドを保存するかどうかを尋ねられます。

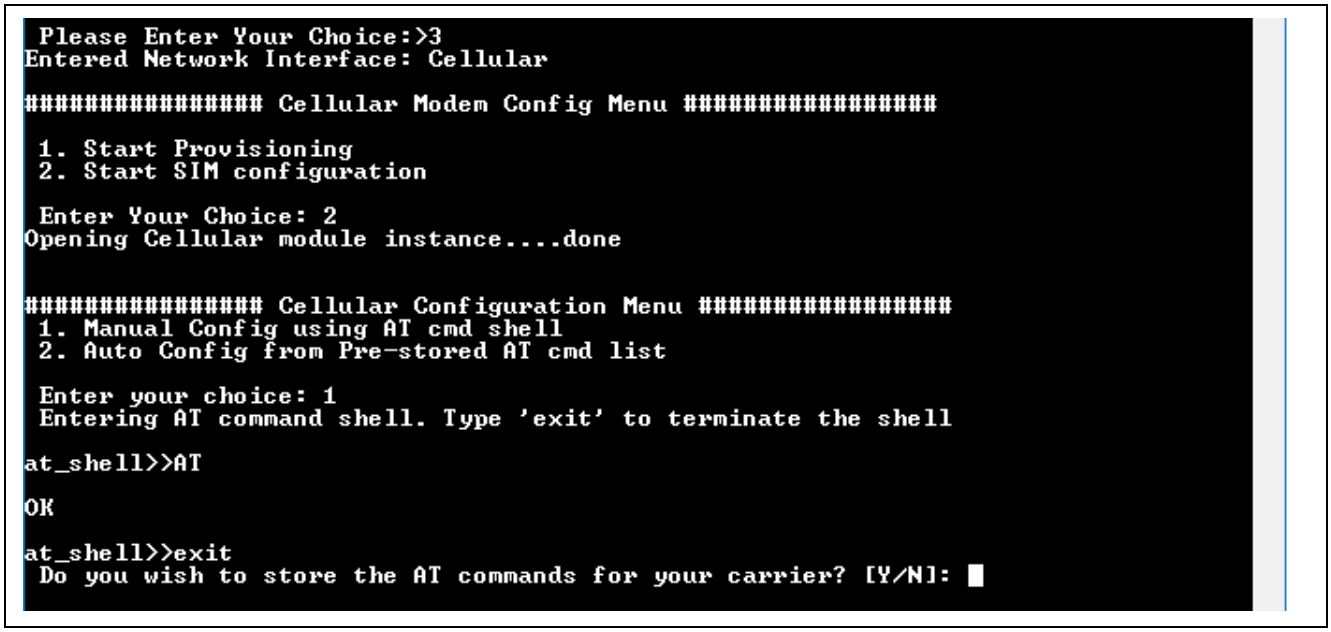

図 35　AT コマンドシェルの使用法

AT コマンドの保存を選択し、後でこれらのコマンドを使用して、新しい SIM カードの自動設定を行おうとする場合、「Y」と入力します。その場合、以下の図のように、AT コマンドの詳細を入力するように表示されます。

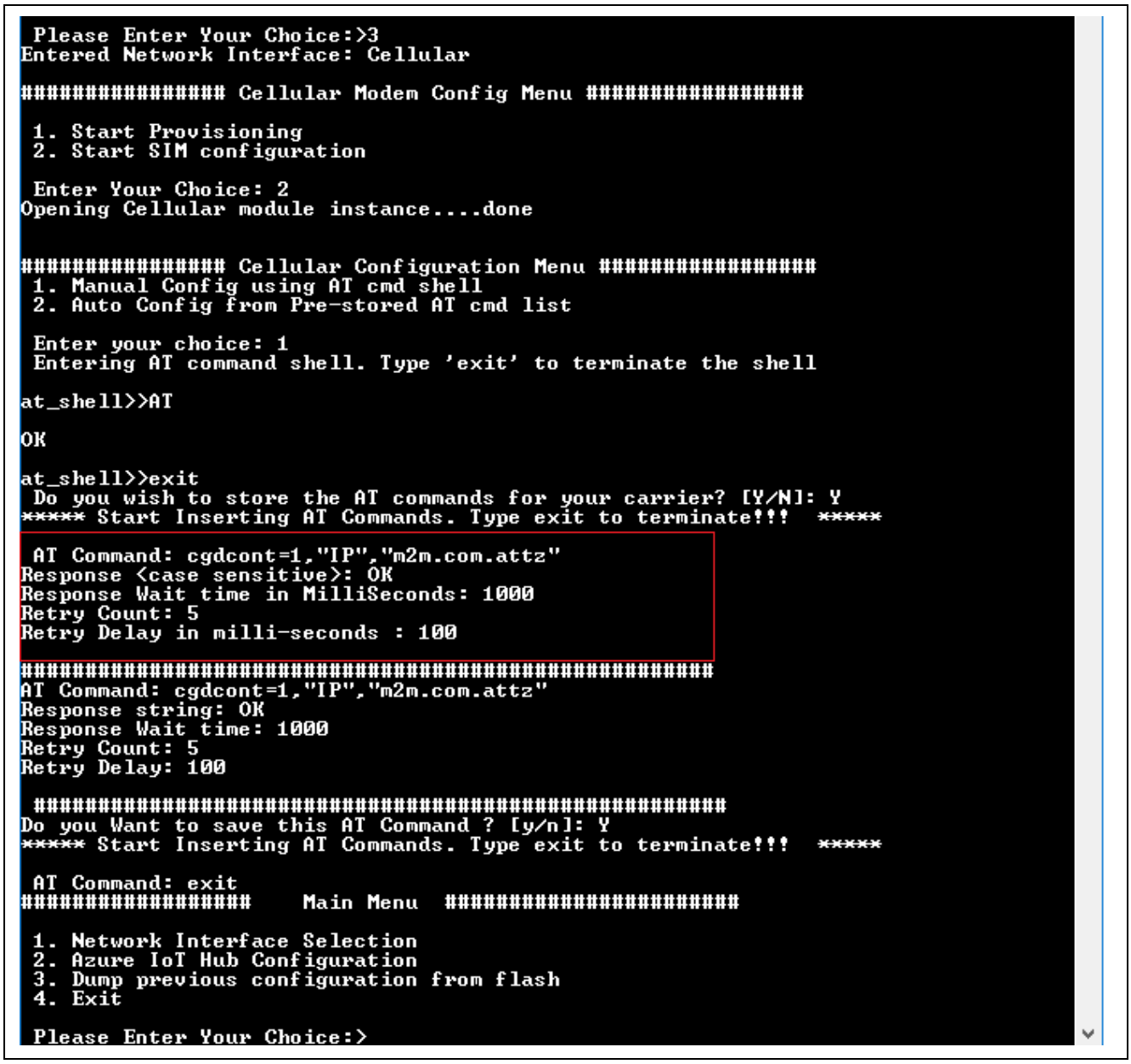

図 36　AT コマンドシェルを使用した手動モード設定

事前保存した AT コマンドリストによる自動設定（Auto Configuration from pre-stored AT command list）

[Cellular Configuration] (セルラーの設定) メニューで、以下の図のようにオプション **[2]** を選択し、**[Auto configuration from pre-stored AT command list menu]** (事前保存した AT コマンドリストから選択を行う自動設定) メニューに入ります。

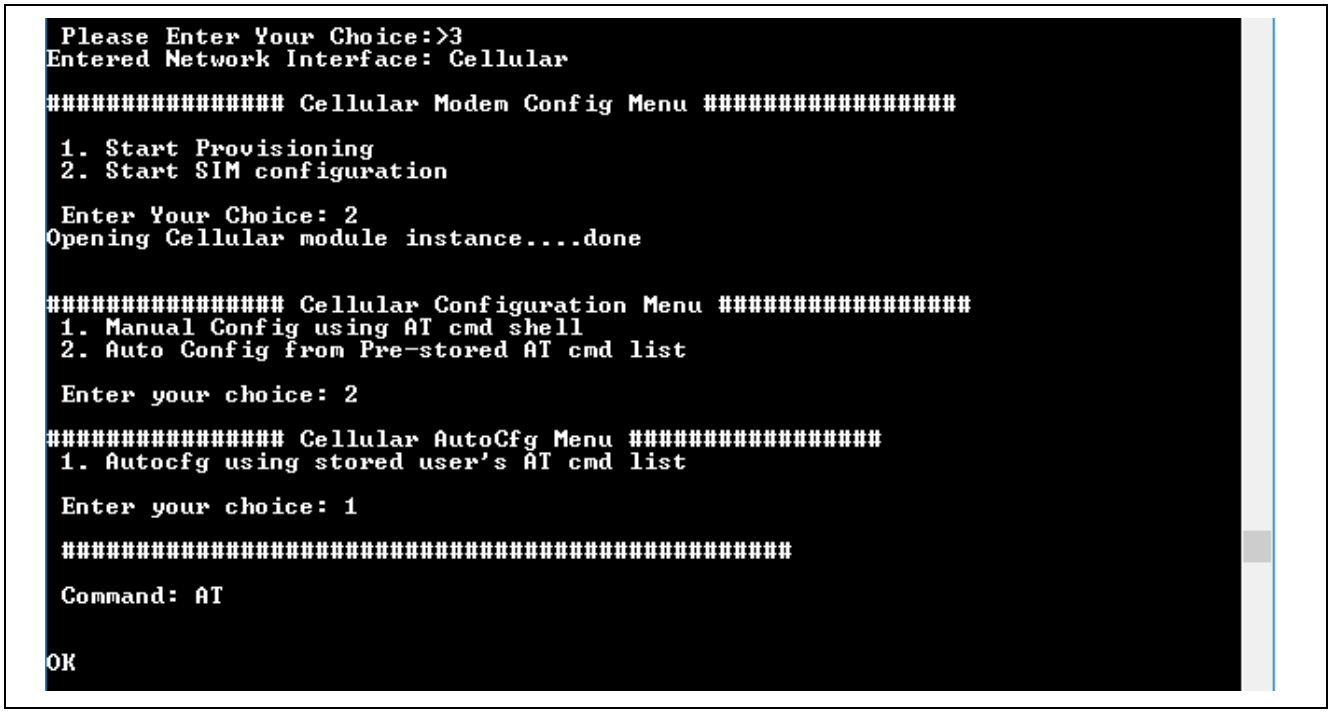

図 37　事前保存した AT コマンドリストから選択を行う自動設定

事前保存した AT コマンドがセルラーモデム宛に送信され、モデムからの応答がコンソールウィンドウに表示されます。

4.4.1.2　Azure IoT Hub の設定（Azure IoT Hub Configuration）

この時点で、3.3 章で説明した手順を既に実行し、Azure Cloud Platform 内でデバイスを作成し、4.4.1.1 章で説明されている使用したい通信インタフェースの選択が完了したことを想定しています。まだ作成していない場合、先に進む前に、3.3 章および 4.4.1.1 章で説明されている手順を完了させてください。

以下の図のように **[Main Menu]** (メインメニュー) で、「**2**」を押し、**Enter** キーを押して Azure Cloud IoT Core サービスを設定します。

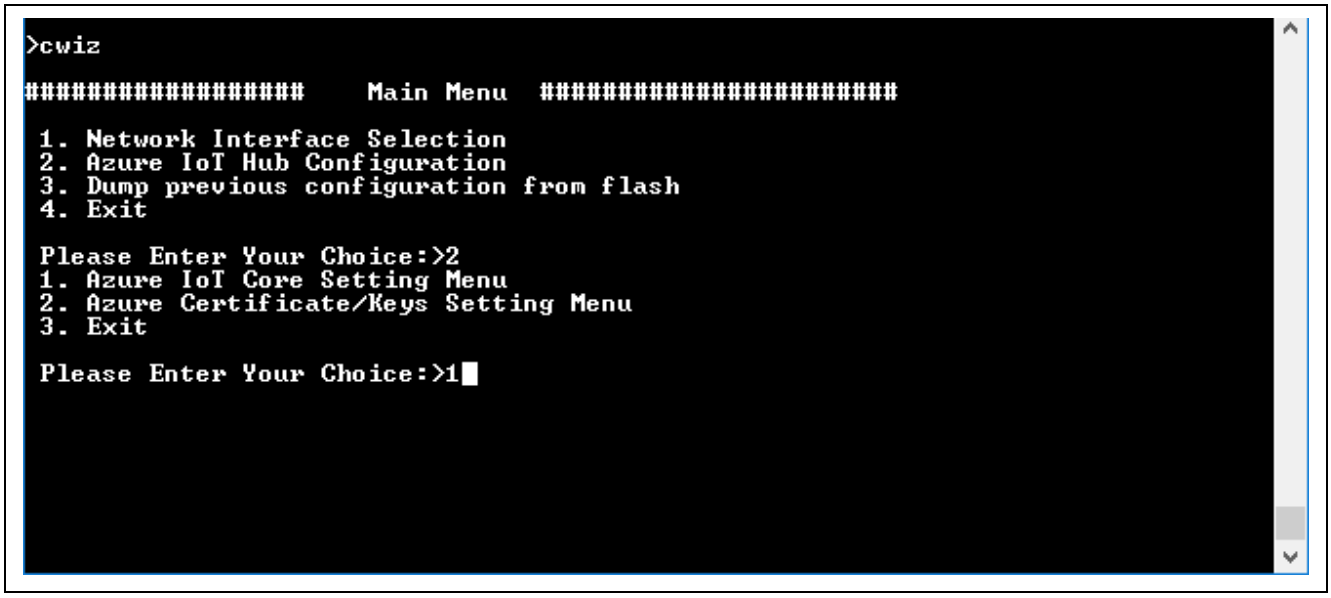

図 38　Azure IoT Hub 設定メニュー

(1)　Azure IoT Core 設定メニュー（Azure IoT Core Setting Menu）

[Azure IoT Core configuration] (Azure IoT Core 設定) メニューで「**1**」と **Enter** キーを押し、Azure IoT Cloud の設定を行います。[Azure IoT settings] (Azure IoT 設定メニュー) には、以下の図に表示されているように、以下の情報を入力するためのオプションがあります。

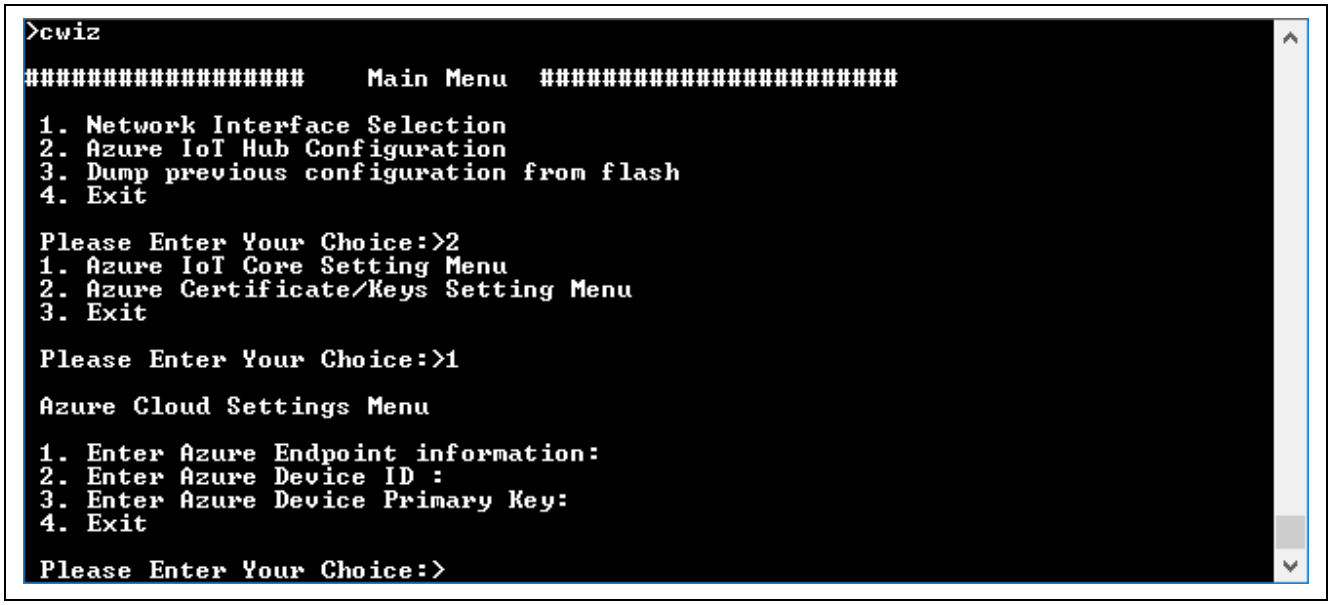

図 39　Azure IoT Hub デバイス設定メニュー

注記：

1. オプション **[1]** の [Enter Endpoint Information] (エンドポイント情報の入力) を選択した場合、「**IoTHubName'.azure-devices.net**」と入力します。この情報は、デバイス詳細のスナップショット（図 19 を参照） に表示されているように、IoT Hub 上のデバイス作成の一部として作成された接続文字列から取得することができます。
   注記：　「IoTHubName」は IoT ハブの名前です。
2. オプション **[2]** の **[Enter Device ID Information]** (デバイス ID 情報の入力) を選択した場合、「**DeviceId**」と入力します。この情報は、**[Device details]** (デバイス詳細) のサンプルスナップショット **図 19** に表示されているように、接続文字列から取得することもできます。
3. オプション **[3]** の **[Primary Key Information]** (プライマリ鍵の情報) を選択した場合、「**Primary Key**」と入力します。この情報は、**図 19** のスナップショットの [**Device Details**] (デバイス詳細) に表示されているように、接続文字列から取得することもできます。
4. 文書の一部としてスナップショットに示されている Azure クレデンシャル情報はサンプル情報でありデモンストレーションの目的のためだけのものです。これらの情報は 3.3.3 章の一部として作成したクレデンシャル情報とともに使用される必要があります。
5. Azure IoT 設定のスナップショットを以下の図 に示します。

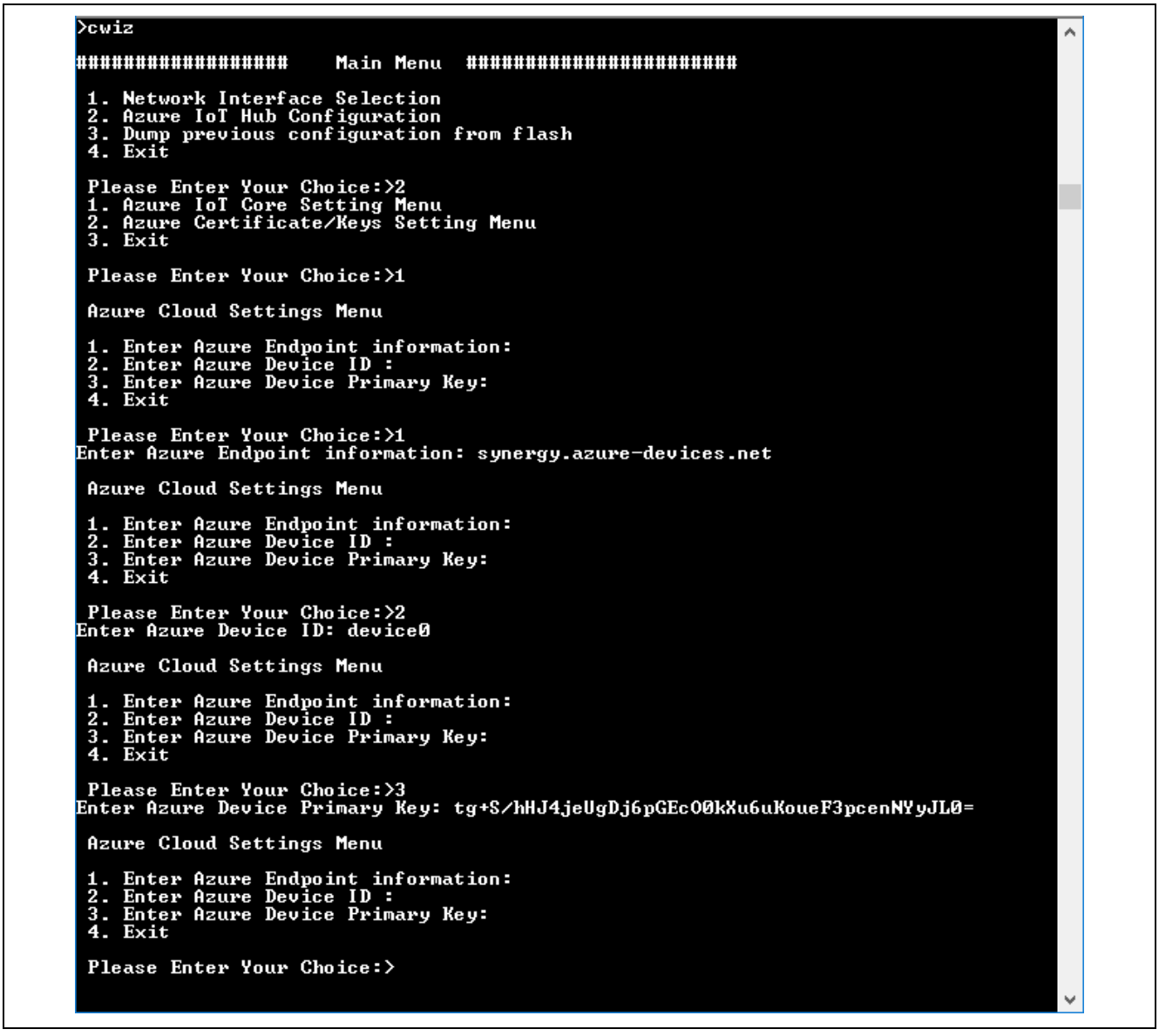

図 40　Azure IoT Hub のデバイス設定のスナップショット

4.4.1.3　Azure の証明書/鍵の設定メニュー（Azure Certificate/Key Settings Menu）

[Azure IoT Hub configuration] (Azure IoT ハブ設定) メニューで「**2**」と **Enter** キーを押し、[Azure Certificate/Keys] (Azure 証明書と鍵) の設定を行います。
[Azure Certificate/Keys settings Menu] (Azure 証明書/鍵の設定) には、ルート CA (認証局) を .pem 形式で入力するためのオプションがあります。
このアプリケーションプロジェクトの一部として付属している汎用の証明書をテキストエディタで開き、コピーしてシリアルコンソールに貼り付け、**Enter** キーを押します。
Azure Cloud に対応するルート CA 証明書 (**azure_rootCA.pem**) は、パッケージの一部として同梱されています。

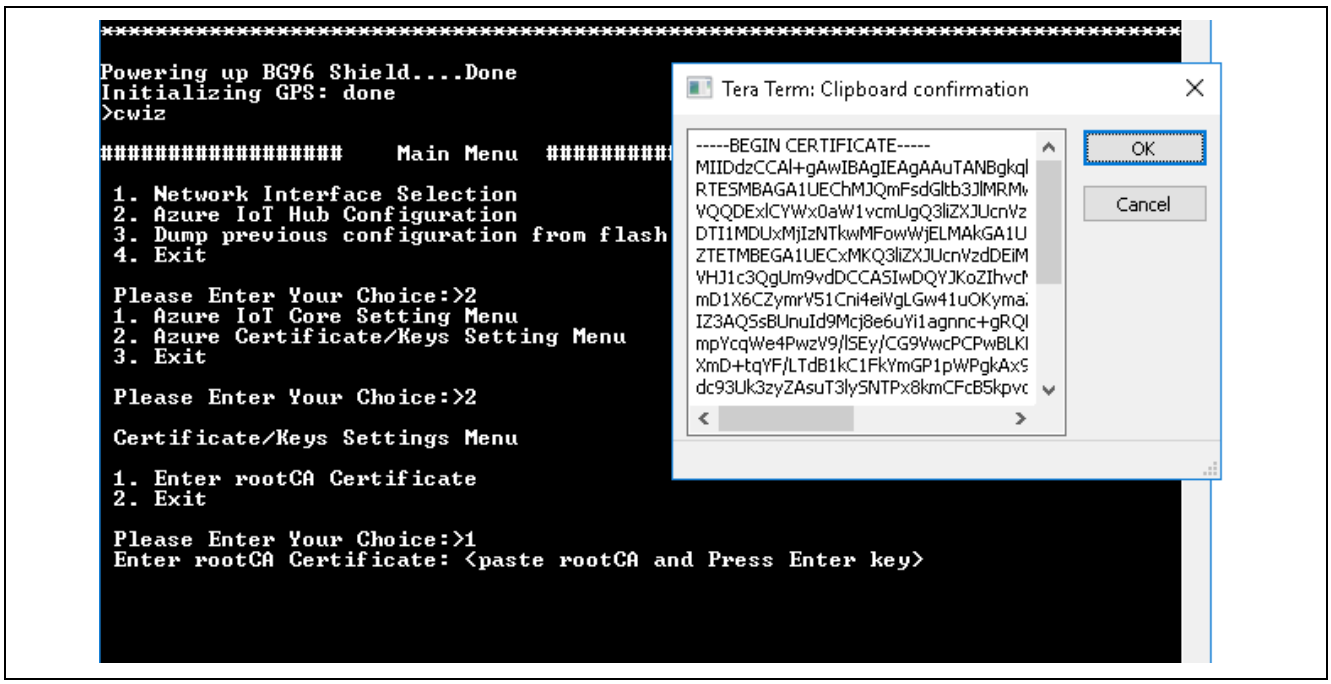

図 41　証明書/鍵の設定メニュー

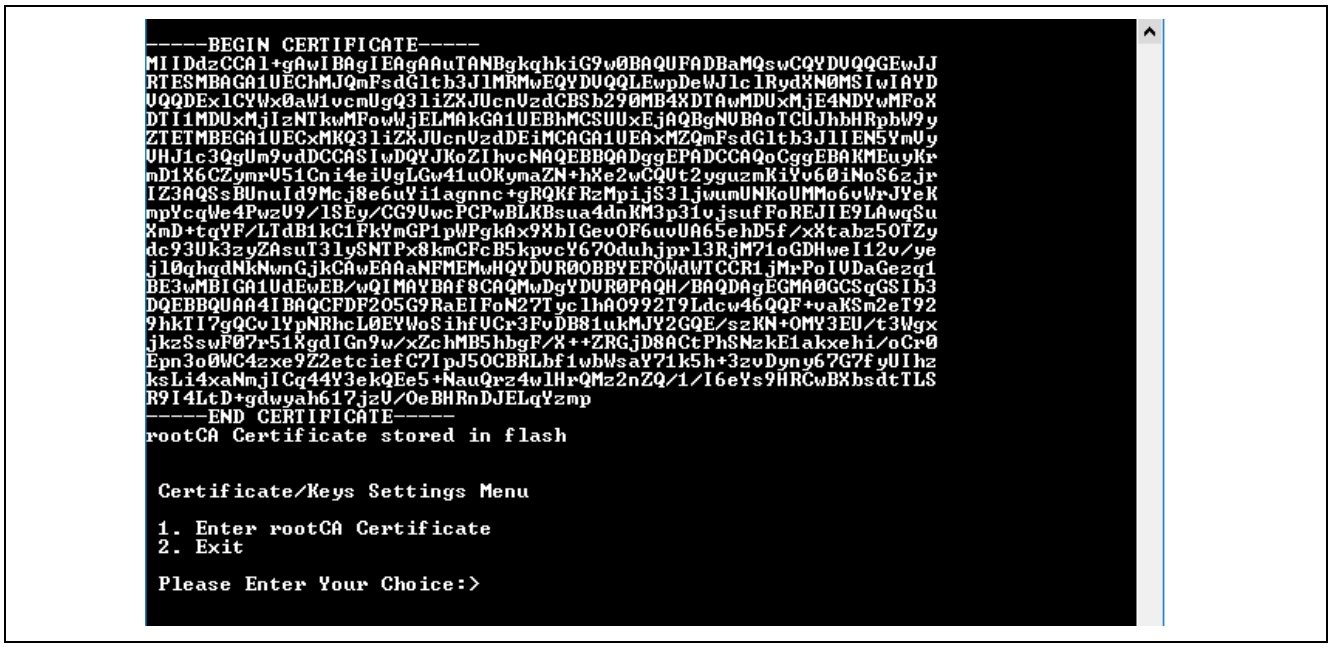

図 42　Azure IoT Hub のルート CA のスナップショット

選択した設定項目が内部フラッシュに保存されます。後で、デバイスが Azure IoT Hub に接続する際にこの設定項目が使用されます。

4.4.1.4　以前の設定のダンプ（Dump Previous Configuration）

[Main Menu] (メインメニュー) からオプション **[3]** を選択すると、以下の図 のように、以前に選択したネットワークや、Azure IoT Hub サービス設定の各種オプションが内部フラッシュからダンプされます。

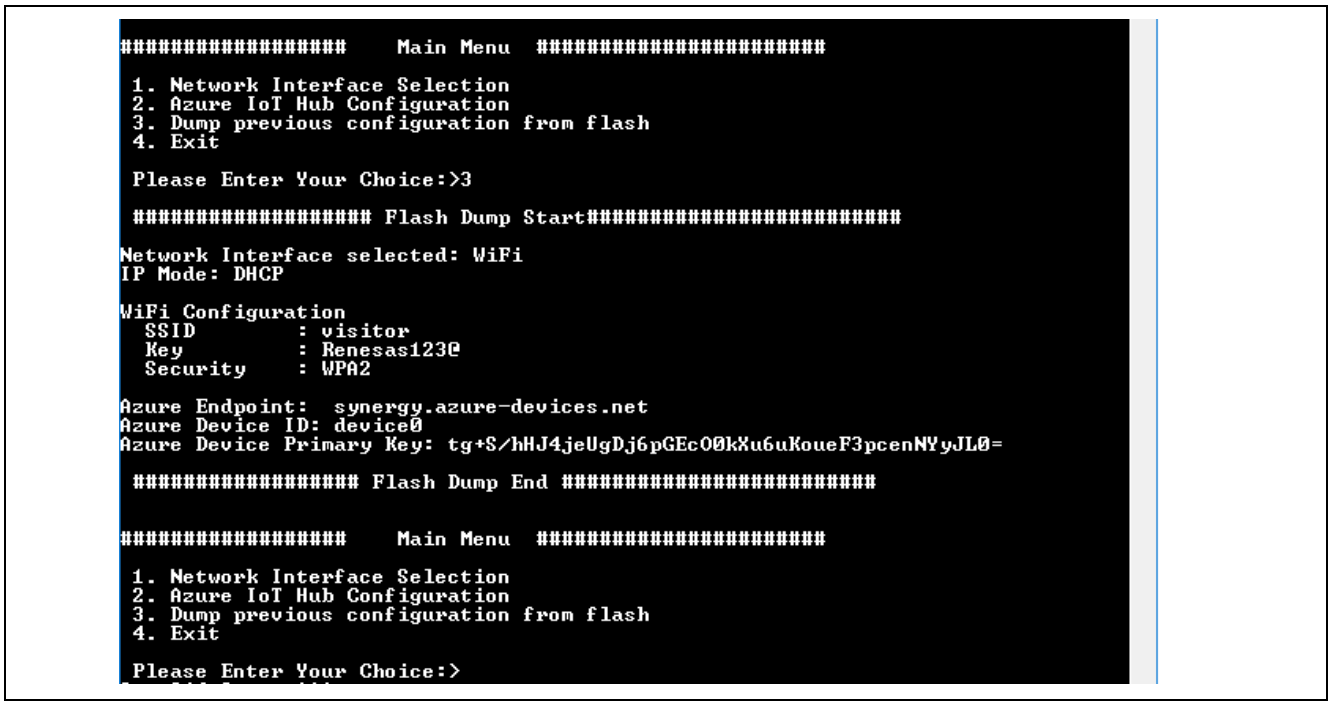

図 43 設定メニューのダンプ

4.4.2　デモの開始/終了コマンド（Demo Start/Stop Command）

Synergy Cloud 接続アプリケーションのデモを開始するには、CLI コンソールで「**demo start**」コマンドを入力します。

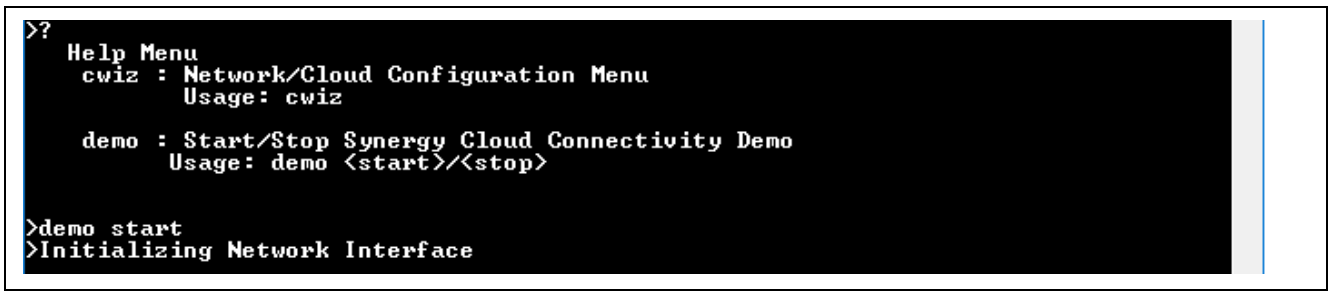

図 44　Azure でデモを実行

アプリケーションフレームワークは、事前に設定したネットワークインタフェースや、IoT サービス設定の各種オプションを内部フラッシュから読み出し、それらの設定の妥当性を検証します。内容が妥当な場合、アプリケーションフレームワークはネットワークインタフェースを初期化し、Azure IoT Hub を使用して MQTT 接続を確立します。

このアプリケーションは定期的 (5 秒ごと) にウェイクアップし、入力イベントフラグの状態を確認します。CLI に「**demo start/stop**」コマンドを入力すると、フラグの状態がセットされて 1 になります。ユーザが「**demo stop**」コマンドを入力するまで、このアプリケーションは以下の機能を定期的に実行します。

1. 選択に基づいて、通信インタフェース (イーサネットまたは Wi-Fi またはセルラー) の初期化。
2. Azure IoT Cloud インタフェースの初期化。
3. センサデータの読み出しと MQTT トピックへのセンサデータの定期的な発行。
4. 受信した MQTT メッセージのタイプに基づいて LED の状態を更新。

「**demo stop**」コマンドが発行された場合、IoT クラウドインタフェースの終了処理の後、MQTT メッセージの発行を停止し、自らの内部キューに保存されている保留中の MQTT メッセージすべてをクリアします。

4.5　デモの確認（Verifying the Demo）

以下の説明に従い、 この Synergy Cloud 接続アプリケーションプロジェクトの機能を確認してください。

注記：3.3 章の説明に従って、Azure Cloud IoT アカウントの作成、Azure IoT Hub に合わせたデバイスのセットアップ、ネットワーククレデンシャルの設定、CLI を使用した Azure IoT 証明書の作成、アプリケーションプロジェクトのコンパイルと PK-S5D9 キットあるいは AE-CLOUD2 および AE-CLOUD1 キットへのダウンロードが完了していることを想定しています。

4.5.1　Synergy Cloud 接続デモの実行（Running the Synergy Cloud Connectivity Demonstration）

このアプリケーションのデモを実行するには、シリアルコンソールに「**demo start**」コマンドを入力します。

「**demo start**」を実行した後、以下の画像のように、このアプリケーションはネットワークインタフェースの設定、Azure IoT Hub との接続の確立、定期的な (5 秒ごと) センサデータの発行を開始します。

```
>?
   Help Menu
    cwiz : Network/Cloud Configuration Menu
           Usage: cwiz

    demo : Start/Stop Synergy Cloud Connectivity Demo
          Usage: demo <start>/<stop>


>demo start
>Initializing Network Interface

Stopping DHCP client.
done

Waiting for IP address.done


IP Configuration
================
  Interface   : Ethernet
  Mode        : DHCP

  IP Address  : 143.103.16.38
  Netmask     : 255.255.255.128
  Gateway     : 143.103.16.2
  DHCP Server : 143.103.10.56
  DNS Server  : 143.103.10.62
**************************************
Initializing Cloud Interface:Enter MQTT Azure open
connection to MQTT Endpoint Successfull

 Subscribed to the following topics
devices/device0/messages/devicebound/#
 Publish to the following topic
$iothub/twin/PATCH/properties/reported/?$rid=0
{
"xacc" : "-0.09",
"yacc" : "0.34",
"zacc" : "9.96",
"temperature" : "94.33",
"pressure" : "997.72",
"humidity" : "28.49",
"longitude" : "",
"latitude" : "",
"spl" : "0.00"
}

{
"xacc" : "-0.18",
"yacc" : "0.40",
"temperature" : "94.37",
"pressure" : "997.74",
"humidity" : "28.43",
"longitude" : "",
"latitude" : "",
"spl" : "0.00"
}
```

図 45　CLI から実行したアプリケーションのデモのスナップショット

4.5.2　デバイスと Azure MQTT ブローカーにおける MQTT メッセージのモニタ（Monitoring MQTT messages on Device and Azure MQTT Broker）

デモを開始した後は、センサのデータが定期的に Azure IoT Hub 宛に発行されます。発行されたデータを表示するには、2 つのオプションが利用できます。**1) デバイス側で CLI ログを使用します。2) IoT Hub 側で、デバイスに対応する [Device Twin] (デバイスツイン) を使用します。**

4.5.2.1　デバイス側でのモニタ（Monitoring on the Device Side）

デバイス側でコマンドラインインタフェース (**CLI**) を使用して **demo start** を実行した時点で、コンソールログはセンサデータの収集を開始し、これらのデータがクラウド宛に発行されます。IoT Hub からデバイスを制御する、つまり LED の ON/OFF (点灯/消灯) を切り替える場合、ユーザはコンソールでメッセージを目視することができます。

4.5.2.2　Azure IoT Hub 側でのモニタ（Monitoring on the Azure IoT Hub）

デバイスツイン（device twin）を使用して、IoT Hub 側でセンサデータを表示することもできます。これらのデータは、「reported」タグの下に JSON 形式で記録されています。センサデータに加えて、デバイスツインに関する詳細も利用できます。まず、[IoT Hub] で[Device] (デバイス) をクリックし、[Device details] (デバイス詳細) ブレードを開きます。[Device details] (デバイス詳細) ブレードの **[Device Twin]** (デバイスツイン) をクリックすると、**以下の図 のように、[Device Twin] (デバイスツイン) ブレードが開きます。**

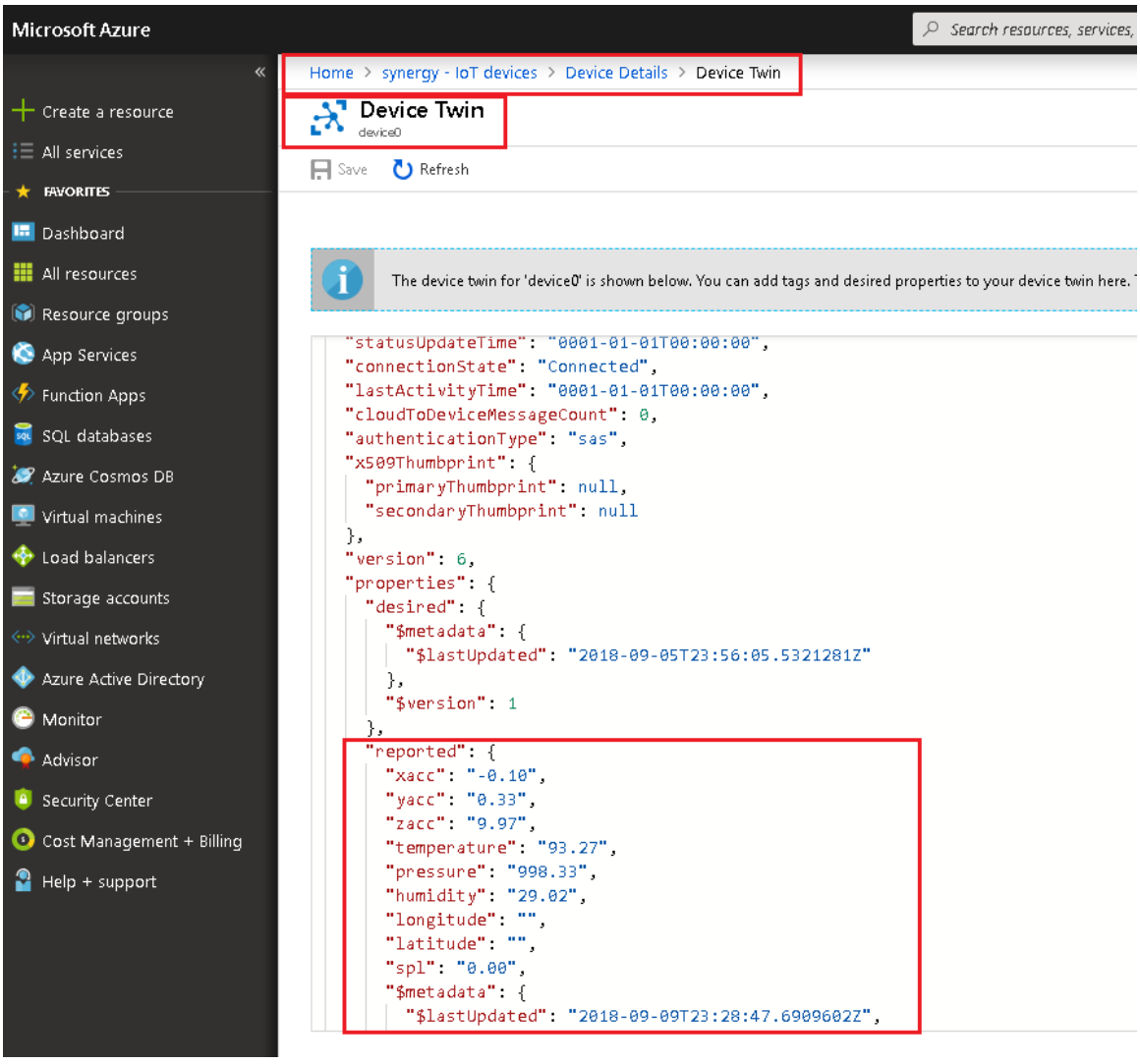

図 46　デバイスツイン内のセンサデータのスナップショット

4.5.3 Azure IoT Hub からの MQTT メッセージの発行（Publishing the MQTT message from Azure IoT Hub）

以下の表に、 LED のさまざまな点灯状態を指示する MQTT メッセージを示します。このメッセージを発行して、PK-S5D9 キットあるいは AE-CLOUD2 および AE-CLOUD1 キット上の LED の ON/OFF を切り替えることができます。

注記：以下の表で [Message] 列に掲載されている各メッセージは**大文字と小文字を区別する**ので、これらのメッセージを使用して LED の ON/OFF (点灯/消灯) を切り替えるときに、大文字小文字に注意を払う必要があります。

表 1　PK-S5D9 キットあるいは AE-CLOUD1 および AE-CLOUD2 キット上にあるユーザ LED の ON/OFF の切り替え

LED 状態	メッセージ
赤の LED が点灯	{"state":{"desired":{"Red_LED":"ON"}}}
赤の LED が消灯	{"state":{"desired":{"Red_LED":"OFF"}}}
黄色の LED が点灯	{"state":{"desired":{"Yellow_LED":"ON"}}}
黄色の LED が消灯	{"state":{"desired":{"Yellow_LED":"OFF"}}}
緑の LED が点灯	{"state":{"desired":{"Green_LED":"ON"}}}
緑の LED が消灯	{"state":{"desired":{"Green_LED":"OFF"}}}

メッセージを発行するには、以下の手順を使用します。

1. [IoT Hub] で[Device] (デバイス) をクリックし、**[Device details]** (デバイス詳細) ブレードを開きます。**[Device details]** (デバイス詳細) ブレードの **[Message to Device]** (デバイス宛のメッセージ) をクリックします。以下の図のように、[Message to Device] (デバイス宛のメッセージ) ブレードが開きます。
2. [Message Body] (メッセージ本文) の下に、LED の ON/OFF (点灯/消灯) の切り替えに対応するメッセージを貼り付けます。たとえば、**{"state":{"desired":{"Red_LED":"ON"}}}** です。その後、**[Send Message]** (メッセージ送信) をクリックします。

   注記:[Key] (鍵) と **[Value]** (値) の各フィールドは空白のままにしてください。
3. 実際のデバイスでメッセージに対応する LED が ON (点灯) したことを確認できます。また、コンソールに「**“Red LED ON**」(赤い LED が点灯) というメッセージが出力されます。

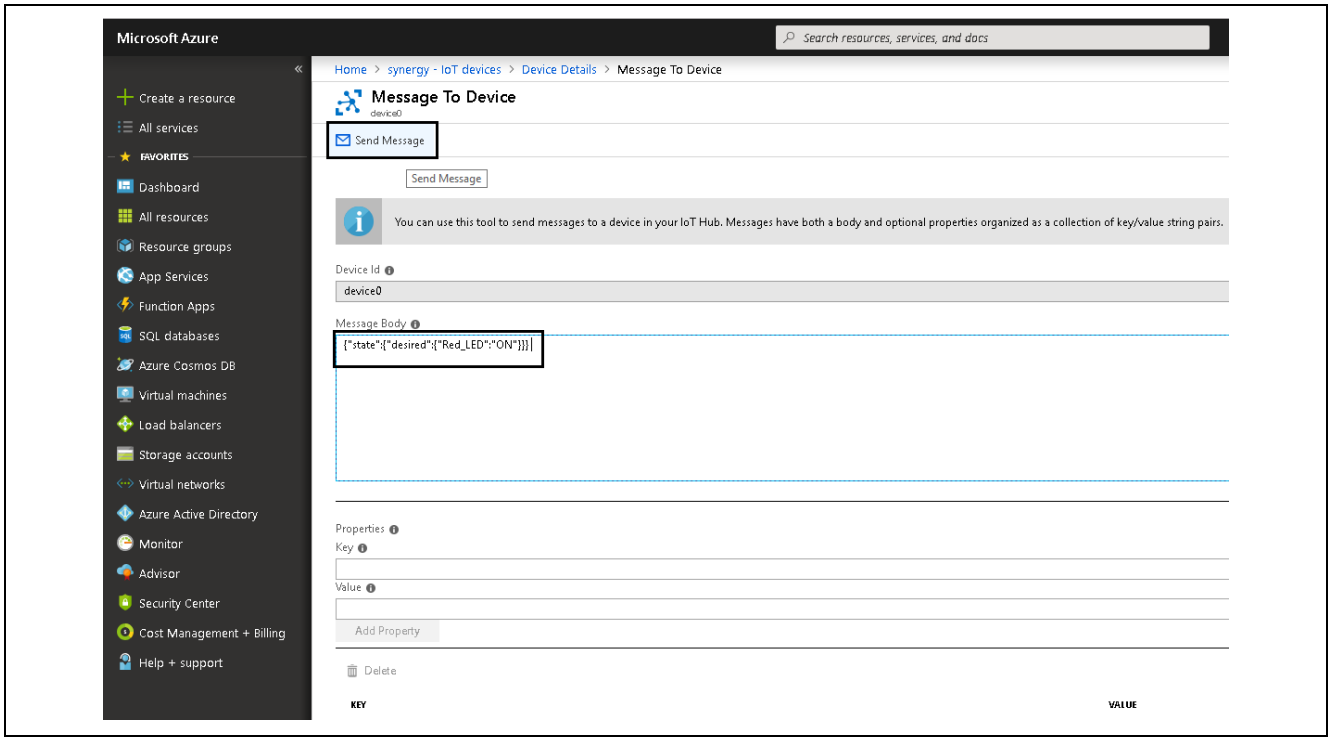

図 47　Azure IoT Hub からデバイス宛のメッセージ送信のスナップショット

4.5.4 Synergy Cloud 接続デモの停止（Stopping the Synergy Cloud Connectivity Demonstration）

デモを停止するには、「**demo stop**」 コマンドを入力します。このコマンドを発行すると、IoT クラウドインタフェースの初期化の取り消し、MQTT メッセージの発行の停止、自らの内部キューに保存されている保留中の MQTT メッセージすべてのクリアが実施されます。

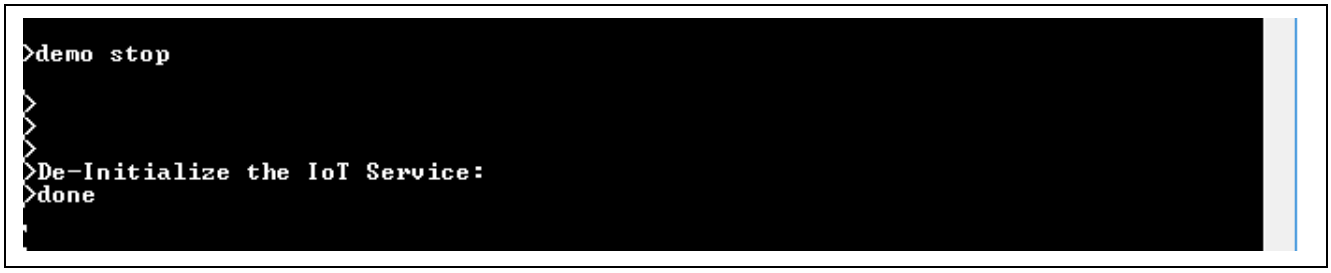

図 48　アプリケーションのデモ停止のシーケンス

5.　次の手順（Next Steps）

開発ツールとユーティリティの詳細については、

https://www.renesas.com/jp/ja/products/synergy/software/tools.html をご覧ください。

開発ツールとユーティリティをダウンロードするには、

https://www.renesas.com/jp/ja/products/synergy/gallery.html をご覧ください。

Renesas Synergy Module Guides 関連リンク：https://www.renesas.com/jp/ja/products/synergy.html

6.　MQTT/TLS の参考資料（MQTT/TLS Reference）

- SSP 1.5.3 ユーザーズマニュアルは、Renesas Synergy™ WEB (https://www.renesas.com/jp/ja/products/synergy/software/ssp.html#) からダウンロードできます
- Azure IoT のドキュメント https://docs.microsoft.com/en-us/azure/iot-hub/

7.　既知の問題と制限（Known Issues and Limitations）

Wi-Fi およびセルラー接続を使用しながらこのデモを実行しているユーザが、**demo stop** コマンドを使用してデモを停止した後に **demo start** コマンドを使用してデモをもう一度実行しようとすると、デモは Google Cloud IoT MQTT ブローカーへの再接続に失敗します。

現時点で、Azure IoT ハブを使用したデバイス相互間の直接通信はサポートされていません。

ホームページとサポート窓口

以下の URL にアクセスし、Synergy プラットフォームの詳細を確認し、関連するドキュメントをダウンロードし、サポートをご活用ください。

Synergy ソフトウェア　https://www.renesas.com/jp/ja/products/synergy/software.html
Synergy ソフトウェアパッケージ　https://www.renesas.com/jp/ja/products/synergy/software/ssp.html
ソフトウェアアドオン　https://www.renesas.com/jp/ja/products/synergy/software/add-ons.html
ソフトウェア用語集　https://www.renesas.com/jp/ja/products/synergy/software/ssp/glossary.html
開発ツール　https://www.renesas.com/jp/ja/products/synergy/software/tools.html

Synergy ハードウェア　https://www.renesas.com/jp/ja/products/synergy/hardware.html
マイクロコントローラ　https://www.renesas.com/jp/ja/products/synergy/hardware/microcontrollers.html
MCU 用語集　https://www.renesas.com/jp/ja/products/synergy/hardware/microcontrollers/glossary.html
主要パラメータでの検索　https://www.renesas.com/jp/ja/search/parametric-search.html
キット　https://www.renesas.com/jp/ja/products/synergy/hardware/kits.html

Synergy ソリューション Gallery　https://www.renesas.com/jp/ja/products/synergy/gallery.html
パートナープロジェクト　https://www.renesas.com/jp/ja/products/synergy/gallery/partner-projects.html
アプリケーションプロジェクト　https://www.renesas.com/jp/ja/products/synergy/gallery.html
（上記 WEB ページの中間部を参照）

セルフサービスサポートリソース：
ドキュメント　https://www.renesas.com/jp/ja/products/synergy/support.html
ナレッジベース/FAQ　https://ja-support.renesas.com/knowledgeBase/category/30643
フォーラム（英語）　https://renesasrulz.com/synergy/
フォーラム（日本語）　https://japan.renesasrulz.com/cafe_rene/
トレーニング　https://www.renesas.com/jp/ja/support/training.html
ビデオ　https://www.youtube.com/playlist?list=PLgUXqPkOStPu_uZCwn_1tM2QZIRDhcbCR
技術質問 問い合わせ先(MyRenesas 登録必要)　https://ja-support.renesas.com/dashboard

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2018.12.21	-	初版 英文版 R11EU0337EU0100 Rev.1.00 を翻訳
1.01	2019.01.10	-	Renesas Synergy™ USB CDC ドライバに変更 AE-wifi1 のサポート終了を追記 http リンク先を修正
1.02	2019.05.07	-	英文版 R11EU0337EU0101 Rev.1.01 の内容をフィードバック

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。お客様の機器・システムの設計において、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には、お客様の責任において行ってください。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、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、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。
5. 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。
6. 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。
7. 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。
8. 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。
9. 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。
10. 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。
11. 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。
12. 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.4.0-1　2017.11)

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