

#1 FPGA回路をリモートで書換 → 壮大なLチカハンズオン

C. Microsoft Azureの設定とIoT Edgeデバイスの登録

1. この章の内容とやること

この章では、今回利用するMicrosoft AzureのサービスであるContainer RegistryとIoT Hubの設定を行います。その後、DE10-NanoボードをIoT Edgeデバイスとして登録し、IoT Hubとの疎通を確認します。最後に、DE10-Nanoボードに出来合いのIoT Edgeモジュールを配置し、その動作を確認します。

本ハンズオンで使用するAzureサービスは、無料アカウントでは受講できません。利用料金の詳細については[G. 2. 今回のAzureサービスの利用料金](#)で解説します。

下記の項目の情報については、資料通り進める場合はこちらで指定した情報を入力いただきます。もちろん変更いただいて構いませんが、異なる情報を入力される場合は、ご自身で情報を適宜読み替えてください。

- リソースグループ名
- IoT Hub名
- IoT Edgeデバイス名
- IoT Edgeモジュール名

ということで、まずはWebブラウザでAzure Portalにサインインしておきましょう。

<https://portal.azure.com/>

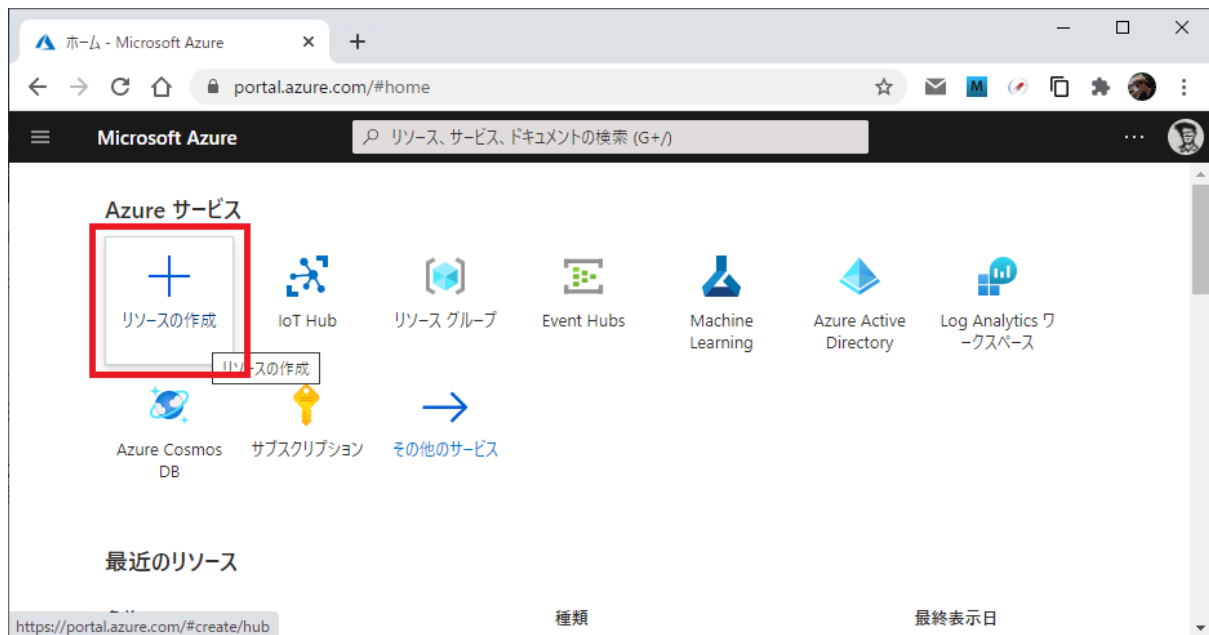
2. Container Registryの作成と設定

Azure Container Registry (ACR) とは、一言であらわすとAzure専用のコンテナイメージの格納庫です。 Dockerを含む [OCI \(Open Container Initiative\)](#) の標準に準拠したコンテナを対象として、イメージの更新やコンテナのビルド、配置適用の自動化を支援します。

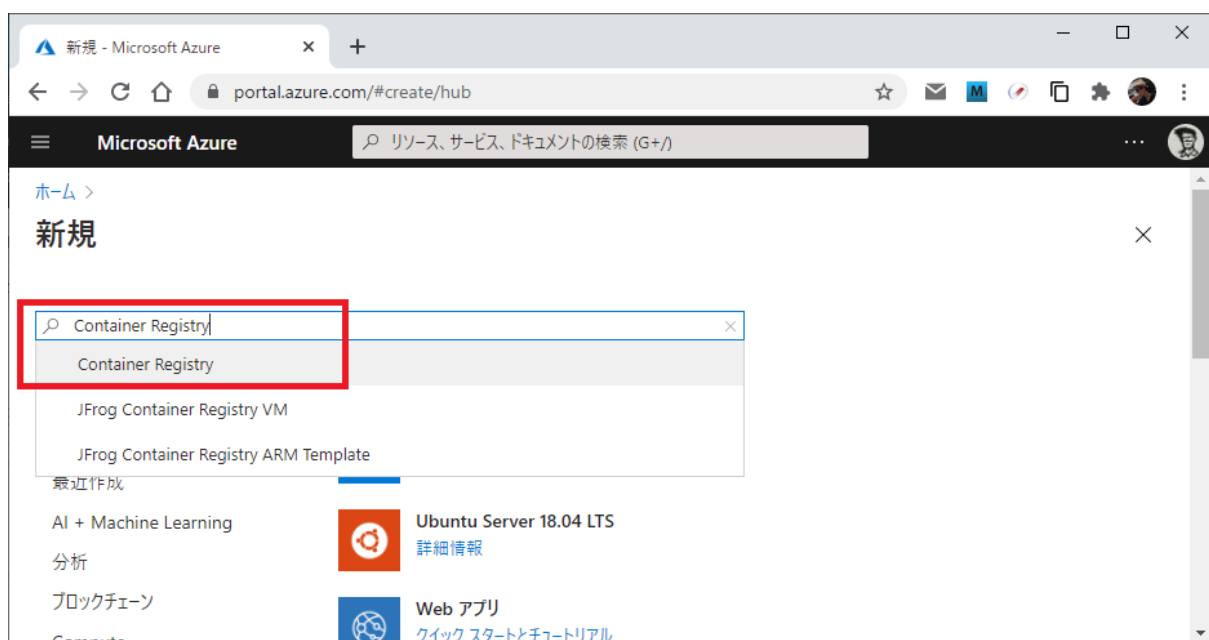
参考: Azure Container Registry | Microsoft Azure

<https://azure.microsoft.com/ja-jp/services/container-registry/>

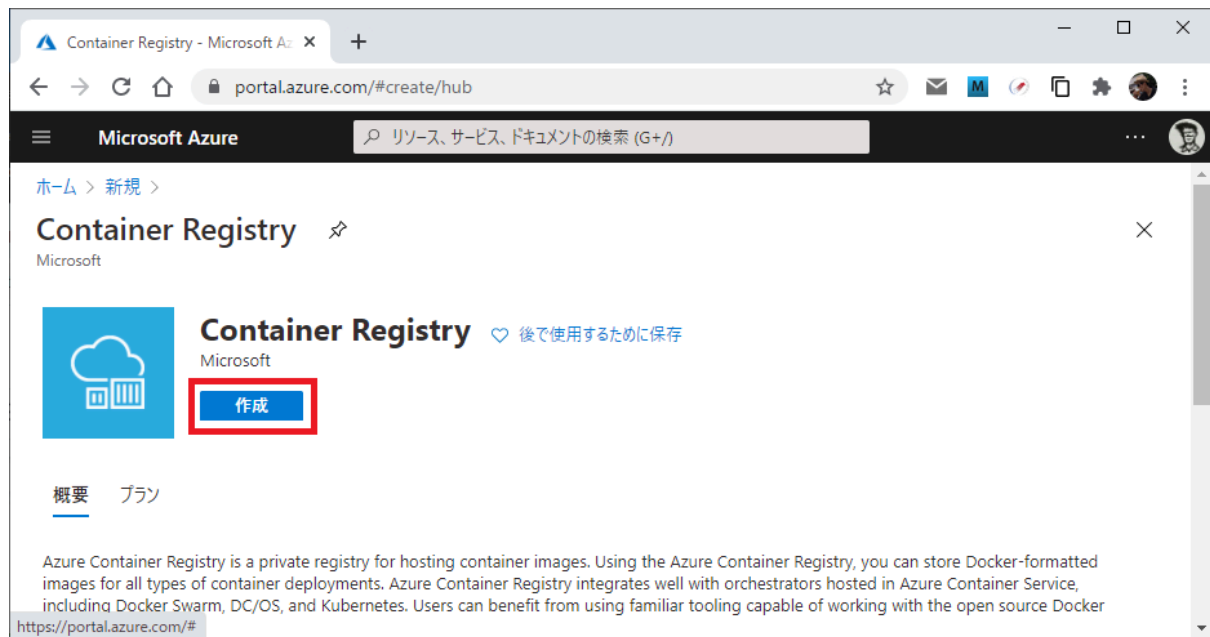
Azure PortalのTOPページにある **【+ リソースの作成】** をクリックします。



"**Marketplace を検索**" のフィールドに、 **Container Registry** を入力して、Enterキーを押すか表示された候補をクリックします。



下記の画面になったら【作成】をクリックします。



「コンテナレジストリの作成」画面になります。

[基本]の設定を行います。

- プロジェクトの詳細
 - ①【サブスクリプション】は、ご自身のものを選択してください。
例では **Microsoft Azure** を選択しています。
 - ②【リソースグループ】は、**【新規作成】** をクリックし、
 - ③【名前】に **intelfpga** を入力して **【OK】** をクリックしてください。
- インスタンスの詳細
 - ④【レジストリ名】は、**グローバルに一意のものを付ける**必要があります。
5文字以上50文字以下で、英字と数字のみ使用できます。
 - 例えば **intelfpgatakasehideki** などハンドル名を入れましょう。
 - ※のところが緑のチェックになっていることを確認してください。
 - ⑤【場所】は **東日本** を選択してください。
 - ⑥【SKU】は **Basic** を選択してください。
 - ACRの料金プランの選択です。今回はBasicで十分です。
- ⑦**【確認および作成】** をクリックします。

The screenshot shows the 'Container Registry creation' page in the Azure portal. A modal window titled '新規作成' (Create New) is open, showing the '名前' (Name) field with 'intelfpga' entered and a green checkmark. The modal also has 'OK' and 'キャンセル' (Cancel) buttons. Red circles and arrows highlight the steps: ① points to the 'サブスクリプション' (Subscription) dropdown, ② points to the '新規作成' (Create New) link for the resource group, ③ points to the '名前' (Name) field in the modal, ④ points to the 'レジストリ名' (Registry Name) field with 'intelfpgatakasehideki' and a green checkmark, ⑤ points to the '場所' (Location) dropdown with '東日本' (East Japan), ⑥ points to the 'SKU' dropdown with 'Basic', and ⑦ points to the '確認および作成' (Review and Create) button at the bottom.

コンテナ レジストリの作成 - Microsoft Azure

portal.azure.com/#create/Microsoft.ContainerRegistry

Microsoft Azure リソース、サービス、ドキュメントの検索 (G+/I)

ホーム > 新規 > Container Registry >

コンテナ レジストリの作成

基本 ネットワーク 暗号化 タグ 確認および作成

Azure Container Registry を使用すると、すべての種類のコンテナ配置についてプライベート保存、管理できます。Azure コンテナ レジストリは、既存のコンテナ開発と配置パイプラインを使用すると、オンデマンドで Azure にコンテナ イメージをビルドしたり、ソース コードの更新、コンテナ イメージのビルドをトリガーしたりすることができます。 [詳細情報](#)

プロジェクトの詳細

サブスクリプション *

リソース グループ *

① Microsoft Azure

(新規) intelfpga

② 新規作成

③ 名前 *

intelfpga

OK キャンセル

④ レジストリ名 *

intelfpgatakasehideki

⑤ 場所 *

東日本

⑥ SKU *

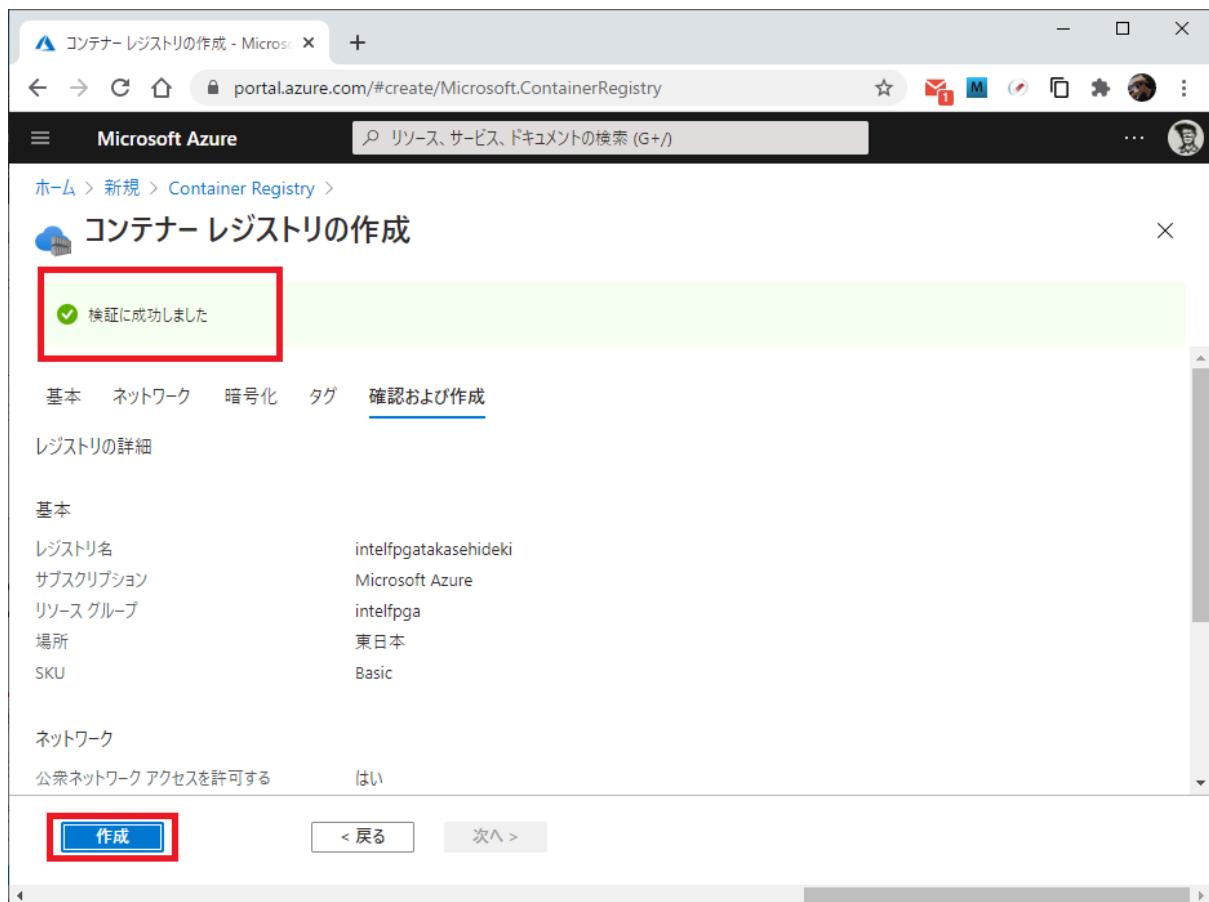
Basic

⑦ 確認および作成

< 戻る 次へ: ネットワーク >

リソース作成のための設定情報を確認します。

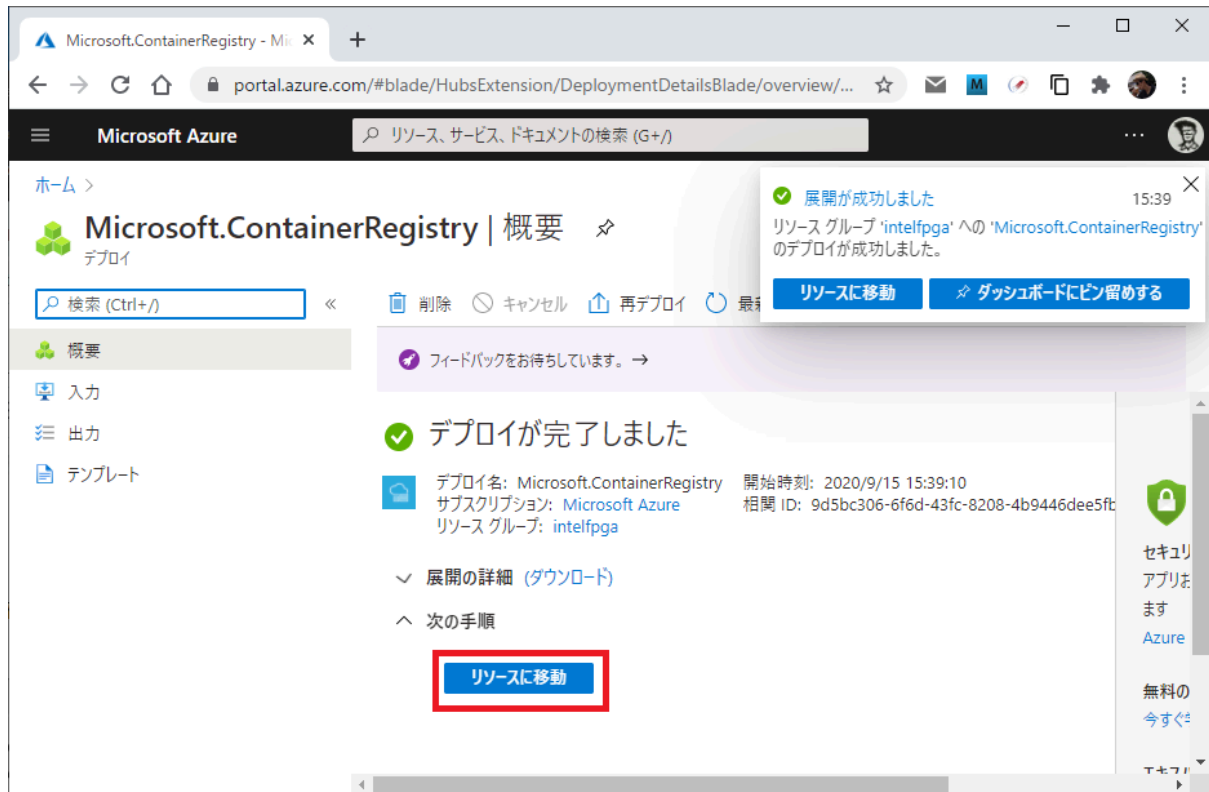
- 「検証に成功しました」と表示されることを確認してください。
- 【作成】をクリックします。



Container Registryのデプロイが進行します。少し待ちましょう。

下記の画面になったら、デプロイが完了しました。

- **【リソースに移動】**をクリックします。



展開されたリソースの設定情報を確認しておきましょう。

- ① レジストリ名: 例では **intelfpgatakasehideki**
 - ② リソースグループ: **intelfpga**
 - ③ サブスクリプション: **Microsoft Azure**
- など

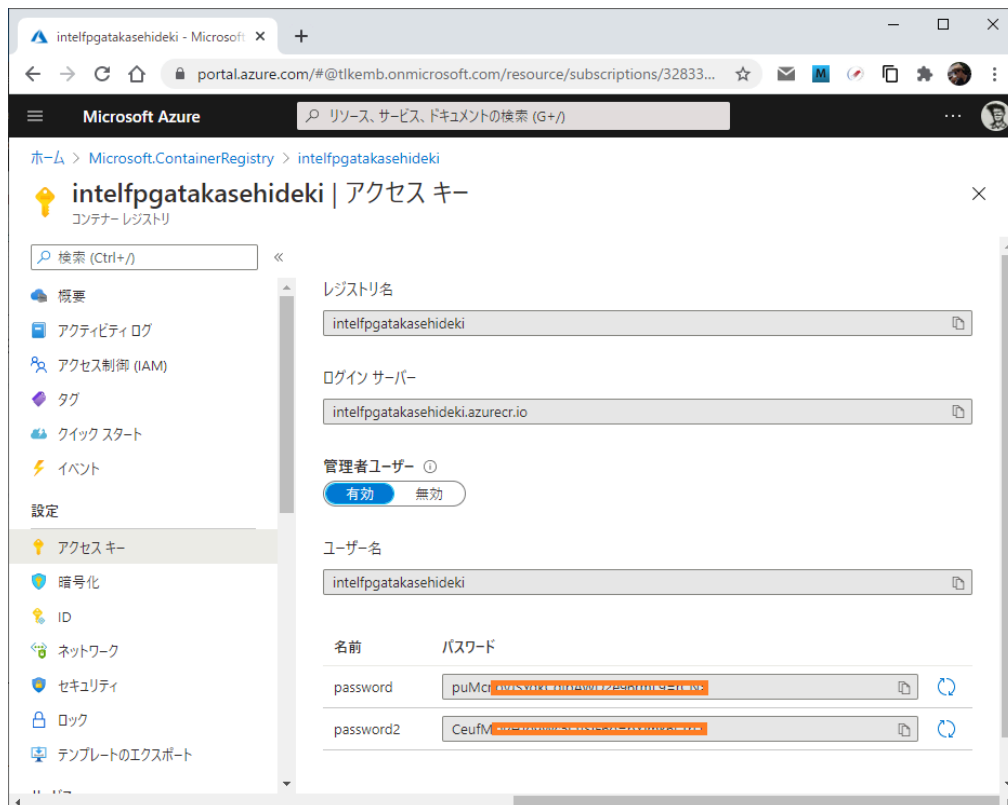


最後に、ACRの管理者ユーザーを有効にして、アクセスキーのパスワードを生成します。

- ① 左横メニューの「設定」>[アクセスキー]を開きます。
- ②【管理者ユーザー】の【有効】をクリックして変更します。

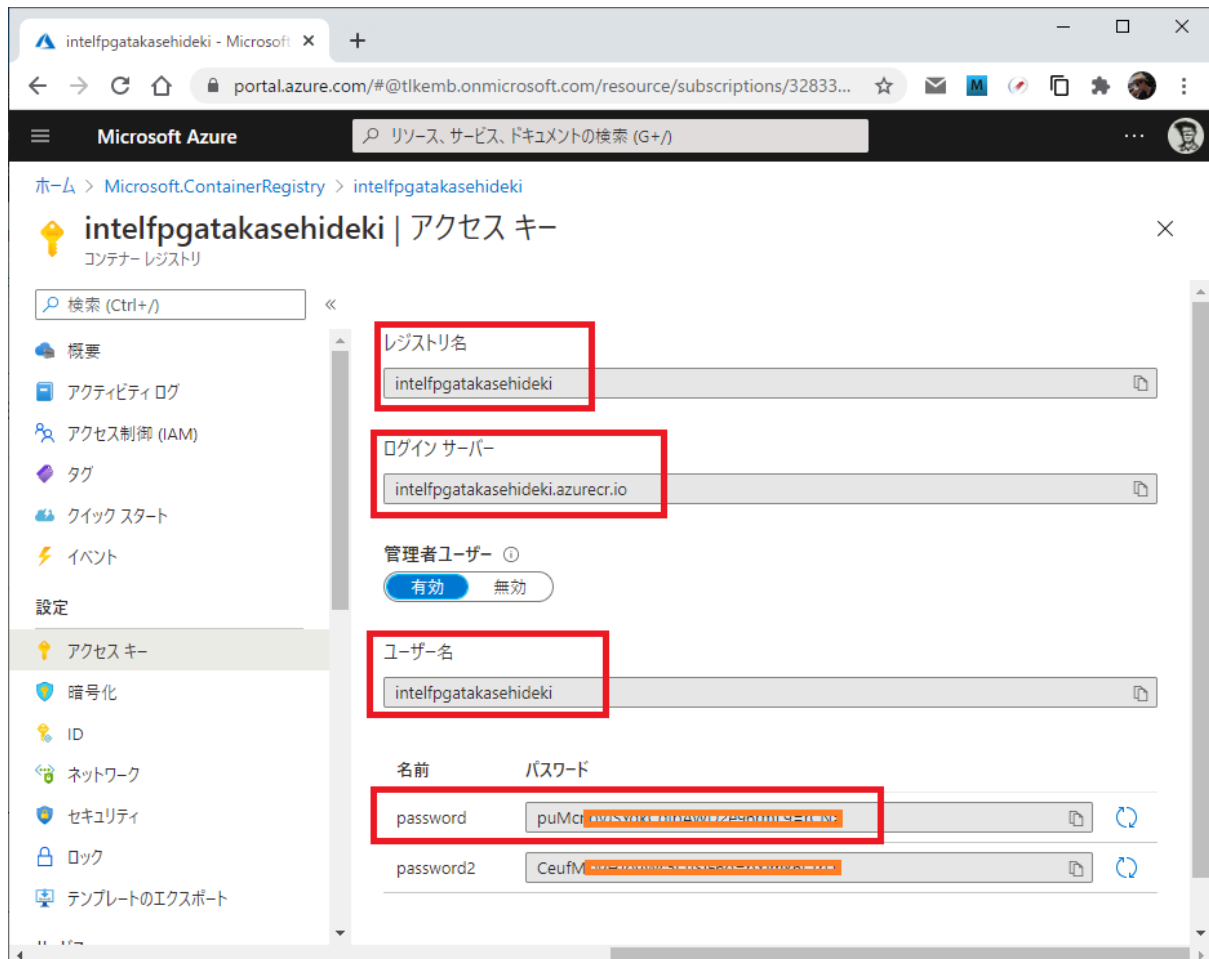


下の画面のようにパスワードが表示されたらOKです。



ここに表示されている次の情報は、VS Codeで開発用のプロジェクトを作成するときに必要となります。メモ帳などにコピーして控えておくとい良いでしょう。

- 「レジストリ名」
- 「ログイン サーバー」
- 「ユーザー名」
- 「password」 ※password2は不要です



ここまでの作業に関するMicrosoftの公式ドキュメントは、下記をご参照ください。

参考: クイックスタート - ポータルでのレジストリの作成 - Azure Container Registry | Microsoft Docs

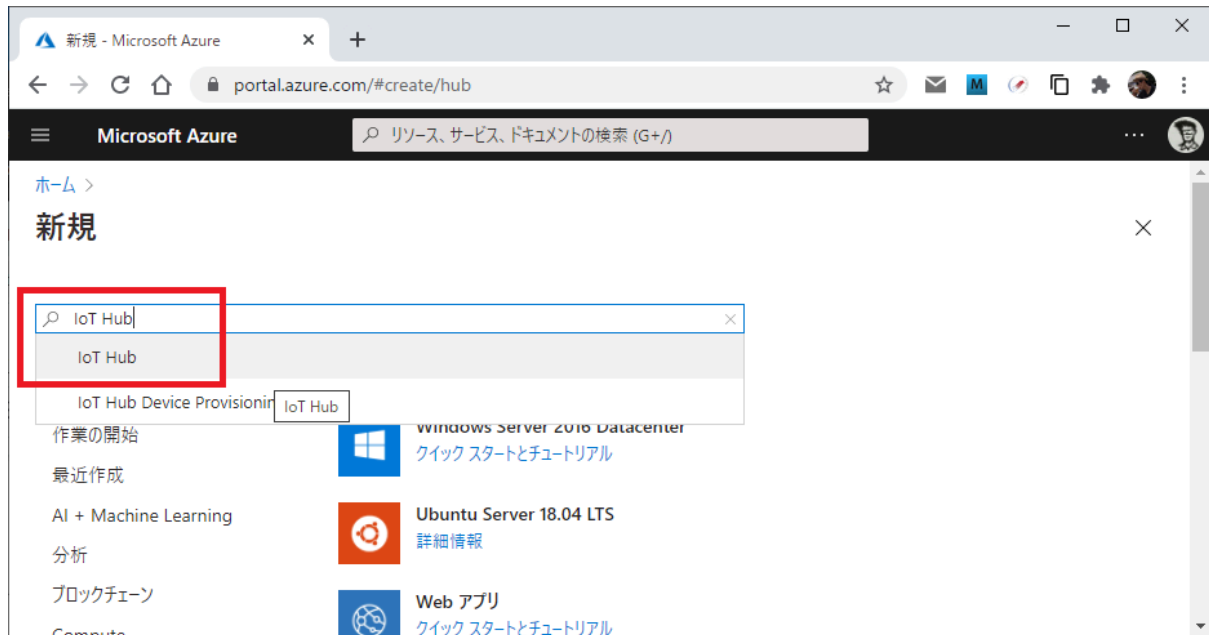
<https://docs.microsoft.com/ja-jp/azure/container-registry/container-registry-get-started-portal>

3. IoT Hubの作成

次はIoT Hubを作成します。

Azure PortalのTOPページに戻って【+ リソースの作成】をクリックします。

"Marketplace を検索" のフィールドに、IoT Hub を入力して、Enterキーを押すか表示された候補をクリックします。



下記の画面になったら【作成】をクリックします。



「IoT ハブ」画面になります。

[基本]の設定を行います。

- プロジェクトの詳細
 - ①【サブスクリプション】は、ご自身のものを選択してください。
例では **Microsoft Azure** を選択しています。
 - ②【リソースグループ】は、先ほど作成した **intelfpga** を選択します。
 - ③【領域】は、**東日本** を選択してください。
 - ④【IoT Hub 名】は、**グローバルに一意のものを付ける必要があります。**
英数字とハイフンが使えます。
 - 例えば **intelfpga-hub-takasehideki** などハンドル名を入れましょう
 - ※のところが緑のチェックになっていることを確認してください。
- 次は[サイズとスケール]を設定します。⑤のところをクリックしてください。
 - まだ 確認および作成 はクリックしません。

IoT ハブ - Microsoft Azure

portal.azure.com/#create/Microsoft.IoTHub

Microsoft Azure

ホーム > 新規 > IoT Hub >

IoT ハブ

Microsoft

基本 ネットワーク **⑤ サイズとスケール** タグ 確認および作成

何十億にも及ぶ IoT アセットに接続し、監視および管理できるように、IoT ハブを作成します。 [詳細情報](#)

プロジェクトの詳細

デプロイとコストを管理するために使用するサブスクリプションを選択します。フォルダーなどのリソース グループを使用して、リソースの整理と管理を行うことができます。

サブスクリプション * ① Microsoft Azure

リソース グループ * ② intelfpga
[新規作成](#)

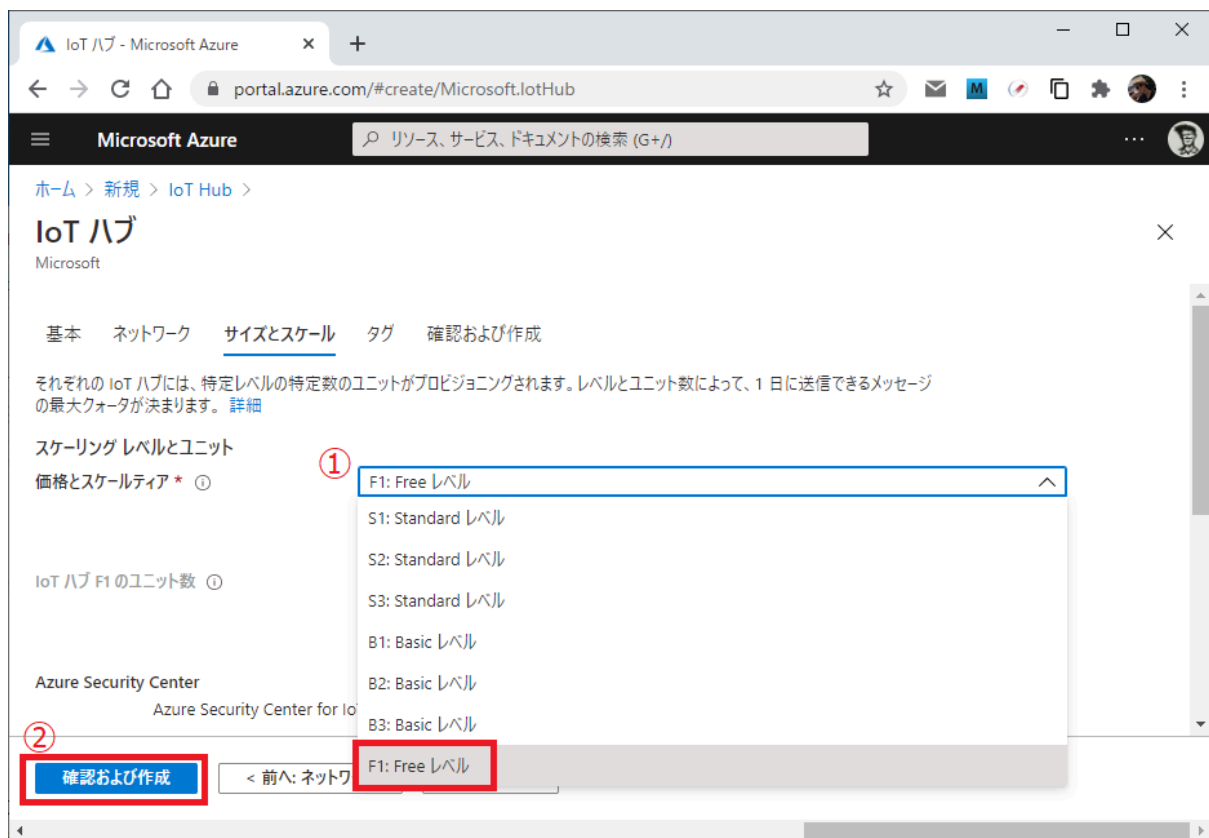
領域 * ③ 東日本

IoT Hub 名 * ④ intelfpga-hub-takasehideki ✓

[確認および作成](#) < 前へ 次へ: ネットワーク > [Automation オプション](#)

[サイズとスケール]の設定を行います。

- スケーリング レベルとユニット
 - ①【価格とスケールティア】は、**F1: Freeレベル** を選択してください。
 - F1は無料ですが、サブスクリプションごとに1つしか作成することができません。赤字のエラーメッセージが表示される場合は、既存のF1のIoT Hubを削除するか **S1: Standardレベル** を選択してください。
 - ここでF1を選択して作成したIoT Hubは、あとから有料のレベルに変更することはできません。
 - 参考: Azure IoT Hub クォータと調整について | Microsoft Docs
<https://docs.microsoft.com/ja-jp/azure/iot-hub/iot-hub-devguide-quotas-throttling>
 - ②【確認および作成】をクリックします。



リソース作成のための設定情報を確認します。

- **【作成】** をクリックします。

IoT ハブ - Microsoft Azure

portal.azure.com/#create/Microsoft.IoTHub

Microsoft Azure

リソース、サービス、ドキュメントの検索 (G+/)

ホーム > 新規 > IoT Hub >

IoT ハブ

Microsoft

基本 ネットワーク サイズとスケール タグ 確認および作成

基本

サブスクリプション	Microsoft Azure
リソース グループ	intelfpga
領域	東日本
IoT Hub 名	intelfpga-hub-takasehideki

ネットワーク

接続方法	パブリック エンドポイント (すべてのネットワーク)
IP フィルター規則	なし
プライベート エンドポイント接続	なし

サイズとスケール

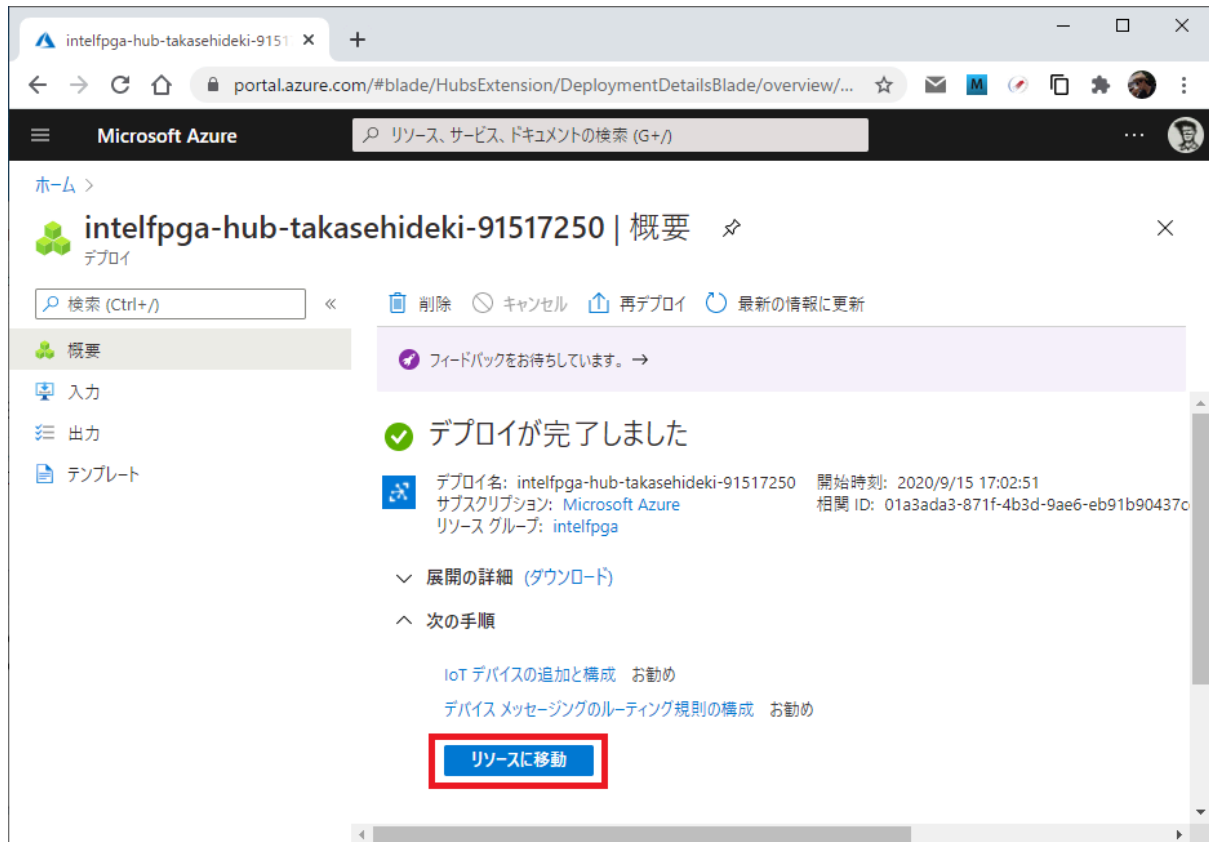
価格とスケールティア	F1
IoT ハブ F1 のユニット数	1
1 日あたりのメッセージ	8,000
Device-to-cloud パーティション	2
月あたりのコスト	0.00 USD
Azure Security Center	機能が有効になっていません

作成 < 前へ: タグ 次へ > Automation オプション

IoT Hubのデプロイが進行します。少し待ちましょう。

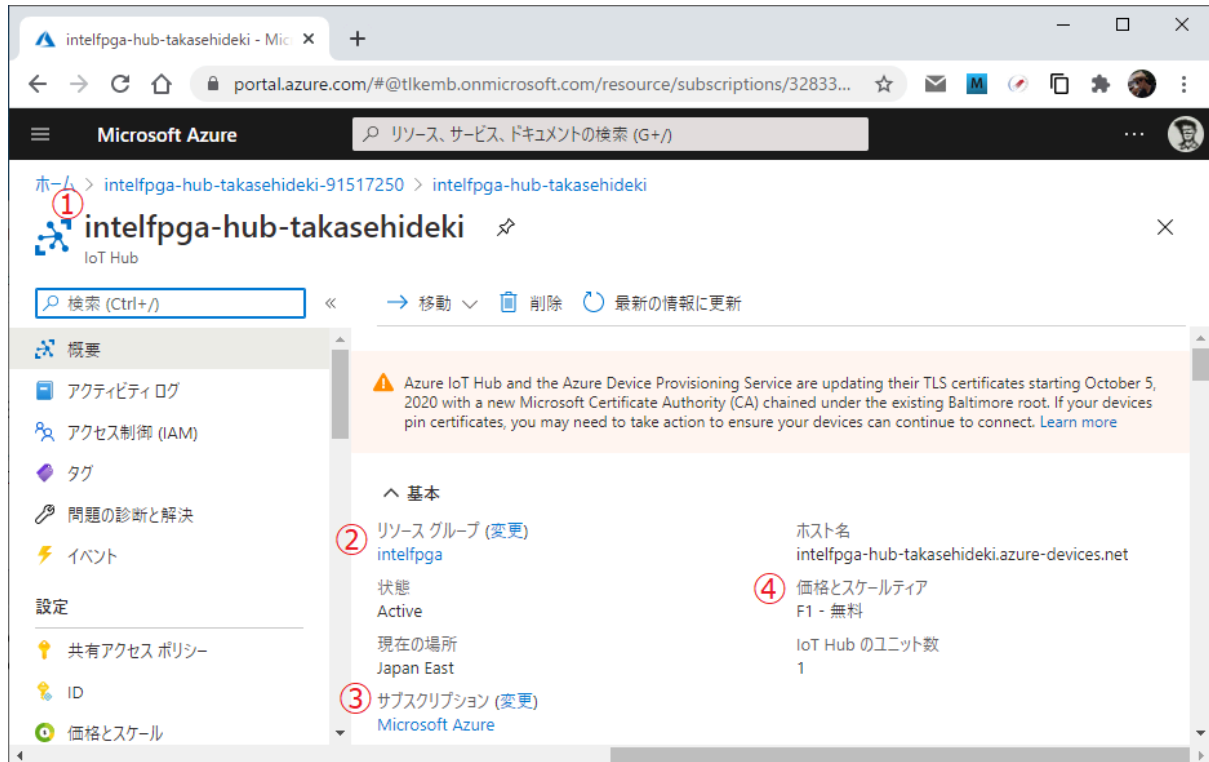
下記の画面になったら、デプロイが完了しました。

- **【リソースに移動】**をクリックします。



展開されたリソースの設定情報を確認しておきましょう。

- ① IoT Hub名: 例では **intelfpga-hub-takasehideki**
- ② リソースグループ: **intelfpga**
- ③ サブスクリプション: **Microsoft Azure**
- ④ 価格とスケールティア: **F1 - 無料**
など



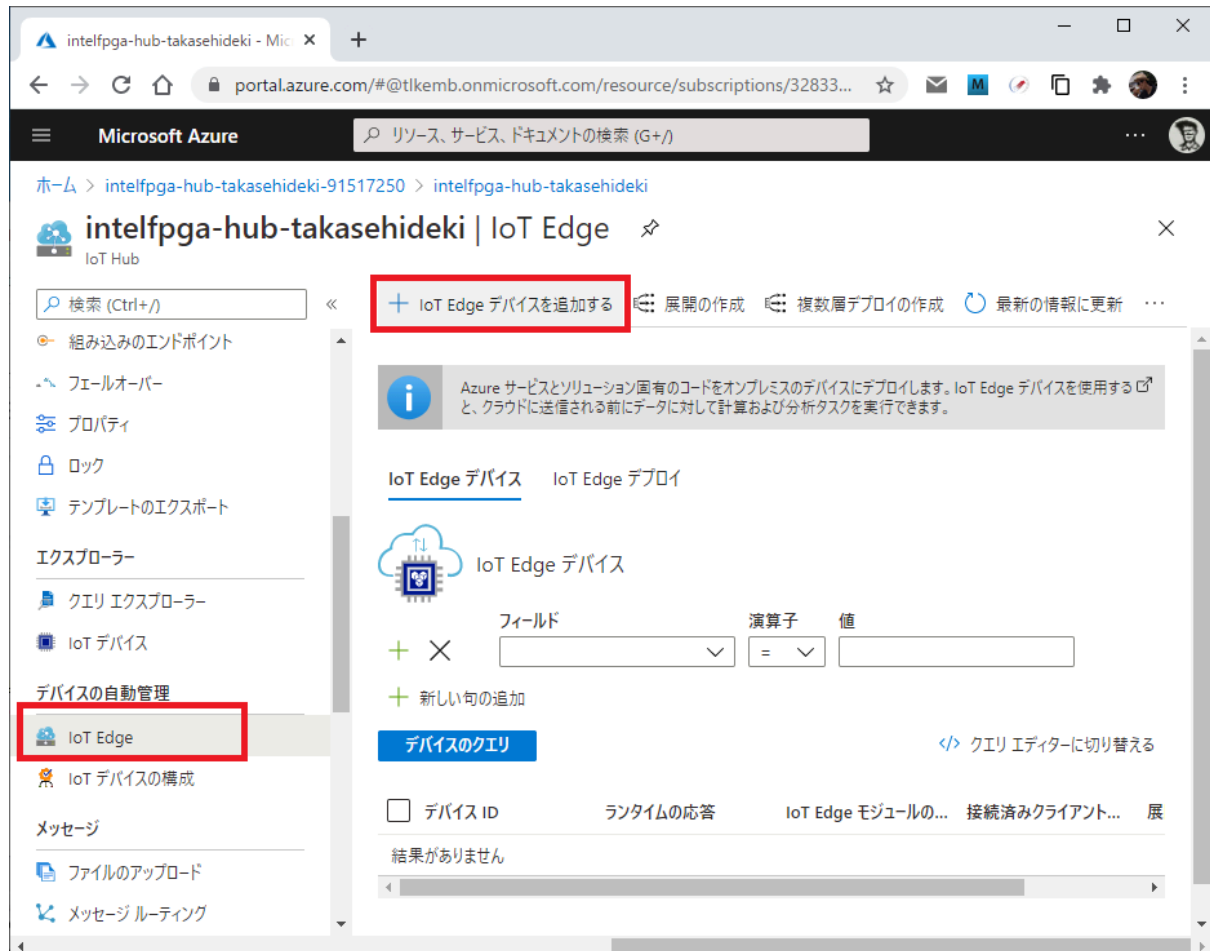
IoT Hubの作成は以上です。

4. IoT Edgeデバイスの作成

次はIoT Hub上にIoT Edgeデバイスを追加します。

先ほど作成したIoT Hubのリソースのページ(例では **intelfpga-hub-takasehideki**)に移動して、左横メニューの「デバイスの自動管理」>[IoT Edge]を開きます。

【+ IoT Edge デバイスを追加する】をクリックします。



「デバイスの作成」画面になります。

- 【デバイス ID】に **de10-nano** と入力します。 任意のものでも構いません。
- 他の設定はデフォルトのままで構いません。
- 【保存】をクリックします。

デバイス ID *

de10-nano

認証の種類

対称キー X.509 自己署名済み

主キー

プライマリ キーを入力してください

セカンダリ キー

セカンダリ キーを入力してください

自動生成キー

このデバイスを IoT ハブに接続する

有効化 無効化

子デバイス

0

子デバイスの選択

保存

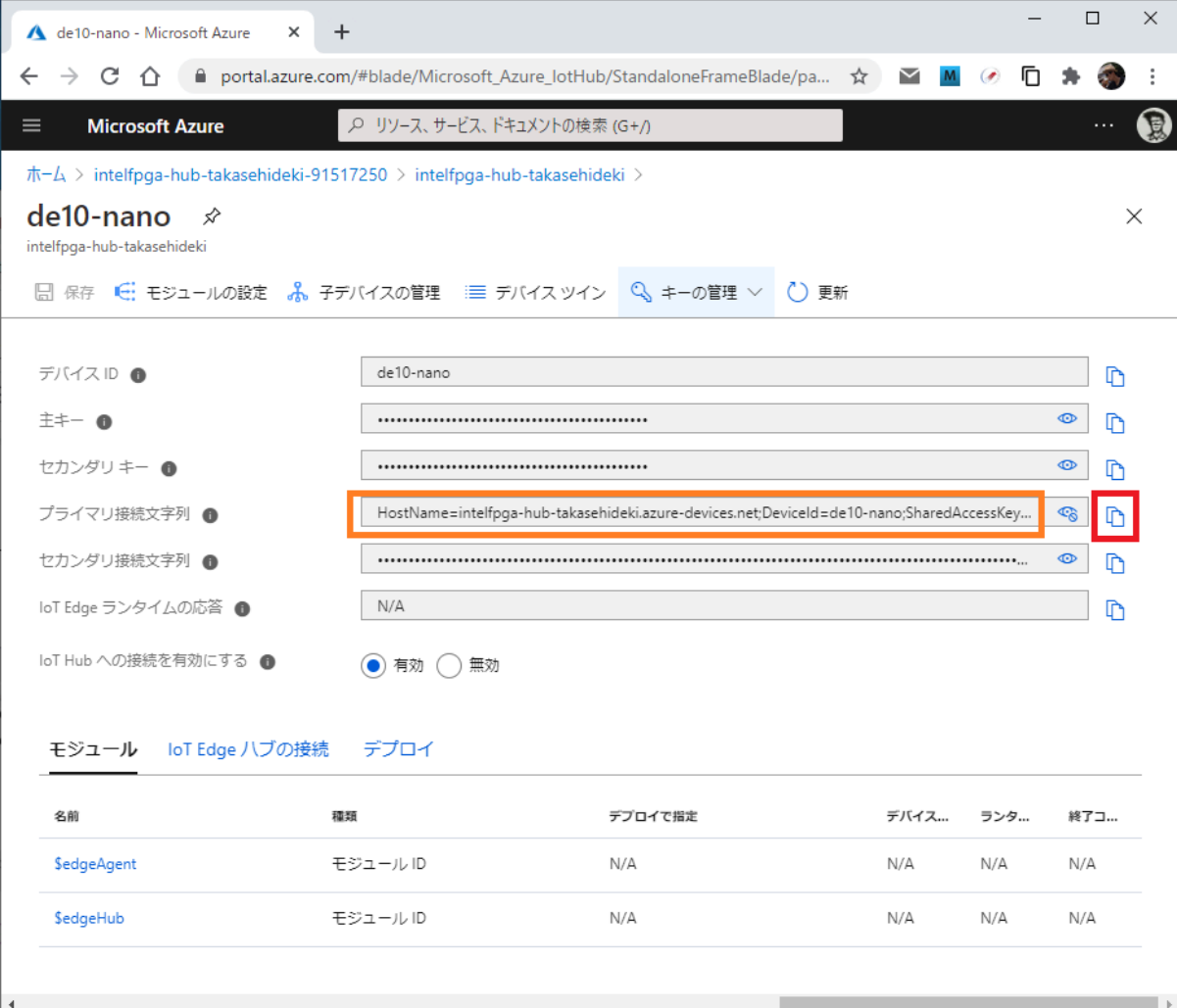
ページが下記のように戻り, 追加したデバイス IDである **de10-nano** が表示されていることを確認します. ここをクリックして開きます.
(このページは左横メニューの「デバイスの自動管理」>[IoT Edge]で開けます)

The screenshot shows the Microsoft Azure portal interface for an IoT Hub named 'intelfpga-hub-takasehideki'. The page is titled 'intelfpga-hub-takasehideki | IoT Edge'. The left sidebar contains a navigation menu with the following items: '検索 (Ctrl+/)', '組み込みのエンドポイント', 'フェールオーバー', 'プロパティ', 'ロック', 'テンプレートのエクスポート', 'エクスプローラー', 'クエリ エクスプローラー', 'IoT デバイス', 'デバイスの自動管理', 'IoT Edge', 'IoT デバイスの構成', 'メッセージ', 'ファイルのアップロード', and 'メッセージ ルーティング'. The 'IoT Edge' item is selected. The main content area shows a list of IoT Edge devices. The device 'de10-nano' is highlighted with a red box. The table has the following columns: 'デバイス ID', 'ランタイムの応答', 'IoT Edge モジュールの...', '接続済みクライアント...', and '展開'. The row for 'de10-nano' shows '該当なし' for 'ランタイムの応答', '0' for 'IoT Edge モジュールの...', '0' for '接続済みクライアント...', and '0' for '展開'.

デバイス ID	ランタイムの応答	IoT Edge モジュールの...	接続済みクライアント...	展開
de10-nano	該当なし	0	0	0

追加されたIoT Edgeデバイスの情報を確認します。

【プライマリ接続文字列】の情報が特に重要です。次に使用しますので、赤枠のアイコンをクリックしてエディタなどにコピーしておいてください。



The screenshot shows the Azure portal interface for an IoT Edge device named 'de10-nano'. The 'Primary Connection String' field is highlighted with a red box, and the copy icon next to it is also highlighted with a red box. The connection string is: `HostName=intelfpga-hub-takasehideki.azure-devices.net;DeviceId=de10-nano;SharedAccessKey=...`

Below the configuration fields, there is a table showing the modules deployed to the device:

名前	種類	デプロイで指定	デバイス...	ランタ...	終了コ...
\$edgeAgent	モジュール ID	N/A	N/A	N/A	N/A
\$edgeHub	モジュール ID	N/A	N/A	N/A	N/A

ここまでの作業に関するMicrosoftの公式ドキュメントは、下記をご参照ください。

参考: Azure Portal を使用して IoT Hub を作成する | Microsoft Docs

<https://docs.microsoft.com/ja-jp/azure/iot-hub/iot-hub-create-through-portal>

5. DE10-NanoボードのIoT Edgeデバイスとしての登録

次に、DE10-NanoボードをIoT Edgeデバイスとして登録します。

DE10-Nanoボードにシリアル／ssh接続して、ターミナルを立ち上げます。

まずはDE10-Nanoボードで稼働しているIoT Edge Runtimeの設定を行います。

なお、以降は本資料作成時点のAzure IoT Edge 1.0.9.4の情報となります。最新版では設定方法やログが異なることがあります。

IoT Edge Runtimeの設定情報は、DE10-Nanoボード上にある `/etc/iotedge/config.yaml` というファイルの編集によって行います。

ターミナル上でのファイルの編集には、Vimが使用できます。

Vimやターミナルの操作が苦手な方は、[準備篇F.4.2 ssh接続](#)で紹介したようにTera Termでssh接続してから、[ファイル(F)] > [SSH SCP...] の機能を使うとよいでしょう。ホストのPCにDE10-Nanoボードの `/etc/iotedge/config.yaml` のファイルを[Receive]してから、メモ帳などのWindowsアプリで編集し、DE10-Nanoボードの `/etc/iotedge/` に[Send]してください。あるいは、[準備篇F.4.3 リモートデスクトップ接続](#)で紹介したように、リモートデスクトップを開いてLeafpadを使うこともできます。

編集すべき箇所は55行目の `"<ADD DEVICE CONNECTION STRING HERE>"` のところです。

```
(省略)
# Manual provisioning configuration
provisioning:
  source: "manual"
  device_connection_string: "<ADD DEVICE CONNECTION STRING HERE>"

# DPS TPM provisioning configuration
# provisioning:
(省略)
```

例えば【プライマリ接続文字列】が下記でしたら、

`HostName=intelfpga-hub-takasehideki.azure-devices.net;DeviceId=de10nano;SharedAccessKey=ABCdefg1234HIJklmn5678PQrstu`

55行目を下記のように修正します。文字列の先頭と末尾に `"` が必要なことに注意してください。

```
device_connection_string: "HostName=intelfpga-hub-takasehideki.azure-devices.net;DeviceId=de10nano;SharedAccessKey=ABCdefg1234HIJklmn5678PQrstu"
```

下記には、Vimでの操作を簡単に示します。

```
# vimでファイルを開く
root@de10nano:~# vim /etc/iotedge/config.yaml
### '/' で "device_connection_string" を検索するか、'55+G' で55行目に移動する
### このファイルの権限は 600 なので、保存時には ':w!' で強制書き込みが必要
```

```
# 記入例の確認
```

```
root@de10nano:~# grep device_connection_string /etc/iotedgedge/config.yaml
device_connection_string: "HostName=intelfpga-hub-takasehideki.azure-devices.net;DeviceId=de10nano;SharedAccessKey=ABCdefg1234HIJklmn5678PQrstu"
```

次に、コンテナ向けのDNSを設定します。

`/etc/docker/daemon.json` のファイルを、新規作成して、下記を記述してください。

(Tera TermでSSH SCPされる方は、このファイルは最初には存在していませんので、ホストPCで作成したファイルをDE10-Nanoボードの `/etc/docker/` に[Send]してください)

```
{
  "dns": ["8.8.8.8", "8.8.4.4"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "10m",
    "max-file": "3"
  }
}
```

IoT Edgeのサービスを再起動して、その稼働状況を確認します。

下記の通り **active (running)** という表示が出ていればOKです。

```
# iotedgedgeサービスを再起動する
```

```
root@de10nano:~# systemctl restart iotedgedge
```

```
# iotedgedgeサービスの稼働状況を確認する
```

```
root@de10nano:~# systemctl status iotedgedge
```

```
● iotedgedge.service - Azure IoT Edge daemon
   Loaded: loaded (/lib/systemd/system/iotedgedge.service; enabled; vendor preset:
   Active: active (running) since Tue 2020-09-15 09:23:17 UTC; 3s ago
     Docs: man:iotedged(8)
   Main PID: 2312 (iotedged)
    Tasks: 2 (limit: 2325)
   CGroup: /system.slice/iotedgedge.service
           mq2312 /usr/bin/iotedged -c /etc/iotedgedge/config.yaml
```

```
Sep 15 09:23:17 de10nano systemd[1]: Started Azure IoT Edge daemon.
```

```
Sep 15 09:23:17 de10nano iotedged[2312]: 2020-09-15T09:23:17Z [INFO] - Starting
```

```
Sep 15 09:23:17 de10nano iotedged[2312]: 2020-09-15T09:23:17Z [INFO] - Version -
```

```
Sep 15 09:23:17 de10nano iotedged[2312]: 2020-09-15T09:23:17Z [INFO] - Using con
```

```
Sep 15 09:23:17 de10nano iotedged[2312]: 2020-09-15T09:23:17Z [INFO] - Configur
```

```
Sep 15 09:23:17 de10nano iotedged[2312]: 2020-09-15T09:23:17Z [INFO] - Configur
```

```
Sep 15 09:23:17 de10nano iotedged[2312]: 2020-09-15T09:23:17Z [INFO] - Transpare
```

```
Sep 15 09:23:17 de10nano iotedged[2312]: 2020-09-15T09:23:17Z [INFO] - Finished
```

```
Sep 15 09:23:17 de10nano iotedged[2312]: 2020-09-15T09:23:17Z [INFO] - Initializ
```

```
lines 1-18/18 (END)
```

```
# 'q'で確認を終了する
```

※トラブルシューティング:

ここで、`Active: failed (Result: exit-code)` となることがあります。原因としては、`/etc/iotedge/config.yaml` の編集を行っていないことが考えられます。少し戻って編集してください。その後、`systemctl restart iotedge` を実行してIoT Edgeのサービスを再起動してください。

IoT Hubとの疎通を確認してみましょう。

初回のIoT Edge Runtime起動時には、システムモジュールの取得や展開などでちょっと時間が掛かります。ちょっと待ってから、`iotedge check` というコマンドを実行します。

3個の警告と1個のエラーメッセージ(edgeHubに関するもの)が出ますが,,, ここではまだ無視して問題ありません。

(エラーはIoT Edgeデバイスのデプロイ構成がまだなにも設定されていないことに起因しています)

```
# IoT Hubとの疎通を確認する
root@de10nano:~# iotedge check
Configuration checks
-----
√ config.yaml is well-formed - OK
√ config.yaml has well-formed connection string - OK
√ container engine is installed and functional - OK
√ config.yaml has correct hostname - OK
√ config.yaml has correct URIs for daemon mgmt endpoint - OK
? latest security daemon - Warning
    Installed IoT Edge daemon has version 1.0.9.4 but 1.0.9.5 is the latest
    stable version available.
    Please see https://aka.ms/iotedge-update-runtime for update instructions.
√ host time is close to real time - OK
√ container time is close to host time - OK
√ DNS server - OK
? production readiness: certificates - Warning
    The Edge device is using self-signed automatically-generated development
    certificates.
    They will expire in 89 days (at 2020-12-14 09:23:30 UTC) causing
    module-to-module and downstream device communication to fail on an active
    deployment.
    After the certs have expired, restarting the IoT Edge daemon will trigger it
    to generate new development certs.
    Please consider using production certificates instead. See
    https://aka.ms/iotedge-prod-checklist-certs for best practices.
√ production readiness: container engine - OK
√ production readiness: logs policy - OK
? production readiness: Edge Agent's storage directory is persisted on the host
    filesystem - Warning
    The edgeAgent module is not configured to persist its /tmp/edgeAgent
    directory on the host filesystem.
    Data might be lost if the module is deleted or updated.
    Please see https://aka.ms/iotedge-storage-host for best practices.
× production readiness: Edge Hub's storage directory is persisted on the host
```

```

filesystem - Error
    Could not check current state of edgeHub container

Connectivity checks
-----
√ host can connect to and perform TLS handshake with IoT Hub AMQP port - OK
√ host can connect to and perform TLS handshake with IoT Hub HTTPS / WebSockets
port - OK
√ host can connect to and perform TLS handshake with IoT Hub MQTT port - OK
√ container on the default network can connect to IoT Hub AMQP port - OK
√ container on the default network can connect to IoT Hub HTTPS / WebSockets
port - OK
√ container on the default network can connect to IoT Hub MQTT port - OK
√ container on the IoT Edge module network can connect to IoT Hub AMQP port - OK
√ container on the IoT Edge module network can connect to IoT Hub HTTPS /
WebSockets port - OK
√ container on the IoT Edge module network can connect to IoT Hub MQTT port - OK

19 check(s) succeeded.
3 check(s) raised warnings. Re-run with --verbose for more details.
1 check(s) raised errors. Re-run with --verbose for more details.

```

※トラブルシューティング:

ここで、上記以外で**2個以上のエラー**または**4個以上の警告**のメッセージが表示される場合は、下記の原因が考えられます。

① `/etc/iotedge/config.yaml` において編集した55行目

`device_connection_string: "<ADD DEVICE CONNECTION STRING HERE>"`

に「プライマリ接続文字列」が正しく貼り付けされていないことが考えられます。少し戻って編集内容を見直してみてください。その後、`systemctl restart iotedge` を実行してIoT Edgeのサービスを再起動してください。

② コンテナ向けのDNS設定 `/etc/docker/daemon.json` が正しく作成・編集できていないことが考えられます。少し戻って編集内容を見直してみてください。その後、

`systemctl restart iotedge` を実行してIoT Edgeのサービスを再起動してください。

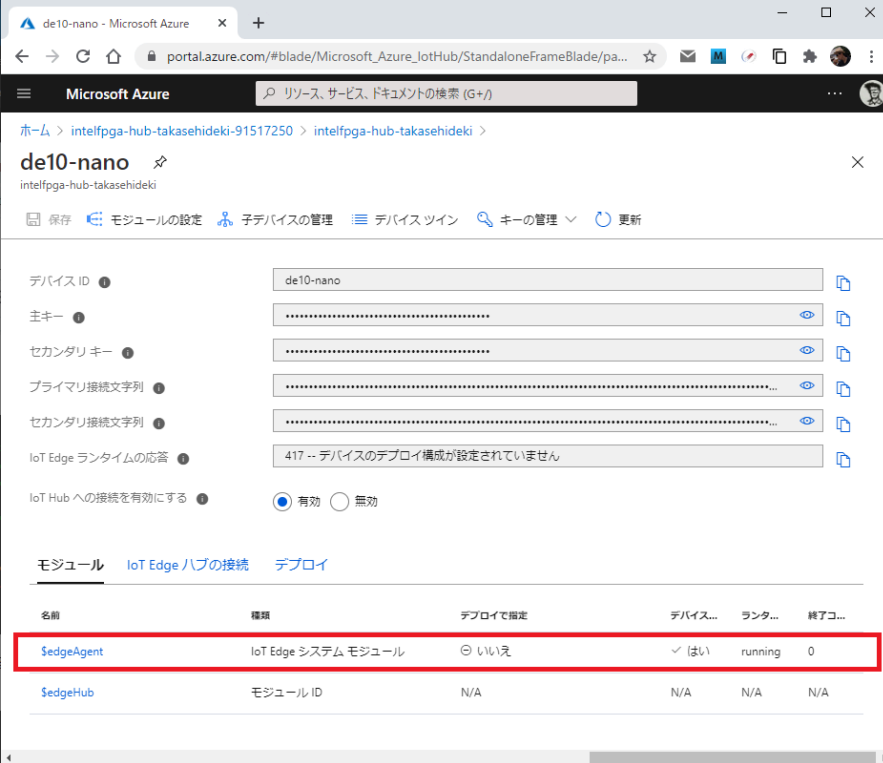
IoT Edge Runtimeのシステムモジュールの1つである edgeAgent が配置されました。

```

# IoT Edgeモジュールの動作状況を確認する
root@de10nano:~# iotedge list
NAME                STATUS      DESCRIPTION           CONFIG
edgeAgent            running    Up 24 minutes
mcr.microsoft.com/azureiotedge-agent:1.0

```

Azure PortalからもIoT Edgeデバイスの動作状況を確認してみましょう。
(このページは, IoT Hubリソースページの左横メニューから「デバイスの自動管理」>
[IoT Edge]を選択して, "de10-nano" をクリックして開きます)



The screenshot displays the Azure portal interface for an IoT Edge device. The top navigation bar shows the device name 'de10-nano' and the resource type 'Microsoft Azure'. The main content area is titled 'de10-nano' and includes a sub-header 'intelfpga-hub-takasehideki'. Below this, there are several configuration fields: 'デバイス ID' (Device ID) set to 'de10-nano', '主キー' (Primary Key), 'セカンダリキー' (Secondary Key), 'プライマリ接続文字列' (Primary Connection String), and 'セカンダリ接続文字列' (Secondary Connection String). The 'IoT Edge ランタイムの応答' (IoT Edge Runtime Response) field shows '417 -- デバイスのデプロイ構成が設定されていません' (417 -- Device deployment configuration is not set). The 'IoT Hub への接続を有効にする' (Enable connection to IoT Hub) checkbox is checked.

The 'モジュール' (Modules) section is active, showing a table of installed modules. The '\$edgeAgent' module is highlighted with a red box, indicating it is running.

名前	種類	デプロイで指定	デバイス...	ランタ...	終了コ...
\$edgeAgent	IoT Edge システム モジュール	いいえ	はい	running	0
\$edgeHub	モジュール ID	N/A	N/A	N/A	N/A

6. Option: Azure PortalからのIoT Edgeモジュールのデプロイ

IoT Edgeデバイスとして登録されたDE10-Nanoボードに、IoT Edgeモジュールをデプロイしてみましょう。

今回は、Docker Hubで講師が公開している出来合いのコンテナイメージを使用して、これをIoT Edgeモジュールとして実行します。このモジュールは、1秒ごとにG-Sensorで加速度の3次元の値を取得して、この値をIoT Hubにメッセージとして300回送信します。

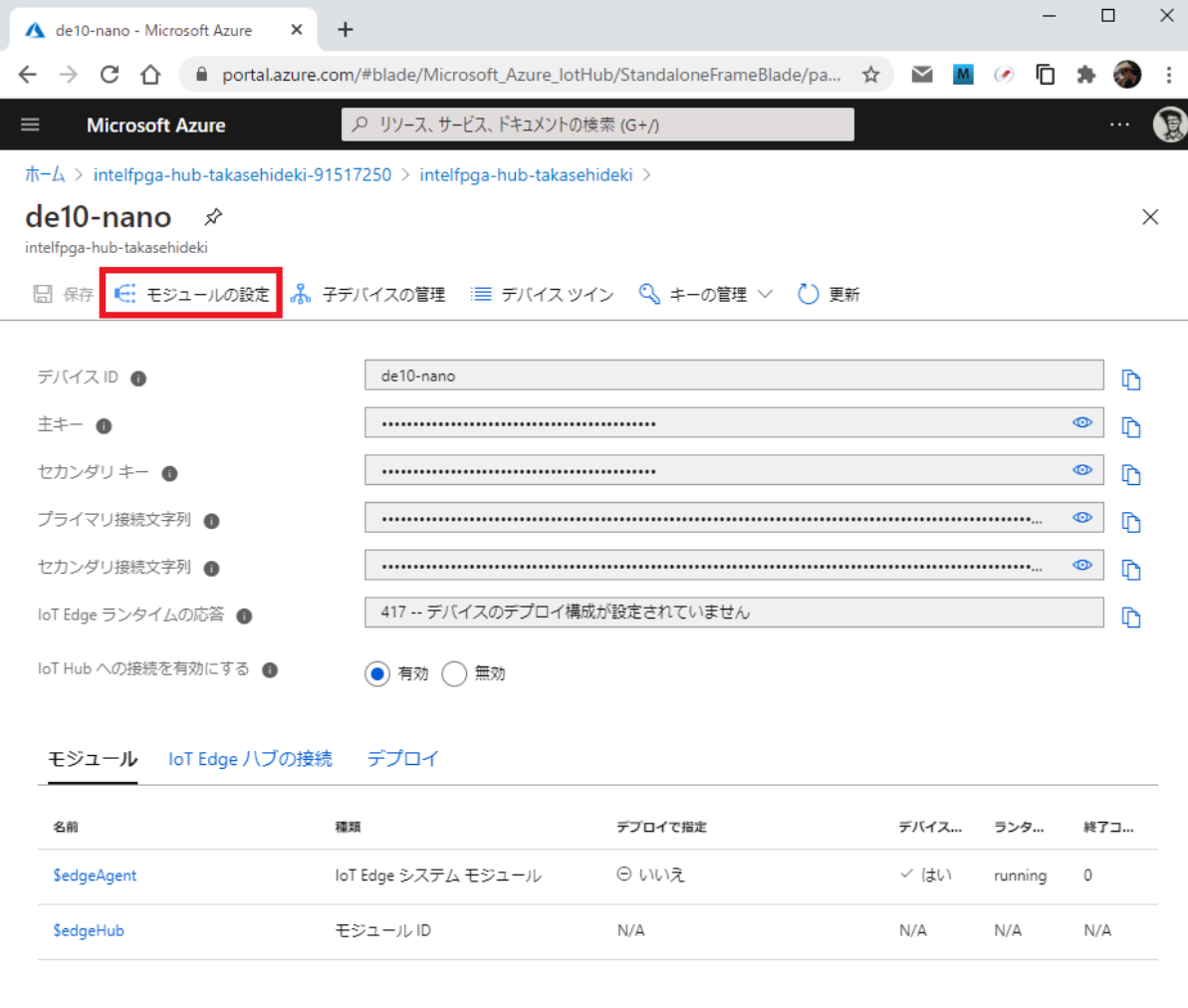
takasehideki/de10nano-g-sensor - Docker Hub

<https://hub.docker.com/r/takasehideki/de10nano-g-sensor>

IoT Edgeデバイスにモジュールをデプロイする方法は、主に、Azure Portalからの操作と、VS Codeでの操作の2種類があります。ここでは、前者の方法を紹介します。（後者の方法は、[E.7 . Option: VS CodeからのIoT Edgeモジュールのデプロイ](#)で紹介しします）

Azure Portalにて、IoT Hubリソースページの左横メニューから「デバイスの自動管理」> [IoT Edge]を選択して、"de10-nano" をクリックしてください。

【モジュールの設定】をクリックして開きます。



The screenshot shows the Azure Portal interface for an IoT Edge device named 'de10-nano'. The 'Module Settings' button is highlighted with a red box. Below the settings, there is a table showing the deployed modules.

名前	種類	デプロイで指定	デバイス...	ランタ...	終了コ...
\$edgeAgent	IoT Edge システム モジュール	⊖ いいえ	✓ はい	running	0
\$edgeHub	モジュール ID	N/A	N/A	N/A	N/A

[デバイスのモジュールを設定してください: de10-nano]の画面になります。

- **+ Add** をクリックし、続けて **+ IoT Edge Module** をクリックします。

Microsoft Azure portal interface showing the "デバイスのモジュールを設定してください: de10-nano" (Configure device modules: de10-nano) page. The page is in the "Modules" tab.

Container Registry Credentials

You can specify credentials to container registries hosting module images. Listed credentials are used to retrieve modules with a matching URL. The Edge Agent will report error code 500 if it can't find a container registry setting for a module.

NAME	ADDRESS	USER NAME	PASSWORD
<input type="text" value="Name"/>	<input type="text" value="Address"/>	<input type="text" value="User name"/>	<input type="text" value="Password"/>

IoT Edge Modules

An IoT Edge module is a Docker container you can deploy to IoT Edge devices. It communicates with other modules and sends data to the IoT Edge runtime. Using this UI you can import Azure Service IoT Edge modules or specify the settings for an IoT Edge module. Setting modules on each device will be counted towards the quota and throttled based on the IoT Hub tier and units. For example, for S1 tier, modules can be set 10 times per second if no other updates are happening in the IoT Hub.

[詳細情報](#)

+ Add (highlighted with a red box) **Runtime Settings**

- + IoT Edge Module** (highlighted with a red box)
- + Marketplace Module
- + Azure Stream Analytics Module

Buttons at the bottom: **Review + create**, **< Previous**, **Next: Routes >**

「Add IoT Edge Module」画面になります。

- ①【IoT Edge Module Name】は、 **DE10NanoGSensor** を入力してください。
- 【Module Settings】タブ
 - ②【Image URI】は、 **takasehideki/de10nano-g-sensor:0.0.1-arm32v7** を入力します。
 - **docker pull** においてイメージのURIを指定することに相当します。
 - 【Restart Policy】では、モジュールの障害時の動作を指定します。
always (常に再起動する)のままとしてください。
 - 【Desired Status】は、デプロイ時の状態を指定します。
running (デプロイ時に実行する)のままとしてください。
- 次は【Container Create Options】の設定を行います。
③のところをクリックして移動してください。

デバイスのモジュールを設定してください × +

portal.azure.com/#blade/Microsoft_Azure_IoTHub/StandaloneFrameBlade/pa...

Microsoft Azure リソース、サービス、ドキュメントの検索 (G+/)

ホーム > intelfpga-hub-takasehideki-91517250 > intelfpga-hub-takasehideki > de10-nano >

デバイスのモジュールを設定してください: de10-nano ×

intelfpga-hub-takasehideki ×

Mod ×

Cont

You ca
with a

NAME Specify the settings for an IoT Edge custom module.
詳細情報

No

① IoT Edge Module Name *

DE10NanoGSensor ③

IoT E

An IoT
sends
for an
Hub
the IoT
詳細情報

Module Settings Environment Variables Container Create Options Module Twin Settings

② Image URI *

takasehideki/de10nano-g-sensor:0.0.1-arm32v7

Restart Policy

always ▾

NAME

Desired Status

running ▾

There

Add Cancel

「Add IoT Edge Module」画面の続きです。

- [Container Create Options]タブ
 - フィールドに次の記述を入力してください。

```
{
  "HostConfig": {
    "Privileged": true
  }
}
```

- コンテナイメージの作成時のオプションをJSON形式で指定しています。
- 今回のモジュールはG-Sensorの接続されている `/dev/i2c-1` のデバイスにアクセスします。そのアクセス権限(実行特権)を与えています。
- Dockerコマンドのオプション `--privileged` に相当します。
- 下部の **Add** をクリックします。

Microsoft Azure

portal.azure.com/#blade/Microsoft_Azure_IoTHub/StandaloneFrameBlade/pa...

ホーム > intelfpga-hub-takasehideki-91517250 > intelfpga-hub-takasehideki > de10-nano >

デバイスのモジュールを設定してください: de10-nano

intelfpga-hub-takasehideki

Add IoT Edge Module

Specify the settings for an IoT Edge custom module.

NAME [詳細情報](#)

IoT Edge Module Name *

DE10NanoGSensor

[Module Settings](#) [Environment Variables](#) **[Container Create Options](#)** [Module Twin Settings](#)

Create options direct the creation of the IoT Edge module Docker container.

[View all options](#)

```
1 {
2   "HostConfig": {
3     "Privileged": true
4   }
5 }
```

Add Cancel

最初の画面に戻ります。

- IoT Edge Modules の一覧に **DE10NanoGSensor** が表示されていることを確認してください。
- 下部の **【Review + create】** をクリックします。

デバイスモジュールを設定してください

portal.azure.com/#blade/Microsoft_Azure_IoTHub/StandaloneFrameBlade/pa...

Microsoft Azure リソース、サービス、ドキュメントの検索 (G+/)

ホーム > intelfpga-hub-takasehideki-91517250 > intelfpga-hub-takasehideki > de10-nano >

デバイスのモジュールを設定してください: de10-nano

intelfpga-hub-takasehideki

Modules Routes Review + create

Container Registry Credentials

You can specify credentials to container registries hosting module images. Listed credentials are used to retrieve modules with a matching URL. The Edge Agent will report error code 500 if it can't find a container registry setting for a module.

NAME	ADDRESS	USER NAME	PASSWORD
<input type="text" value="Name"/>	<input type="text" value="Address"/>	<input type="text" value="User name"/>	<input type="text" value="Password"/>

IoT Edge Modules

An IoT Edge module is a Docker container you can deploy to IoT Edge devices. It communicates with other modules and sends data to the IoT Edge runtime. Using this UI you can import Azure Service IoT Edge modules or specify the settings for an IoT Edge module. Setting modules on each device will be counted towards the quota and throttled based on the IoT Hub tier and units. For example, for S1 tier, modules can be set 10 times per second if no other updates are happening in the IoT Hub.

[詳細情報](#)

+ Add Runtime Settings

NAME	DESIRED STATUS
DE10NanoGSensor	running

Review + create < Previous Next: Routes >

デプロイの設定の最終確認をします。

- "Validation passed." と緑色のメッセージが出ていることを確認してください。
- 下部の **【Create】** をクリックします。

Microsoft Azure

ホーム > intelfpga-hub-takasehideki-91517250 > intelfpga-hub-takasehideki > de10-nano >

デバイスのモジュールを設定してください: de10-nano

intelfpga-hub-takasehideki

Modules Routes Review + create

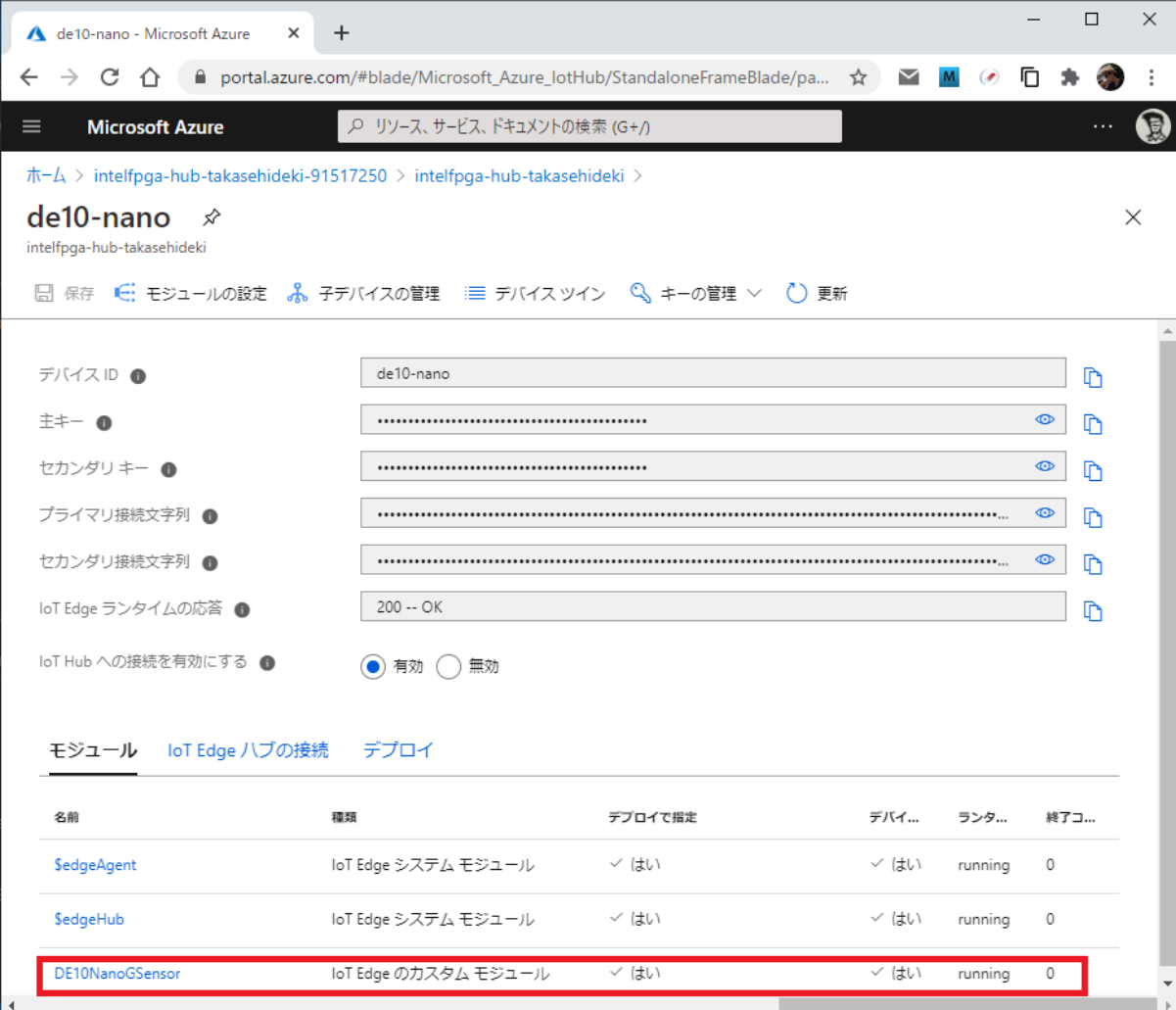
✓ Validation passed.

Deployment
The text box below displays the deployment to be submitted.

```
1 {
2   "modulesContent": {
3     "$edgeAgent": {
4       "properties.desired": {
5         "modules": {
6           "DE10NanoGSensor": {
7             "settings": {
8               "image": "takasehideki/de10nano-g-sens
9               "createOptions": "{\"HostConfig\":{\"F
10            },
11            "type": "docker",
12            "status": "running",
13            "restartPolicy": "always",
14            "version": "1.0"
15          }
16        },
17        "runtime": {
18          "settings": {
```

Create < Previous Next >

Dockerコンテナイメージが展開され、DE10-Nanoボードで稼働します。
しばらく待ってから下記の画面の右上にある【更新】をクリックしてください。
すると、下部の[モジュール]の一覧に **DE10NanoGSensor** が表示されます。そして、その[ランタイムの状態]が "running" になっていることが確認できます。



The screenshot shows the Azure portal interface for an IoT Edge device named 'de10-nano'. The 'Modules' section contains the following table:

名前	種類	デプロイで指定	デバイ...	ランタ...	終了コ...
\$edgeAgent	IoT Edge システム モジュール	✓ (はい)	✓ (はい)	running	0
\$edgeHub	IoT Edge システム モジュール	✓ (はい)	✓ (はい)	running	0
DE10NanoGSensor	IoT Edge のカスタム モジュール	✓ (はい)	✓ (はい)	running	0

DE10-Nanoボード上のターミナルで動作を確認してみましょう。
IoT Edge Runtimeのシステムモジュールである edgeHub が配置されており、さらに今回デプロイした **DE10NanoGSensor** も展開されたことが表示されます。

```
# IoT Edgeモジュールの一覧と動作状況を確認する
root@de10nano:~# iotedge list
NAME                STATUS      DESCRIPTION           CONFIG
DE10NanoGSensor     running    Up 4 minutes         takasehideki/de10nano-g-sensor:0.0.1-arm32v7
edgeAgent            running    Up 2 hours           mcr.microsoft.com/azureiotedge-agent:1.0
edgeHub              running    Up 44 minutes        mcr.microsoft.com/azureiotedge-hub:1.0
```

このモジュールの動作を再掲します。1秒ごとにG-Sensorで加速度の3次元の値を取得します。そして、その値をIoT Hubにメッセージとして300回送信します。

`iotedge logs <モジュール名>` コマンドで、対象モジュールの実行ログ (printfメッセージ)を確認できます。 `DE10NanoGSensor` のログを表示してみましょう。
危険が危なくない範囲でボードを振ってみると楽しいでしょう。

```
# IoT Edgeモジュールの動作ログを確認する
root@de10nano:~# iotedge logs DE10NanoGSensor | tail
IoTHubModuleClient_LL_SendEventAsync accepted message [15] for transmission to
IoT Hub.
Confirmation[14] received for message tracking id = 14 with result =
IOTHUB_CLIENT_CONFIRMATION_OK
[17]X=-20 mg, Y=-12 mg, Z=1004 mg
IoTHubModuleClient_LL_SendEventAsync accepted message [16] for transmission to
IoT Hub.
Confirmation[15] received for message tracking id = 15 with result =
IOTHUB_CLIENT_CONFIRMATION_OK
[18]X=-20 mg, Y=-12 mg, Z=1000 mg
IoTHubModuleClient_LL_SendEventAsync accepted message [17] for transmission to
IoT Hub.
Confirmation[16] received for message tracking id = 16 with result =
IOTHUB_CLIENT_CONFIRMATION_OK
[19]X=-16 mg, Y=-12 mg, Z=1004 mg
IoTHubModuleClient_LL_SendEventAsync accepted message [18] for transmission to
IoT Hub.
```

※トラブルシューティング:

このとき、下記のようにFailedしている場合は、「**Add IoT Edge Module**」画面での[**Container Create Options**]タブの記述にtypoがあった可能性が高いです。
Portal画面で見直してみて、デプロイを再度行ってみてください。

```
root@de10nano:~# iotedge logs DE10NanoGSensor | tail
(省略)
Failed to open the i2c bus of gsensor: No such file or directory
===== gsensor test =====
Failed to open the i2c bus of gsensor: No such file or directory
===== gsensor test =====
```

なお、IoT Edgeモジュールはコンテナとして実行されています。IoT Edgeでは、コンテナエンジンとしてOSSである [moby-engine](https://moby-engine.org/) が採用されています。

参考: サポートされているオペレーティング システム、コンテナ エンジン - Azure IoT Edge | Microsoft Docs

<https://docs.microsoft.com/ja-jp/azure/iot-edge/support>

つまり、Dockerの一般的なコマンドで、コンテナやイメージの状態を確認することもできます。

```
# イメージの一覧を表示する
root@de10nano:~# docker images
REPOSITORY                                TAG                                IMAGE ID
CREATED                                  SIZE
takasehideki/de10nano-g-sensor            0.0.1-arm32v7                    0a1edabe8550
3 days ago                               187MB
mcr.microsoft.com/azureiotedge-hub        1.0                              d6d6a399d448
5 days ago                               224MB
mcr.microsoft.com/azureiotedge-agent      1.0                              fb8e08219594
5 days ago                               210MB
mcr.microsoft.com/azureiotedge-diagnostics 1.0.9.4                          1acd9bbac7da
8 weeks ago                              7.25MB
# コンテナの一覧を表示する('docker container ls'と同様)
root@de10nano:~# docker ps
CONTAINER ID   IMAGE                                COMMAND
CREATED       STATUS      PORTS
NAMES
94a7d2a2e4bc   takasehideki/de10nano-g-sensor:0.0.1-arm32v7
"/iothub_client_sam..."   10 minutes ago   Up 10 minutes
DE10NanoGSensor
6d0351d98594   mcr.microsoft.com/azureiotedge-hub:1.0      "/bin/sh -c
'echo \"$...."   51 minutes ago   Up 51 minutes   0.0.0.0:443->443/tcp,
0.0.0.0:5671->5671/tcp, 0.0.0.0:8883->8883/tcp   edgeHub
79055a382f6c   mcr.microsoft.com/azureiotedge-agent:1.0     "/bin/sh -c
'echo \"$...."   2 hours ago     Up 2 hours
edgeAgent
# Dockerコンテナの資源使用量を確認する('--no-stream'は一度のみ表示)
root@de10nano:~# docker stats --no-stream
CONTAINER ID   NAME                CPU %      MEM USAGE / LIMIT
MEM %          NET I/O          BLOCK I/O  PIDS
94a7d2a2e4bc   DE10NanoGSensor    0.01%     1.285MiB / 1002MiB
0.13%         36.7kB / 97.1kB   0B / 0B    1
6d0351d98594   edgeHub            0.13%     82.01MiB / 1002MiB
8.19%        339kB / 297kB     0B / 0B    24
79055a382f6c   edgeAgent          0.00%     24.31MiB / 1002MiB
2.43%        60.5kB / 47.4kB   0B / 0B    17
```

ちなみに、IoT Hubとの疎通を **iotedge check** で再度確認してみましょう。

IoT Edgeデバイスのデプロイ構成が設定されてモジュールが配置されましたので、先ほどのedgeHubに関するエラーメッセージは無くなりました(警告に変わりました)

```
# IoT Hubとの疎通を確認する
root@de10nano:~# iotedge check
Configuration checks
-----
(省略)
√ production readiness: logs policy - OK
```



```
? production readiness: Edge Agent's storage directory is persisted on the host
filesystem - Warning
    The edgeAgent module is not configured to persist its /tmp/edgeAgent
    directory on the host filesystem.
    Data might be lost if the module is deleted or updated.
    Please see https://aka.ms/iotedge-storage-host for best practices.
? production readiness: Edge Hub's storage directory is persisted on the host
filesystem - Warning
    The edgeHub module is not configured to persist its /tmp/edgeHub directory
    on the host filesystem.
    Data might be lost if the module is deleted or updated.
    Please see https://aka.ms/iotedge-storage-host for best practices.

Connectivity checks
-----
(省略)
19 check(s) succeeded.
4 check(s) raised warnings. Re-run with --verbose for more details.
```

7. Tips: DE10NanoGSensorのソースコード

出来合いの品として今回利用したモジュールのソースコードは、下記で公開してあります。時間があれば解説したいと思います。

algyan/IntelFPGA_AzureOTA at vsc_g-sensor

https://github.com/algyan/IntelFPGA_AzureOTA/tree/vsc_g-sensor

ベースに下記を使用して実装しました。 [commit log](#)を追って見たらなにをやったのか分かって楽しいかもしれません。

- SystemCD:
DE10-Nano_v.1.3.8_HWrevC_SystemCD\Demonstrations\SoC\hps_gsensors\
- azure-iot-sdk-c/iothub_client/samples/iothub_client_sample_module_sender at master · Azure/azure-iot-sdk-c
https://github.com/Azure/azure-iot-sdk-c/tree/master/iothub_client/samples/iothub_client_sample_module_sender

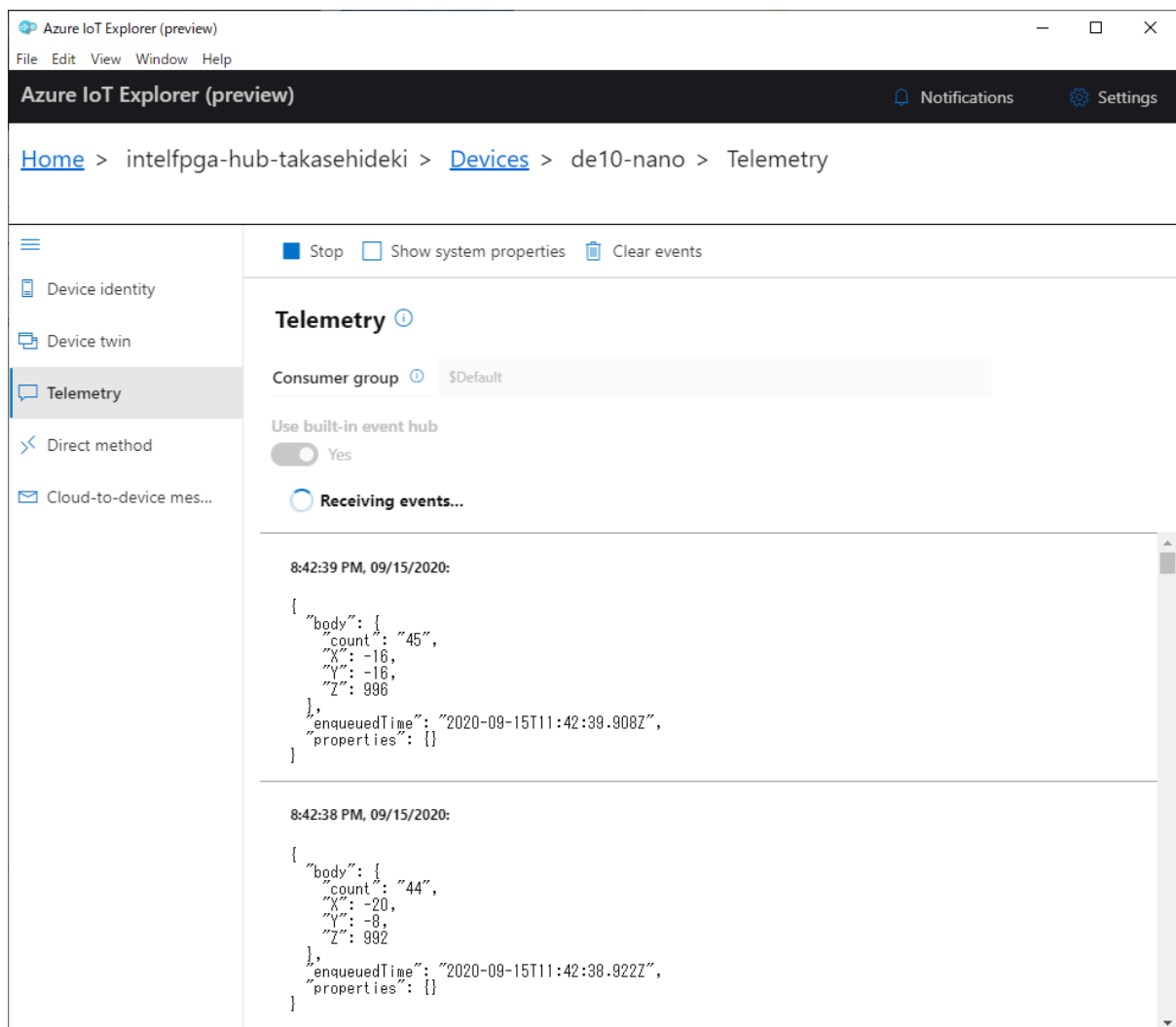
8. Tips: Azure IoT エクスプローラーでIoT Hubの受信メッセージを確認する

IoT Edgeデバイスから送信したメッセージ(IoT Hubの受信したメッセージ)を確認するには、Azure IoT Explorerを使用するのがおすすめです。

Azure IoT エクスプローラーをインストールして使用する | Microsoft Docs

<https://docs.microsoft.com/ja-jp/azure/iot-pnp/howto-use-iot-explorer>

IoT Hubの共有アクセスポリシー "iothubowner" の接続文字列を設定するだけで、すぐに使えます。まだpreview版ではありますが、IoT Plug and Playにすでに対応していますし、macOSでも動作するという嬉しさもあります。



次の章に進む: [D. Visual Studio Codeの設定](#)

目次に戻る: [A. 3. ハンズオンの内容](#)