

LPWA 技術検証用 IoT システム仕様書

1.0.1 版

■ 1. 概要

IoT ゲートウェイは、インターネット接続用に LTE Cat1 モジュールを搭載したゲートウェイです。センサーノードとの無線インタフェースは、429MHz の LoRa/FSK に対応した無線モジュール SLR429(Circuit Design 製)、920MHz の LoRa/FSK に対応した無線モジュール RM-92A(RF Link 社製)、EnOcean、IEEE802154k に対応しており、それぞれ2つのセンサーノードからデータを同時に受け取ることが出来ます。

また、ゲートウェイ側に RTK/GNSS モジュール C94-M8P(u-blox 社製)のベース機を接続することで、無線経由で位置補正情報(RTK)をセンサーノードに送信することができ、センサーノードから位置補正済の GPS データ(GNRMC)を受け取ることが出来ます。センサーノードは温度、湿度、気圧、位置情報(EnOcean は温度、湿度のみ)を送信し、ゲートウェイはそれらのデータを受け取ったら、データをファイルに保存するとともに、Toami for DOCOMO のクラウドに保存します。

■ 2. システム概要

● ゲートウェイ (ハードウェア)

- LTE: セイコーソリューションズ製 MM-M510 (M.2 コネクタ搭載、LTE Cat.1 モジュール)を搭載可能
- Wi-Fi: Raspberry Pi3 の内蔵 Wi-Fi でデザリングが可能
Wi-Fi の ON/OFF スイッチを搭載
- LPWA:
 - RF Link 社製 RM-92A を搭載 (920MHz LoRa/FSK 無線モジュール)
 - Circuit Design 製 SLR-429D を接続 (429MHz LoRa/FSK 無線モジュール)
 - Lapis Semiconductor 製 MK74040 を搭載 (IEEE802.15.4k 無線モジュール)
 - EnOcean 温度・湿度(a5-04-01)、温度プロファイル(a5-02-01)に対応
- GPS:
 - u-blox 製 C94-M8P を搭載 (RTK/GNSS GPS モジュール)
- リモートデスクトップ:
 - リモートデスクトップ機能を搭載。PC からゲートウェイの画面を閲覧することが出来ます。

● ゲートウェイ (ソフトウェア)

- システムソフトウェア: Node-RED にて実現
- 動作概要:
 - (送信側) u-blox 製 C94-M8P の RTK 情報を RM-92A で送信可能
SLR-429D に対しては、トリガー情報を発信(RTK 情報への変更も可能)
 - (受信側)
 - ① RM-92A が送信する温度、湿度、気圧、位置情報を受信
 - ② SLR-429D が送信する温度、湿度、気圧情報、位置情報を受信
 - ③ EnOcean が送信する温度、湿度を受信
 - ④ IEEE802154K が送信する温度、湿度、気圧、位置情報情報を受信
 - ⑤ ゲートウェイの位置情報を C94-M8P から取得可能①～⑤のデータをファイルに保存するとともに、Toami for DOCOMO に送信可能

● センサーノード (RM-92A)

- CPU ボード: Lazurite SubGHz(Rev3) 3V 設定
- RF モジュール: RF LINK 社 RM-92A を接続可能
- GPS モジュール: u-blox 社製 C94-M8P(Rover 設定)を接続可能
- 温度、湿度、気圧センサーを搭載
- Grove コネクタ (3V)の2系統を搭載

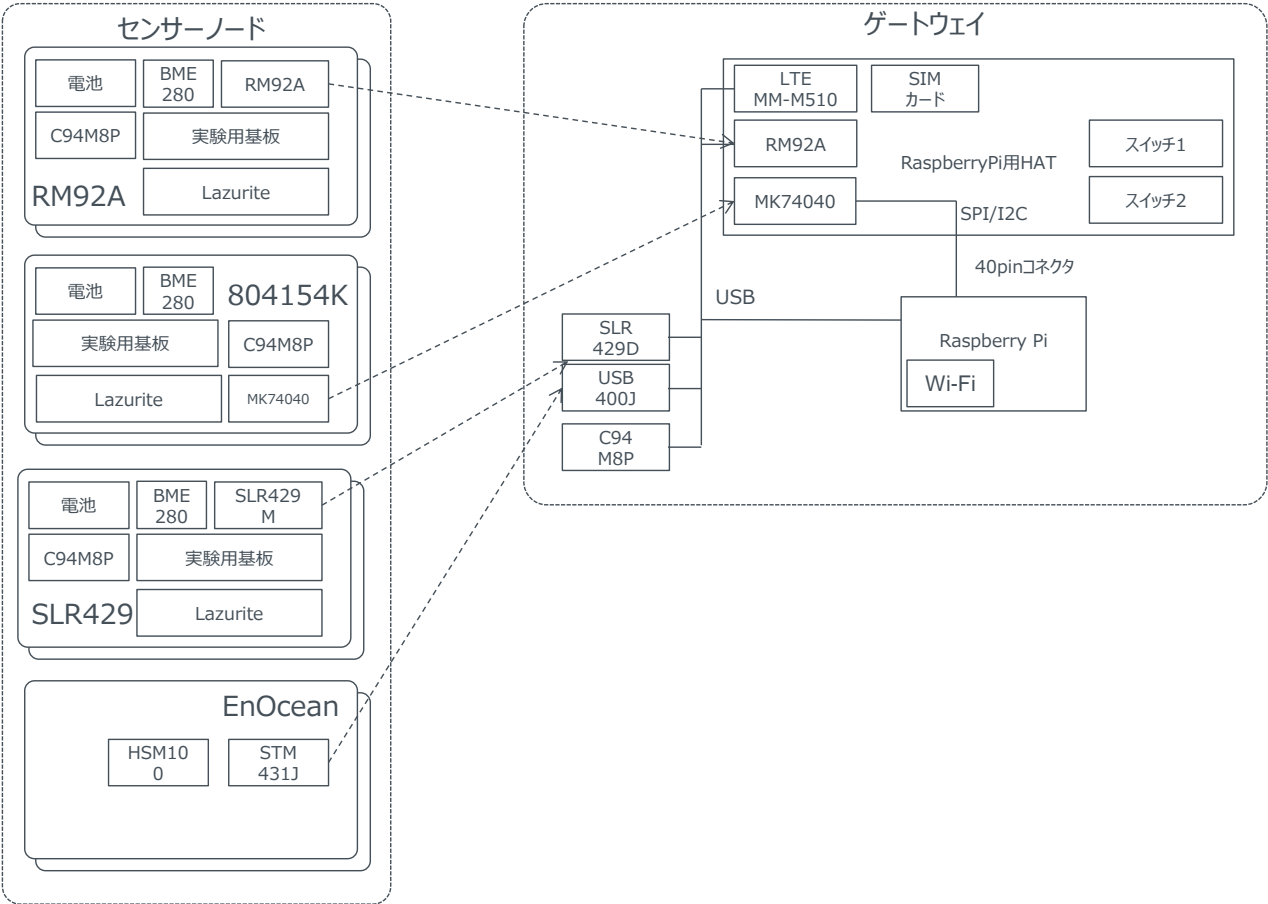
- センサーノード (SLR-429M)
 - ・ CPU ボード: Lazurite SubGHz(Rev3) 5V 設定
 - ・ RF モジュール: Circuit Design 社製 SLR-429M を接続可能
 - ・ GPS モジュール: u-blox 社製 C94-M8P(Rover 設定)を接続可能
 - ・ 温度、湿度、気圧センサーを搭載
 - ・ Grove コネクタ (5V, 3V)の2系統を搭載

- センサーノード (EnOcean)
 - ・ 温度、湿度センサー(STM 431J と HSM100)

- センサーノード (802154K)
 - ・ CPU ボード: Lazurite SubGHz(Rev3) 3V 設定
 - ・ RF モジュール: ラピスセミコンダクタ製 MK74040 を搭載
 - ・ GPS モジュール: u-blox 社製 C94-M8P(Rover 設定)を接続可能
 - ・ 温度、湿度、気圧センサーを搭載
 - ・ Grove コネクタ (3V)を2系統を搭載

3. システム構成

ハードウェア

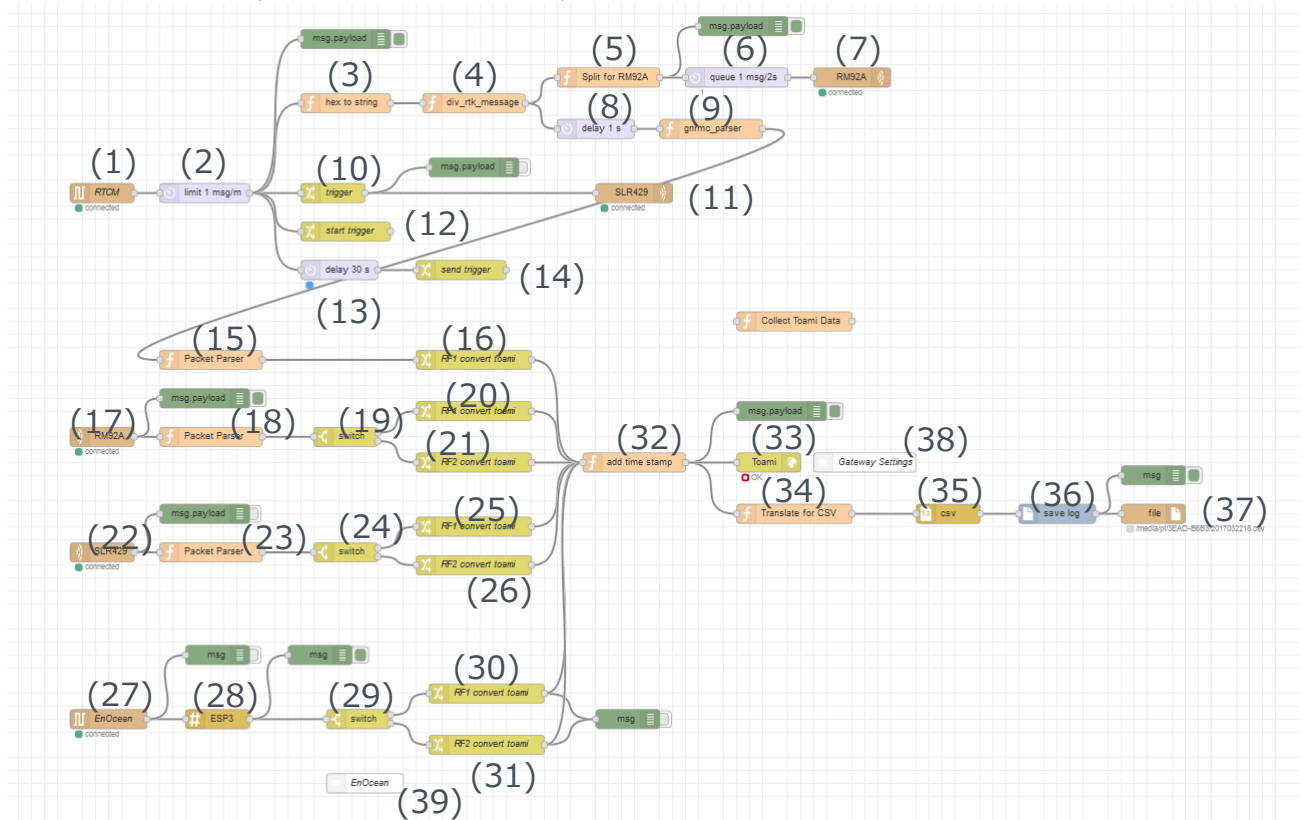


システム設定のまとめ

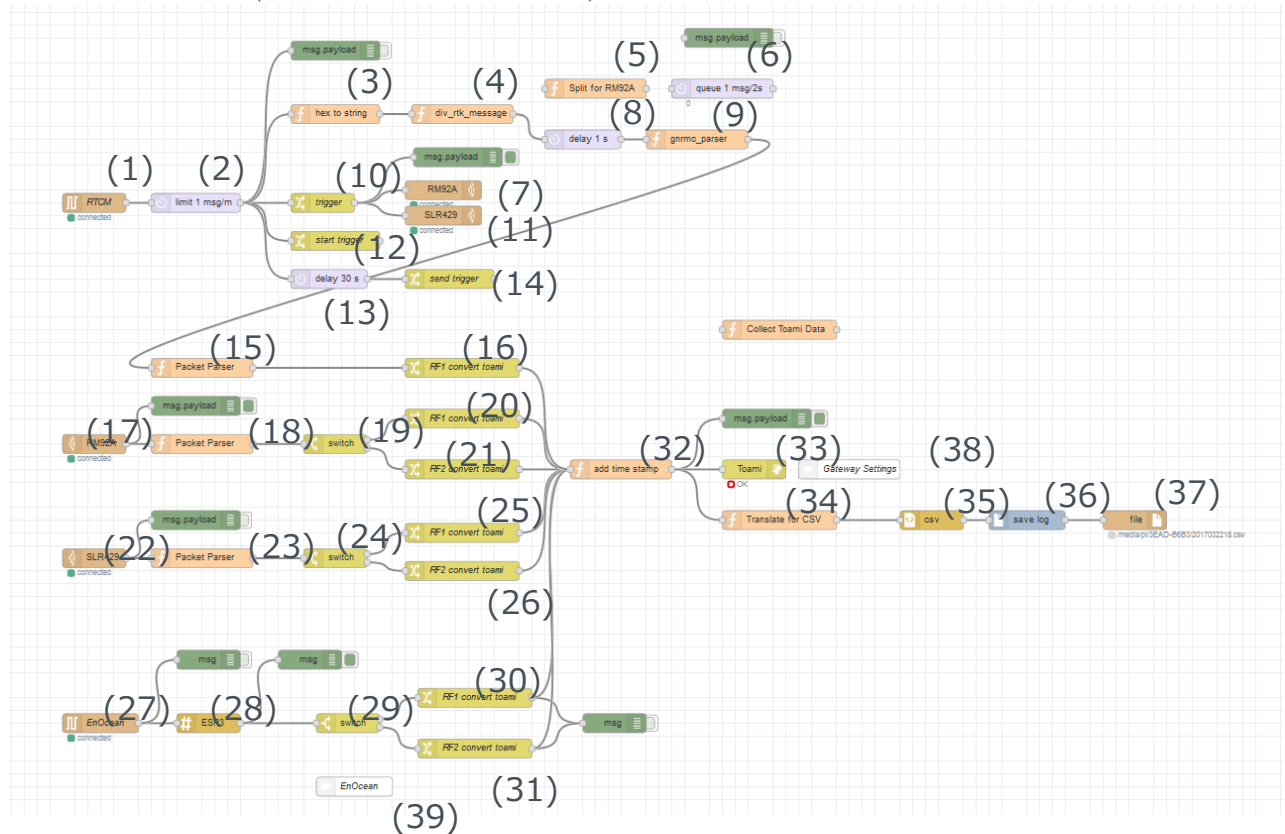
Category	Items	parameters
Wi-Fi 関連	アクセスポイント名	dcm_lpwapgw_xxx
	passphase	dcm201612
	IP アドレス	192.168.42.1
Node-RED 関連	URL	192.168.42.1:1880
リモートデスクトップ	PC アプリケーション	UltraVNC
	アクセス先	192.168.42.1:5900
	パスワード	dcm20161
	画面設定	1240 x 1024 60Hz

● ゲートウェイ ソフトウェア

- ・ Node-RED のフロー図(RTK 情報を発信する場合)



- ・ Node-RED のフロー図(RTK 情報を発信しない場合)



・ 機能説明

	ノード	表示	機能
1	serialport	RTCM	RTCM のデータを受信するノードです。
2	delay	limit 1msg/m	1 分に 1 メッセージを流します。
3	function	hex to string	RM92A/SLR429 とともにバイナリでデータを送信することが出来ないため、文字列に変換しています。
4	function	div_rtk_message	RTK とそれ以外を分離します。rtk は msg.payload に、それ以外は msg.gnrmc にして出力します。
5	function	Split for RM92A	RM92A は 1 パケットに送信できるデータ量が 100 バイトのため、データを分割しています。
6	delay	queue 1msg/2s	2 秒おきにメッセージを出力します。(LoRa で 100 バイトを送信するのに 2 秒明けないと通信が成立しなかったため)
7	RM92A output	RM92A	RM92A からデータを送信するノードです。
8	delay	delay 1s	1 秒間のディレイです。
9	function	gnrmc_parser	msg.gnrmc から GNRMC のみを取り出すとともに、packet perser の仕様であるデータ書式に合わせるための書式変換を行います。
10	switch	trigger	SLR429 にセンサー情報を送信するためのトリガーを生成します。データは「@trigger」です。
11	SLR429 output	SLR429	SLR429 でデータを送信するノードです。
12	switch	start trigger	未使用 新しいシーケンスを開始するためのトリガーです。toami へのデータ送信後に受信したメッセージをクリアしています。
13	delay	delay 30s	未使用。RM92A による RTK 送信完了タイミングを設定しています。
14	switch	send trigger	未使用。toami へのデータ送信を開始します。
15 18 23	Function	packet_perser	受信データから Toami に送信するためのデータ抽出、並びに書式変換を行います。
16 20 21 25 26 30 31	change	RFx convert Toami	各センサーの情報を toami の REST 番号に置き換えます。
17	RM92A input	RM92A	RM92A のデータを受信するノードです。
19 24 29	switch	switch	送信機のアドレスに応じてデータの出力先を振り分けます。
27	serialport	EnOcean	シリアルポートから EnOcean のデータを取り出します。
28	esp3	ESP3	EnOcean のバイナリデータから、温度・湿度・RSSI の情報を取り出します。
32	function	Add time stamp	Toami のデータに timestamp を追加します。
33	Toami	Toami	Toami for DOCOMO にデータを push します。
34	function	Translate for CSV	時間の書式を Toami から CSV 出力したときと同じにします。
35	CSV	CSV	Toami に送信した JSON データを CSV に戻します。
36	log file	log file	CSV ファイルを保存するためのファイル名を生成します。
38	Comment	Gateway setting	メモ用ノードです。gateway の ID と App key が記録されています。
39	Comment	EnOcean	メモ用ノードです。EnOcean の識別子が記録されています。

・ゲートウェイに搭載しているソフトウェアの一覧

Package	ソース	ノードのパッケージ名	主な機能／補足
Delay	Node-RED 標準		ディレイ
Switch	Node-RED 標準		条件分岐
Function	Node-RED 標準		内部のソフトウェアはすべて非公開 個別機能
Csv	Node-RED 標準		CSV ファイルの解析
File	Node-RED 標準		ファイル保存
Injection	Node-RED 標準		タイミング生成
Serialport	npm	serialport@1.7.4	シリアル入出力
es6-promise	npm	es6-promise	JavaScript の非同期処理用
javascript-state-machine	npm	javascript-state-machine	JavaScript の状態遷移用
ESP3	新規ノード／公開予定	Node-RED-contrib-enocean(仮)	EnOcean モジュールから受信したパケットの解析
RM92A	新規ノード／公開予定	Node-RED-contrib-rm92a(仮)	<ul style="list-style-type: none"> ・RM92A の初期設定 ・受信したパケットの解析を行い、src, rssi, payload に分離する。 ・payload を設定された条件で送信する。
SLR429	新規ノード／公開予定	Node-RED-contrib-slr429(仮)	<ul style="list-style-type: none"> ・SLR429 の初期設定 ・受信したパケットの解析を行い、src, rssi, payload に分離する。 ・payload を設定された条件で送信する。
Toami	新規ノード	非公開	Host、ゲートウェイ、key を指定して toami for docomo にデータを push します。
log file	新規ノード／公開予定	node-red-contrib-logfile	ゲートウェイの内部時間で 1 時間毎にファイル名を生成して msg.filename を追加します。後段に file(output)のノードを置くことで本ノードによって生成したファイル名にログファイルを保存することが出来ます。

● Toami for DOCOMO のセンサー設定(REST 番号)とセンサーの関係

無線モジュール	自機アドレス	データ	REST 番号	名称
ゲートウェイ	-	位置情報	l01	RM92A_2_LOC
RM92A	2	温度	n01	RM92A_2_TEMP
		湿度	n02	RM92A_2_HUM
		気圧	n03	RM92A_2_PRES
		受信感度	n23	RM92A_2_RSSI
		位置情報	l01	RM92A_2_LOC
	3	温度	n04	RM92A_3_TEMP
		湿度	n05	RM92A_3_HUM
		気圧	n06	RM92A_3_PRES
		受信感度	n24	RM92A_3_RSSI
		位置情報	l01	RM92A_2_LOC
SLR429	2	温度	n07	SLR_2_TEMP
		湿度	n08	SLR_2_HUM
		気圧	n09	SLR_2_PRES
		受信感度	n25	SLR_2_RSSI
		位置情報	l01	RM92A_2_LOC
	3	温度	n10	SLR_3_TEMP
		湿度	n11	SLR_3_HUM
		気圧	n12	SLR_3_PRES
		受信感度	n26	SLR_3_RSSI
		位置情報	l01	RM92A_2_LOC
EnOcean	未定	温度	n13	EOC_2_TEMP
		湿度	n14	EOC_2_HUM
		気圧	-	-
		受信感度	n27	EO_2_RSSI
		位置情報	-	-
	未定	温度	n15	EOC_3_TEMP
		湿度	n16	EOC_3_HUM
		気圧	-	-
		受信感度	n28	EO_3_RSSI
		位置情報	-	-
IEEE802154k	未定	温度	n17	4K_2_TEMP
		湿度	n18	4K_2_HUM
		気圧	n19	4K_2_PRES
		受信感度	n29	4K_2_RSSI
		位置情報	l01	RM92A_2_LOC
	未定	温度	n20	4K_2_TEMP
		湿度	n21	4K_2_HUM
		気圧	n22	4K_2_PRES
		受信感度	n30	4K_3_RSSI
		位置情報	l01	RM92A_2_LOC

● データ送信フォーマット

・ Toami for DOCOMO に送信する JSON のサンプル

```
{ "n01": 25.6, "n02": 37.36, "n03": 999.77, "l01": { "latitude": 35.51221, "longitude": 139.61753361111113 },  
"gwtimestamp": 1488775382350 }
```

・ ゲートウェイから RTK を送信する場合のデータのフォーマット

```
@F:0:10:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

“:”はデータを分割するための splitter です。

先頭の“@F”は、RTK の情報を送信することを示しているフラグです。

次の数字は、パケット分割して送信するときの番号を示しています。合計 10 パケットを送信する場合、0 から 9 まで順番に数字が増えていきます。子機側ではすべて正しく受信したら位置情報、温度、湿度、気圧データをゲートウェイに送信します。

3 番目の数字は、パケットの合計数を示しています。上のサンプルのデータでは合計 10 パケット送信することを指しています。

以降のデータ(XXXX....)は、すべて RTK のデータになります。データは、RTK(バイナリ)をテキスト形式で HEX 変換した文字列で送信します。

子機が一度に受け取ることが出来る RTK のデータは 1K バイトです。それ以上になったらデータを破棄し、すべての受信が完了した時点で、位置情報と温度、湿度、気圧データを送信します。

・ ゲートウェイから RTK を送信しない場合のペイロード

“@START”のみを送信します。

・ SLR429/RM92A からゲートウェイに送信するデータのフォーマット

CSV 形式で、「自機アドレス, GNRMC のデータ, BME280, 温度, 湿度, 気圧」の順番に送信します。

(例)

```
3,$GNRMC,044357.00,A,3530.43460,N,13937.03116,E,0.047,,060317,,,A*65,BME280,25.5,38.26,1001.22
```

GNRMC を送信しないときは、次のようになります。

(例)

```
3,BME280,25.5,38.26,1001.22
```

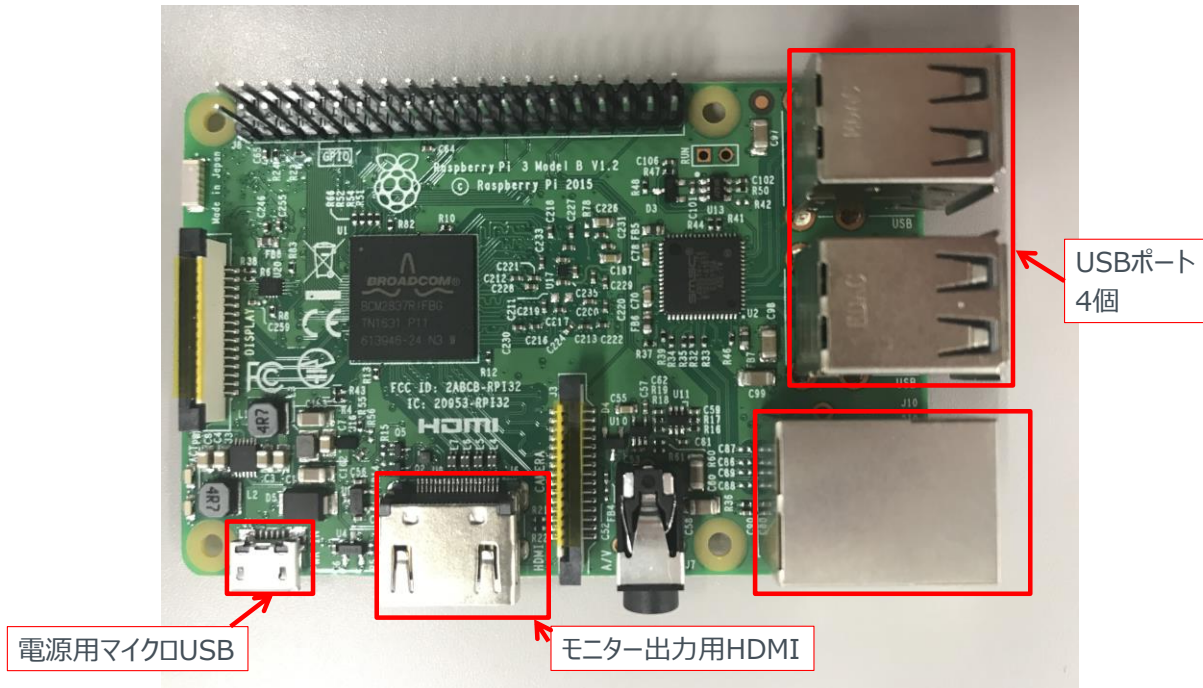
・ 802154k からゲートウェイに送信する場合のペイロード

未定

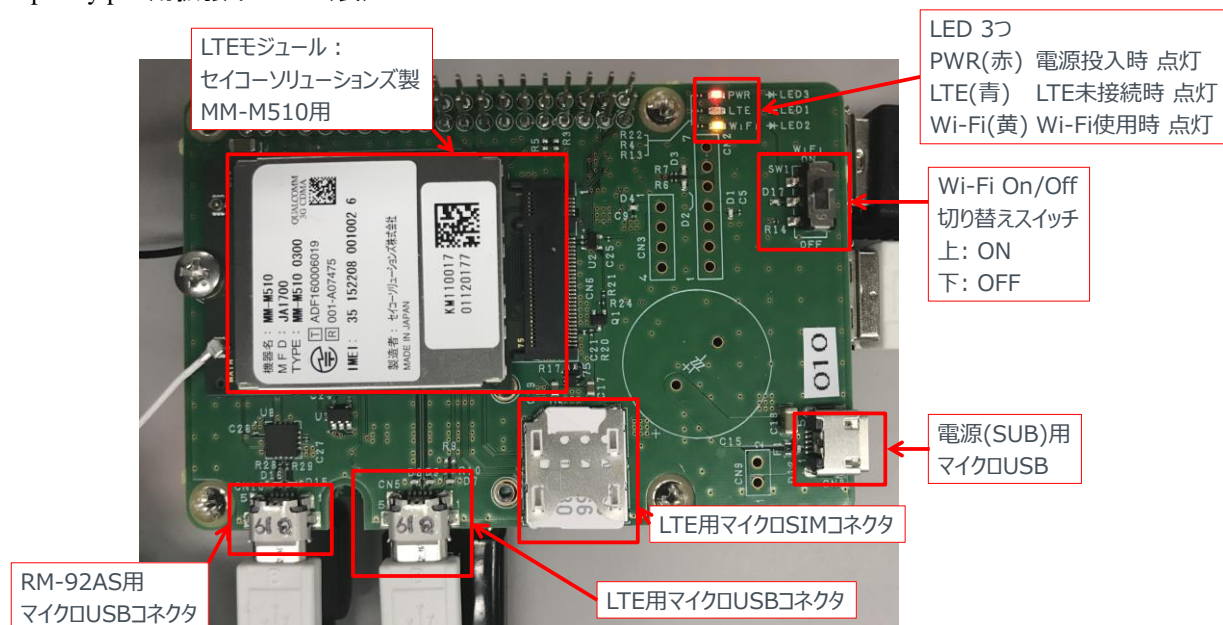
■ 4. ゲートウェイの使用方法

● ボードの構成と各部の設定

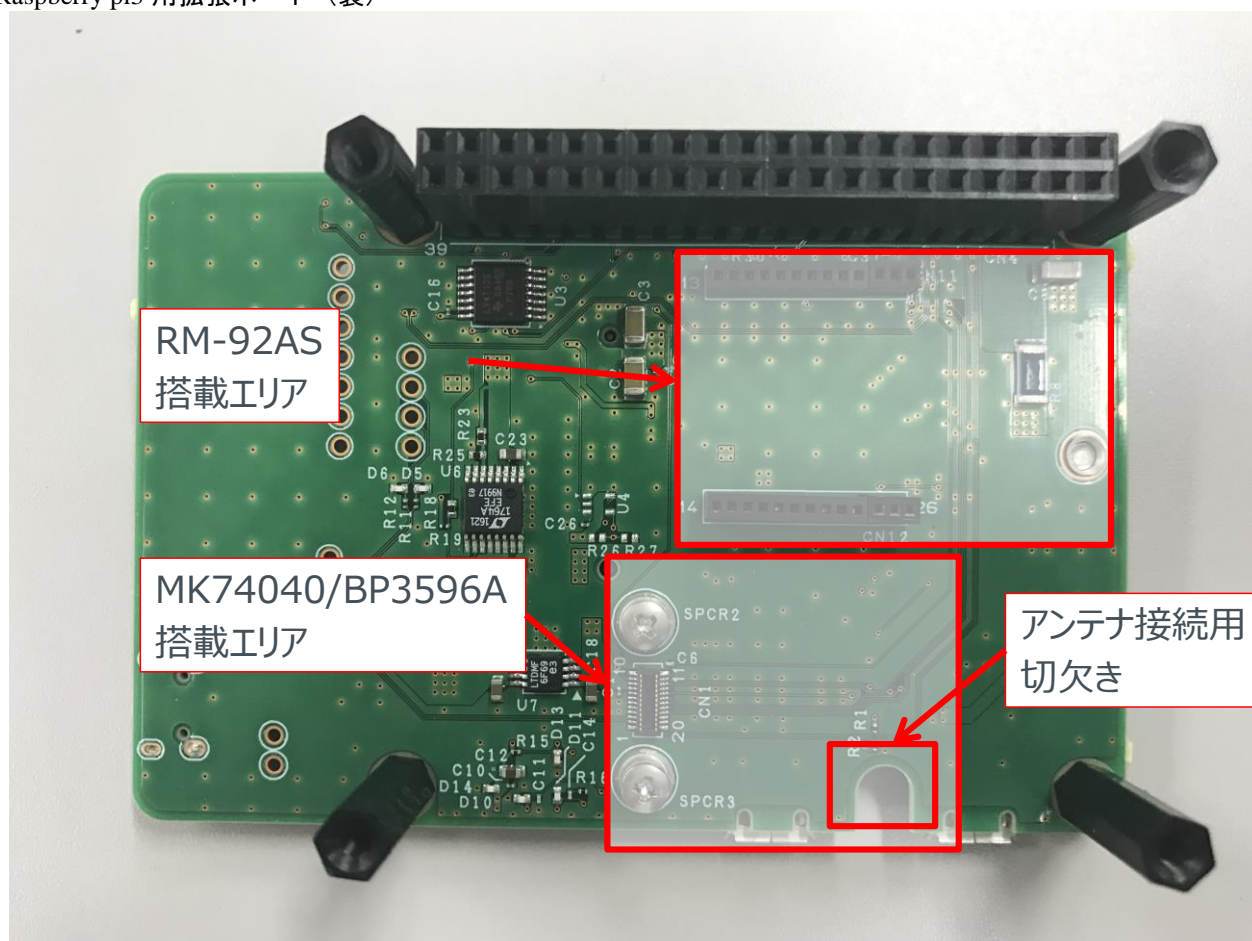
- ・ Raspberry pi3 type B



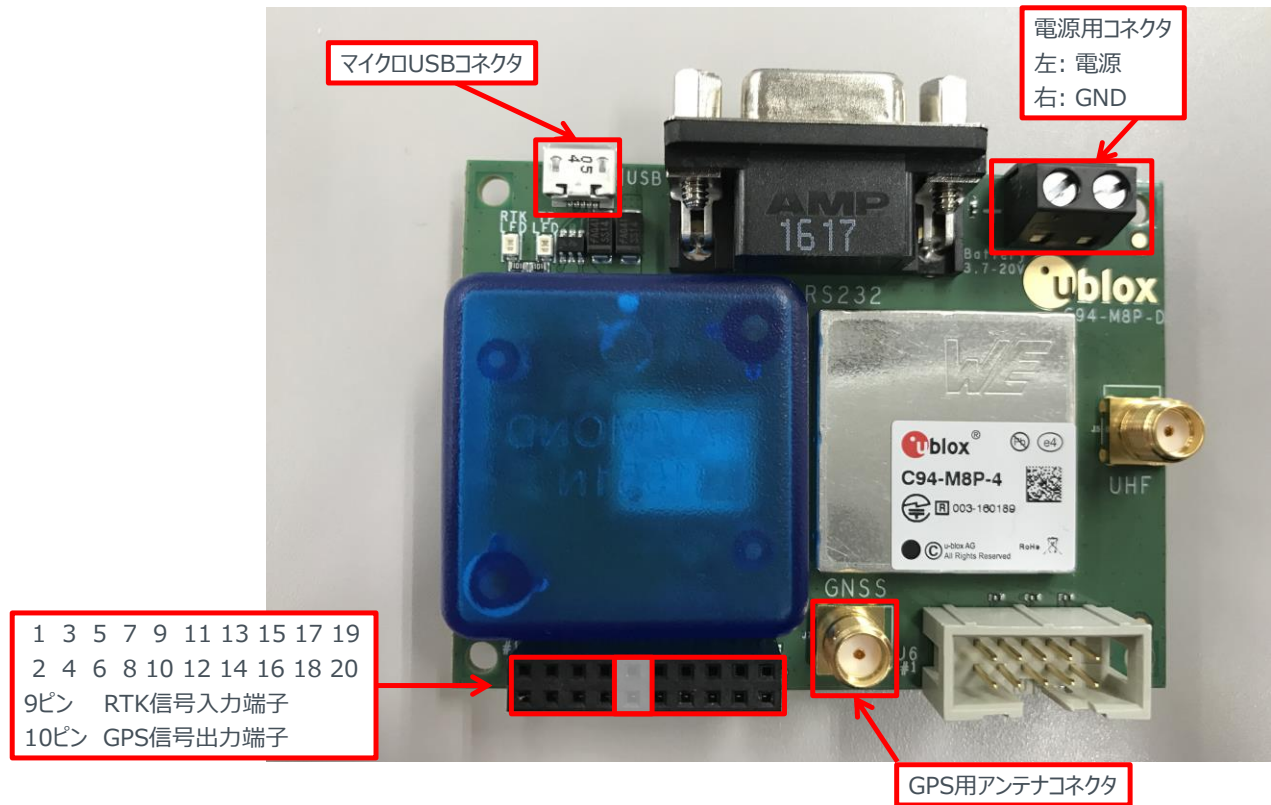
・ Raspberry pi3 用拡張ボード (表)



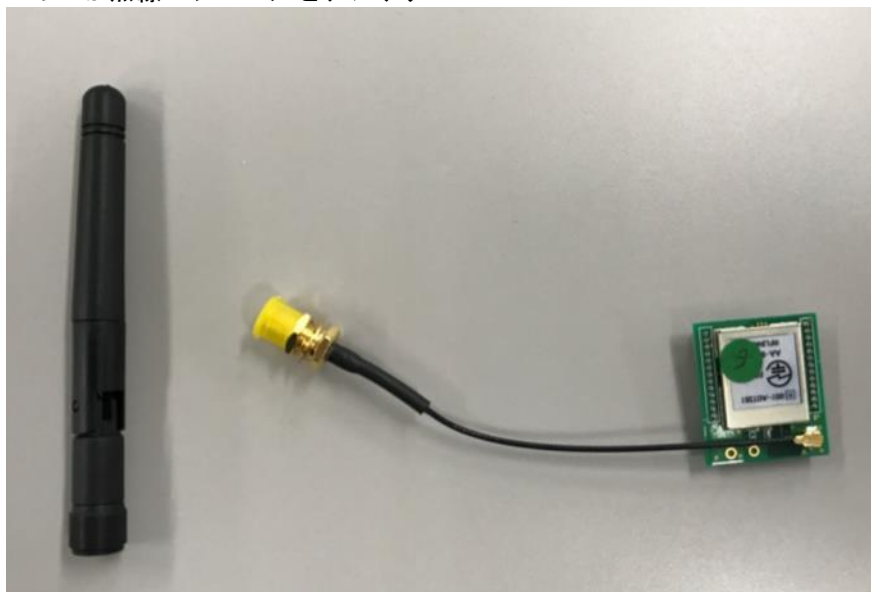
・ Raspberry pi3 用拡張ボード (裏)



・ GPS モジュール C94-M8P



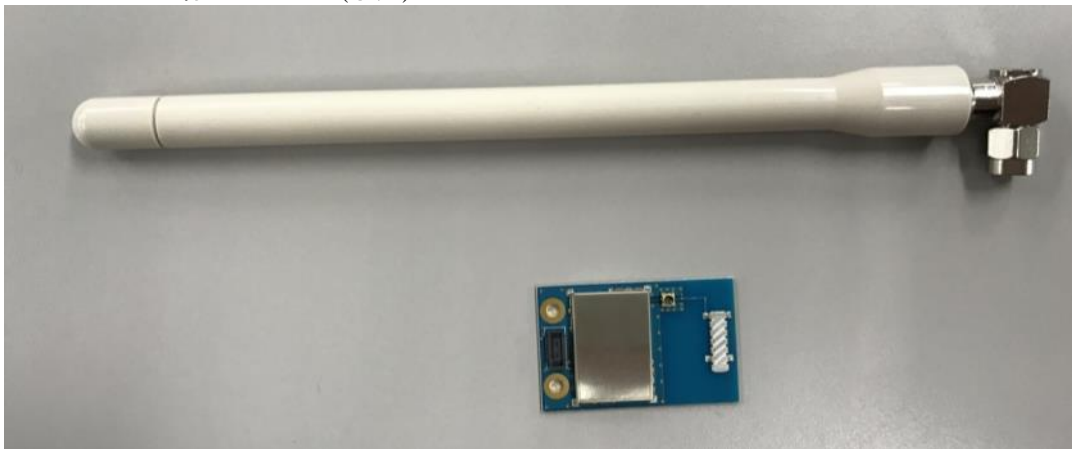
・ RM-92AS 無線モジュールとアンテナ



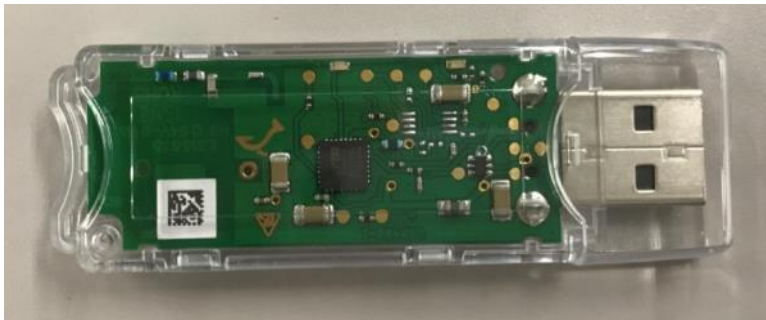
- ・ SLR-429D 無線モジュールとアンテナ



- ・ IEEE802154K 無線モジュール(予定)



- ・ EnOcean 無線モジュール



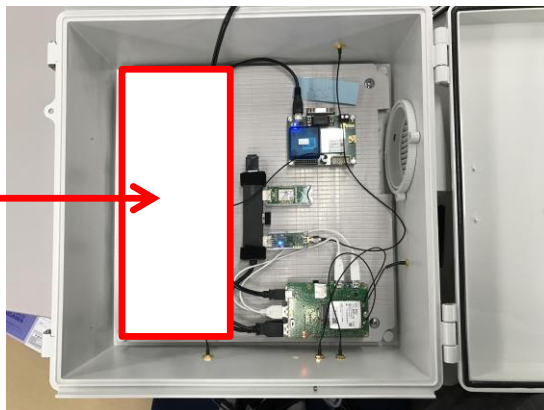
● 接続方法

ゲートウェイの接続図を以下に示します。各 SMA コネクタに、対応したアンテナを接続してください。

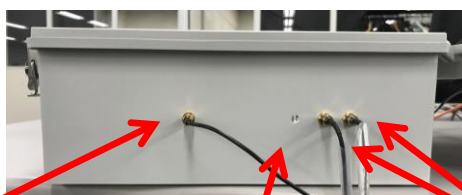


429MHz用アンテナ

電池
取り付け
エリア



LoRa(920MHz)用
アンテナ



GPS用
アンテナ

IEEE802154K用
アンテナ

LoRa(920MHz)用
アンテナ(2 個)

● 起動方法

電源用マイクロ USB または電源用(SUB)マイクロ USB にマイクロ USB 経由で電源を投入してください。

リチウムイオンバッテリーから起動する場合は、電池を満充電の状態にしてから使用してください。ゲートウェイの消費電流は起動時が最も多い為、起動が成功する前にシャットダウンしてしまう可能性があります。

また、Wi-Fi は消費電力が大きい為、使用しないときは Wi-Fi をスイッチで OFF しておくことを推奨します。

● 停止方法

・ Node-RED の起動と停止方法

Node-RED を起動する方法は 2 種類あります。

モニタリングを行う場合は、node-red-start で起動することをお勧めします。

コマンド	機能
node-red-start	node-red を service で起動しています。エラーが発生した場合には自動で再起動がかかります。node-red のコンソール画面上で Ctrl+C を押して動作を停止しても、バックグラウンドで処理が動いています。停止をするためには、node-red-stop のコマンドを実行してください。
node-red	node-red を起動します。Ctrl+C で処理を停止すると、その時点で node-red は終了します。

本システムを起動した時の Node-RED は node-red-start で起動しています。コンソール画面上で Ctrl+C を画面が消えますが、service は停止していません。

停止をするためにはコンソール画面上で

\$ node-red-stop

のコマンドを実行してください。

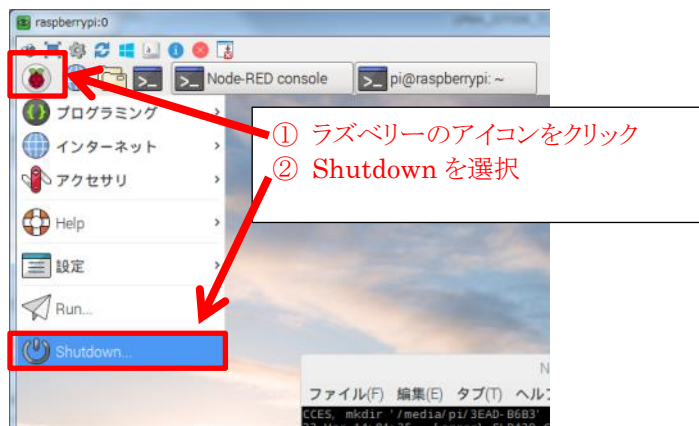
・ ゲートウェイの電源停止方法

以下の2つの電源停止を推奨します。

1) コンソール画面にコマンド入力をしてください

\$ sudo shutdown -h now

2) 画面上からシャットダウンを選択してください。



- Wi-Fi デザリングの使用方法

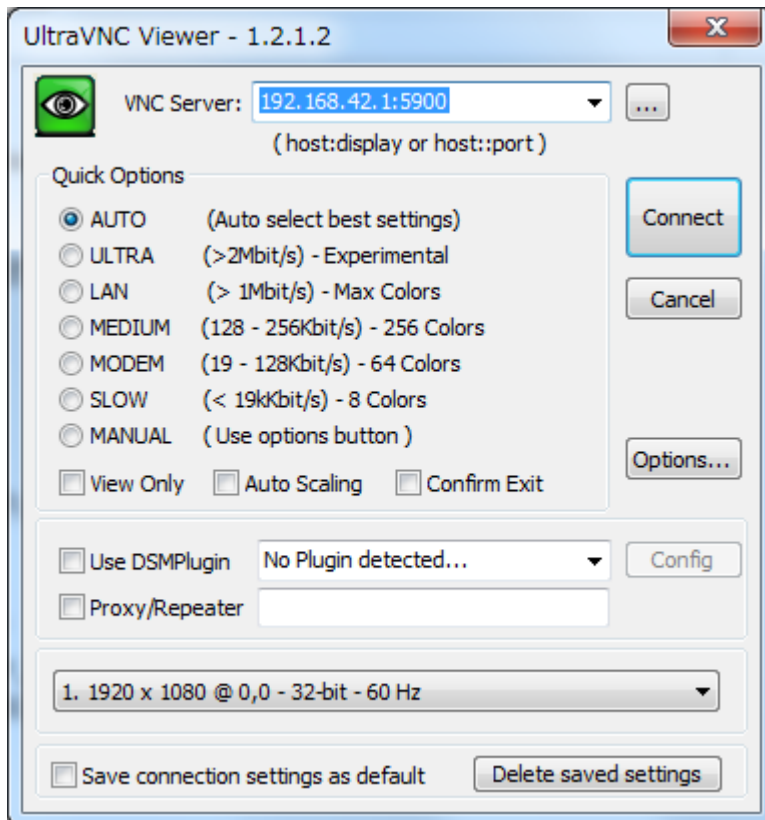
アクセスポイントで dcm_lpwapgw_xxx(xxx は 002～006)にアクセス
passphase の初期値は dcm201612

- Node-RED からの閲覧方法

Wi-Fi の接続先に「dcm_lpwapgw_xxx」にした PC でブラウザを起動し、次のアドレスを入力してください
192.168.42.1:1880

- リモートデスクトップ機能の使用方法

UltraVNC viewer に次の IP アドレスを入力し、x11vnc で設定したパスワードを入力してください。



connect を押し、パスワードを入力します。



初期値:dcm20161

● ゲートウェイについているデバイスを抜き差しする方法

・ C94-M8P(GSP モジュール)の場合

C94-M8P を外したときは 170324_noGPS.txt のフローを使用してください。

170324_noRTK.txt(RTK 送信無し)や、170324_RTK.txt(RTK 送信)は RTK の情報を受けてからシステムが動作し始めるため、C94-M8P を外すとシーケンスがスタートしません。

モジュールを外した後、170324_noGPS.txt を読み込んで、node-red で deploy をしてください。

・ RM-92A/SLR429/EnOcean の場合

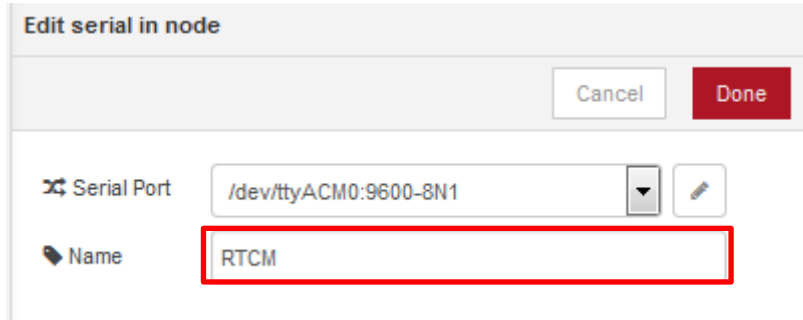
これらの USB デバイスはいつでも抜いて大丈夫ですが、接続をした場合は Node-RED を再度 deploy するか、Node-RED を再起動をしてください。deploy または Node-RED の再起動で無線モジュールが初期化されて使用できるようになります。

■ 5. ゲートウェイ ソフトウェア仕様

本項ではゲートウェイに搭載されている各ノードの機能について説明します。ノードの説明の図の中で赤線で囲ってあるパラメータについては、変更可能です。それ以外は、変更しないでください。

● RTCM ノード

1) シリアル設定



The screenshot shows a dialog box titled "Edit serial in node". At the top right are "Cancel" and "Done" buttons. Below the title bar, there are two fields: "Serial Port" with a dropdown menu showing "/dev/ttyACM0:9600-8N1" and a "Name" field with the text "RTCM". The "Name" field is highlighted with a red rectangular border.

① Serial Port

シリアルポートの設定を新規に作成、または設定済みのものを選択します。ボタンを押下することで、「シリアル詳細設定」へ遷移します。

② Name

Node-RED 上に表示される名称です。

2) シリアル詳細設定

① Serial Port

GPS モジュールのシリアルポートを指定します。

② Settings

シリアルポートのパラメータを設定します。

※変更しないでください。

③ Split input

シリアルポート入力のパケットの区切りを設定します。

※変更しないでください。

④ and deliver

シリアルポートノードの出力形式を設定します。

※変更しないでください。

● delay ノード

Edit delay node

Cancel Done

Action Limit rate to

Rate 1 msg(s) per 1 Minute

☒ drop intermediate messages

Name Name

① Action

ノード動作を指定します。

※変更しないでください。

② Rate

指定した時間に通過可能なメッセージ数を設定します。

時間を変更することで、RTCMを送信する頻度を変更することが出来ます。

※メッセージ数は変更しないでください。

③ drop intermediate messages

Rateを超過したメッセージの扱いを設定します。

※変更しないでください。

④ Name

Node-RED 上に表示される名称です。

● change ノード

Edit change node

Cancel Done

Name trigger

Rules

Set msg.payload

to a_z @trigger

※変更しないでください。

● delay ノード

Edit delay node

Cancel Done

Action Topic based fair queue

Rate 1 msg(s) per 2 Seconds

Name Name

※変更しないでください。

● RM92A ノード

1) RM-92A 設定

Dialog box titled "Edit rm92a out node".

Buttons: Cancel, Done

Fields:

- SerialPort: /dev/ttyRM92A
- Config: mode:2 ch:24 panid:0x1234 src:0x0001
- Default Dst: 0xFFFF

① SerialPort

シリアルポートの設定を新規に作成、または設定済みのものを選択します。
ボタンを押下することで、「シリアル詳細設定」へ遷移します。

② Config

RM-92A の設定を新規に作成、または設定済みのものを選択します。
ボタンを押下することで「RM-92A 詳細設定」へ遷移します。

③ Default Dst

ノードの入力メッセージに宛先が指定されていない場合に指定する宛先を設定します。

2) シリアル詳細設定

Dialog box titled "rm92a out > Edit rm92a-serial node".

Buttons: Delete, Cancel, Update

Fields:

- SerialPort: /dev/ttyRM92A

① SerialPort

RM-92A のシリアルポートを指定します。

3) RM-92A 詳細設定

rm92a out > Edit rm92a-config node

Delete Cancel Update

Reset

Mode

Channel

Pan ID

Src

Dst

TX Power mW

Band Width kHz ※ LoRa only

Factor ※ LoRa only

Bitrate bps ※ FSK only

ACK ☐

ACK timeout sec

ACK retry

RTC clock

① Reset

RM-92A 用のリセットスクリプトを指定します。

※変更しないでください。

② Mode

RM-92A の動作を FSK / LoRa から選択します。

③ Channel

無線のチャンネルを指定します。

有効範囲は 24 ～ 61 です。

④ Pan ID

無線の Pan ID を指定します。

有効範囲は 0x0000 ～ 0xFFFFC です。

⑤ Src

自身のアドレスを入力します。

有効範囲は 0x0000 ～ 0xFFFE です。

⑥ Dst

宛先のアドレスを入力します。

有効範囲は 0x0000 ～ 0xFFFF です。

0xFFFF はブロードキャスト用のアドレスです。

⑦ TX Power

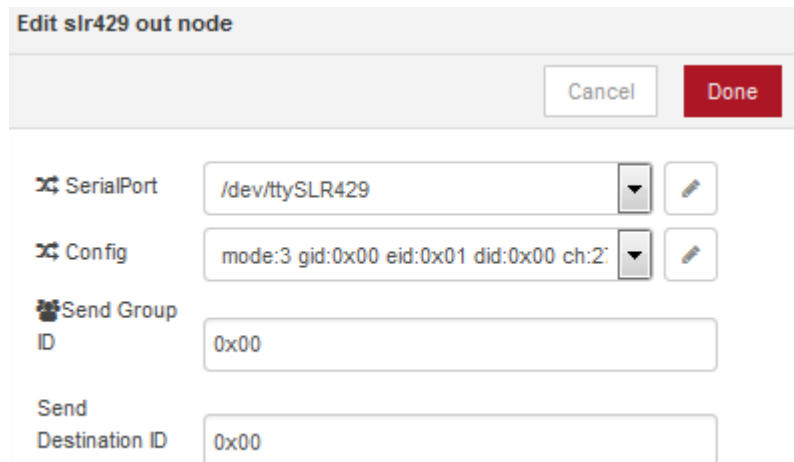
送信出力を指定します。

⑧ Band Width

帯域幅を指定します。

● SLR429 ノード

1) SLR429D 設定



① SerialPort

シリアルポートの設定を新規に作成、または設定済みのものを選択します。
ボタンを押下することで、「シリアル詳細設定」へ遷移します。

② Config

RM-92A の設定を新規に作成、または設定済みのものを選択します。
ボタンを押下することで「RM-92A 詳細設定」へ遷移します。

③ Default Dst

ノードの入力メッセージに宛先が指定されていない場合に指定する宛先を設定します。

2) シリアル詳細設定



① SerialPort

SLR429 のシリアルポートを指定します。
本プロジェクトでは/dev/ttySLR429 を指定してください。

3) SLR-429D 詳細設定

slr429 out > Edit slr429-config node

Delete Cancel Update

Mode LoRa Command Mode

Group ID 0x00

Equipment ID 0x01

Destination ID 0x00

Channel 27

Chip 128chip (245bps)

① Mode

LoRa Comannd Mode と FSK Comand Mode を選択することが出来ます

② Group ID

Group ID を指定してください。0x00-0xFF が指定可能です。

③ Equipment ID

自機の ID を指定します。有効範囲は 0x01 ～ 0xFF です。

④ Destination ID

送信先の ID を指定します。有効範囲は 0x00 ～ 0xFF です。

0x00 は broadcast で、0x01～0xFF が unicast です。

⑤ Channel

送信周波数を示す Channel を指定します。有効範囲は、7～46 です。

⑥ Chips

LoRa 通信モード時の Chip 数(Scale Factor)を設定します。次の中から選択をしてください。

128chip (245bps)

256chip (146bps)

512chip (86bps)

1024chip (49bps)

2048chip (27bps)

4096chip (15bps)

● EnOcean

• serialport

USB シリアルの設定を行うノードです。

serial in > Edit serial-port node

Delete

Cancel

Update

Serial Port

/dev/ttyEnOcean

Q

Settings

Baud Rate

57600

Data Bits

8

Parity

None

Stop Bits

1

Input

Split input

after a timeout of

100

ms

and deliver

binary buffers

Tip: In timeout mode timeout starts from arrival of first character.

•Serial Port: 変更しないでください。

•Settings: 変更しないでください

•Input: 変更しないでください

・ ESP3

EnOcean のプロファイルを解析するノードです。

Edit esp3 in node

Cancel

Done

Name

Name

EEPs

ID

04017d2a

EEP

a5-04-01

ID

0400d4ff

EEP

a5-04-01

ID

000004017d2a

+ add

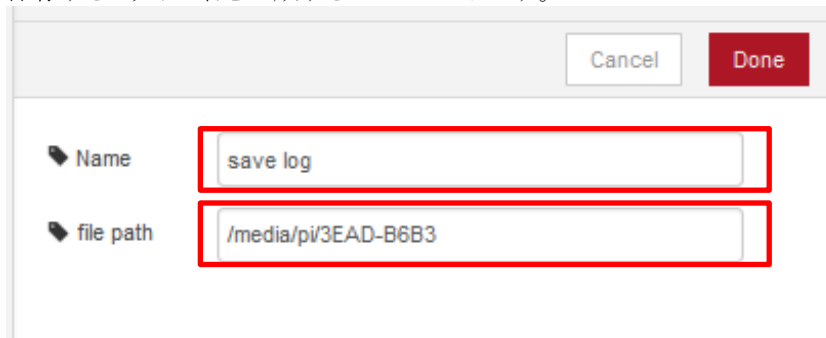
・ ESPs :

ID と EEP は接続するモジュールによって変更をしてください。
初期値は、納入したモジュールの ID と EEP が設定されています。

ID	EnOcean のモジュールに書き込まれているモジュール固有の ID を指定してください。
EEP	EnOcean のプロファイルを指定してください。対応しているプロファイルは、次の 2 種類のみです。 a5-02-01: 温度プロファイル a5-04-01: 温度・湿度プロファイル

● log file

保存するファイル名を生成するためのノードです。



name: 任意です。

file path: ログを保存するファイルのパスを指定してください。

log file のノードの後ろに file(output)のノードを置くことで、
file path で指定したフォルダに、”年月日時+msg.id.csv”の命名則に基づくファイル名にログが保存されていきます。

たとえば、時刻が yyyy 年 mm 月 dd 日 HH 時で msg.id が無い場合、ファイル名は”yyyymmddHH.csv”になります。

■ 6. センサーノードの使用方法

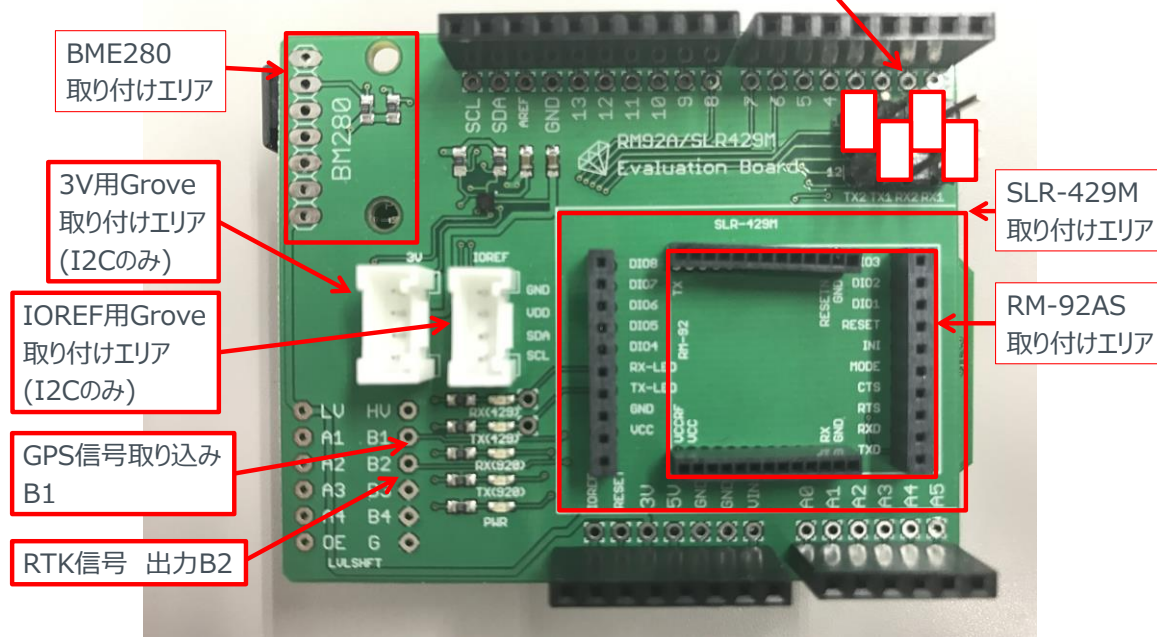
● ボードの構成と各部の説明

・RM92A/SLR429M 用シールド基板

基板の概要を以下に示します。各無線モジュールに併せた設定をしてください。

ショートプラグ領域

Serial(TX)	Serial(TX)	Serial(RX)	Serial(RX)
RTK出力	RFのRX	GPS入力	RFのTX
Serial1(TX)	Serial1(TX)	Serial1(RX)	Serial1(RX)



・RM92AS と SLR429M を使用する場合のショートプラグ設定:

使用時は以下のようにショートプラグを接続してください。

Serial(TX)	Serial(TX)	Serial(RX)	Serial(RX)
RTK出力	RFのRX	GPS入力	RFのTX
Serial1(TX)	Serial1(TX)	Serial1(RX)	Serial1(RX)

・IEEE802154K を使用する場合のショートプラグ設定:

使用時は以下のようにショートプラグを接続してください。IEEE802154K 使用時は GPS 信号を Serial1(RX)/Serial1(TX) 経由でやり取りをするため、プログラム書き込み時にショートプラグの設定を変更する必要があります。

Serial(TX)	Serial(TX)	Serial(RX)	Serial(RX)
RTK出力	RFのRX	GPS入力	RFのTX
Serial1(TX)	Serial1(TX)	Serial1(RX)	Serial1(RX)

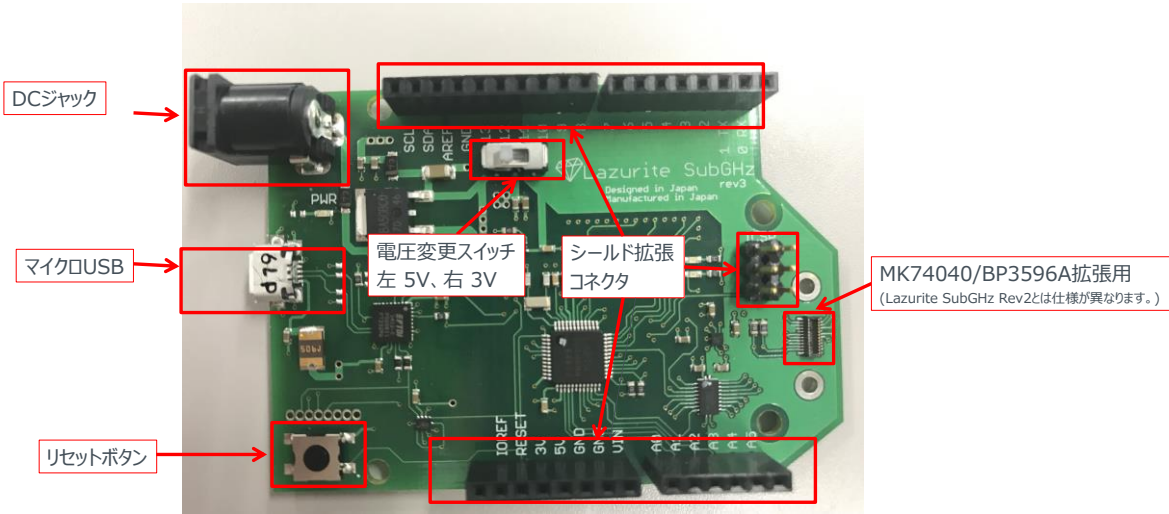
・プログラム書き込み時のショートプラグ設定 (IEEE802154K 使用時を除く):

LazuriteIDE からプログラムを書き込みするときに Serial(TX)と Serial(RX)を使用しています。そのため、プログラム書き込み時は、Serial(RX)と Serial(TX)の配線がオープンになるように、ショートプラグを外してください。

Serial1(RX)と Serial1(TX)は接続されていても問題はありません。

Serial(TX)	Serial(TX)	Serial(RX)	Serial(RX)
RTK出力	RFのRX	GPS入力	RFのTX
Serial1(TX)	Serial1(TX)	Serial1(RX)	Serial1(RX)

- ・マイコンボード Lazurite SubGHz Rev3
Lazurite SubGHz Rev3 基板の概要を示します。

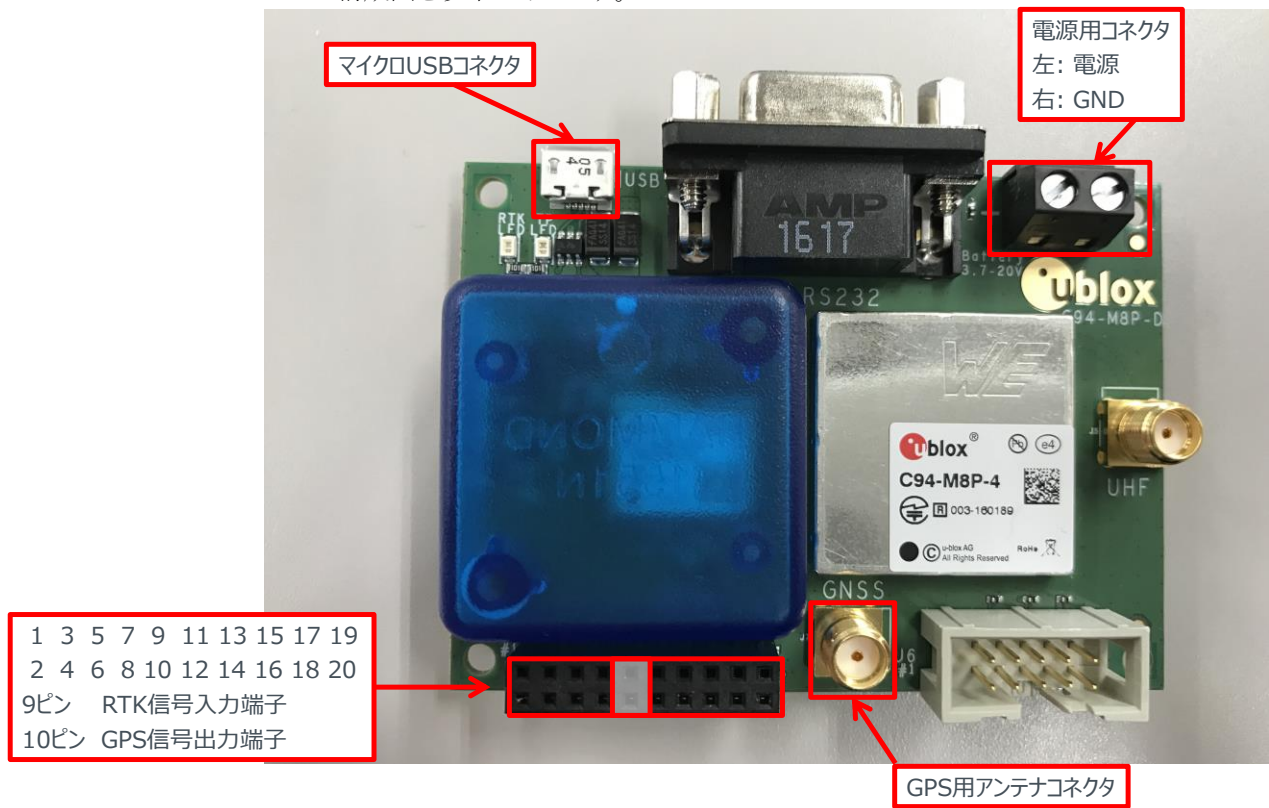


電圧変更スイッチについて :
使用する無線モジュールに併せて次の通り設定を変更してください。

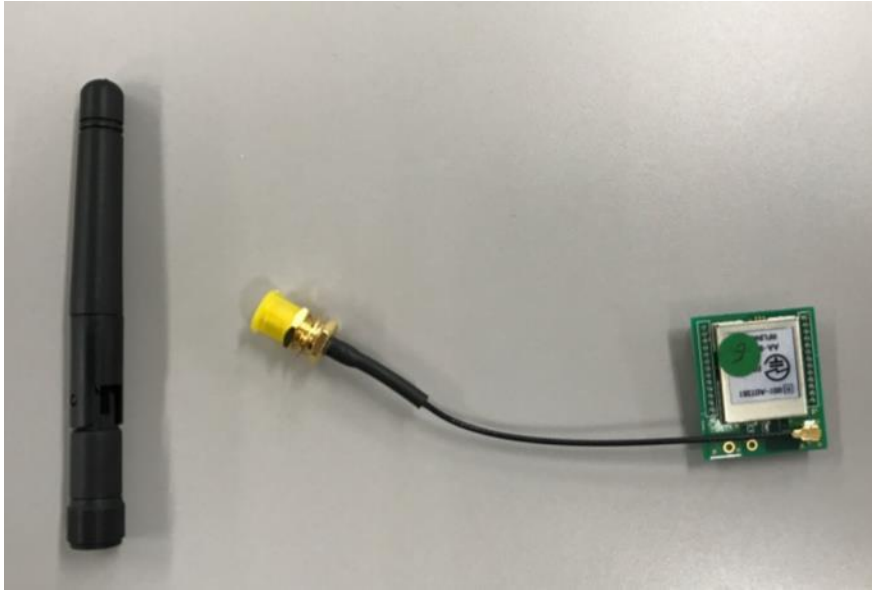
無線モジュール	設定電圧	スイッチの向き
RS-92AS	3V	右
SLR-429M	5V	左
IEEE802154K	3V	右

・ GPS モジュール/C94-M8P

GPS モジュール C94-M8P の構成図を以下に示します。



- ・ RM-92AS 無線モジュールとアンテナ
RM-92AS 取り付けエリアにモジュールを接続してください。



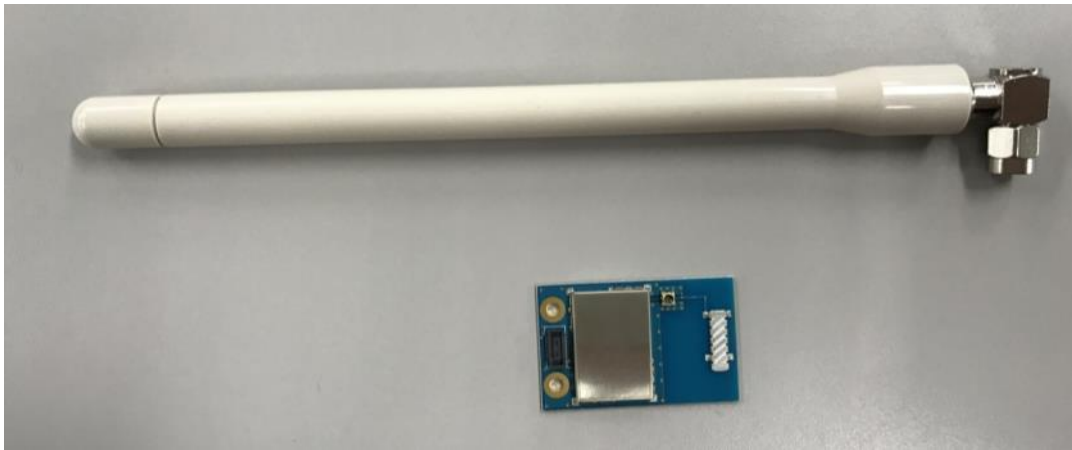
- ・ SLR-429M 無線モジュールとアンテナ

SLR-429M 取り付けエリアにモジュールを接続してください。



- ・ IEEE802154K 無線モジュール

MK74040/BP3596A 取り付けコネクタに接続をしてください。



● RM-92AS センサーノードについて

・プログラム書き込み方法

プログラムの書き込みは、LazuriteIDE を介して行うことができます。

プログラムの書き込みを行う際は、

- ・Lazurite SubGHz Rev3 から、RM92A/SLR429M 用シールド基板を外す。
- ・RM92A/SLR429M 用シールド基板のディップスイッチをプログラムが書き込める状態にするのいずれかを行ってください。

・組み立て方法

- ・Lazurite SubGHz rev3 の電源電圧設定を 3V にしてください。
- ・RM92A/SLR429M 用シールドに RM-92AS を接続してください。
- ・Lazurite SubGHz rev3 と RM92A/SLR429M 用シールド基板を接続します。
- ・RM92A/SLR429M 用シールドに BME280 モジュールを接続してください。
- ・RM92A/SLR429M 用シールド基板の B1 の信号を、C94-M8P 拡張 IO の 10 ピンに接続してください。
- ・RM92A/SLR429M 用シールド基板の B2 の信号を、C94-M8P 拡張 IO の 9 ピンに接続してください。

● SLR-429M センサーノードについて

・プログラム書き込み方法

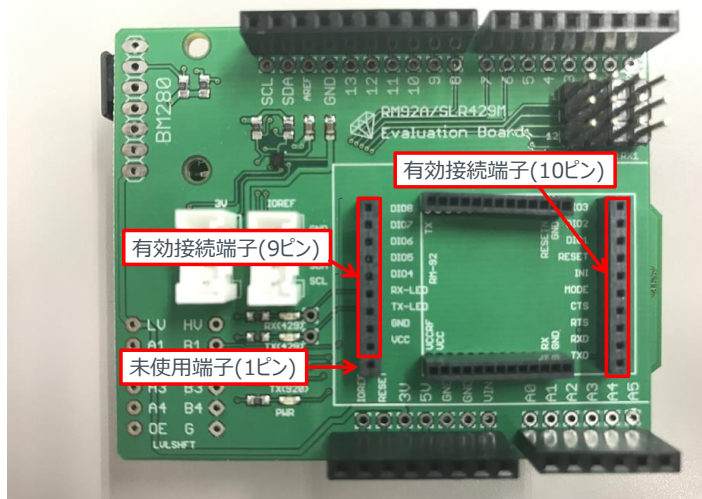
プログラムの書き込みは、LazuriteIDE を介して行うことができます。

プログラムの書き込みを行う際は、

- ・Lazurite SubGHz Rev3 から、RM92A/SLR429M 用シールド基板を外す。
- ・RM92A/SLR429M 用シールド基板のディップスイッチをプログラムが書き込める状態にするのいずれかを行ってください。

・組み立て方法

- ・Lazurite SubGHz rev3 の電源電圧設定を 5V にしてください。
 - ・RM92A/SLR429M 用シールドに SLR-429M を接続してください。
- SLR429M を接続するときには、下図のように左下の端子が 1 つ余りますので注意してください。



- ・Lazurite SubGHz rev3 と RM92A/SLR429M 用シールド基板を接続します。
- ・RM92A/SLR429M 用シールドに BME280 モジュールを接続してください。
- ・RM92A/SLR429M 用シールド基板の B1 の信号を、C94-M8P 拡張 IO の 10 ピンに接続してください。
- ・RM92A/SLR429M 用シールド基板の B2 の信号を、C94-M8P 拡張 IO の 9 ピンに接続してください。

● IEEE802154K センサーノードについて

・プログラム書き込み方法

プログラムの書き込みは、LazuriteIDE を介して行うことができます。IEEE802154K モジュール使用時のショートプラグ設定どおりに接続をすれば、ショートプラグの設定を変更するひうがありません。

・組み立て方法

- ・Lazurite SubGHz rev3 の電源電圧設定を 3V にしてください。
- ・Lazurite SubGHz rev3 に MK74040 モジュールを取り付けてください。
- ・Lazurite SubGHz rev3 と RM92A/SLR429M 用シールド基板を接続します。
- ・RM92A/SLR429M 用シールドに BME280 モジュールを接続してください。
- ・RM92A/SLR429M 用シールド基板の B1 の信号を、C94-M8P 拡張 IO の 10 ピンに接続してください。
- ・RM92A/SLR429M 用シールド基板の B2 の信号を、C94-M8P 拡張 IO の 9 ピンに接続してください。

● センサーノードのソフトウェア開発

・開発環境のセットアップ

センサーノードのプログラムは Lazurite IDE を用いて行うことができます。

Lazurite IDE は Lazurite のホームページからダウンロードをしてください。

URL: www.lapis-semi.com/lazurite-jp

センサーノードのソフトウェアは github のプロジェクトに含まれています。

LazuriteIDEをインストールすると、c:\¥LazuriteIDEのフォルダに開発環境がインストールされますので、そのフォルダに github からダウンロードしたフォルダの node 以下のファイルをすべてコピーしてください。

フォルダ	ライブラリ名	ファイル	機能
node/		LazuriteIDE.pref	Lazurite SubGHz rev3 を使用するための環境設定ファイル
node/ hardware		**, *	Lazurite SubGHz rev3 を使用するためのファイル
node/libraries	RM92ALIB	RM92ALIB.C RM92ALIB.H	RM92A 用のライブラリです。
	SLR429LIB	SLR429LIB.C SLR429LIB.H	SLR429 用のライブラリです。
	C94M8PLIB	C94M8PLIB.C C94M8PLIB.H	C94M8P 用のライブラリです。
node/examples/00.lpwasn	RM92A_2	RM92A_2.C RM92A_2.SSF RM92A_2_IDE.H	RM92A の子機用メインプログラムで、子機のアドレスが 2 に設定されています。LazuriteIDE でソフトウェアの編集、ビルド、センサーノードへの書込みが可能です。
	RM92A_3	RM92A_3.C RM92A_3.SSF RM92A_3_IDE.H	RM92A の子機用メインプログラムで、子機のアドレスが 3 に設定されています。LazuriteIDE でソフトウェアの編集、ビルド、センサーノードへの書込みが可能です。
	SLR429_2	SLR429_2.C SLR429_2.SSF RM92A_2_IDE.H	SLR429 の子機用メインプログラムで、子機のアドレスが 2 に設定されています。LazuriteIDE でソフトウェアの編集、ビルド、センサーノードへの書込みが可能です。
	SLR429_3	SLR429_3.C SLR429_3.SSF SLR429_3_IDE.H	SLR429 の子機用メインプログラムで、子機のアドレスが 3 に設定されています。LazuriteIDE でソフトウェアの編集、ビルド、センサーノードへの書込みが可能です。

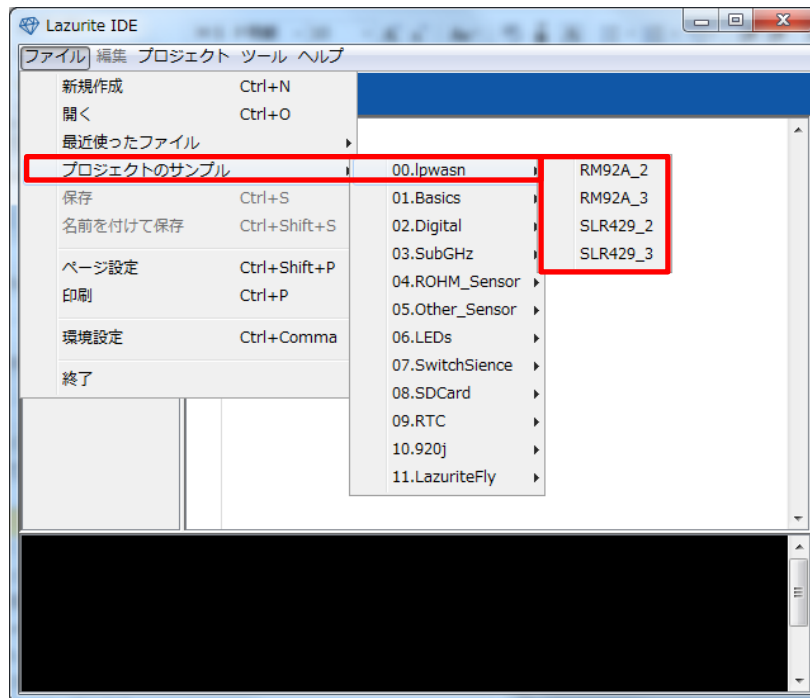
(補足)

~.ssf : LazuriteIDE で開くことが出来るプロジェクトファイルです。

~_IDE.H : LazuriteIDE が自動で生成するプロジェクトのヘッダファイルです。

・ センサーノードのプログラム

センサーノードのプログラムは、上記の作業を行うことで「プロジェクトのサンプル」 → 「00.lpwasn」から選択を行うことが出来ます。



・ センサーノードの設定変更

センサーノードのプログラムは次のようになっています。パラメータの部分のみを変更して使用してください。

```

#include "RM92A_2_defs.h" // Additional Header

#define RF_MODE FSK920 // Chose from LORA429, FSK429, LORA920, FSK920

#if RF_MODE == FSK920
#define RF_MODULE rm92a
#define RM92A
#define DEBUG 1 // must be remove Q34M8P
#elif RF_MODE == LORA429
#define RF_MODULE slr429
#define SLR429
#define DEBUG 1
#elif RF_MODE == LORA920
#define RF_MODULE rm92a
#define RM92A
#define DEBUG 1
#elif RF_MODE == FSK429
#define RF_MODULE slr429
#define SLR429
#define DEBUG 1
#else
#error RF_MODE must be set
#endif

#define RF_SRC 2 // First byte of payload show the address. Same data start

// COMMON PARAMETER
#define ORANGE 25
#define BLUE 26

// LORA 329 Initial Parameter
#define LORA920_BOOT0 4 // GPIO setting
#define LORA920_RST 5 // GPIO setting

#define RF_CH 24 // RF frequency. Chose from 24-61
#define RF_PANID 0x1234 // PANID choose from 0x0000 - 0xFFFF
// #define RF_SRC 0x2 // my address. choose from 2-3
#define RF_DST 0x1 // gateway address. (it should be 1)

static const t_RM92A_CONFIG config =
{
    LORA920_BOOT0,
    LORA920_RST
};

static struct s_rm92a_settings param;

// LORA429 Initial Parameter
#define SLR429
#define LORA429_RTS 2 // GPIO setting
#define LORA429_DTS 3 // GPIO setting
#define LORA429_INI 4 // GPIO setting
#define LORA429_RSTB 5 // GPIO setting

#define RF_CH 27 // RF frequency. Chose from xxx-xxx
#define RF_PANID 0 // PANID choose from 0 - FF
// #define RF_SRC 3 // my address. choose from 2 - 3
#define RF_DST 1 // gateway address. (it should be 1)

```

RF_MODE 使用する無線モードを指定します。
次の4つから使用するモードを選択してください。
LORA429、FSK429、LORA920、FSK920

RF_SRC
子機の自機アドレスを示しています。

RM92A 用の周波数、PANID、送信先を示しています。

RF_CH: RM92A 用の周波数
RF_PANID: PANID
RF_DST: ゲートウェイのアドレス

RM92A 用の周波数、PANID、送信先を示しています。

RF_CH: RM92A 用の周波数
RF_PANID: PANID
RF_DST: ゲートウェイのアドレス

① RF_MODE

無線のモードを指定します。次の4種類から選択をすることが出来ます。

設定値	無線モジュール	モード
LORA429	SLR-429M	429MHz の LORA モードで起動します。
FSK429	SLR-429M	429MHz の FSK モードで起動します。
LORA920	RM-92A	920MHz の LORA モードで起動します。
FSK920	RM-92A	920MHz の FSK モードで起動します。

② RF_SRC

子機のアドレスです。2 または 3 をセットしてください。

③、④ RM92A/SLR-429 の無線設定

変数	パラメータの意味	設定可能値	
		RM-92A	SLR-429M
RF_CH	チャンネル(周波数)	24-61	7-46
RF_PANID	PANID / グループ ID	0-0xFFFFC	0-0xFF
RF_SRC	自機アドレス	2 または 3	2 または 3
RF_DST	送信先アドレス (ゲートウェイアドレス)	1	1

・無線パラメータの詳細変更

・RM92A のパラメータを変更する方法

214 行目と 215 行目の間、もしくは 217 行目と 218 行目の間でパラメータを変更してください。

```
213
214 RF_MODULE.getMode(&param);
215 #ifdef DEBUG
216 param.debug = true;
217 #endif
218 RF_MODULE.setMode(&param);
219 result = RF_MODULE.init(&Serial1, %config);
220
```

どちらかに追加をしてください

変更可能なパラメータ

param.ack.retry	0: No retry, 1~5 : retry, 6~ : error
param.rf.txpwr	1: TX-Power Set[0:20mW[+13dBm] 1:4mW[+6dBm] 2:1mW[+0dBm]
param.rf.bw	Bandwidth Set[0:125kHz 1:250kHz 2:500kHz]
param.rf.sf:	Factor(SF) Set[0:SF6 1:SF7 2:SF8 3:SF9 4:SF10 5:SF11 6:SF12]

・RM92AS のデフォルト設定を変更する方法

LazuriteIDE¥libraries¥RM92ALIB¥RM92ALIB.C の 105 行目~132 行目

以下のパラメータを変更してください。

```
struct s_rm92a_settings rm92a_settings = {
  0, // unsigned char debug; // 0: normal 1: debug
  2, // unsigned char routing_mode; // 0: Fixation 1: AutoRouting 2:NonRouting
  1, // unsigned char unit_mode; // 0: parent 1: child
  1, // unsigned char dt_mode; // Data Transfer Mode[0:Discharge 1:Frame 2:TimerSend 3:SleepTimerSend(Non Routing Only)]
  {
    1, // bool enable;
    1, // unsigned char timeout;
    5, // unsigned char retry;
  }, // } ack;
  {
    1, // bool rssi;
    1, // bool src;
  }, // } output;
  {
    1, // bool enable; // Carrier Sense Enable[0:Not Use 1:Use]
    5, // unsigned char retry; // Carrier Sense Retry Counter [0:Not Retry 1-9:Retry Count]
  }, // } cca;
  {
    0, // unsigned char tx_pwr; // RF Settings[1:TX-Power Set 2:Transmit Time-Total Set 3:Transmit Down-Time Set
    1, // unsigned char tts; // 1: TX-Power Set[0:20mW[+13dBm] 1:4mW[+6dBm] 2:1mW[+0dBm]
    0, // unsigned char dts; // 2: Transmission Time-Total Set[0:Not Use 1:Use]
    0, // unsigned char bw; // 3: Transmission Time-Total Set[0:Not Use 1:2msec 2:50msec 3:TransmitTime*10]
    4, // unsigned char sf; // 4: Bandwidth Set[0:125kHz 1:250kHz 2:500kHz]
    5, // Factor(SF) Set[0:SF6 1:SF7 2:SF8 3:SF9 4:SF10 5:SF11 6:SF12]
  }, // } rf;
  1, // unsigned char rf_mode; // 1: Lora 2: FSK
};
```

SLR429 のパラメータ変更方法

・SLR429 のパラメータを変更する方法

```
213 RF_MODULE.setMode(%param);
214 #ifdef DEBUG
215 param.debug = true;
216 #endif
217 RF_MODULE.setMode(%param);
218 result = RF_MODULE.init(&Serial1, %config);
219
220
```

どちらかに追加をしてください

param.chips = @SF 設定値

- 0: 128chip(実測値: 245bps)
- 1: 256chip(実測値: 146bps)
- 2: 1024chip(実測値: 86bps)
- 3: 2048chip(実測値: 49bps)
- 4: 4096chip(実測値: 15bps)

param.txretry=再送回数 0～255 任意。初期値は 5。

・SLR429 のデフォルト設定を変更する方法

LazuriteIDE¥libraries¥SLR429LIB¥SLR429LIB.C の 34 行目～40 行目

以下のパラメータを変更してください。

```
static t_SLR429_CONFIG slr429_config;
struct s_slr429_settings slr429_settings = {
    0, // debug
    0, // chips
    5, // txretry
    8, // minbe 2^n[ms]
    9, // maxbe 2^n[ms]
};
```

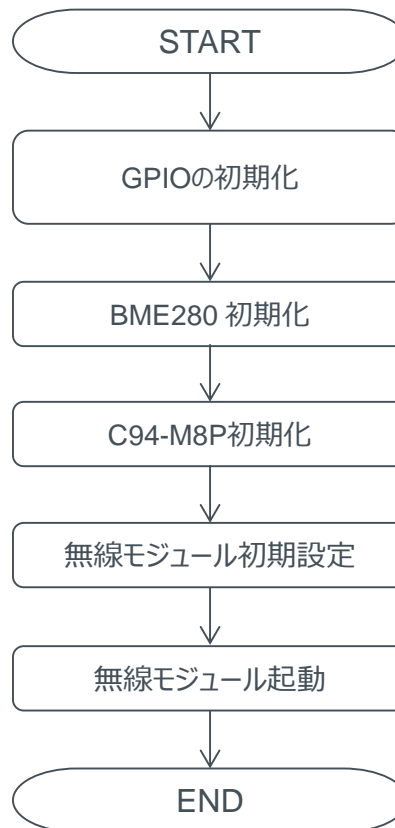

■ 7. センサーノード ソフトウェア仕様

● ソフトウェア構成図

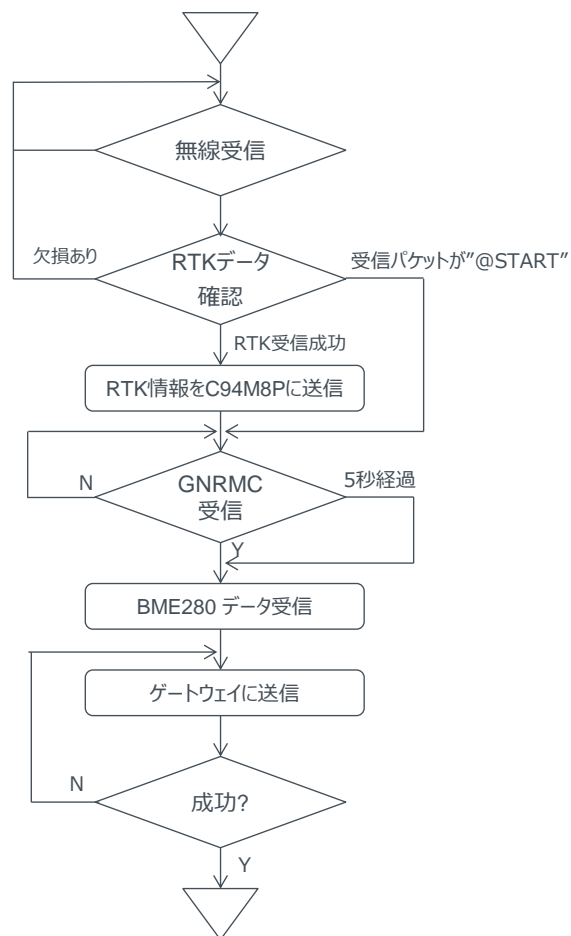
Application	LPWA_SN		
New libraries	RM92ALIB_NEW SLR429LIB	C94M8PLIB	BME280
Lazurite Standard	Lazurite firmware/libraries		
	Serial	Serial1	Wire (I2C)
External Modules	RM-92A SLR-429M	1 C94-M8P	BME280

非公開
公開予定
公開済
ハードウェア

● 初期化プログラム (setup)のフローチャート



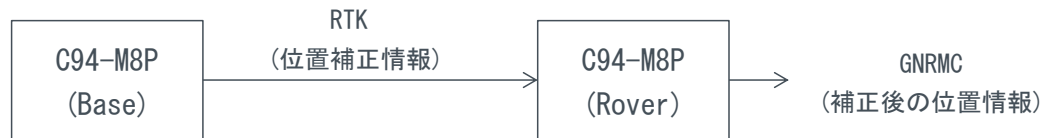
● メインルーチン(loop)のフローチャート(RM92A／SLR429 共通)



■ 8. GPS モジュール C94-M8P の設定

● C94-M8P 概要

C94-M8P は正確な位置情報を設定した BASE 機から送信される位置補正情報(RTK)を Rover に入力することで、通常の GPS よりも正確な位置情報を得ることが出来るシステムです。



● ゲートウェイ側の設定

ゲートウェイ側の C94-M8P は Base 側の設定を行う必要があります。ベース側は参照となる位置を決める必要があるため、本作業はアンテナの取り付けが完了して位置が確定した状態で作業を行う必要があります。

・ 環境セットアップ

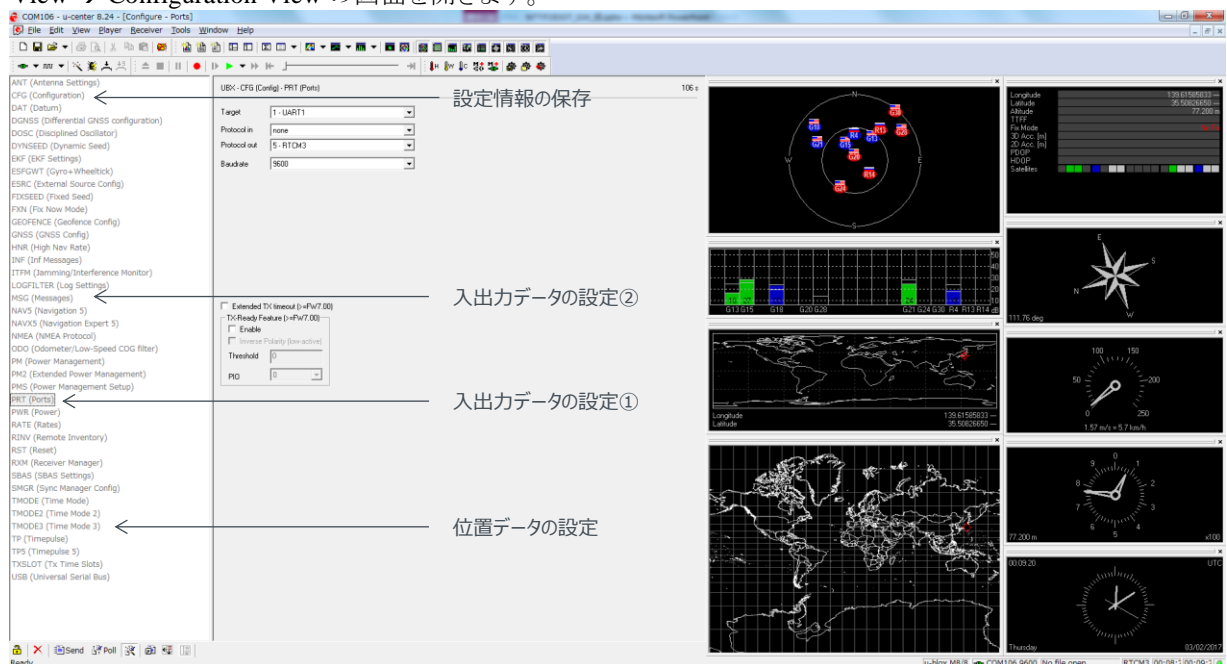
設定作業は Windows PC を用いて行ってください。

本マニュアルは、u-blox 社が配布するソフトウェア u-center 8.24 を用いて作業を行っています。

u-center 8.24 のインストールや USB ドライバ等のインストールは u-blox が提供するマニュアルを参照してください。出荷時には、①～④は行われています。設置時は、アンテナの位置を決めた後に⑤以降の設定を行ってください。

・ 作業手順

- ① Windows PC に Base 側の設定をする C94-M8P を接続して、u-center 8.24 を起動します。
- ② View → Configuration View の画面を開きます。



・ ゲートウェイへの接続について

C94-M8P を外したときは 170324_noGPS.txt のフローを使用してください。

170324_noRTK.txt(RTK 送信無し)や、170324_RTK.txt(RTK 送信)は RTK の情報を受けてからシステムが動作し始めるため、C94-M8P を外すとシーケンスがスタートしません。

モジュールを外した場合は、170324_noGPS.txt のフローを Node-RED に読み込んで、deploy をしてください。

③ 出力データの設定①

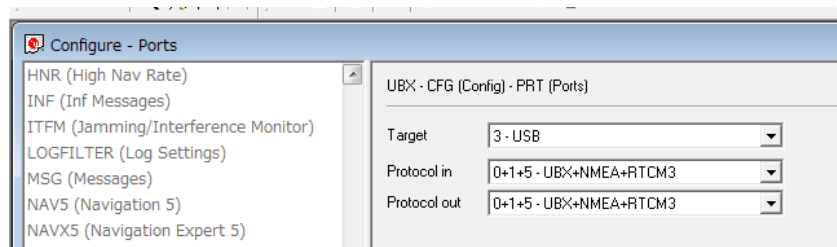
メニューから PRT(Ports)をクリックし、各ポートから出力する信号を決定します。

Target を USB にして、Protocol out を「0+1+5 UBX+NMEA+RTCM3」に設定してください。

④ 出力データの設定②

メニューから MSG(Message)を選択し、各メッセージの出力先として UART1、USB を選択していきます。

設定が完了したら、Send ボタンを押してください。



- 1) Target: USBに設定
- 2) Protocol in : 変更しない
- 3) Protocol out: 0+1+5-RTCM3に設定

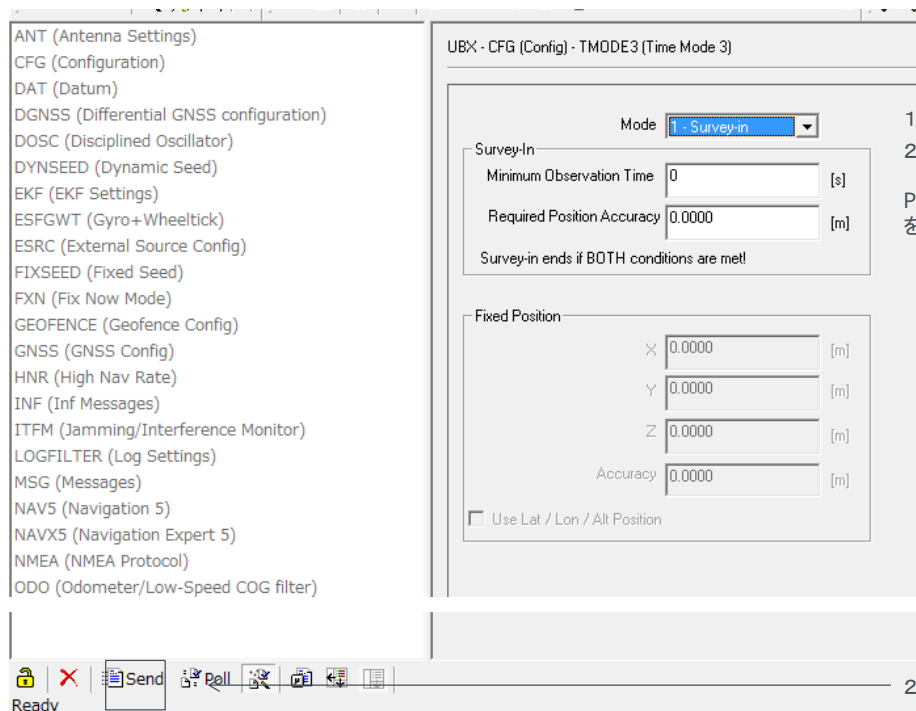


- 4) 最後に Sendを押す

⑤ 位置データの設定

ここでは、GPS から取得された情報を使用して位置補正を行う「Servay-in」を実施します。

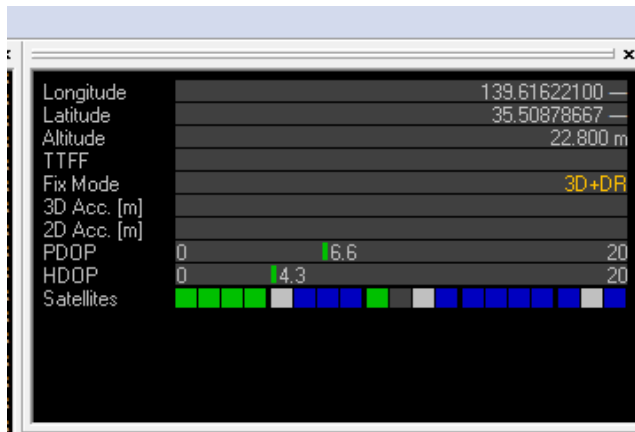
メニューから TMODE3(Time Mode 3)を選択して、位置を固定し、Send ボタンを押します。



- 1) Survey-inを選択
- 2) Minimum Observation Time (最小計測時間)と、Required Position Accuracy(要求位置精度)を指定します。

- 2) 最後に Sendを押す

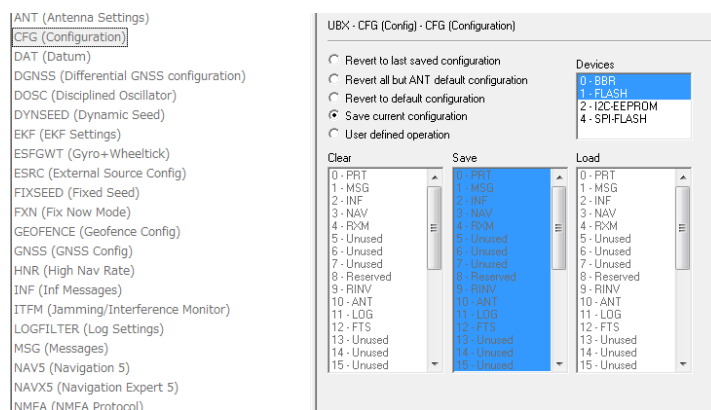
Servay-in が完了すると、Fix Mode がこのようになります。



Servay-inが完了すると、
Fix Modeがこのようになります。

⑥ 設定情報の保存

設定した情報の保存を行います。CFGを選択し、Send ボタンを押してください。

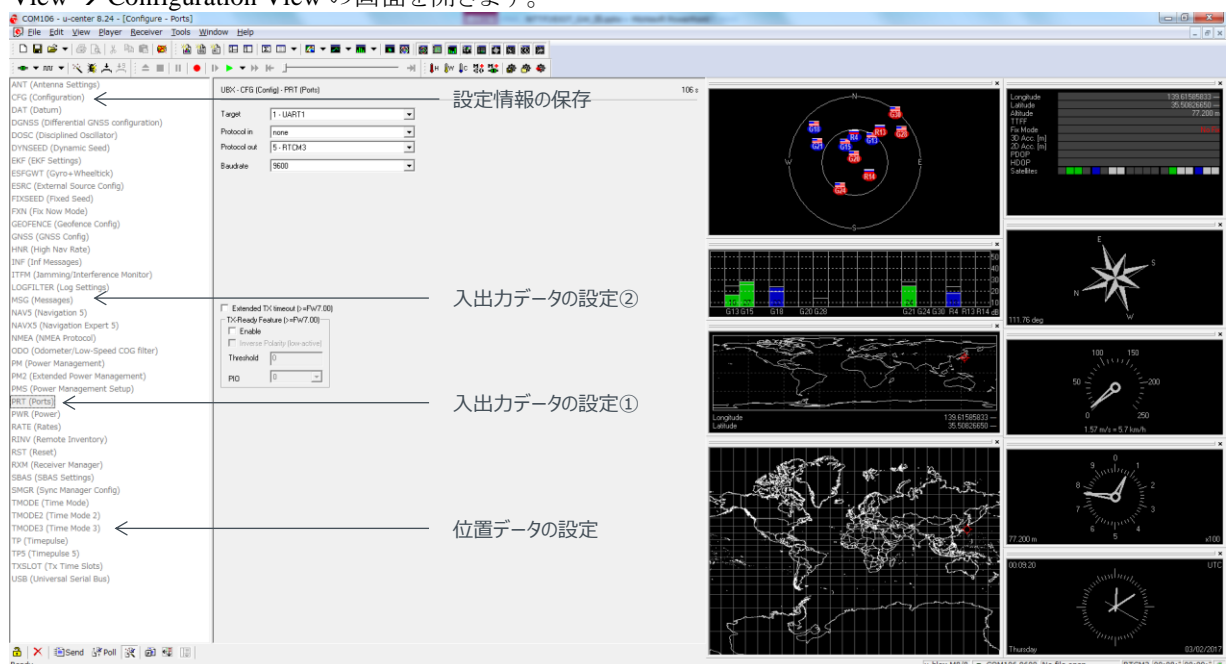


Sendを押す

● センサー側の設定

・ 作業手順

- ① Windows PC に Base 側の設定をする C94-M8P を接続して、u-centor 8.24 を起動します。
- ② View → Configuration View の画面を開きます。



設定情報の保存

入力データの設定②

入力データの設定①

位置データの設定

③ 入力データの設定①

メニューから **PRT(Ports)**をクリックし、各ポートから入出力する信号を決定します。

以下のように設定を行ってください。

設定が完了したら、**Send** ボタンを押してください。

UBX - CFG (Config) - PRT (Ports)	
Target	1 - UART1
Protocol in	5 - RTCM3
Protocol out	1 - NMEA
Baudrate	9600

- 1) Target: UART1に設定
- 2) Protocol in : RTCM3に設定
- 3) Protocol out: NMEAに設定
- 4) Baudrate: 9600に設定



Sendを押す

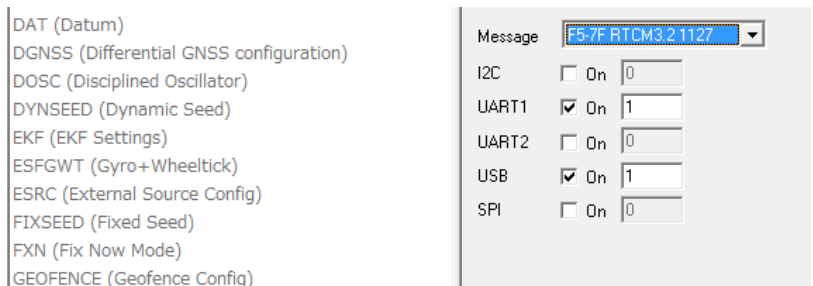
・センサーノードへの接続について

センサーノードは **C94-M8P(GSP モジュール)**の有無に関わらずに動作します。センサーノード側で **GPS** 信号を 5 秒間受信できない場合は受信を停止し、センサーの値のみを送信します。

外すときはいつでもOKですが、接続する場合は、安全のために一旦電源を落として **C94-M8P** を接続した後、再度電源を投入してください。

④ 入力データの設定②

メニューから MSG(Message)を選択し、各メッセージの出力先として UART1、USB を選択していきます。

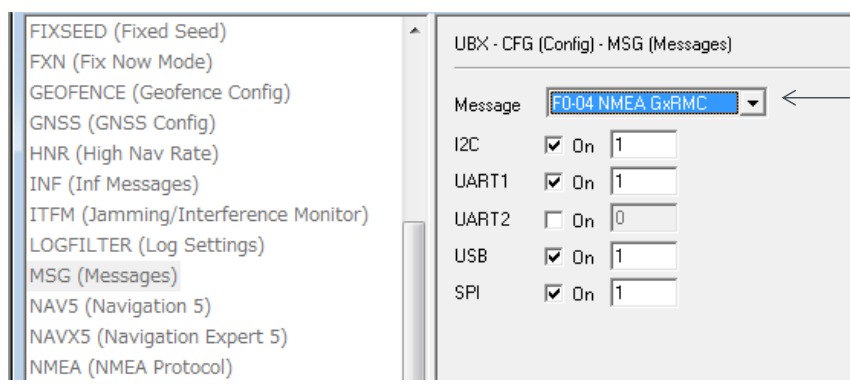


1) Messageから次のメニューを選択し
全てUART1、USBを有効にしています。

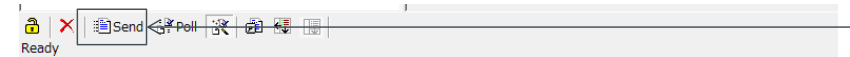
F05-05 RTCM3.2	1005
F05-04 RTCM3.2	1077
F05-57 RTCM3.2	1087
F05-7F RTCM3.2	1127
F05-E6 RTCM3.2	1230

続いて、NMEA GxRMC を選択し、UART1 を有効にしてください。

最後に、Send ボタンを押してください。



F0-04 NMEA GxRMC
を選択し、UART1を有効にしてください。



Sendを押す

以上で、設定は完了です。

■ 9. ゲートウェイの詳細設定

● プログラムの保存場所

~/docomo/lpwagw	Github のサーバーとリンクしたフォルダです。 プログラムが保存してあるだけで実際にこのフォルダ内ではプログラムを動かしていませんが、この中のプログラムを更新すると update 用スクリプトにより GitHub サーバーからプログラムの更新が出来なくなるため、注意してください。
~/gateway	Gateway の GPIO 設定や監視用プログラムが保存されている炉札です。
~/node-red/node_modules	Node-red のライブラリが保存されています。

● プログラムの更新方法

プログラムを更新する場合は次のコマンドを実行してください。GitHub に保存されているプログラムを取得し、上記の 3 つのフォルダに保存されているプログラムをすべて更新し、その後システムを再起動します。

“~/docomo/lpwagw”の内部を変更すると更新プログラムが正常に動作しなくなります。その時は、手動で github サーバーからダウンロードできるようになるようプロジェクトのフォルダの修正を行ってください。

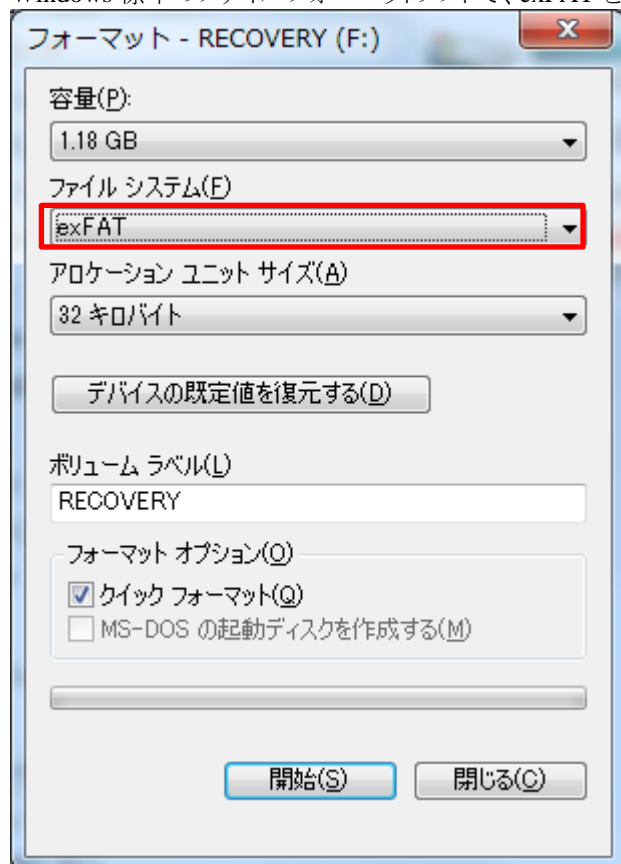
```
$ cd ~  
$ ./update.sh
```

(注意) update.sh を変更しました

● USB メモリにログを保存する方法

1) USB メモリのフォーマット

ログを保存する USB メモリは、Windows と Linux の双方で Read/Write を行うことができる exFAT を推奨します。Windows 標準のファイルフォーマットソフトで、exFAT を指定することが出来ます。



2) ゲートウェイに USB メモリをマウントする

USB メモリを Raspberry Pi に装着しても最初は自動で認識されないために手動でマウントをする必要があります。

USB にマウントできているか調べるときは、”df -l”コマンドを入力してください。

以下のログの場合は”/dev/sda1”が USB デバイスであり、”/media/pi/3EAD-B6B3”がマウントポジションになります。

本システムでログファイルを保存するとき、log file のノードに”/media/pi/3EAD-B6B3”を指定すると、USB メモリの直下に日付+時間のファイルが保存されていきます。

```
pi@raspberrypi:~$ df -l
ファイルシステム 1K-ブロック 使用 使用可 使用% マウント位置
/dev/root          12428816 4065084  7806412   35% /
devtmpfs           437052      0  437052    0% /dev
tmpfs              441384    8764  432620    2% /dev/shm
tmpfs              441384    6224  435160    2% /run
tmpfs               5120         4    5116    1% /run/lock
tmpfs              441384      0  441384    0% /sys/fs/cgroup
/dev/mmcblk0p1     64456    21312  43144   34% /boot
tmpfs              88280      0    88280    0% /run/user/1000
/dev/sda1          15699424  3264 15696160    1% /media/pi/3EAD-B6B3
```

USB メモリが自動的にマウントされない場合は、手動でマウントをしてください。

3) 手動で USB メモリをマウントする方法

usb メモリを挿入して dmesg のコマンドを実行すると、以下のような log が表示されます。この log の結果から sda1 が USB デバイスになります。

```
$ dmesg
[ 1926.779749] sd 3:0:0:0: [sda] 31422464 512-byte logical blocks: (16.1 GB/15.0 GiB)
[ 1926.780297] sd 3:0:0:0: [sda] Write Protect is off
[ 1926.780316] sd 3:0:0:0: [sda] Mode Sense: 23 00 00 00
[ 1926.780847] sd 3:0:0:0: [sda] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[ 1926.785459] sda: sda1
[ 1926.789861] sd 3:0:0:0: [sda] Attached SCSI removable disk
```

続いて、以下のように実行してください。ここでは”/mnt/usb0”を USB デバイスとして認識します。

```
$ sudo mkdir /mnt/usb0
$ sudo mount /dev/sda1 /mnt/usb0
```

なお 2 回目以降は自動認識するようになりますが、初回と 2 回目以降でマウント先が変わる可能性がありますので注意してください。2)の確認で使用した”df -l”のコマンドでマウント先が表示されたら成功です。

● カーネルファイルの作成

1) Lazurite の初期設定を行う

以下のホームページに従い、Lazurite に対応した Raspberry Pi のカーネルファイルを作成します。

<http://www.lapis-semi.com/lazurite-jp/raspberry-pi%E9%96%A2%E9%80%A3/10083.html>

● ゲートウェイに必要なモジュールのインストール

1) 必要なプログラムのインストール

```
$ sudo apt-get update
$ sudo apt-get install cu npm node-gyp hostapd isc-dhcp-server sysv-rc-conf x11vnc
```

cu	シリアルコンソール用アプリケーション。LTE の SIM カードを設定するのに使用。
npm	Node-RED のパッケージ管理用ソフトウェア
node-gyp	Node-RED のパッケージをビルドするためのソフトウェア
hostapd	Wi-Fi デザリング用アプリケーション
isc-dhcp-server	Wi-Fi デザリング用アプリケーション
sysv-rc-conf	Wi-Fi デザリングのサービス自動起動管理アプリケーション

x11vnc	
--------	--

2) ソースファイルを取得

github からプロジェクトをダウンロードします。Raspberry Pi が起動したら次のフォルダを作成し、「LapisIoTGateway」のレポジトリをダウンロードしてください。

```
$ mkdir ~/docomo
$ cd docomo
$ git clone git://github.com/LapisIoTGateway/LapisIoTGateway lpwagw
$ cd lpwagw
```

● LTE の設定

1) ペリフェラルの初期化

次のコマンドを実行して下さい。

```
$ cd ~/docomo/lpwagw/Gateway/
$ sudo ./gw_peri &
```

2) LTE モジュールの USB ドライバをロード

“usbserial”の USB ドライバを一旦無効にし、LTE モジュール MM-M510 に usbserial を割り付けます。

```
$ sudo rmmod ftdi_sio
$ sudo rmmod usbserial
$ sudo modprobe usbserial vendor=0x2a9e product=0x0103
```

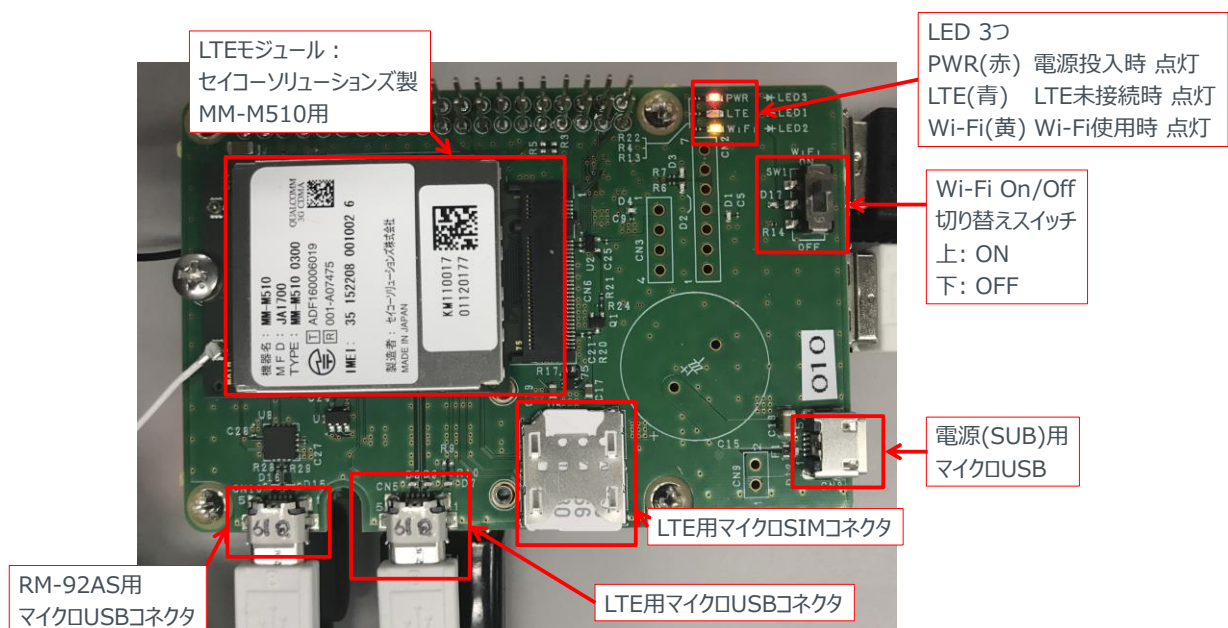
[確認]

```
$ ls /dev/ttyUSB*
```

を実行して、USB0～2 が見えていれば OK です。

ftdi_sio や usbserial の USB ドライバを無効にすることが出来たことを、“lsmod”を実行し、ftdi_sio や usbserial のドライバが無いことを確認してください。lsmodを実行して使用しているドライバがある場合はrmmodによりドライバを無効にすることが出来ません。

無効にすることが出来ない場合は下図の「LTE 用マイクロ USB コネクタ」以外の USB を一旦外して、再起動をした後、「\$ sudo modprobe usbserial vendor=0x2a9e product=0x0103」を実行してください。



(3) LTE モジュールへの SIM 設定

次のコマンドを実行してシリアルターミナルを開きます。

```
cu -l ttyUSB1
```

(4) AT コマンドによる初期設定

以下は、IIJMIO を使用する場合の例です。

```
=====
AT+IFC=0,0
OK
AT+CGDCONT=2,"IP","iijmio.jp"
OK
AT*PDPP=2,2,"iij","mio@iij"
OK
AT*DPATH_AUTO=1,2
OK
=====
```

ターミナルを抜けるためには終了をするためには、~.(チルダ、ピリオド)を押します。

ドコモの SIM は以下のコマンドで設定可能です。

```
=====
AT+IFC=0,0
OK
AT+CGDCONT=2,"IP","mopera.net"
OK
AT*PDPP=2,2," "," "
OK
AT*DPATH_AUTO=1,2
OK
=====
```

ダブルクォーテーションの間にスペースが必要です。

● Wi-Fi デザリングの設定

1) Wi-Fi の IP アドレスを設定

/etc/dhcpd.conf を編集

最後に以下の 2 行を追加

```
-----
interface wlan0
static ip_address=192.168.42.1/24
-----
```

2) DHCP サーバーの設定

/etc/dhcp/dhcpd.conf を編集

2-1) 13 行目と 14 行目をコメントアウト

----- 変更前 -----

```
option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;
```

----- 変更後 -----

```
# option domain-name "example.org";
# option domain-name-servers ns1.example.org, ns2.example.org;
```

2-2) 最後に追記

W-Fi の IP アドレスは、「192.168.42.1」に設定されます。必要に応じて変更してください。

---- ここから ----

```
subnet 192.168.42.0 netmask 255.255.255.0 {
    range 192.168.42.10 192.168.42.50;
    option broadcast-address 192.168.42.255;
    option routers 192.168.42.1;
    default-lease-time 600;
    max-lease-time 7200;
    option domain-name "local";
    option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

---- ここまで ----

3) hostapd の設定

/etc/hostapd/hostapd.conf を新規作成し、以下の内容を貼り付けます

```
---- ここから ----
interface=wlan0
driver=nl80211
ssid=dcm_lpwagw_001
hw_mode=g
channel=6
ieee80211n=1
wmm_enabled=1
ht_capab=[HT40][SHORT-GI-20][DSSS_CCK-40]
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_passphrase=dcm201612
rsn_pairwise=CCMP
---- ここまで ----
```

[補足]

- channel は任意で変更してください。
- Wi-Fi ルーターとしての名前は ssid で指定した値になります。
- Wi-Fi の passphrase は、wpa_passphrase で指定した値になります。

4) hostapd デモン化

/etc/default/hostapd を編集

DAEMON_CONF に先ほどのファイルを指定します。

---- 変更後 ----

DAEMON_CONF="/etc/hostapd/hostapd.conf"

5) iscp-dhcp-server のインタフェースを設定

/etc/default/isc-dhcp-server を編集

INTERFACE に wlan0 を指定

---- 変更後 ----

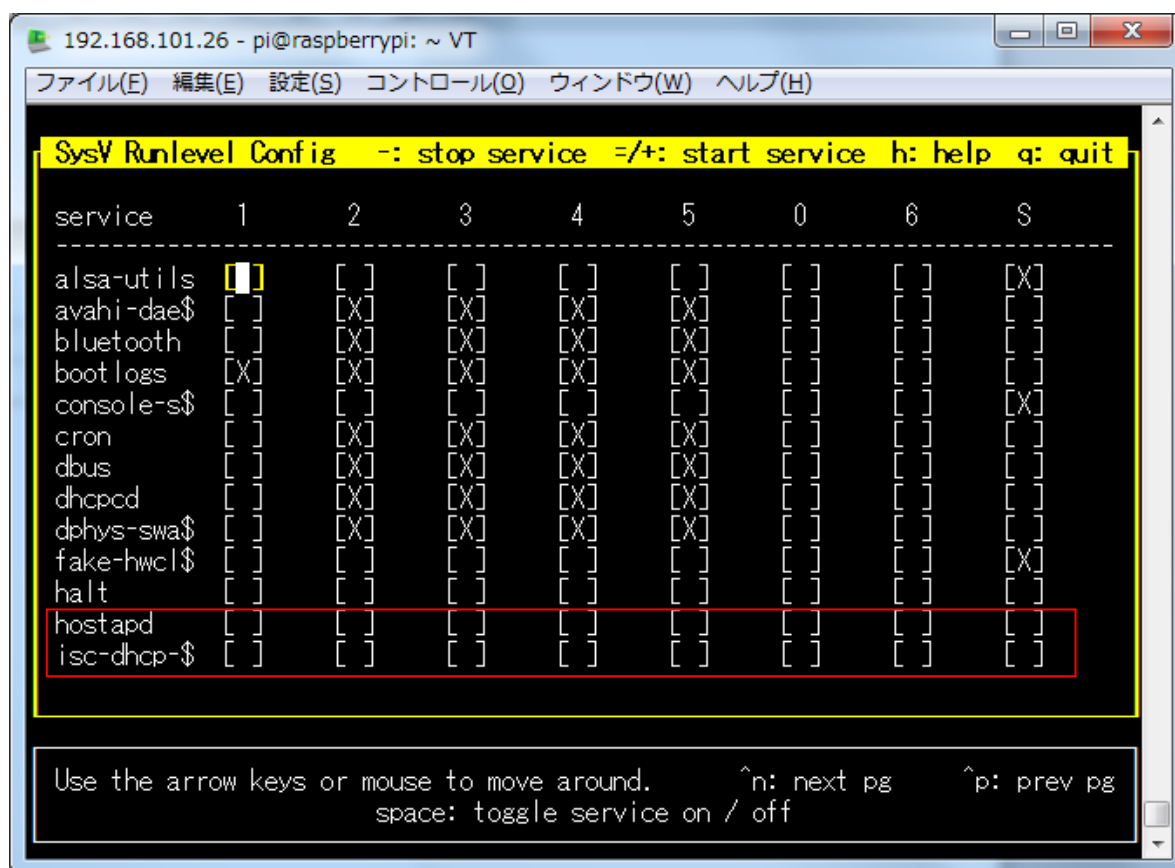
INTERFACE="wlan0"

6) サービスの自動起動を無効化

次のコマンドで hostapd と isc-dhcp-service の自動起動を停止します。

\$ sudo sysv-rc-conf

sysv-rc-conf での設定完了画面は以下の通りです。



7) Wi-Fi 自動接続のサービスを停止する

/etc/network/interfaces を編集

```
iface wlan0 inet manual
#    wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
... # ↑ コメントアウト
```

● 使用するフォルダを作成

- 1) ゲートウェイ用設定を格納するためのフォルダを作成

```
$ mkdir ~/gateway
```

- 2) Node-RED 用フォルダの生成

Node-RED が使用するフォルダを生成

メニューから Node-RED を起動

● Node-RED の外部パッケージをインストール

```
$ cd ~/.node-red
```

```
$ sudo npm install serialport@1.7.4 es6-promise javascript-state-machine
```

● リモートデスクトップ環境の設定

```
$ x11vnc -storepasswd
```

パスワードを入力します。

初期値: dcm20161

● モニター設定

Raspberry Pi の画面サイズを固定します。

/boot/config.txt を編集します。

- 1) 21、22 行目のフレームバッファサイズを変更

```
framebuffer_width=1280
```

```
framebuffer_height=1024
```

- 2) 25 行目の HDMI 出力設定を HDMI 固定に変更

```
hdmi_force_hotplug=1
```

- 3) 28、29 行目の画面のサイズ 1280x1024(60Hz)に固定

```
hdmi_group=2
```

```
hdmi_mode=35
```

● データ保存用 USB メモリの設定

- ・ exFAT 用ライブラリのインストール

```
$ sudo apt-get install exfat-utils exfat-fuse
```

● 自動更新スクリプトのコピーと実行

```
$ cd
```

```
$ cp docomo/lpwagw/Gateway/memo/update.sh .
```

```
$ sudo ./update.sh
```

実行すると、システムを自動で再起動します。

■ 10. 改版履歴

ドキュメント No.	発行日	変更内容
000000	2017.3.17	暫定 0.0.0 版
000001	2017.3.	暫定 0.0.1 版
000002	2017.3.22	<ul style="list-style-type: none">・EnOcean の RSSI 対応・USB メモリにデータを保存する方法を追記 「9. USB メモリにログを保存する方法」 「9. データ保存用 USB メモリの設定」・log file のノード追加に伴い、「5.- log file」を追記・RTK を送信しないフローを追加 「3.Node-RED のフロー図(RTK を発信しない場合)」・「9.- プログラムの保存場所」を追記・「9.- プログラムの更新方法」を追記・「6. プログラムの保存場所」を追記・「9. LTE の設定」に、LTE モジュールが認識されない場合の対処方法を追記
010000	2017.3.23	初版発行(1.0.0 版)
010001	2017.3.27	「6.センサーノードの使用方法」-「センサノードのソフトウェア開発」を追記しました。

Copyright 2017 LAPIS Semiconductor Co., Ltd.