

Raspberry Pi 用 IM 無線 HAT
IM920-HAT 用 Node-RED
取扱説明書

インタープラン株式会社

〒102-0072 東京都千代田区飯田橋 3-3-12 石原ビル 5F

TEL : 03-5215-5771 FAX : 03-5215-5772 URL : <http://www.interplan.co.jp>

目 次

1. インストール方法.....	3
2. 必要なライブラリ.....	4
3. 使用方法.....	4
3-1. HAT-Read.....	4
3-2. HAT-Write.....	4
4. サンプルフロー.....	5
4-1. サンプルフローの使い方.....	5
4-2. 各フローの説明.....	7
5. 改訂履歴.....	7

interplan

この取扱説明書は、Raspberry Pi 用 IM 無線 HAT（IM920-HAT）に使用できる Node-RED の使用方法を説明しています。Raspberry Pi と組み合わせた無線通信を簡単に実現できますので、多彩なアプリケーションにご活用いただけます。

ご注意

- ・本ソフトウェアは無保証のデモ用ソフトウェアです。
- ・本ソフトウェアの不具合により損害が発生した場合でも補償は致しません。
- ・本ソフトウェアに関するサポートは行っておりません。

1. インストール方法

IM920-HAT を Node-RED で動作させるために必要なノード `node-red-contrib-im920-hat` をインストールします。まず、`im_node.tar.gz` をダウンロードし解凍します。

その後以下の手順でインストールします。

```
$ cd im_node/node-red-contrib-im920-hat
$ sudo npm link
```

インストールが完了すると、パレットに「IM920 HAT」が追加され、その中に「HAT-Read」と「HAT-Write」ノードがあります。



2. 必要なパッケージ

本ノードを使用するには、i2c-bus モジュール¹が必要になります。

以下を Node-RED のユーザデータディレクトリ内で行ってください。

```
$ cd .node-red
$ npm install i2c-bus
```

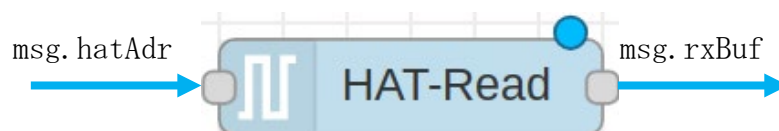
3. 使用方法

各ノードの使い方について説明します。

3-1. HAT-Read

本ノードは受信データを読み込むノードになります。入力として、「msg.hatAdr」に I2C アドレス設定ピンによって設定した I2C アドレスを入力してください。

受信データはノードより出力される「msg.rxBuf」に書き込まれています。



3-2. HAT-Write

本ノードは無線モジュールにコマンドを送信するノードになります。入力引数として、「msg.hatAdr」と「msg.txBuf」があります。「msg.hatAdr」には HAT-Read 同様 I2C アドレスを入力、「msg.txBuf」には string 型か buffer 型でコマンドを入力してください。



¹ i2c-bus : <https://github.com/fivdi/i2c-bus>

4. サンプルフロー

サンプルフローでは受信データの表示とデータ送信が行えます。

4-1. サンプルフローの使い方

初めに ipcflow.json をダウンロードします。

次に Node-RED のメニューオプションの「読み込み」を選択し、「読み込むファイルを選択してください」から先程ダウンロードしたファイルを読み込んでください。



フローをクリップボードから読み込み

クリップボード

JSON形式のフローデータを貼り付けてください

読み込むファイルを選択してください

ライブラリ

サンプル

```
[
  {
    "id": "1aalc3cb.1dd39c",
    "type": "tab",
    "label": "送受信フロー",
    "disabled": false,
    "info": ""
  },
  {
    "id": "345dle45.036a32",
    "type": "function",
    "z": "1aalc3cb.1dd39c",
    "name": "Busy設定",
    "func": "sts = msg.payload;\nif (sts === 1) {\n  msg.busy = 'busy';\n} else {\n  msg.busy = 'free';\n}\n\nglobal.set('IPCLibs.busy_sts', sts);\n\nreturn msg;",
    "outputs": 1,
    "wires": []
  }
]
```

読み込み先

現在のタブ

新規のタブ

中止

読み込み

interplan

「関数定義」ノードをダブルクリックし、コード内の HAT_ADR に IM920-HAT の I2C アドレスを入力し、完了を押した後にデプロイを行ってください。



デプロイが完了するとデバックコンソールに「設定完了」と表示されます。これでサンプルフローの準備は完了です。

サンプルでは以下の動作ができます。

- ・受信データ(レスポンスを含む)が存在する場合、デバックコンソールに表示されます。
- ・タイムスタンプノードの左側にあるボタンを押すと TXDA コマンドと送信回数が送信されます。送信内容はデバックコンソールに表示されます。

4-2. 各フローの説明

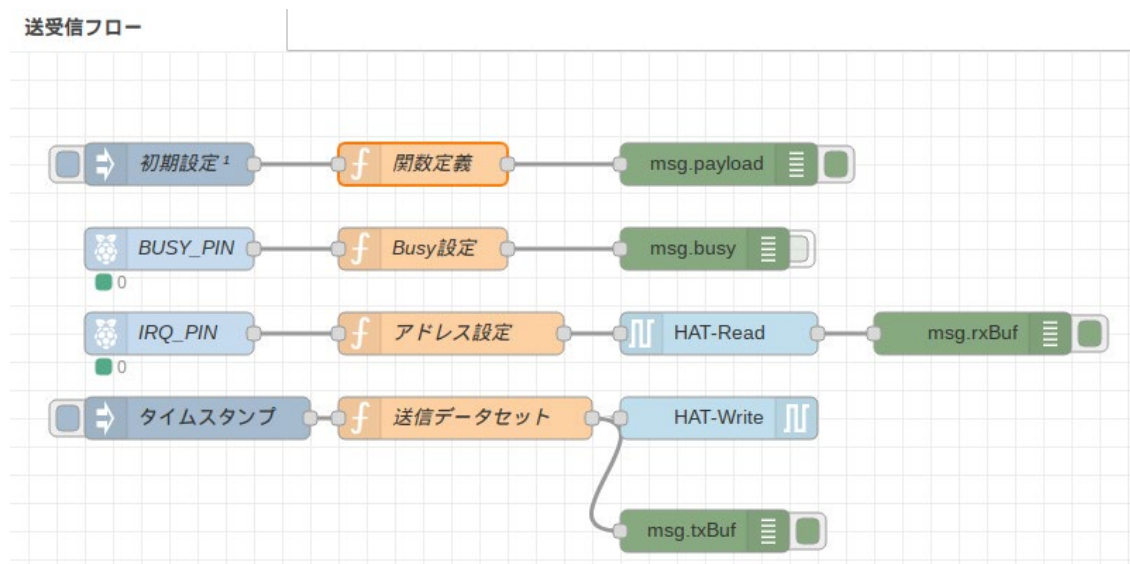
「初期設定」から始まるフローでは、msg.hadAdr で必要となる I2C アドレスと本サンプルで使用する送信データを作成するための関数(itoH)を定義しています。このフローは node-red 起動時にのみ実行されます。

次に「BUSY_PIN」で始まるフローでは、無線モジュールの Busy ピンの Hi/Lo を監視しています。「Busy 設定ノード」でピン状態をグローバル変数に書き込んでいます。

「IRQ_PIN」で始まるフローでは無線モジュールの受信データを読み込みます。アドレス設定ノードでは IRQ_PIN が Hi になると、HAT-Read ノードに msg.hadAdr を入力します。

「HAT-Read」では受信データが msg.rxBuf に保存されます。

最後に「タイムスタンプ」で始まるフローでは無線モジュールに TXDA コマンドを送信します。



5. 改訂履歴

初版制定 2019 年 11 月 21 日

以上