M5Stack NanoC6 で試す Thread Matter デバイス

M5Stack が製造・販売している無線通信機能付きマイコン・ユニット **NanoC6** を用いて、スマートホーム通信規格 **Matter** に対応したデバイスを作ります。

ESP32-C6

ESP32-C6 は Espressif が製造・販売している Wi-Fi6(2.4GHz), Thread, Zigbee, Bluetooth5 に対応した無線通信機能付きマイクロ・コントローラです。Espressif が提供するソフトウェア開発環境 **ESP-IDF** を用いてソフトウェアを開発することにより、前述の無線通信機能を用いたデバイスに用いることができます。

また、Espressif は自社デバイスを使った Matter デバイスの開発環境として **esp-matter** を用意しています。esp-matter を用いれば、ESP32-C6 を使って Wi-Fi や Thread で接続する Matter デバイスをつくることができます。

M5Stack NanoC6

NanoC6 は M5Stack が製造・販売している ESP32-C6 を搭載した無線通信機能付きのユニットです。24x13x10mm^3 の小さな筐体のなかに、ESP32-C6、USB Type-C コネクタ、電源回路、GROVE 互換接続コネクタ、カラー LED、プッシュ・ボタンを含んでおり、単体で Wi-Fi、Thread、Bluetooth5、Zigbee を活用したデバイスのプロトタイプを作成できます。

無線に関しては日本の電波法に基いた工事設計認証を取得しており、日本国内で合法的に使用できます。国内ではスイッチサイエンス[*1]などから購入できます。2024 年 5 月 6 日時点での価格は 1100 円です。

NanoC6 をつかった Thread Matter Contact Sensor デバイス

NanoC6 といくつかの拡張ユニットをつかって、Contact Sensor として動作する Thread Matter デバイスを構成します。Contact Sensor の代わりにプッシュボタンを接続し、ボタンを離している状態を開いている、ボタンを押している状態を閉じているとして扱います。

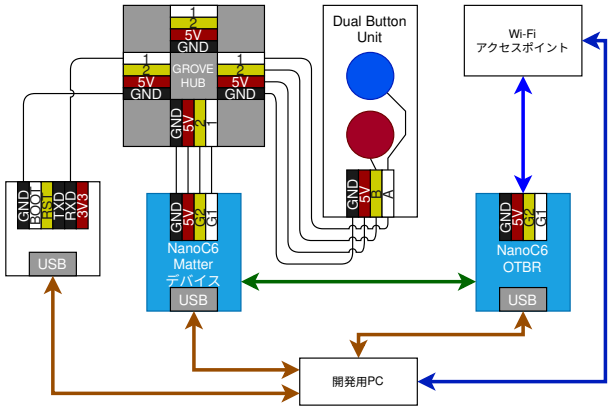

図 1　システム構成

M5Burner によるファームウェアの書き込み

M5Stack は自社のマイコン付きユニットに対してファームウェアを書き込むための **M5Burner** [*2] を提供しています。対応するプラットフォームの M5Burner をダウンロードして起動します。

機材名	個数	購入元	価格
M5Stack NanoC6	2	スイッチサイエンス	1100
USB シリアル変換 (Stamp ISP 等)	1	スイッチサイエンス	913
M5Stack 用デュアルボタンユニット	1	スイッチサイエンス	528
M5Stack 用拡張ハブユニット	1	スイッチサイエンス	572
Grove - デュポン 変換ケーブル 20 cm	1	スイッチサイエンス	726

1 つ目の NanoC6 には **NanoC6 OpenThread Border Route (Single NanoC6)** を書き込みます。2 つ目の NanoC6 には **NanoC6 Thread Matter Contact Sensor** を書き込みます。

OpenThread Border Router (OTBR)

Thread で通信するデバイスを Wi-Fi や Ethernet で通信するデバイスと中継するデバイスを**ボーダールーター (Border Router)** と呼びます。Thread の通信スタックの実装として OpenThread があり、ボーダールーターの実装 Open Thread Border Router (OTBR) が含まれています。ESP32 の開発環境 ESP-IDF には ESP32 シリーズ向けの OTBR の実装が含まれており、NanoC6 向けに設定を変更してビルドして書き込むだけで動作します。Thread ボーダールーターを持っている場合は不要ですが、持っていない場合のために、NanoC6 で OTBR を構築しておきます。

動作確認

OTBR の設定

OTBR を書き込んだ NanoC6 のシリアルポートを開いて Wi-Fi の接続処理と Thread の処理の起動を行います。

```
wifi connect -s <SSID> -p <PASS>
ifconfig up
thread start
```

その後、Thread のネットワークキーを確認しておきます。

```
dataset active -x
```

```
# 実際には1行で出てくる
0e080000000000000000000300000b35060004001fffe00208
dead00beef00cafe0708fddead00beef000005107443e9e3e3
f9bf4ea87009b17455b039030a4f70656e5468726561640102
6f64041043e769eacf5354de6d9ce281ab1b3bdd0c0402a0f7f8
Done
```

Matter デバイスのコミッショニング

Matter デバイスのコミッショニングを試すために、まずは connectedhomeip をビルドします。connectedhomeip のリポジトリにあるドキュメント[*3]の手順にしたがってビルドします。

```
sudo apt-get install git gcc g++ pkg-config libssl-dev libdbus-1-dev \
     libglib2.0-dev libavahi-client-dev ninja-build python3-venv python3-dev \
     python3-pip unzip libgirepository1.0-dev libcairo2-dev libreadline-dev
```

[*1] https://www.switch-science.com/products/9570
[*2] https://docs.m5stack.com/en/download
[*3] https://github.com/project-chip/connectedhomeip/blob/v1.3.0.0/docs/guides/BUILDING.md

```
git clone https://github.com/project-chip/connectedhomeip -b v1.3.0.0 --depth 1 --recurse-submodules connectedhomeip_v1.3.0.0
cd connectedhomeip_v1.3.0.0
./scripts/examples/gn_build_example.sh examples/chip-tool $PWD/out/linux
```

connectedhomeip_v1.3.0.0/out/linux/chip-tool に Matter のテストツールである chip-tool のバイナリが生成されます。

chip-tool をつかって作成した Matter デバイスの接続処理 (コミッショニング) を行います。以下のコマンドの <DATASET> を、OTBR で dataset active -x を実行したときの出力の 16 進文字列で置き換えて実行します。<DATASET> の前に hex: をつけるのを忘れないようにします。

```
cd out/linux
./chip-tool pairing ble-thread 1 hex:<DATASET> 20202020 3841
```

成功すれば

```
CHIP:CTL: Successfully finished commissioning step 'Cleanup'
CHIP:TOO: Device commissioning completed with success
```

のように Device commissioning completed with success というメッセージが出力されます。

Matter デバイスの動作確認

コミッショニング後も chip-tool を使ってデバイスの操作が可能です。

```
# エンドポイント番号を取得
./chip-tool descriptor read parts-list 1 0
# CHIP:TOO: Endpoint: 0 Cluster: 0x0000_001D
    Attribute 0x0000_0003 DataVersion: 1997397239
# CHIP:TOO:   PartsList: 2 entries
# CHIP:TOO:     [1]: 1
# CHIP:TOO:     [2]: 2
#
# 全エンドポイントのデバイスタイプリストを取得
./chip-tool descriptor read device-type-list 1 0xffff
# CHIP:TOO: Endpoint: 0 Cluster: 0x0000_001D
    Attribute 0x0000_0000 DataVersion: 1997397239
# CHIP:TOO:   DeviceTypeList: 1 entries
# CHIP:TOO:     [1]: {
# CHIP:TOO:       DeviceType: 22
# CHIP:TOO:       Revision: 1
# CHIP:TOO:      }
# CHIP:TOO: Endpoint: 1 Cluster: 0x0000_001D
    Attribute 0x0000_0000 DataVersion: 1890457468
# CHIP:TOO:   DeviceTypeList: 1 entries
# CHIP:TOO:     [1]: {
# CHIP:TOO:       DeviceType: 256
# CHIP:TOO:       Revision: 2
# CHIP:TOO:      }
# CHIP:TOO: Endpoint: 2 Cluster: 0x0000_001D
    Attribute 0x0000_0000 DataVersion: 1876874978
# CHIP:TOO:   DeviceTypeList: 1 entries
# CHIP:TOO:     [1]: {
# CHIP:TOO:       DeviceType: 21
# CHIP:TOO:       Revision: 1
# CHIP:TOO:      }
# コンタクトセンサーの現在状態を読み出し
./chip-tool booleanstate read state-value 1 2
# CHIP:TOO: Endpoint: 2 Cluster: 0x0000_0045
    Attribute 0x0000_0000 DataVersion: 446174921
# CHIP:TOO:   StateValue: FALSE
```

デバイスの値の変更通知を受け取ることもできます。変更通知をテストするには、chip-tool をインタラクティブモードで起動します。インタラクティブモードでは、chip-tool の引数で指定していたコマンドを直接実行できます。

```
# インタラクティブモード
./chip-tool interactive start
```

コンタクトセンサーの変更通知を受け取ってみます。

```
# Contact SensorのエンドポイントのBoolean State ClusterのState Valueを、最小1，最大30[s]間隔で変更通知を受け取る
booleanstate subscribe state-value 1 30 1 2
# CHIP:DMG: Subscription established with SubscriptionID = 0xfb2a2ebe MinInterval = 1s MaxInterval = 60s Peer = 01:0000000000000001
# 赤い方のボタンを押すとStateValueがTRUEの通知が届く
# CHIP:TOO: Endpoint: 2 Cluster: 0x0000_0045
    Attribute 0x0000_0000 DataVersion: 446174922
# CHIP:TOO:   StateValue: TRUE
```

その他

デモで使用しているファームウェアのソースコードは筆者の GitHub リポジトリに置いてあります。https://github.com/ciniml/m5stack_matter_examples

Nature の Engineering Blog でも Matter 関連の記事を公開しています。https://engineering.nature.global/archive/category/-Matter

参考文献

- Matter Device Library Specification 1.3, Connectivity Standards Alliance, https://csa-iot.org/developer-resource/specifications-download-request/
  - Matter のデバイス・タイプの仕様

筆者について

- Kenta Ida
- twitter: [@ciniml]
- GitHub: https://github.com/ciniml/
- 最近はだいたい ESP32 のファームウェア書いて生きてます。
  - Nature Remo というスマートホーム・デバイスのファームを書いてます。

M5Stack NanoC6 で試す Thread Matter デバイス

初版　2024 年 05 月 12 日

発行	Kenta IDA
著者	Kenta IDA (@ciniml)
連絡先	fuga@fugafuga.org
Twitter	@ciniml
GitHub	https://github.com/ciniml/