RN4020 を操作しよう

山本 恵理

RN4020 を操作しよう：準備編

RN4020 の操作を試す際の手軽な構成を図 1 に示します。以降は、この構成で使用例を紹介します。

図 1　RN4020 を試すオススメのシステム構成

- RN4020 をブレットボードで使う準備

RN4020 のピッチパターンは表面実装タイプです。一般的なブレッドボードの 2.54mm ピッチで使用するときは、ピッチ変換の基板を使用するか、直接ピンに銅線をハンダ付けするなどの方法があります。今回、筆者の山本が KiCAD で RN4020 の 2.54mm ピッチ変換基板を製作したので、以降はそれを使って一部機能を試していきます。この変換基板にはインジケータの LED を乗せる場所もあります。付録の URL にピッチ変換基板の CAD データを UP したのでぜひ活用してください。

- RN4020 と PC の接続準備

PC と RN4020 を、USB シリアル変換デバイスを介して接続します。BLE のモニターにはスマートフォン(スマホ)を使用します。RN4020 と USB シリアル変換デバイスの配線は表 1 のようにし、表 1 で指示のないピンは未接続です。

表 1　使用例のピン接続

ピン番号	ピン機能	接続
1,9,16,24	GND	GND
5	UART TX	FT232 RX
6	UART RX	FT232 TX
7	WAKE_SW	HIGH
8	CMD/MLDP	LOW
15	WAKE_HW	HIGH
10	PIO1/SCK	緑 LED
11	MLDP_EV/PIO2/CS	赤 LED
12	WS/PIO3/MOSI	青 LED
23	VDD	FT232 3.3V

表の通りに接続を行うと、コマンドモードとして動作します。コマンド操作は PC から行うため、シリアル通信モニターアプリを使います。Windows の場合は TeraTarm、Mac の場合は CoolTarm というアプリを使用すると便利です。または、Arduino を普段から使う方は、ArduinoIDE のシリアル通信モニターを使用するのをオススメします。RN4020 のシリアル通信の初期設定はボーレート:115200bps、データビット:8 、パリティ:なし、ストップビット:1、フロー制御：なし、デリミタ：CR となっているので、PC アプリの設定も合わせておきます。スマホのモニターアプリは、iOS の場合 Light Blue、Android の場合は、BLE Scanner というアプリがオススメです。

RN4020 を操作しよう：実践編

ハードウェアの接続を終え、電源を投入し、PC のシリアルモニタに CMD と出てきたら UART 通信が成功している証拠です。接続が完了したら、＋と入力します。Echo On と返ってきたらコマンド操作も正常に動作しています。+コマンドはエコー機能が ON になるコマンドです。エコーがあるとモジュールの反応が分かるのでコマンド操作がしやすくなります。

- コマンドモードでバッテリーレベルを周囲に知らせよう

まずは、バッテリーサービスをアドバタイズし、スマートフォンで値を確認する例を紹介します。RN4020 ではバッテリーレベルの場合、サービス名：Battery、ビットパターン：0x40000000 として用意されています。一方、Bluetooth SIG で策定されているバッテリーサービスはサービス名:Battery Service、サービス UUID：0x180F、キャラクタリスティックタイプ UUID(キャラクタ UUID)：2A19 として登録されていて、RN4020 でもこれに準拠しています。バッテリーレベルは 0%～100%のパーセンテージ表記で示す事が要求されているので値を書き込む際は気をつけて下さい。

以上のことを踏まえ、表 2 に示したコマンドを 7 番まで順に入力します。スマホ側で RN4020_TEST が見つかればアドバタイズが成功しています。

表 2　コマンドモードで使用するコマンド

番号	送信コマンド	コマンドの意味	受信コマンド（成功時）
1	＋	Echo オン	Echo ON/Echo Off
2	SF,1	工場出荷時の既定値にリセット	AOK
3	SN,RN4020_TEST	デバイス名を RN4020_TEST にする	AOK
4	SS,40000000	サーバーサービスを Battery にする	AOK
5	R,1	リブート(設定を反映させる)	Reboot→CMD
6	LS	サーバーサービスの一覧表示	(例)バッテリーサービス登録時 180F 2A19,000B,V 2A19,000C,C END
7	A	アドバタイズ開始	AOK
8	SUW,2A19,32	UUID で値を書き込み/バッテリーレベルを 50%に設定	AOK
9	SHW,000B,32	ハンドルで値を書き込み/バッテリーレベルを 50%に設定	AOK
10	K	通信遮断	Connection End
11	Y	アドバタイズ開始停止	AOK

- バッテリーレベルの値を書き換えてみよう

スマホで、RN4020 の中身をみると、バッテリーレベルが 0%になっているはずです。この値を書き換えるには、LS コマンドを打った際に返ってきた情報を活用します。LS コマンドを打つと、180F、2A19,000B,V、2A19,000C,C、END の文字が返ってきます。これは現在 RN4020 に登録されているサーバーサービスの一覧です。キャラクタ UUID である 2A19 の後ろにくっついている 000B はバッテリーレベルのリファレンスハンドル、000C はバッテリーサービス設定のリファレンスハンドルとなります。サービス、キャラクタ、ハンドルの関係をまとめると、図 2 のイメージです。

図 2　バッテリーサービスの構成図

では、実際にバッテリーレベルの値を書き換えてみます。書き換える方法は 2 通りあります。1 つ目が、バッテリーサービスのキャラクタ UUID である 2A19 を使用する方法、2 つ目がバッテリーレベルのリファレンスハンドルである 000B を使用する方法です。ここでお勧めなのは、Bluetooth SIG で策定されているサービスはキャラクタ UUID を、オリジナルサービスの場合はリファレンスハンドルを使用して値を書き換える方法です。理由としては、後に紹介するオリジナルサービスの場合、キャラクタ UUID を 128bit で指定しなければならないので、値を書き換える際のコマンドがとても長くなってしまい、コマンドの打ち間違えが起こる可能性があります。一方リファレンスハンドルは、サーバーサービスの内容を追加するとシステムによって変更されてしまうことがあるので、Bluetooth SIG で策定されているサービスのキャラクタ UUID を指定するとリファレンスハンドルの変更で左右されません。よって今回は、表 2 の 8 番、SUW コマンドを使用して値を書き換えます。

スマホでバッテリーレベルの再読み込みを行い、50%と表示されていれば書き込み成功です。これで、バッテリーサービスをアドバタイズし、スマホで値を確認するところまで出来ました。

- スクリプト機能とユーザーオリジナルサービスについて

ユーザーオリジナルサービス設定はその名の通り、Bluetooth SIG で策定されていないオリジナルのサービスを登録できるものです。RN4020 は最大 1 つのオリジナルサービスに最大 10 個のキャラクタリスティックを登録できるようになっています。

一方、スクリプト機能とは、RN4020 の NVM(不揮発性メモリ)にスクリプトを書き込んで、電源を投入すると書き込んだスクリプトに従って動作する機能です。簡単なフロー制御であればマイコン等の MCU を使用せずに、このスクリプト機能を使って RN4020 単体での動作が可能です。

まずは、オリジナルサービスの設定をします。ここでの注意点は、オリジナルサービスの UUID は 128bit で登録し、他のサービスの UUID とかぶってはいけない点です(Universally Unique IDentifier というくらいなので当たり前ですが)。UUID の生成方法は、macOS と Linux の場合、ターミナルで uuidgen コマンドを打つと簡単に生成できます。Windows の場合は UUID を生成する WEB サービスを使用すると良いでしょう。

今回は使用例として、オリジナルサービスにひとつのキャラクタリスティックを持たせます。キャラクタリスティックのプロパティは、0x02(読み出し可)とし、データは最大 20 バイトまで設定可能となっていますが、今回は 4 バイトのデータをもたせることにします。この場合、図 3 のようにサービスの UUID とキャラクタリスティックの UUID の合計 2 つが必要になるので準備します。表 3 のコマンドを 4 番まで順に打ち、リブートで設定を読み込ませ、サービスの一覧を表示してオリジナルの設定が反映されていれば完了です。

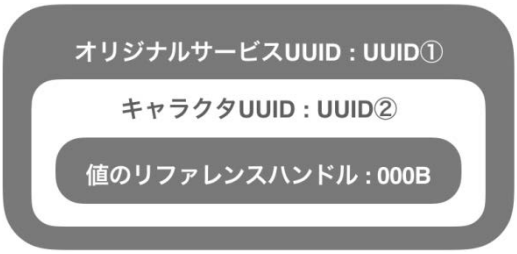

図 3　追加するサービスの構成図

表 3　スクリプトモードで使用するコマンド

番号	送信コマンド	コマンドの意味	受信コマンド
1	SS,00000001	サーバーサービスにオリジナルサービスを設定する	AOK
2	PZ	ユーザーオリジナルサービス内容をクリア	AOK
3	PS,UUID(1)	UUID(1)のユーザーオリジナルサービスを設定する	AOK
4	PC,UUID(2),02,1A	オリジナルサービスのキャラクタリスティックに UUID(2)を設定/プロパティは読み出し(0x02)/データ長 4Byte(0x04)	AOK
5	WC	スクリプトをクリア	AOK
6	WW	スクリプトの書き込み開始	AOK
7	スクリプトをクリア入力/入力終了はESC		
8	LW	入力したスクリプトを表示	AOK
9	SR,01000000	スクリプト機能を有効にする	AOK

- スクリプト機能でユーザーオリジナルのサービスを周囲に知らせよう

次に、スクリプト機能を使用して、自動的に約 15 秒間隔でアドバタイズ/アドバタイズ停止を行うように設定します。スクリプトは、最大 512 バイト、50 行未満とする必要があります。スクリプトはイベントによって駆動され、現在 11 個のイベントが定義されています。今回は表 4 に示す、電源 ON、タイマー1 タイムアウト、タイマー2 タイムアウトの 3 つのイベントを使用します。送信する情報は半固定抵抗で分圧して作った電圧値をアナログポートで読み込んだものとします。図 4 の部分を RN4020 の AIO0(4 ピン)に追加してください(アナログポートの入力は最大 1.65V)。制御フローは以下のようにします。

(1)電圧値をアナログポートで読む

(2)読んだ電圧値をアドバタイズ

(3)アドバタイズの停止

(4)1～3 の繰り返し

以上のことを踏まえ、表 3 のコマンドを 5 番以降から順に入力し、スクリプトはスクリプト 1 に示した内容を入力します。スクリプト内のコマンドについては、表 5 を参考にしてください。

表 4　スクリプトで使用するイベントラベル

イベント	イベントラベル
電源 ON	@PW_ON
タイマー1 タイムアウト	@TMR1
タイマー2 タイムアウト	@TRM2

スクリプト 1

```
@PW_ON
SM,2,00000001
@TMR1
Y
AM,2,01000000
@TMR2
$VAR1 = @I,O
SUW,UUID(2),$VAR1
A
SM,1,01000000
```

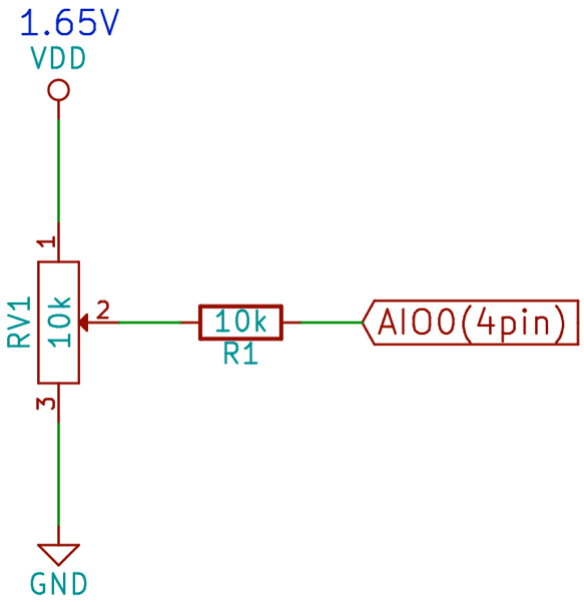

図 4　追加回路

表 5　スクリプトで使用するコマンド

スクリプトコマンド	意味
SM,(N①),(T①)	タイマーN①を使用/1us×0xT①秒でタイムアウト
$VAR = @I,0	変数 VAR1 に、AIO0 に入力されている電圧値を代入

RN4020 をリブートすると自動的にアドバタイズ/アドバタイズ停止を繰返すようになります。スマホで RN4020_TEST が約 15 秒おきに見えたり消えたりしていて、かつ、任意の値がアドバタイズされていれば成功です。これで、ユーザーオリジナルサービス設定とスクリプト機能を確認することができました。

最後に

今回紹介できたのは RN4020 の機能の一部です。MLDP 機能は主に RN4020 同士の通信で使用する機能なので、RN4020 を手元に 2 台準備しなければならず少々ややこしくなるので、今回は説明を省略しました。まず手始めに、本記事で RN4020 の使用感を確かめ、次は Microchip 公式の仕様書を読みながら紹介しきれなかった機能を試してみてください。

読者様のオリジナルのシステムで RN4020 が活用できることを願い、本記事が少しでもその手助けになれば幸いです。

参考文献

Bluetooth : https://www.bluetooth.com

RN4020 仕様書(日本語版) ：
http://ww1.microchip.com/downloads/jp/DeviceDoc/70005191A_JP.pdf

付録

RN4020 ピッチ変換基板 URL：
http://sendagi3chome.servebeer.com/wordpress/?p=41