参考

https://github.com/sekigon-gonnoc/BLE-Micro-Pro
https://salicylic-acid3.hatenablog.com/entry/BMP-Introduction
https://log.brdr.jp/post/395
https://note.com/huaa/n/n5df773cd8707
https://github.com/sekigon-gonnoc/BLE-Micro-Pro/tree/master/CoinCellHolder#%E7%B5%84%E3%81%BF%E7%AB%8B%E3%81%A6%E4%B8%8A%E3%81%AE%E6%B3%A8%E6%84%8F%E7%82%B9
https://qiita.com/koktoh/items/1c246d56d7fa8c121b7f

Runner3680

自作キーボード 2 Runner3680 で作成したキーボードを無線化する。

注文

注文したもの	URL	備考
BLE Micro Pro	https://yushakobo.jp/shop/ble-micro-pro/	分離型の場合は 2 個必要
コンスルー	https://yushakobo.jp/shop/a01mc-00/	BLE Micro Pro 数 x 2。12pin で良い。
BLE Micro Pro 用電池基板	https://yushakobo.jp/shop/ble-micro-pro-battery-board/	BLE Micro Pro と同じ数だけ必要。別な電源を検討している場合は不要。難易度高めなので不安な場合は予備でいくつか注文しておく。

その他にあったほうがいいもの

- ハンダ吸取器
- ハンダ吸取線
- ピンセット
- テスター

組み立て

https://github.com/sekigon-gonnoc/BLE-Micro-Pro/tree/master/CoinCellHolder#%E7%B5%84%E3%81%BF%E7%AB%8B%E3%81%A6%E4%B8%8A%E3%81%AE%E6%B3%A8%E6%84%8F%E7%82%B9
https://log.brdr.jp/post/395
に書いてある通り。

BLE Micro Pro とコンスルー

Pro Micro と同様にハンダ付する。

電池基盤

部品が小さめなのでなくさないように気をつける。

パーツ	取り付け方法

スイッチ	基盤上部にハンダ付け。
ダイオード x 2	黒くて小さいパーツ。向きがあるので注意。よ〜〜〜く見るとラインが入っている。ラインがある側を基盤の「●」側にハンダ付。基盤側に少しハンダを乗せてから、ピンセットを使ってハンダ付けすると比較的楽にできる。
コンデンサ	基板の裏にハンダ付けする。向きはない。ダイオード同様に基板側にハンダを乗せてから、ピンセットを使ってハンダ付けすると楽。
電池ホルダ	周りのパーツと干渉しないように取り付ける。電池を出し入れする向きに注意。

電池基盤と BLE Micro Pro は、

・電池基盤の「＋」-> BLE Micro Pro の BAT
・電池基盤の「ー」-> BLE Micro Pro の GND

につなぐ。

プログラム

前提

ファームウェア ver4.2 を利用する。
最新を使う場合は自作キーボードキットを無線化するを参考にする
最新はさらに楽になっているので、そちらを使うことをオススメします。

環境作成 (CentOS)

■ ModemManager の停止

Linux で nrfutil を使う場合、ModemManager が動作していると

```
can't open device "/dev/ttyACM0": Device or resource busy
```

が出て失敗する。また、docker 使う場合も ModemManager が動作していると

```
serial.serialutil.SerialException device reports readiness to read but returned no data
```

とエラーが出る。

```
sudo systemctl stop ModemManager
```

で ModemManager を停止する。

```
sudo systemctl disable ModemManager
```

で ModelManager を無効化しておくか、削除してしまっても良い。

■ /dev/ttyACM0: Permission denied が出る

ttyACM0 にパーミッションがなくて書き込みできないことがある。
ユーザを dialout グループに追加する必要がある。

```
sudo usermod -a -G dialout <username>
sudo chmod a+rw /dev/ttyACM0
```

■ docker 作成

nrfutil の環境が必要になるので docker で作成しておく。
Dockerfile_ble

```
FROM qmkfm/base_container

VOLUME /qmk_firmware
WORKDIR /qmk_firmware

ENV LC_ALL=C.UTF-8
ENV export LANG=C.UTF-8
RUN git clone https://github.com/yyuu/pyenv.git ~/.pyenv
ENV PYENV_ROOT=/root/.pyenv
ENV PATH=/root/.pyenv/bin:${PATH}
ENV PATH=/root/.pyenv/shims:${PATH}
RUN echo 'eval "$(pyenv init -)"' >> ~/.bashrc

RUN apt remove -y python3.5
 RUN apt update && apt install -y make build-essential libssl-dev zlib1g-dev libbz2-dev
libreadline-dev libsqlite3-dev wget curl llvm libncurses5-dev libncursesw5-dev xz-utils
RUN . ~/.bashrc && pyenv install 3.7.7
RUN . ~/.bashrc && pyenv global 3.7.7
RUN . ~/.bashrc && pip3 install nrfutil
```

docker ビルド

```
docker build  ./ -t ble -f Dockerfile_ble
```

流れ

1. ファームウェアの書き込み
2. アプリケーションの書き込み
3.CONFIG.JSN 編集
4.KEYMAP.JSN 編集

ファームウェアアップデート

https://github.com/sekigon-gonnoc/BLE-Micro-Pro/releases
から 4.2 をダウンロードし、作成した docker 環境から nrfutil でファームウェアをアップデートする。
以下の作業を左右両方に行う。

```
  docker run --rm -it --privileged -v /dev:/dev -w /qmk_firmware -v
/home/centos/data/source/qmk_firmware_bmp:/qmk_firmware -e ALT_GET_KEYBOARDS=true -e SKIP_GIT= -e
MAKEFLAGS= ble bash
 nrfutil dfu usb-serial -pkg ble_micro_pro_bootloader_0_4_2.zip -p /dev/ttyACM0
```

アプリケーションアップデート (独自キーの定義や自分でプログラムをしていな

い場合)

https://github.com/sekigon-gonnoc/qmk_firmware/releases
から 4.2 をダウンロードし、作成した docker 環境から nrfutil でアプリケーションをアップデートする。
以下の作業を左右両方に行う。

```
    docker run --rm -it --privileged -v /dev:/dev -w /qmk_firmware -v
/home/centos/data/source/qmk_firmware_bmp:/qmk_firmware -e ALT_GET_KEYBOARDS=true -e SKIP_GIT= -e
MAKEFLAGS= ble bash
 nrfutil dfu usb-serial -pkg ble_micro_pro_default_0_4_2.zip -p /dev/ttyACM0
```

アプリケーションアップデート (独自キーの定義や自分でプログラムをする場合)

https://github.com/sekigon-gonnoc/BLE-Micro-Pro/blob/v0.4.2
/AboutDefaultFirmware/doc/define_new_keyboard.md

1. 新しいキーボードを定義する
    1. ./util/new_keyboard.sh
2. qmk_firmware 同様にプログラムを組む
    1. 通常の qmk_firmware と違う点があるので注意
        1. custom_keys_user の定義
        2. custom_keycodes の開始が、SAFE_RANGE ではなく、BMP_SAFE_RANGE
        3. 参考 ()
        4. キーマップは、後述の KEYMAP.JSN で変更するので keymap.c は適当な内容で良い。
        5. 右手用、左手用の設定は後述する CONFIG.JSN で設定するので config.h は気にしなくて良い。同じ設定を左右に書き込んで問題ない。
        6. レイアウトは最大 6 つまでしか登録できないので注意。
3. コンパイルと nrfutil で転送
    1. docker run --rm -it --privileged -v /dev:/dev -w /qmk_firmware -v /home/centos/data/source/qmk_firmware_bmp:/qmk_firmware -e ALT_GET_KEYBOARDS=true -e SKIP_GIT= -e MAKEFLAGS= ble make runner3680_ble:default:nrfutil

CONFIG.JSN 編集

アプリケーションをアップデート後、USB ケーブルで接続するとストレージとして認識され

・CONFIG.JSN
・KEYMAP.JSN

などのファイルなどが見える。
CONFIG.JSN を以下のように編集する。

■左手

```
  {"config":{"version":2,
                "device_info":{"vid":"0       xfeed","pid":"0       x0000       ","name":"runner3680
_ble","manufacture":"funatake","description":"A custom keyboard"},
     "matrix":{"rows":10 ,"cols":8 ,"device_rows":5 ,"device_cols":8 ,"debounce":1 ,"is_left_hand":1
,"diode_direction":0,"row_pins":[7,8,9,10,11],"col_pins":[20,19,18,17,16,15,14,13],
  "layout":[
    1,  2,  3,  4,  5,  6,  7,  8, 48, 47, 46, 45, 44, 43, 42, 41,  0,
    9, 10, 11, 12, 13, 14, 15, 16, 56, 55, 54, 53, 52, 51, 50, 49,  0,
   17, 18, 19, 20, 21, 22, 23, 24, 64, 63, 62, 61, 60, 59, 58, 57,  0,
   25, 26, 27, 28, 29, 30, 31, 32, 72, 71, 70, 69, 68, 67, 66, 65,  0,
   33, 34, 35, 36, 37, 38, 39, 40, 80, 79, 78, 77, 76, 75, 74, 73]},
```

```
  "mode":"SPLIT_MASTER","startup":1,
  "peripheral":{"max_interval":50,"min_interval":20,"slave_latency":7},
  "central":{"max_interval":50,"min_interval":20,"slave_latency":0},
  "led":{"pin":1,"num":80},
  "keymap":{"locale":"US","use_ascii":0},
  "reserved":[0,0,0,0,0,0,0,0]}}
```

■右手

```
  {"config":{"version":2,
                "device_info":{"vid":"0       xfeed","pid":"0       x0000       ","name":"runner3680
_ble","manufacture":"funatake","description":"A custom keyboard"},
     "matrix":{"rows":10 ,"cols":8 ,"device_rows":5 ,"device_cols":8 ,"debounce":1 ,"is_left_hand":0
,"diode_direction":0,"row_pins":[7,8,9,10,11],"col_pins":[20,19,18,17,16,15,14,13],
  "layout":[
    1,  2,  3,  4,  5,  6,  7,  8, 48, 47, 46, 45, 44, 43, 42, 41,  0,
    9, 10, 11, 12, 13, 14, 15, 16, 56, 55, 54, 53, 52, 51, 50, 49,  0,
   17, 18, 19, 20, 21, 22, 23, 24, 64, 63, 62, 61, 60, 59, 58, 57,  0,
   25, 26, 27, 28, 29, 30, 31, 32, 72, 71, 70, 69, 68, 67, 66, 65,  0,
   33, 34, 35, 36, 37, 38, 39, 40, 80, 79, 78, 77, 76, 75, 74, 73]},
  "mode":"SPLIT_SLAVE","startup":1,
  "peripheral":{"max_interval":50,"min_interval":20,"slave_latency":7},
  "central":{"max_interval":50,"min_interval":20,"slave_latency":0},
  "led":{"pin":1,"num":80},
  "keymap":{"locale":"US","use_ascii":0},
  "reserved":[0,0,0,0,0,0,0,0]}}
```

KEYMAP.JSN

```
  {"keyboard":"runner3680_ble",
  "keymap":"",
  "layout":"LAYOUT",
  "layers":
    [["KC_TRNS","KC_ESC","KC_1 ","KC_2 ","KC_3 ","KC_4 ","KC_5 ","KC_6 ","KC_7 ","KC_8 ","KC_9 ","KC_0
","KC_MINS","KC_EQL","KC_JYEN","KC_BSPC",

"KC_TRNS","KC_TAB","KC_LSFT","KC_Q","KC_W","KC_E","KC_R","KC_T","KC_Y","KC_U","KC_I","KC_O","KC_P","KC_LBRC","KC_RBRC",
```

CLI

com ポートにシリアル通信すると状態確認や bluetooth の制御、KEYMAP.JSN などのファイル削除が行える
ツールは cu などを使うと良い。
https://github.com/sekigon-gonnoc/BLE-Micro-Pro/blob/master/docs/cli.md

```
cu -l /dev/ttyACM0
```

主に使うもの。

show
　接続している blutooth 機器の情報表示
del
　ペアリングしている機器をすべて削除する。ファームウェアアップデートしたりして、マスタとスレーブが接続できなくなったら両方 del すると繋がるようになる。