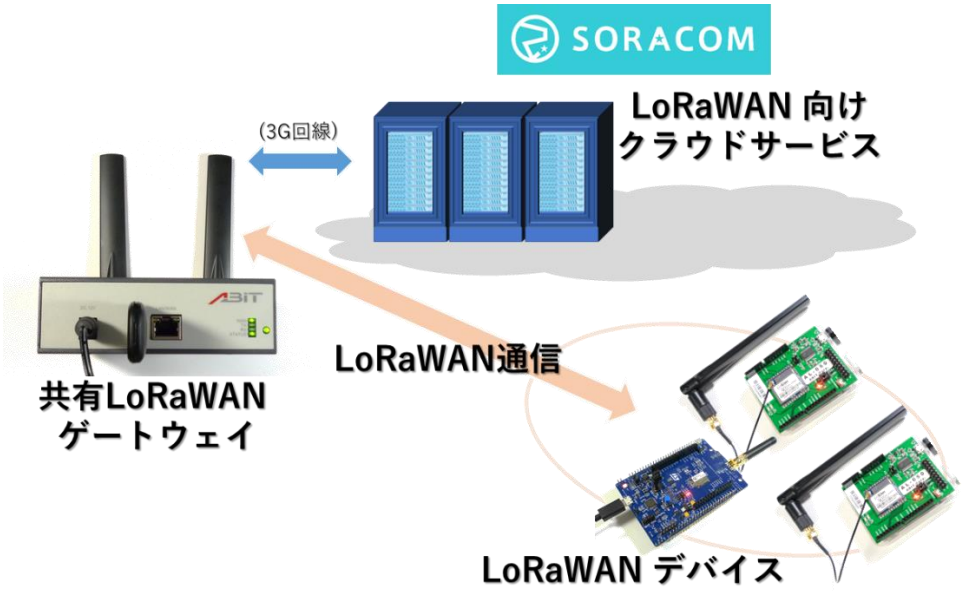
ボクにもわかる Sub-GHz 屋外通信
SORACOM LoRaWAN で簡単 IoT 機器製作



国野　亘

内容

ご注意：

本書に掲載している情報は，執筆時点（2018 年 4 月～5 月）のものです．

ボクにもわかる Sub-GHz 屋外通信
SORACOM LoRaWAN で簡単 IoT 機器製作

1. 手軽に始める SORACOM LoRaWAN

本稿では SORACOM の LoRaWAN 製品を使って，手軽に IoT 機器の製作を行う方法について説明します．他にも LoRaWAN に対応した製品が販売されていますが，実験や試作を行うには，電波法への対応や基地局に相当するゲートウェイ，サーバの設置などに多くの手間や労力を要します．とくに，Sub-GHz（後述）の機器の設定方法や使用方法を誤ると，他の無線設備の利用者や事業者へ被害や損害を与えるリスクや，懲役，罰金が科せられるリスクがあるので，慎重にシステムを構築する必要があります．

SORACOM の LoRaWAN サービス(図 1-1)であれば, ユーザ・コンソール(https://console.soracom.io/)から LoRaWAN デバイス製品（写真 1-1）を購入するだけで，共有ゲートウェイや，同社のサーバを利用することができます．ただし，執筆時点では利用可能な共有ゲートウェイが多くありません．必ず同社のウェブページ（https://lora-space.soracom.jp/map/）でカバーエリアを確認してください．共有ゲートウェイを SORACOM からレンタル（有償）して利用することも可能です．なお，SORACOM 以外で購入した LoRaWAN デバイスを，SORACOM の LoRaWAN サービスへ接続することは出来ません．

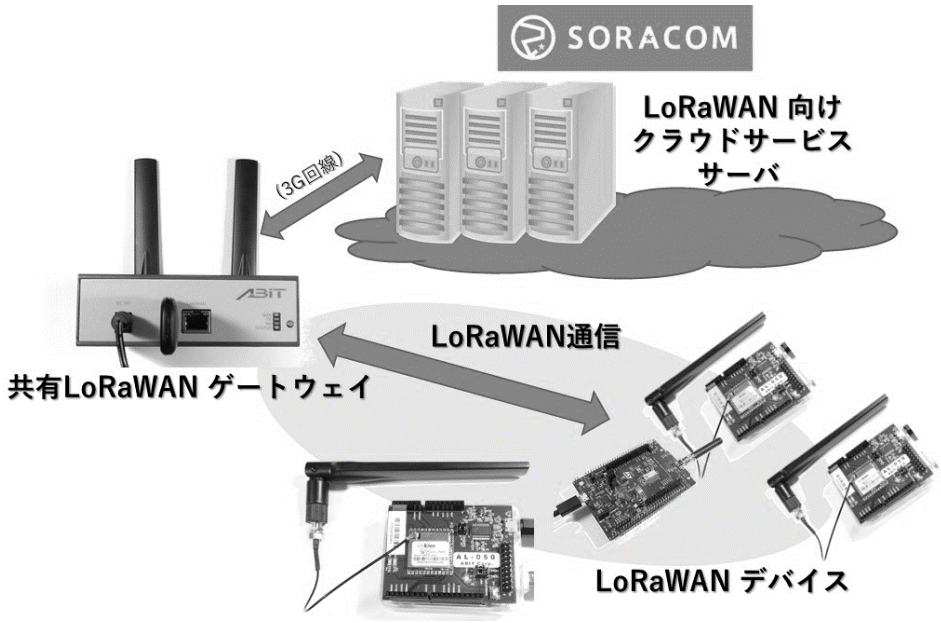


図 1-1 SORACOM LoRaWAN の概略
SORACOM の LoRaWAN デバイスを購入すれば，共有の LoRaWAN ゲートウェイを経由して同社が提供するサーバへ接続できる．

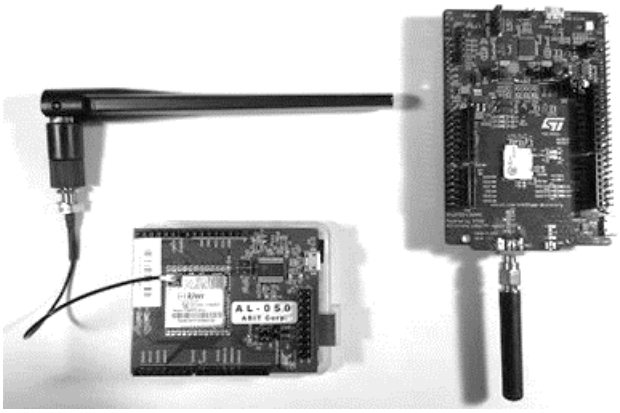


写真 1-1 SORACOM から販売されている LoRaWAN 製品の一例
Arduino へ装着して使用する LoRa Arduino 開発シールド AL-050（左）と単体動作が可能な STM32L0 LoRa Discovery Kit（右）が販売されている．どちらも SORACOM ユーザ・コンソールから購入できる．

2. SIM カード不要で屋外ネットワーク接続

IoT 機器の普及が急速に進みつつありますが，その多くは家電機器や，住設機器，業務用システム機器といった分野に限られています．ここでは，その他の分野への展開例として，歩行時に発電したエネルギーを使ってインターネットへ接続可能な IoT 対応の靴の実現性について，考えてみます．

これまで IoT 機器をインターネット接続するには，屋外では携帯電話事業者などが提供する 4G などのモバイル回線を，家庭内では無線 LAN を使用してきました．4G モバイル回線の場合は，端末ごとに回線契約済みの SIM カードが必要であり，保有している靴の数だけ回線契約を行うか，外出前に SIM カードの差し替えを行うことになります．無線 LAN の場合は，外出すると接続出来なくなる課題があります．また，購入者の自宅まで靴屋さん又はその代行業者が訪れて，無線 LAN アクセスポイントの接続設定をするといったサービスも必要になるでしょう．このように，4G モバイル回線や無線 LAN を使った場合，コストや契約，使い勝手などに見合った IoT 対応の靴サービスの実現性が考え難い状況でした．

LoRaWAN をはじめとする LPWA では，回線に関する契約や SIM カードが不要で，屋外での通信が可能となります．このため，靴メーカが出荷時に LPWA 接続の設定をしておけば，購入した靴を履いて歩行するだけで，靴から得られた情報をインターネットへ送信することができるようになります．例えば，日々の歩行数や，使用者のおよその歩行エリア，現在地などの情報が利用できるでしょう．

ただし，この段階では，収集・蓄積したデータを統計的に利用する程度の活用にとどまります．靴メーカが靴の情報に関する IoT サービスを靴の所有者へ提供し，所有者を特定したサービスへと展開するには，所有者と靴の MAC アドレスとを紐づけする必要があるからです．その方法としては，製造番号などを含めた QR コード（デンソーウェーブの登録商標）を靴箱に同封しておき，購入者が QR コードから靴メーカのウェブサイトへアクセスして，ユーザ登録を行うなどが考えられます．

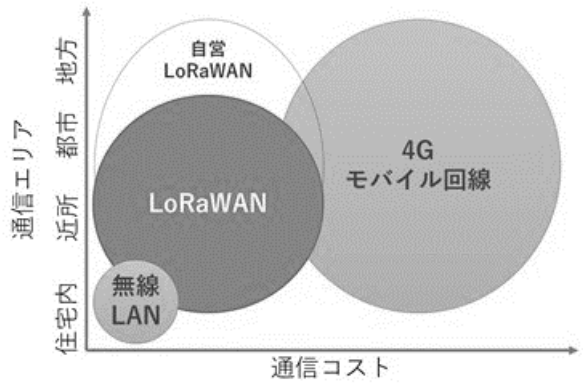


図 2-1 通信コストと通信エリアの予想図
主に住宅内などで使われる無線 LAN は通信コストが低いが，通信エリアが少ない．4G モバイル回線は主に外出先で利用できるが，通信コストが高い．LoRaWAN は低い通信コストで幅広い通信エリアが確保できると言われている．

3. Sub-GHz 無線と低速通信で最大 10km をカバー

LoRaWAN では，Sub-GHz 帯の利用と，LoRa 変調と呼ばれる通信方式による低い伝送速度により，最大 10km 程度の長距離のワイヤレス通信を可能にしています．

Sub-GHz 帯とは，1GHz に満たない 900MHz 付近の周波数帯のことで，主にモバイル通信で使用されてきました．自由空間における電波の伝搬距離は，周波数の二乗に反比例するので，周波数が低いほど有利になりますが，アンテナの大きさも周波数に反比例するので，周波数が低すぎるとアンテナ性能を維持したまま機器を小型化することが難しくなります．およそクレジットカードの大きさから胸ポケットに収まるくらいの機器で長距離のワイヤレス通信を実現するためには，プラチナバンドとも呼ばれている

約 700～950MHz 付近の周波数帯が向いており，LoRaWAN でも Sub-GHz 帯の 920.6MHz（24ch）～928.0MHz（61ch）の周波数帯（計 38ch）を使用しています．

また，LoRa では，チャープ・スペクトラム拡散を行い，伝送速度を落とすことで，受信感度を高めています．一例として，バンド幅 125kHz の SF10（拡散度 1024cps・伝送速度 976bps）で-131dBm の感度が得られ，送信出力 20mW（13dBm）と低い場合であっても，モバイル通信回線に迫る 144dB ものリンクバジェットを得ることが出来ます．

ただし，国内では ARIB STD-T108 の規定により，923.6MHz（39ch）以上の場合，1 回に送信可能な通信時間は 400ms 以下で，パケットに付与可能な情報量は 11 バイトしかありません．しかも，送信後は送信持続時間の 10 倍以上の通信停止期間が規定されており，実際のデータレートとしては最大 20bps しか得られなくなります．主にセンサ等の数値情報を送信するのであれば，11 バイトで 2～3 個のセンサ値を 4.4 秒間隔で送ることが出来るので十分でしょう．

例えば，拡散度を下げると，SF9 で 53 バイト，SF8 で 125 バイト，SF7 だと 242 バイトまで送ることが出来るようになります．しかし，拡散度を 1 段階下げるごとに，受信感度が約 3dB ずつ悪化し，到達距離は約 70%（カバーエリアは半分）ずつ狭くなってしまいます．反対に，拡散度を SF10 よりも大きくすると，LoRaWAN のヘッダ情報が 400ms 以内に収まらなくなるので，国内では SF10 が最大値です．

4. LoRa Arduino 開発シールド AL-050 を使うための準備

まずは，Arduino マイコンボードへ装着可能な LoRa Arduino 開発シールド AL-050 を使って，LoRaWAN への接続実験を行います．必要な機器は，開発用の PC と，表 4-1 の Arduino UNO と AL-050，Arduino UNO へプログラムを書き込むための USB ケーブルです．

Arduino 用の開発環境 Arduino IDE は，Arduino のウェブページ（https://www.arduino.cc/）の[SOFTWARE]メニュー内の[DOWNLOAD]のページ（図 4-1）で「Windows Installer」をクリックし，図 4-2 の画面で「JUST DOWNLOAD」をクリックしてダウンロードします．ブラウザ上に，図 4-3 のようなメッセージが表示されたら「実行」を選択してインストールを開始します．インストール時の設定や項目は推奨値のままでかまいません．Arduino 用 USB ドライバも，インストールしておきます．

また，SORACOM 用の LoRaWAN ライブラリと筆者が作成したサンプル・スケッチ（Arduino 言語で書かれたプログラム）を下記からダウンロードしておいてください．

Arduino IDE：
https://www.arduino.cc/en/Main/Software/

ライブラリ・サンプル（GitHub）：
https://github.com/bokunimowakaru/SORACOM-LoRaWAN/archive/master.zip
または https://goo.gl/LhNbFZ

次に Arduino IDE の設定を行います．Arduino IDE を起動し，図 4-4 のようにマイコンボードを[Arduino/Genuino UNO]に設定し，図 4-5 にしたがってダウンロードしたライブラリ・サンプルをインクルードしてください．図 4-6 のように[スケッチ例]内に cqp_ex から始まるサンプル・スケッチが表示されれば設定完了です．

スケッチを Arduino UNO へ書き込むには，Arduino UNO を PC へ USB 接続し，[ツール]メニューの[シリアルポート]で「COM 番号:(Arduino/Genuino Uno)」を選択してから，図 4-7 の右矢印ボタンを

クリックします．もし，コンパイル・エラーや Arduino UNO との接続エラーが発生すると緑色のメッセージ表示欄が橙色に変化し，エラー内容が表示されます．

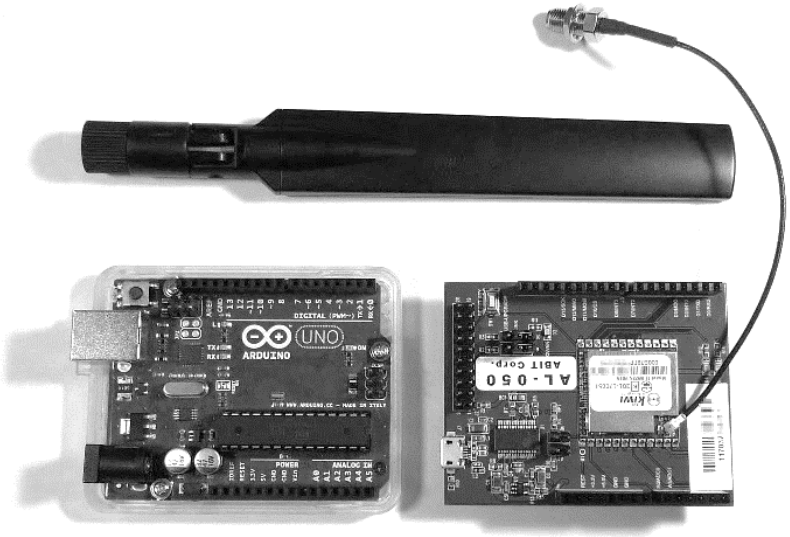


写真 4-1　LoRaWAN の実験に必要な機器

LoRa Arduino 開発シールド AL-050（右下）および動品に付属するアンテナ（上）と，別途，購入が必要な Arduino UNO（左下）．Arduino 側の「ARDUINO」の文字と，シールが側の「AL-50」文字が反対になる方向に装着する．

表 4-1　LoRaWAN の実験に必要な機器

製品名	内容
LoRa Arduino 開発シールド AL-050	Arduino へ装着して使用するシールド．SORACOM ユーザ・コンソール（https://console.soracom.io/）から発注する．
Arduino UNO	AL-050 を制御するためのマイコンボード
USB ケーブル	Arduino をパソコンへ接続するための USB ケーブル（Type A→B）

図 4-1　Arduino IDE のダウンロード画面

「https://www.arduino.cc/en/Main/Software」へアクセスし，「Windows Installer」をクリックしてダウンロードする．

図 4-2　Arduino IDE をダウンロードするときに表示される画面

寄付金額を選択するか，[JUST DOWNLOAD]をクリックしてダウンロードを行う．Arduino 互換機を使用する場合も無料でダウンロード可能であるが，最低額の$3 は払っておきたい．

図 4-3　ブラウザに表示されたダウンロードに関する選択画面の例

「arduino-1.X.X-windows-exe について行う操作を選んでください」などのメッセージが表示されたら「実行」を選択してインストーラを実行する．

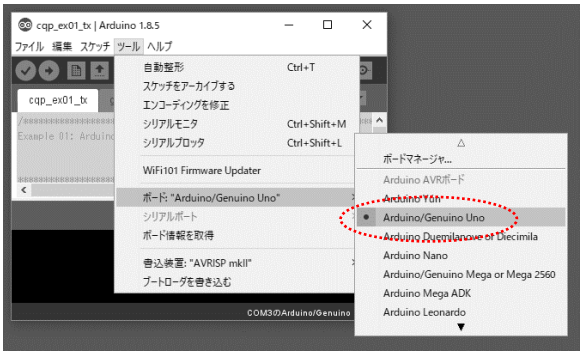


図 4-4　Arduino IDE へ Arduino UNO ボードの設定を行う

Arduino IDE のメニュー[ツール]から[ボード]→[Arduino/Genuino Uno]を選択する.

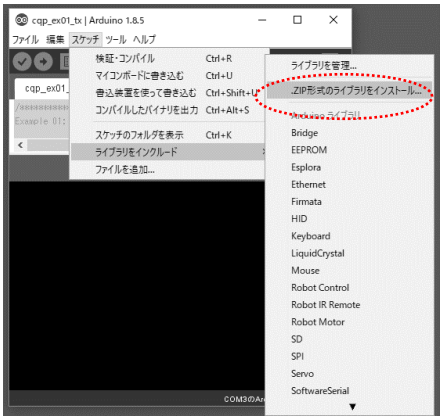


図 4-5　ダウンロードした ZIP 形式のライブラリをインクルードする

Arduino IDE のメニュー[スケッチ]から[ライブラリをインクルード]→[.ZIP 形式のライブラリをインストール]を選択し，ダウンロードした「SORACOM-LoRaWAN-master.zip」を Arduino IDE へ組み込む.

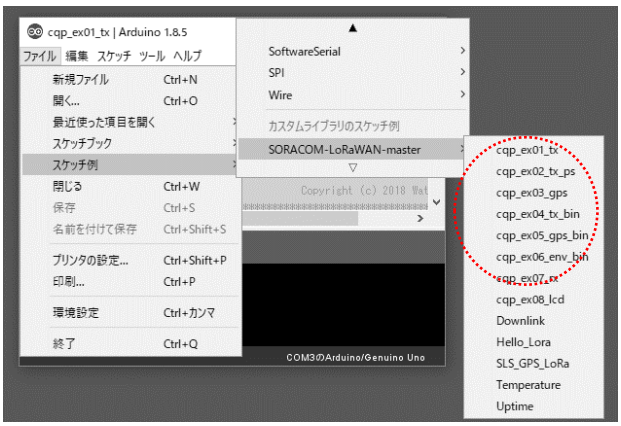


図 4-6　サンプル・スケッチを確認する

Arduino IDE のメニュー[ファイル]から[スケッチ例]→[SORACOM-LoRaWAN-master]を選択すると，「cqp_ex」から始まるサンプル・スケッチが表示される．[cqp_ex_01_tx]を選択してファイルを開く.

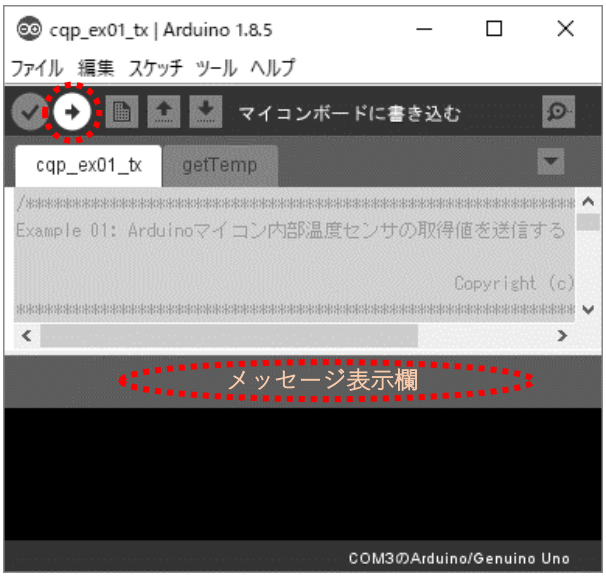


図 4-7　サンプル・スケッチを Arduino UNO へ書き込む

スケッチを Arduino UNO へ書き込むには，[ツール]メニューのシリアル・ポートで「COM 番号:(Arduino/Genuino Uno)」を選択し，Arduino IDE の右矢印ボタンをクリックする.

5. LoRaWAN へ送信したデータをサーバに蓄積する

スケッチ cqp_ex_01_tx を書き込んだ Arduino Uno と AL-050 を使って，LoRaWAN 送信を行い，SORACOM の LoRaWAN 用のサーバへデータを蓄積する実験を行ってみましょう．

SORACOM のサーバ側の設定を行うには，SORACOM ユーザ・コンソールを使用します．Web ブラウザからユーザ・コンソールへログインし，LoRa グループを作成してから，図 5-1 の SORACOM LoRa Space を[利用する]に，SORACOM Harvest 設定を ON に設定し，LoRaWAN デバイスを LoRa グループへ登録してください．詳細な設定方法は，図 5-1 や，SORACOM が提供している LoRaWAN デバイス設定ガイドをご覧ください．

LoRaWAN デバイス設定ガイド(SORACOM 提供)： https://dev.soracom.io/jp/start/lora_uc_device/

LoRa Arduino 開発シールド AL-050 を装着した Arduino UNO を，PC へ接続し，Arduino IDE のシリアル・モニタ（右上の虫眼鏡アイコン）を起動すると，図 5-2 のような Arduino UNO の動作ログが表示されます．LoRaWAN への接続に成功すると，「{“t”:温度値}」のように温度値が表示されるとともに，SORACOM Harvest へデータが送信されます．なお，温度値にはマイコンの内部発熱などが含まれ，また測定精度も良く無いので，温度変化の目安を確認する程度の目的で使用してください．

SORACOM 側が受信したデータの内容を確認するには，SORACOM ユーザ・コンソールの[LoRa デバイス管理]で対象のデバイスにチェックマークを入れてから，[操作]メニューの[データを確認]を選択してください．図 5-3 のように受信データが SORACOM へ蓄積されていることが確認できるでしょう．

なお，SORACOM の LoRaWAN デバイスの月額利用料は発生しませんが，SORACOM のサーバについては利用料が発生します．SORACOM Harvest は 1 日 5 円，受信したデータを他のサーバへ転送する SORACOM Beam は 1 リクエスト 0.0018 円，LoRaWAN デバイスへのダウンリンク通信には 1 リクエスト 0.0054 円などです．回線ごとに固定の月額料金や解約手数料が発生する一般的なモバイル回線と比較して，通信量が少なく，契約・解約件数の多い IoT 機器に適した課金方法となっています．

図 5-1　SORACOM ユーザ・コンソールで LoRa グループの設定を行う

LoRa グループの管理画面で，SORACOM LoRa Space を[利用する]に，SORACOM Harvest 設定を ON に設定する．SORACOM Harvest については利用料（1 日 5 円）が発生するので，実験が終わったら OFF に戻しておく．

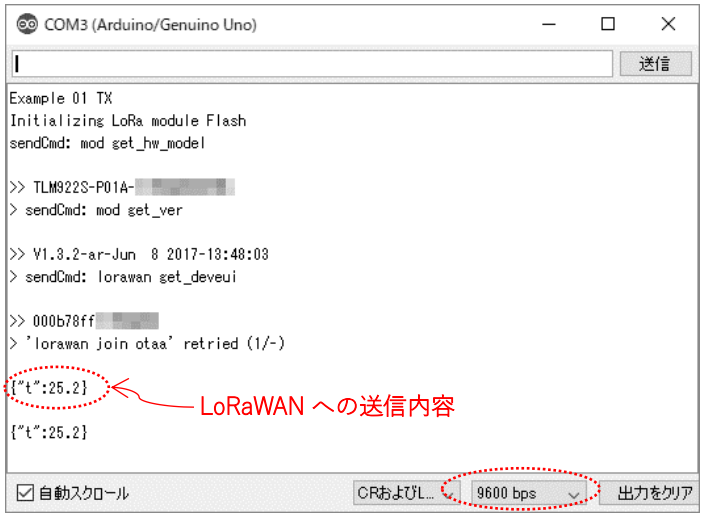


図 5-2 温度データを送信する cqp_ex_01_tx の動作例

シリアル・モニタを起動し，ビットレートを 9600bps に設定すると，動作状態が表示される．LoRaWAN ネットワークへの接続が完了すると，測定した温度値が LoRaWAN へ送信される．

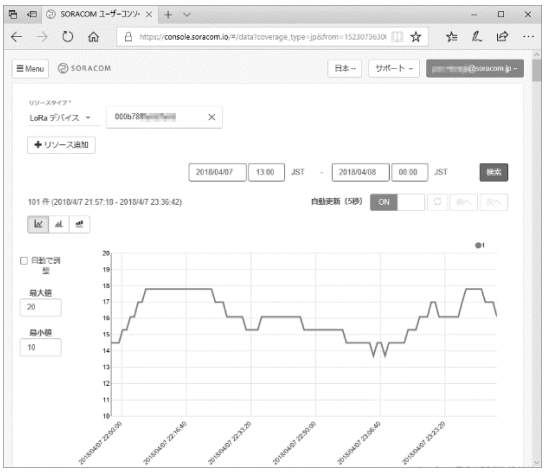


図 5-3 LoRaWAN デバイスの送信結果を SORACOM Harvest で確認する

Arduino から LoRaWAN 送信した温度値が SORACOM Harvest に蓄積された．

6. LoRaWAN 送信用スケッチ（cqp_ex01_tx）

プログラム 6-1 は，LoRaWAN へ温度値を送信するサンプル・スケッチ（Arduino 言語で書いたプログラム）cqp_ex01_tx.ino です．本スケッチ内の主な動作手順①～⑧について以下に説明します．

① #include はライブラリを組み込むための命令です．LoRaWAN 用ライブラリを組み込みます．
② LoRaWAN 用ライブラリ内の関数へアクセスするためのインスタンス変数 client を定義します．
③ Arduino マイコン起動後に一度だけ実行する setup 関数を定義します．
④ LoRaWAN への接続を実行する client.connect 命令を実行します．引数に true を付与することで，LoRaWAN モジュールの設定を初期状態に戻します．
⑤ 継続的に繰り返し処理を行う loop 関数です．
⑥ Arduino マイコン内蔵の温度センサの値を getTemp 命令で取得します．getTemp 命令はファイル名 getTemp.ino に収録されており，タブ「getTemp」をクリックすると表示されます．
⑦ LoRaWAN 送信を行う client.send 命令を実行します．予め送信内容を文字配列変数 data へ代入しておき，引数として渡します．
⑧ delay 命令は，待ち時間処理を行う Arduino 独自の命令です．引数は待ち時間（ms）で，冒頭で定義した 57000（57 秒）を渡します．本ステップを含む手順⑤の loop 関数を約 1 分毎に繰り返し実行します．

プログラム 6-1　LoRaWAN への送信用サンプル・スケッチ cqp_ex01_tx.ino

```
#include <lorawan_client.h>                    // LoRaWAN 用ライブラリの組み込み
#define PIN_LED 13                             // Digital 13 に LED を接続
#define INTERVAL_MS 57000                      // 送信待ち時間(ms)
#define TEMP_OFFSET 0.0                        // 温度補正値（℃）
LoRaWANClient client;                          // LoRaWAN Client を定義

void setup(){                                  // 起動時に一度だけ実行する関数
    pinMode(PIN_LED,OUTPUT);                   // LED を接続したポートを出力に
    digitalWrite(PIN_LED,HIGH);                // LED の点灯
    Serial.begin(9600);                        // 動作確認のためのシリアル出力開始
    Serial.println("Example 01 TX");           // 「Example 01」をシリアル出力表示
    client.connect(true);                      // LoRaWAN のフラッシュメモリの初期化
}

void loop(){                                   // 繰り返し実行する関数
    digitalWrite(PIN_LED,HIGH);                // LoRaWAN モジュールのスリープ解除
    float temp=getTemp()+TEMP_OFFSET;          // 温度を取得し変数 temp へ保存
    char val[6],data[12];                      // 文字配列型変数 val と data を定義
    dtostrf(temp,-1,1,val);                    // 温度値を文字配列型に変換
    snprintf(data,12,"{¥"t¥":%s}",val);        // JSON 形式の送信データに変換
    Serial.println();                          // 改行をシリアル出力表示
    Serial.println(data);                      // 送信データをシリアル出力表示
    client.sendData(data);                     // データを送信(最大 11 文字まで)
    digitalWrite(PIN_LED,LOW);                 // LED の消灯
    delay(INTERVAL_MS);                        // 次回の送信までの待ち時間
}
```

①　②　③　④　⑤　⑥　⑦　⑧

7.　低消費電力動作用スケッチ（cqp_ex02_tx_ps）

従来の長距離通信が可能な通信方式の多くは，そのトレードオフとして消費電力が大きく，乾電池などによる長期間の駆動が難しい傾向がありました．一方，LoRaWAN は，長距離通信の特長を保持しつつ，低消費電力動作が可能です．AL-050 に搭載された LoRaWAN 通信用 IC の送信時の消費電流は 80mA 以下，待機時は 3μA と，乾電池などによる長期間動作が可能な性能を有しています．ただし，AL-050 や Arduino UNO には開発・デバッグ用の USB シリアル変換 IC が実装されており，これらが合計 50mA ほどを消費しているので，省電力性能を発揮させるには，周辺回路の低消費電力化が必要です．また，待機時に LoRaWAN や Arduino マイコンを低消費電力状態へ遷移させるための制御も必要です．

今回はハードウェアについては改造せずに，低消費電力動作に対応したスケッチ cqp_ex02_tx_ps を作成してみました．本スケッチによる低消費動作を実行したときの消費電流（縦軸・50mA/DIV）と経過時間（横軸・1 秒/DIV）のようすを図 7-1 に示します．

待機状態から起動し約 4 秒後に約 0.4 秒の送信を行い，その後，2 度の受信を行い，起動してから約 7 秒で動作を完了しました．また，受信が完了してから，約 1 秒後に待機状態へ遷移しました．待機時の約 50mA の多くは USB シリアル変換 IC や電源などの周辺回路によるものです．仮に，この 50mA を差し引いた場合，マイコン等の制御系の消費電流は約 30mA，送信用の通信部は約 110mA，受信用の通信部は約 30mA，1 分に 1 回の動作での平均電流は約 6mA となり，単 3 アルカリ乾電池 2 本で 2 週間程度の動作期間に相当します．

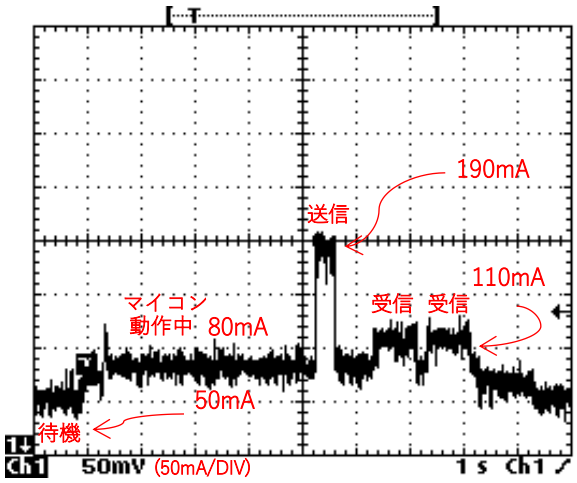


図 7-1　Arduino と AL-050 の低消費動作時の消費電流変化のようす

LoRaWAN 送信したときの消費電流の変化の測定結果．縦軸は電流（50mA/DIV），横軸は経過時間（1 秒/DIV）を表す．

8.　GPS による移動距離測定（cqp_ex03_gps）

GPS を用いることで，LoRaWAN ゲートウェイ（基地局）の伝搬距離を測定することが容易になります．サンプル・スケッチ cqp_ex03_gps は，起動時に GPS で測定した位置情報と，現在地の位置情報から距離を計算し，30 秒ごとに LoRaWAN 送信するプログラム例です．共有ゲートウェイの近くで起動し，ゲートウェイから遠ざかる方向へ移動すると，何 m，離れたかを確認することが出来ます．

ハードウェアは，Arduino UNO および LoRaWAN 開発シールド AL-050 に u-blox 製 GPS モジュール NEO-6M-0-001 を接続して製作しました．図 8-1 のように，GPS モジュールの TXD を Arduino の Digital 8 番ピンへ接続し，電源を供給します．NEO-6M-0-001 の動作電圧は 3.3V ですが，GPS モジュール基板上に電源レギュレータが実装されているので 5V を接続しました．

写真 8-1 は，Arduino シールド用の基板へ GPS モジュールと USB シリアル変換モジュールを実装したときの製作例です．USB シリアル変換モジュールは，u-blox 製の PC 用ソフトウェア u-center を使用するために実装しましたが，本稿の実験を行うだけであれば不要です．

GPS モジュール起動後，位置情報が得られるようになるまで 30 秒以上の時間を要します．また，精度が得られるまでにも時間がかかる場合があります．起点位置の精度が不十分だった場合は，数分の時間を待ってから，Arduino のリセット・ボタンを押下して起点位置を更新してください．

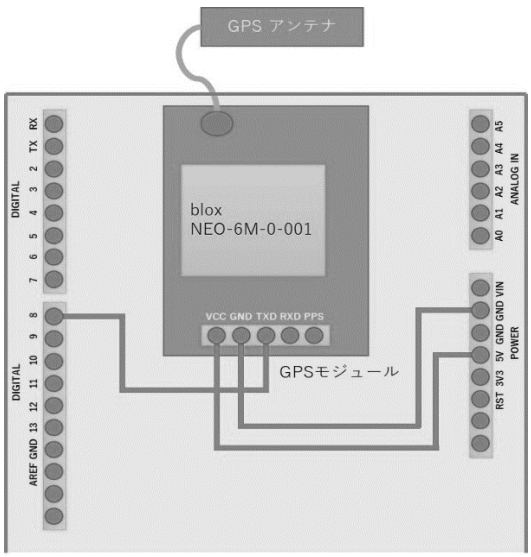


図 8-1　　GPS モジュールの接続図

Arduino の Digital 8 番ピンを GPS モジュール u-blox NEO-6M-0-001 の TXD へ接続し，電源 5V を VCC へ接続する．

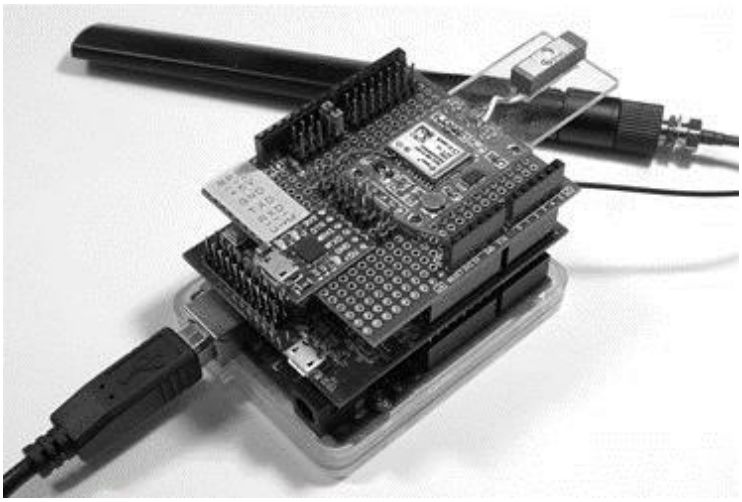


写真 8-1　GPS モジュールを実装した LoRaWAN デバイスの製作例

Arduino UNO および LoRaWAN 開発シールド AL-050 に GPS モジュール u-blox NEO-6M-0-001 を実装した実験のようす.

9. 複数のセンサ値を送信する（cqp_ex04_tx_bin）

これまでの 3 例のサンプル・スケッチは，JSON 形式で送信していましたので，2 値以上のデータを LoRaWAN の 11 バイト以内に収めることが出来ませんでした．ここでは，バイナリ送信を行うことで，3 値のデータを送信する方法について説明します．ハードウェアは Arduino UNO へ LoRa Arduino 開発シールド AL-050 を装着したものを使用します.

プログラム 9-1 では，温度値 4 バイトと，マイコンの電源電圧値 2 バイト，マイコンの動作時間 4 バイトの計 10 バイトのデータをバイナリデータとして送信します．LoRaWAN 用バイナリ送信の処理部分について，以下に説明します.

① struct は複数の変数で構成される構造体の型を定義する命令です．ここでは手順②に示す 3 つの変数で構成する LoRaWAN 送信用のデータ列 LoRaPayload 型を定義します.
② 温度値を保持するための float 型の変数 temp，電源電圧値用の uint16_t 型の変数 vcc，動作時間用の uint32_t 型の変数 time を定義します．（uint16_t 型は符号なし short 型を，uint32_t 型は符号なし long 型を示す.）
③ 手順①で定義した LoRaPayload 型の構造体変数 payload を定義します．以降，payload.temp や payload.vcc，payload.time と記述することで，各変数にアクセスすることが出来ます.
④ 各変数へそれぞれのデータを代入します.
⑤ 構造体変数 payload の内容を LoRaWAN で送信します.

バイナリデータを SORACOM Harvest などのサービスへ連携するには，SORACOM ユーザ・コンソールの LoRa グループの管理画面の[バイナリパーサー設定]の項目へ，バイナリデータの構成を記述する必要があります．フォーマット欄へ以下のような「変数名：：変数型：ビット数：バイトオーダー：演算」の形式で入力します（改行はスペースにして入力）.

```
temp::float:32:little-endian
vcc::uint:16:little-endian:/1000
time::uint:32:little-endian:/1000
```

温度値用の変数名 temp は，float 型（32 ビット）です．また，Arduino マイコンのバイトオーダーはリトルエンディアンなので，little-endian を設定します．また，変数 vcc と time の末尾の「/1000」はデ

ータを 1000 で除算することを示しています．Arduino 側より単位 mV（ミリボルト），ms（ミリ秒）で送信したデータを，V（ボルト），秒として取り扱うために 1000 で除算しました．

プログラム 9-1　LoRaWAN へのバイナリ送信用サンプル・スケッチ cqp_ex04_bin.ino

```
#include <lorawan_client.h>                    // LoRaWAN 用ライブラリの組み込み
#define PIN_LED 13                             // Digital 13 に LED を接続
#define PIN_SLP 7                              // スリープ解除ピン(Digital 7 固定)
#define INTERVAL_MS 51000                      // 送信待ち時間(ms)
#define INTERVAL_WDT 1000                      // スリープ間隔(ms)
#define TEMP_OFFSET 0.0                        // 温度補正値（℃）
#define SLEEP_MODE 1                           // 0:ノーマルスリープ，1:ディープ
LoRaWANClient client;                          // LoRaWAN Client を定義
volatile long wdt_counter = 0;                 // 残スリープ時間(ms)
uint32_t time_prev;                            // 前回の時刻

①
struct LoRaPayload{                            // LoRaWAN 送信用の変数(11 バイト以下)
  float temp;                                  // 温度(float 型 4 バイト)
  uint16_t vcc;                                // 電圧温度(uint16 型 2 バイト)
  uint32_t time;                               // 動作時間(uint32 型 4 バイト)
} payload;
②
③

void setup(){                                  // 起動時に一度だけ実行する関数
                              ～～GPIO 設定部（省略）～～
    time_prev=millis();                        // 起動したときの時刻を記録
    client.connect(true);                      // LoRaWAN のフラッシュメモリの初期化
}

void loop(){                                   // 繰り返し実行する関数
    if(wdt_counter > 0){                       // 残スリープ時間があるとき
        Serial.print(".");                     // 「.」をシリアル出力表示
        delay(3);                              // シリアル出力の完了待ち
        wdt_counter-=INTERVAL_WDT;             // 残スリープ時間を減算
        digitalWrite(PIN_LED,LOW);             // LED の消灯
        wdtSleep(INTERVAL_WDT);                // スリープの実行
        digitalWrite(PIN_LED,HIGH);            // LED の点灯
        return;                                // 繰り返し処理に戻る
    }
    Serial.println();                          // 改行をシリアル出力表示
    digitalWrite(PIN_SLP,HIGH);                // LoRaWAN モジュールのスリープ解除
    delay(2200 * SLEEP_MODE);                  // ディープスリープ復帰時間待ち
④  payload.temp = getTemp()+TEMP_OFFSET;       // 温度を取得
    payload.vcc = (uint16_t)(getVcc()*1000);   // 電圧を取得し，mV へ変換
    payload.time = millis()-time_prev;         // 動作時間を取得し，前回値を減算
    time_prev += payload.time;                 // 今回の CPU 時間を保存
    if( client.connect(false) ){               // 再接続
                    ～～シリアル・モニタ用表示出力部（省略）～～
⑤      client.sendBinary((byte *)&payload,sizeof(payload));    // データ送信
        delay(82);                             // シリアル出力の完了待ち
    }
    wdt_counter=INTERVAL_MS;                   // 残スリープ時間のリセット
    digitalWrite(PIN_SLP,LOW);                 // スリープ解除ピンのリセット
    delay(1);                                  // スリープ信号出力の遷移待ち
    if(SLEEP_MODE) client.deep_sleep();        // ディープスリープの実行
    else client.sleep();                       // ノーマルスリープの実行
}
```

図 9-1　LoRa グループの管理画面内のバイナリ・パーサ設定の一例

温度値データ用の float 型の変数 temp, 電源電圧値用の uint16_t 型の変数 vcc, 動作時間用の uint32_t 型の変数 time を，バイナリ・パーサへ設定した.

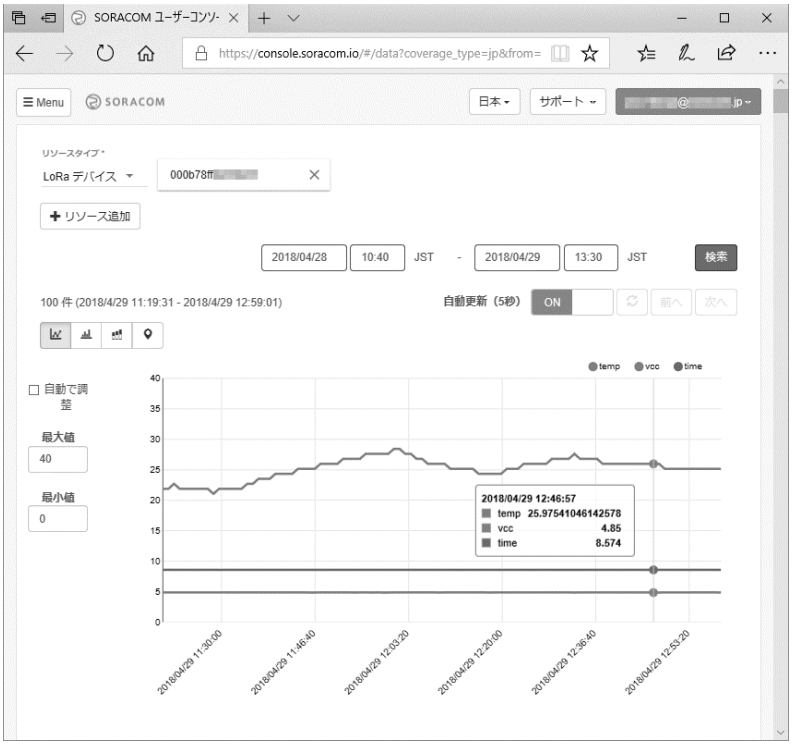


図 9-2　LoRaWAN へバイナリデータを送信し，パーサーによって 3 値を復元したときの受信結果の一例

温度値データ temp, 電源電圧値 vcc, 動作時間 time の 3 値を SORACOM Harvest へ蓄積した.

10. 各種サンプル・スケッチの使い方

本稿用に製作したサンプル・スケッチ 8 本を表 10-1 に示します．本節までに，cqp_ex01～04 について説明しましたので，ここでは cqp_ex05～08 について，紹介します.

表 10-1　サンプル・スケッチの内容と使い方（https://goo.gl/LhNbFZ）

スケッチ名	内容	説明
cqp_ex01_tx	Arduino マイコン内温度センサの取得値を送信	本稿内で説明済
cqp_ex02_tx_ps	ディープスリープモードを使用した低消費動作	本稿内で説明済
cqp_ex03_gps	GPS による移動距離の送信	本稿内で説明済・要 NEO-6M-0-001
cqp_ex04_bin	温度，電圧，時間の 3 値をバイナリデータ送信	本稿内で説明済
cqp_ex05_gps_bin	GPS から得た緯度，経度，標高をバイナリ送信	cqp_ex03_gps と同様の構成. ソース内にバイナリ・パーサ設定例を記載.
cqp_ex06_env_bin	温度，湿度，気圧センサ値をバイナリ送信	要 BME280，ソース内に I2C アドレス設定方法，バイナリ・パーサ設定例を記載.
cqp_ex07_rx	ダウンリンク受信データをシリアルへ出力	受信した 16 進数のデータを出力.
cqp_ex08_lcd	ダウンリンク受信した文字列を LCD へ表示	要 I^2C 接続 LCD．ASCII データを表示.

GPS 位置情報を送信する（cqp_ex05_gps_bin）

サンプル・スケッチ cqp_ex05_gps_bin は，GPS の位置情報を LoRaWAN 送信するプログラムです．ハードウェアは cqp_ex03_gps と同じ構成です．GPS モジュール NEO-6M-0-001 の TXD を Arduino の Digital 8 端子へ入力してください．スケッチを実行すると，GPS から得られた緯度，経度，標高をバイナリで LoRaWAN 送信します. データの先頭には, SORACOM 社のサンプル SLS_GPS_LoRa と同様に，0x21 の識別子を含めました．バイナリ・パーサの設定例はソースへ記載しました．

温度・湿度・気圧センサ（cqp_ex06_env_bin）

サンプル・スケッチ cqp_ex06_env_bin は，温度，湿度，気圧の測定が可能な Bosch 製 BME280 を Arduino の I2C 端子へ接続することで，各センサ値を LoRaWAN へ送信する IoT 環境センサ用のプログラムです．写真 10-1 の製作例では，図 10-2 の配線図のように，BME280 の SDA を Arduino UNO の Analog 4 へ，SCL を Analog 5 へ接続し，3.3V の電源を VIN へ供給しました．送信時の Arduino 側のようすを図 10-3 に，SORACOM Harvest へデータを蓄積したときのようすを図 10-4 に示します．

メッセージを受信する（cqp_ex07_rx と cqp_ex08_lcd）

サンプル・スケッチ cqp_ex07_rx と cqp_ex08_lcd は，LoRaWAN からのメッセージを受信するプログラム例です．SORACOM から送信を行うには，SORACOM コンソールを起動し，[LoRa デバイス管理]から対象デバイスを選択し, [操作]メニューから[ダウンリンク通信]を選び, 16 進数の送信データと fPort 番号「2」を入力してから[送信]ボタンを押します．例えば，図 10-5 のように，送信データに ASCII コードで「48656c6c6f21」を入力すると，「Hello!」のメッセージを LoRaWAN デバイスへ送ることが出来ます．

cqp_ex07_rx は，受信した 16 進数のデータをシリアル・モニタへ出力し，cqp_ex08_lcd は文字列を LCD へ出力します. 製作した IoT 表示器の製作例を写真 10-2 に, LCD の接続例を図 10-6 に示します．なお，AL-050 動作中は 3.3V の電源電圧が高まり，LCD の表示が濃くなる場合があります．気になる場合は，5V 出力に 3.3V の電源 LDO を接続し，安定化した電圧を LCD へ供給してください．

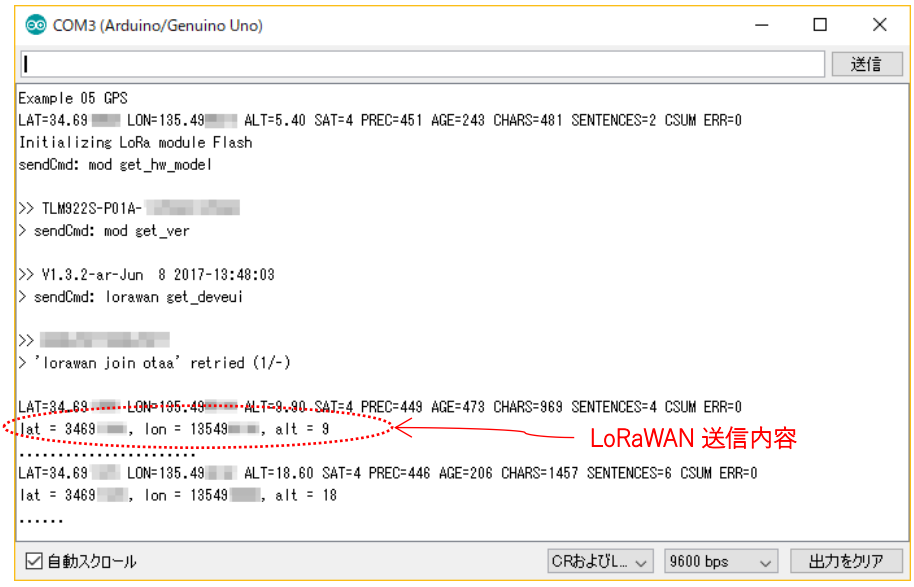


図 10-1　GPS 位置情報を送信するサンプル cqp_ex05_gps の実行例

GPS モジュール NEO-6M-0-001 から得られた緯度（LAT）・経度（LON）・標高（ALT）の計 3 値のデータを LoRaWAN へ送信するときのようす．

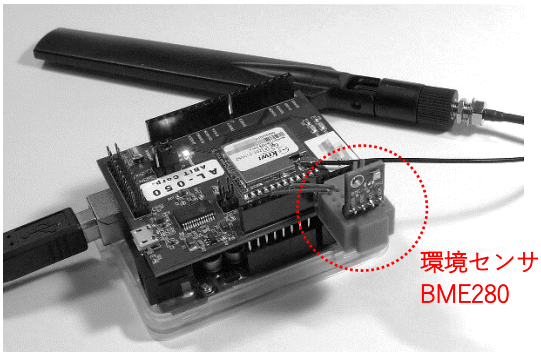


写真 10-1　BME280 を接続した IoT センサの製作例

Arduino UNO へ LoRa Arduino 開発シールド AL-050 を装着し，環境センサ BME280 を接続した．

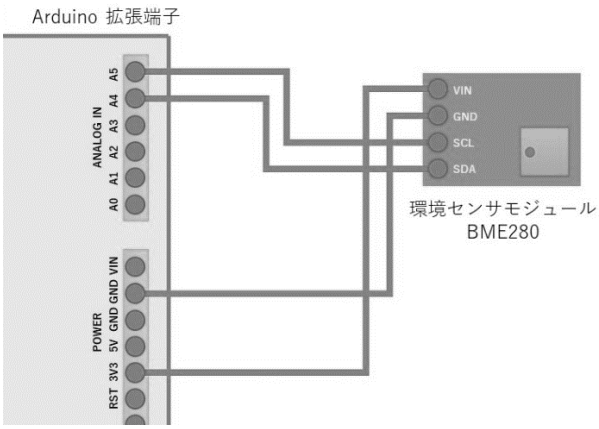


図 10-2　環境センサ・モジュールの接続図

環境センサ Bosch 製 BME280 の SDA を Arduino UNO の Analog 4 へ，SCL を Analog 5 へ接続し，また Arduino の 3.3V の電源を供給する．

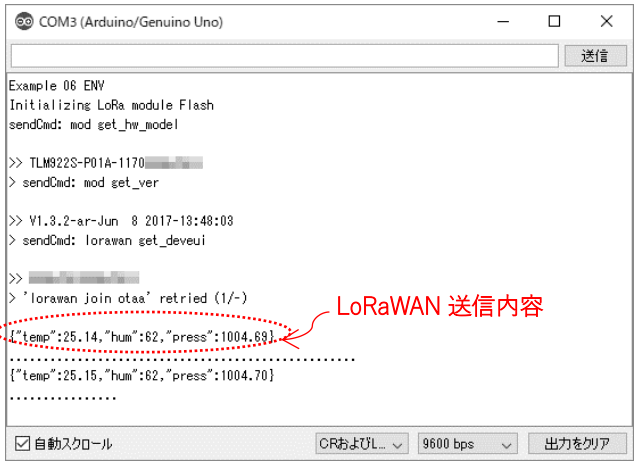


図 10-3　IoT 環境センサ cqp_ex06_env_bin の実行例

環境センサ BME280 から得られた温度（temp）・湿度（hum）・気圧(press)の計 3 値のデータを LoRaWAN へ送信したときのようす．

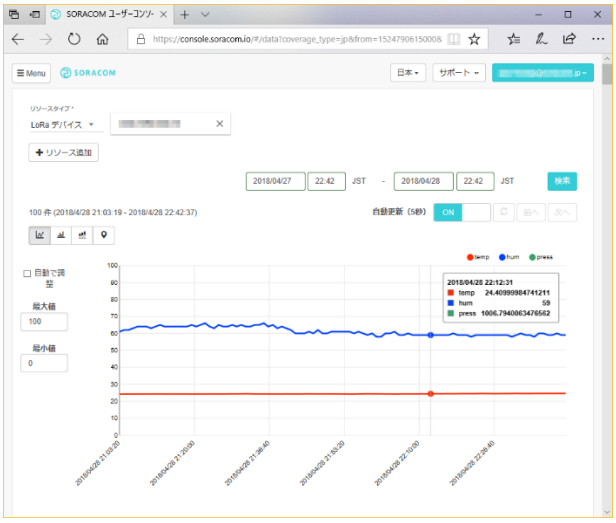


図 10-4　IoT 環境センサのデータを LoRaWAN 送信し，SORACOM Harvest へ蓄積したときの一例

温度値 temp，湿度値 hum，気圧値 press を SORACOM Harvest へ蓄積することができた．

図 10-5　ダウンリンク通信の入力例

LoRaWAN デバイスへ送信したいデータを 16 進数で入力する．本例では「Hello!」の ASCII コードを入力した．fPort には LoRaWAN デバイスが待ち受け中の受信ポート番号（本スケッチでは 2）を入力する．

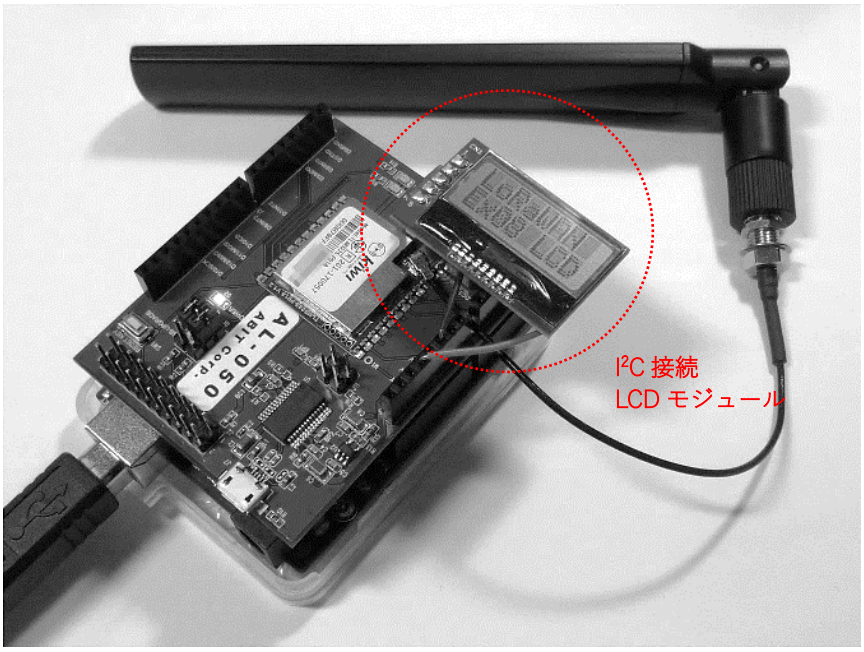


写真 10-2　I²C 接続 LCD モジュール AQM0802A を搭載した IoT 表示器の製作例

Arduino UNO へ LoRa Arduino 開発シールド AL-050 を装着し，I²C 接続 LCD モジュール AQM0802A（秋月電子通商製）を接続した．

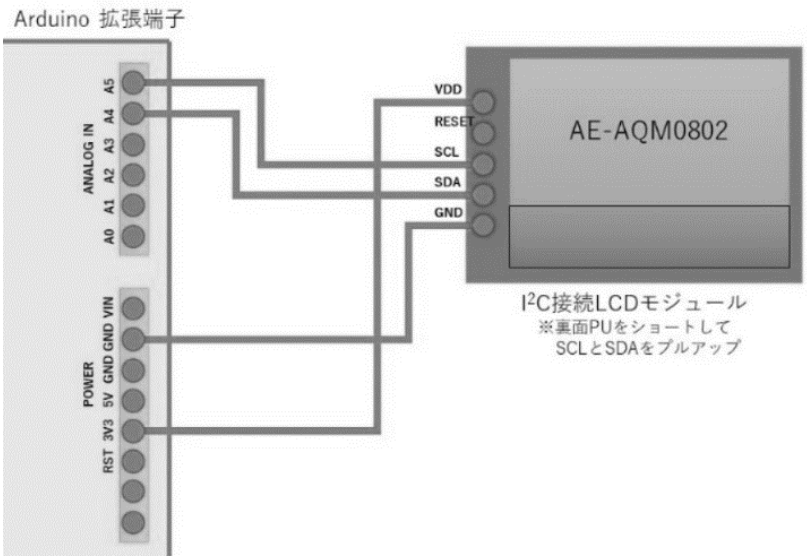


図 10-6　LCD モジュールの接続図

I²C 接続 LCD モジュール（秋月電子通商製）の SDA を Arduino UNO の Analog 4 へ，SCL を Analog 5 へ接続し，VDD へ Arduino の 3.3V の電源を供給する．

11. STM32L0 LoRa Discovery Kit を試してみよう

STM32L0 LoRa Discovery Kit は，村田製作所製の LoRa モジュール CMWX1ZZABZ-091 を搭載した LoRaWAN 評価ボードです．ほぼマイクロ SD と同じ 12.5×11.5mm のサイズのモジュール内に，ST マイクロエレクトロニクス製の MPU と Semtech 製 LoRa トランシーバ SX1276 を内蔵しており，様々な機器への組み込みが可能です．

SORACOM ユーザ・コンソールから購入すると，予め SORACOM へ接続するための情報が書き込まれた状態で送られてきます．本ボードを使うにあたり，以下の 2 つの注意点があります．

LoRa Discovery Kit の注意点
・ボード内に予め設定されている接続情報を消さないこと
・SORACOM が配布するサンプルを使用すること

例えば，ST-LINK Utility を使ってフラッシュの消去を行ってしまうと，接続情報が消えてしまい，SORACOM の LoRaWAN サービスへ接続できなくなってしまいます．また，SORACOM が配布するサンプル以外を使用すると，LoRaWAN へ接続できなくなるだけでなく，電波法に違反してしまう恐れがあります．

開発環境には，ARM 社の Keil MDK を使用します．評価用のライセンスだと LoRaWAN 接続用のプログラム End_Node のコンパイルが出来ないので，評価キットに同封されている PSN（ARM 社 Keil 製品インストール用シリアル番号）から LIC（ライセンス番号）を取得し，アクティベートを実行してください．ARM 社が STM32L0/STM32F0 用に無料で配布している PSN を使うこともできます．また，評価キットに同封されているダウンロード用の URL からサンプル・プログラムの ZIP ファイルをダウンロードし，サンプル End_Node を STM32L0 LoRa Discovery Kit へ書き込みます．詳細については，SORACOM が提供する下記の情報を参考にしてください．

STM32L0 LoRa Discovery Kit 設定方法：
https://dev.soracom.io/jp/start/lora_hw_stm32l0/

End_Node プロジェクトの保存場所：
Lora_soracom_rev2.1 > Projects > Multi > Applications > LoRa > End_Node > MDK-ARM > B-L072Z-LRWAN1 > Lora.uvprojx

ボード上の RESET と書かれた黒色のボタンを押して起動すると，LoRaWAN ネットワークへ参加し，緑色の LED(写真 11-2・右上部の LD1)が点灯し，10 秒ごとに送信が行われます．図 11-1 に，SORACOM Harvest での確認例を示します．なお，執筆時点では，接続に 10 分以上を要する場合や，送信の ACK を受信できない不具合がありました．本記事が出版される頃には改良されているでしょう．

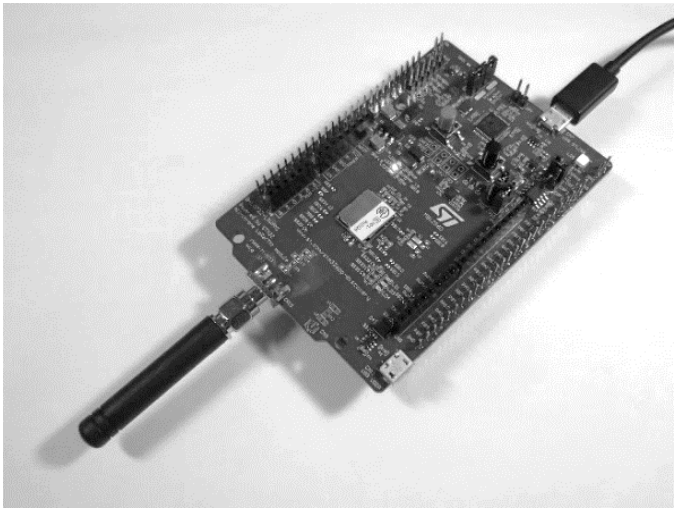


写真 11-1 LoRaWAN 評価キット（STM32L0 LoRa Discovery Kit）

村田製作所製の LoRa モジュール CMWX1ZZABZ-091 を搭載した LoRaWAN 評価ボード STM32L0 LoRa Discovery Kit B-L072Z-LRWAN1（ST マイクロシステムズ製）．予め SORACOM への接続情報が書き込まれている．

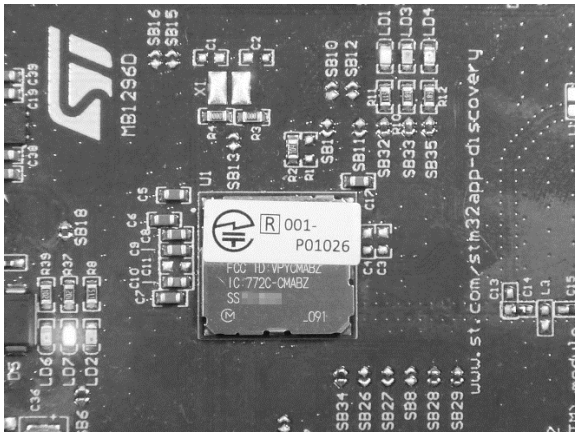


写真 11-2　村田製作所製の LoRa モジュール CMWX1ZZABZ-091 の拡大図

マイクロ SD 並みの 12.5×11.5mm サイズに ARM Cortex-M0+コア・マイコンと LoRa トランシーバを内蔵した村田製作所製の LoRa モジュール（ボード中央部の金属シールドで覆われた部品）.

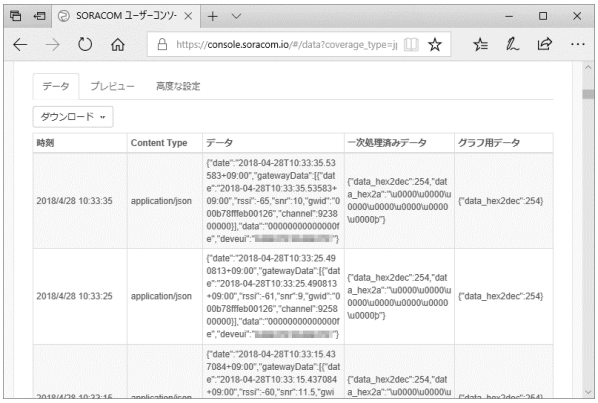


図 11-1　送信したデータを SORACOM Harvest へ蓄積したときのようす

STM32L0 LoRa Discovery Kit 上でサンプル・プログラム End_Node を実行すると，バイナリデータ 0xFE（10 進数で 254）が SORACOM Harvest に蓄積される.

12. バッテリ動作に対応した STM32L0 LoRa Discovery Kit で低消費電力動作

STM32L0 LoRa Discovery Kit の裏面には単 4 アルカリ乾電池 3 本で電源を供給するための電池ボックスが取り付けられており，低消費電力動作の実験が可能です.

プログラムの低消費電力動作機能を有効に設定するには，Keil MDK 上で End_Node のビルドを実行してから，Project:Lora →mlm32l07x01 → Project/End_Node → main.c → hw_conf.h を開き，下記の行を削除するかコメントアウト（行頭に「//」を挿入）してください.

```
/* uncomment below line to never enter lowpower modes in main.c*/
#define LOW_POWER_DISABLE
```

乾電池から電源を供給する場合は，開発およびデバッグ用の USB ST-LINK インタフェース部へ電源が供給されないため，リセット信号の解除ができなくなります．STM32L0 LoRa Discovery Kit の裏面の背面のチップジャンパ抵抗 SB37 を半田ごてで取り外し，USB ST-LINK のリセット信号を切り離してから，電池を取り付けてください.

乾電池による動作時の消費電流の測定結果例を図 12-2 に示します．本例では，待機時に約 10mA，送信時の約 400ms 区間に 33mA，受信時の約 600ms 区間に 21mA の電流が流れ，10 秒間隔で送信した場合の平均消費電流は 12mA でした．LoRa Arduino シールド AL-050 と比較したところ，表 12-1 のように，5 分の 1 から 6 分の 1 の消費電流となりました.

測定結果例図 12-2 の下側のパルス状の波形は，LoRaWAN 通信の検出回路の出力値です．本サンプル End_Node では，送信のたびに送信結果（以降 ACK 応答信号）を要求しているので，毎回，ゲートウェイからの応答があります．パルス波形内の 2 つの High レベル出力のうち，左側は STM32L0 LoRa

Discovery Kit の送信（上り）を検出した波形で，右側はゲートウェイからの ACK 応答信号（下り）です．STM32L0 LoRa Discovery Kit が送信を完了してから 1000ms 後に受信を開始し，ゲートウェイからの ACK 応答を受け取ると，すぐに受信機能をオフします．ACK を要求しなかった場合は，タイムアウトするまで待ち受けるので，1 回の送受信に約 3 秒を要しますが，ACK を要求し，ACK 受信後に受信機能を停止することで，動作時間を約 2 秒に短縮し，平均電力の省力化に寄与することが出来ます．

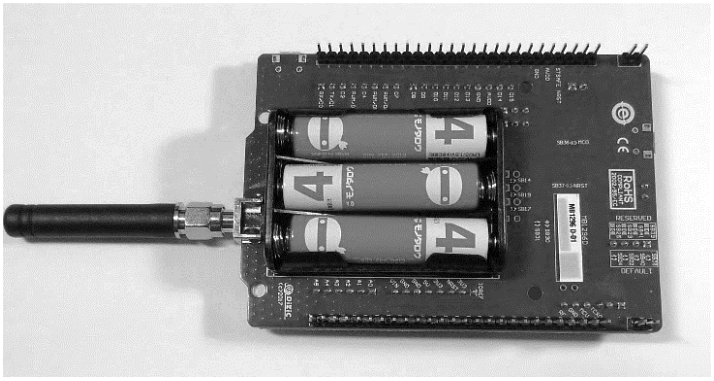


写真 12-1　単 4 型アルカリ乾電池で低消費電力動作させたときのようす

STM32L0 LoRa Discovery Kit の裏面には電池ボックスが取り付けられており，単 4 アルカリ乾電池を使った実験が可能．

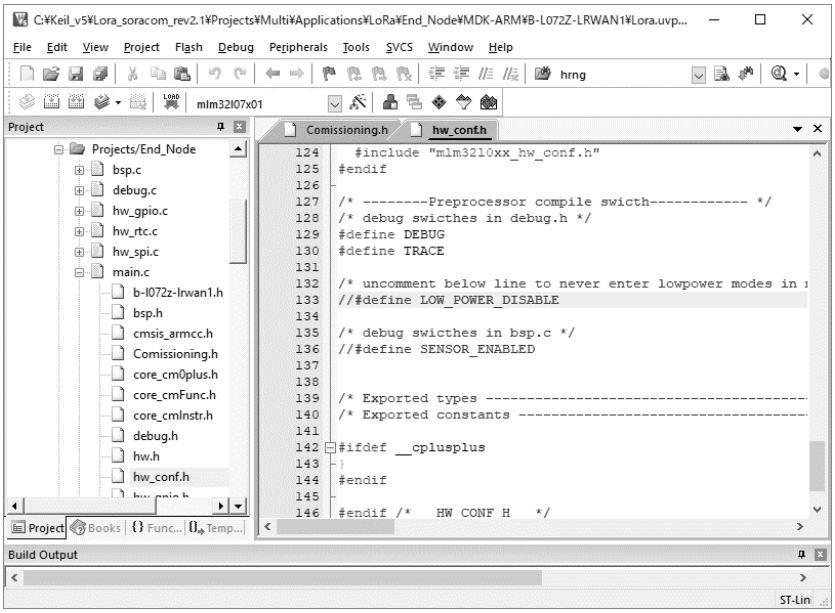


図 12-1　低消費電力動作を行うための Keil MDK での設定例

SORACOM からダウンロードしたサンプル End_Node の hw_conf.h 内の #define LOW_POWER_DISABLE をコメントアウト（行頭に「//」を挿入）すると，低消費電力動作が有効になる．

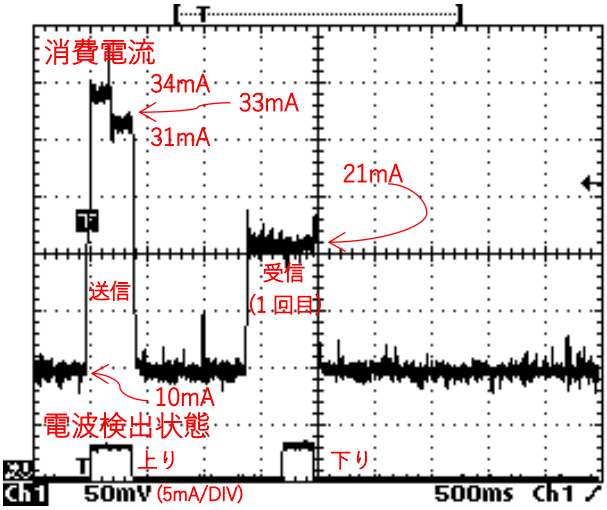


図 12-2　LoRaWAN への送信と受信動作のようす

送信時と受信時の電流の測定例．待機中は約 10mA の電流が流れ，送信時は約 33mA が 400ms，受信時は約 21mA が 600ms，流れることが分かった．

表 12-1　低消費電力動作時の消費電流の比較結果の一例（マイコン含む）

項目		LoRa Arduino 開発シールド AL-050		STM32L0 LoRa Discovery Kit	
		実測値	シリアル無(換算)	実測値	メーカー仕様
送信時		190 mA	(140 mA～)	32 mA	47 mA
受信時		110 mA	(60 mA～)	21 mA	22 mA
待機時		80 mA	(30 mA～)	10 mA	不明
スリープ時		50 mA	不明	10 mA	不明
平均	10 秒※	84 mA	(33 mA～)	12 mA	不明
	60 秒※	57 mA	(6 mA～)	10 mA	不明

※送信間隔

13. 組み込み用 LoRaWAN モデムとして利用する

STM32L0 LoRa Discovery Kit に搭載された村田製作所製 CMWX1ZZABZ-091 は，マイクロ SD カード程度の小さなサイズであることや，低消費電力動作が可能な点で，図 13-1 の一例をはじめ，様々な機器へ組み込むことが可能です．ここでは，STM32L0 LoRa Discovery Kit を LoRaWAN モデムとして使用する方法について説明します．なお，本実験では，写真 13-1 のように Arduino UNO を使用しました．実用時は機器側のマイコンから LoRa モジュールを制御することを想定しています．

まずは，Keil MDK からプロジェクト「AT_Slave」を開き，コンパイル後に，STM32L0 LoRa Discovery Kit へソフトウェアを書き込んでおきます．また，書き込み後，USB ST-LINK インタフェースのシリアル信号とリセット信号を LoRa モジュールから切り離すために，ボード表面の SB28 と裏面の SB37 のチップジャンパ抵抗を半田ごてで取り外してください．

次に，Arduino UNO 側です．既に AL-050 用のソフトウェアを Arduino IDE へ組み込んでいる場合は，[ファイル]メニューの[スケッチ例]からサンプル・スケッチ「cqp_stm32_AT_Master」を Arduino UNO へ書き込んでください．もしくは，https://goo.gl/HKhJyC （GitHub 上の公開フォルダ）へアクセスし，3 つの ino ファイルをダウンロードして，ご利用ください．

以上の準備が出来たら，電源を切り，各ボードの相互配線を行います．Arduino UNO の Digital 12 番ピン（シリアル送信）を RX（受信）へ，Digital 11 番ピン（受信）を STM32L0 LoRa Discovery Kit の TX（送信）へ，電源の 5V 同士，GND 同士を接続します．ただし，Arduino UNO のシリアル信号は 5V で，STM32L0 LoRa Discovery Kit は 3.3V なので，レベル変換が必要です．図 13-2 では，実験用の簡易レベル変換の一例として，Arduino UNO の送信側にダイオード 1N4148 のカソードを接続し，STM32L0 LoRa Discovery Kit の RX 側に同アノードと 3.3V へのプルアップ抵抗 10kΩを接続しました．Arduino UNO の受信側は STM32L0 LoRa Discovery Kit の TX へ直結しました．この簡易レベル変換回路には実験用としての実績は十分にありますが，極端な温度環境や製造ばらつきなどによって，通信エラー発生する可能性があります．製品では，レベル変換 IC などを使用してください．

配線完了後，Arduino UNO を PC へ接続すると，LoRaWAN ネットワーク接続を開始します．動作の流れを，図 13-3 および，以下に示します．

① LoRaWAN モデムとのシリアル（UART）接続を確認するために，モデム制御コマンド「AT」をシリアル送信し，応答値「OK」を確認します．
② モデムを初期化するための制御コマンド「ATZ」をシリアル送信します．
③ LoRaWAN ネットワークへ接続する制御コマンド「AT+JOIN」をシリアル送信します．
④ ネットワーク接続状態を確認するコマンド「AT+NJS=?」を送信します．応答値「0」は未接続状態を，応答値「1」は接続状態を示します．応答値が「1」になるまで，繰り返します．

⑤ Arduino マイコンの内部温度を 30 秒毎に測定し，LoRaWAN へ送信するための制御コマンド「AT+SEND=1:温度値」を，LoRaWAN モデムへシリアル送信します．バイナリを送信したいときは「AT+SENDB=1:バイナリ値（16 進数のテキスト文字列)」を使用します．

Arduino IDE 上のタブ「lora」をクリックすると，制御コマンドを LoRaWAN モデムへ送信するドライバ部のスケッチ lora.ino が表示されます．関数 lora_init は，手順①のシリアル接続確認部です．関数 lora_connect は，手順②～④の LoRaWAN ネットワーク接続部です．関数 lora_send は手順⑤の LoRaWAN へのデータ送信部です．LoRaWAN 送信時にゲートウェイからの ACK 応答を要求したい場合は，lora_connect 内でコメントアウトした制御コマンド「AT+CFM=1」を使ってください．

なお，筆者が作成したサンプルは LoRaWAN を使ったシステムの実験を目的としたものです．実製品へ展開する場合は，異常系の処理の追加や，改良，信頼性確認などが必要です．LoRaWAN を使ったシステム実験を着手するときの参考用として活用いただければと思います．

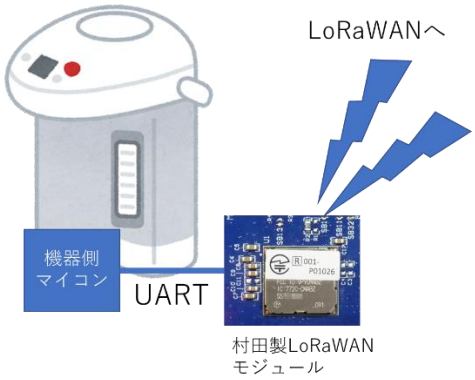


図 13-1　組み込み用 LoRaWAN モデムの使用例

マイクロ SD サイズの小型 LoRaWAN モデムを機器側マイコンと UART で接続する場合の一例

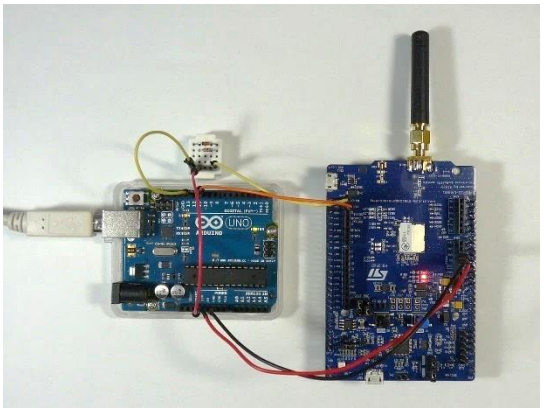


写真 13-1　組み込み用 LoRaWAN モデムの実験のようす

村田製作所製の LoRa モジュールを組み込み用 LoRaWAN モデムとして使用するための実験を行った．

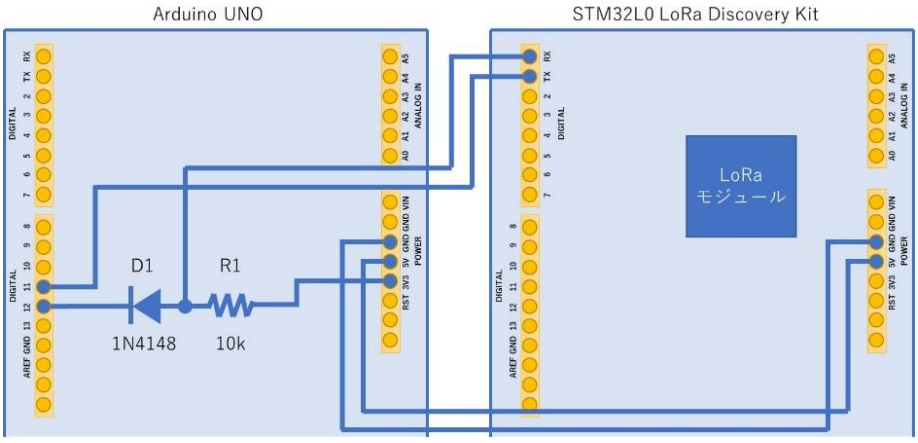


図　13-2　実験用 LoRaWAN モデムの配線図

Arduino と LoRa モデムのシリアル信号は，実験用の簡易レベル変換回路を用いて接続した．

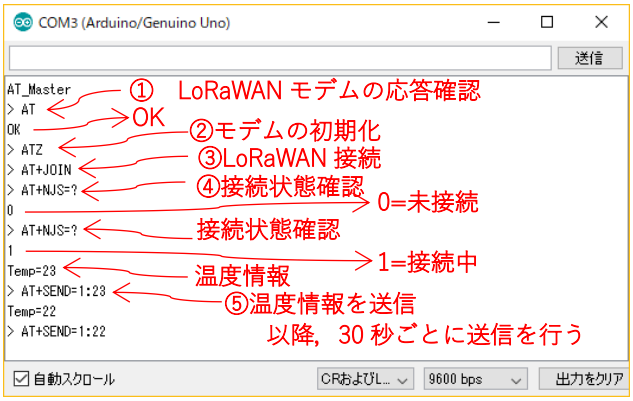


図 13-3　LoRaWAN モデムの動作例

AT コマンドで LoRaWAN モデムを制御したときのようす．モデムの応答を確認後，初期化と接続を行い，接続が完了したら，温度情報を送信する．

むすび

SORACOM の LoRaWAN サービスを利用することで，LoRaWAN の実験を容易に行うことが出来ました．他の方法としては，自営で LoRaWAN ゲートウェイを構築し，LoRaWAN 用のクラウド・サービス The Things Network を使う方法もありますが，冒頭に記したように，日本の電波法への対応を行うには多くの知識や調整が必要です．その手間やリスクを考慮すると，SORACOM のサービスを利用するのが手軽でしょう．

LoRa Arduino 開発シールド AL-050 については，Arduino IDE で試作や開発が行えることから，LoRaWAN の実験や動作確認が容易であり，応用システムの試作などに活用できそうです．ただし，実動作時時に不要なデバッグ用 USB シリアル IC の消費電流が大きい点に注意が必要です．

STM32L0 LoRa Discovery Kit については，様々な製品に組み込まれる可能性を実感することが出来ました．ボード上に搭載されている村田製作所製の LoRa モジュール CMWX1ZZABZ-091 の小型サイズ，ARM コア内蔵，低消費電力などの特長を活かした実製品の開発用に有用でしょう．しかし，システムの試作や実験用としては，国内向け LoRaWAN ライブラリの動作が不安定である点や，開発環境に Keil MDK が必要である点について，少し敷居が高く感じられました．LoRaWAN ライブラリの動作の安定性については，近いうちに解決するでしょう．開発環境については，もし，mbed が使えるようになれば，試作や実験用などに手軽に利用できるようになると思います．

〈国野　亘〉

参考文献

本書を執筆するにあたり，下記の文献を参考にしました．

- SORACOM LoRaWAN（株式会社ソラコム）
  https://dev.soracom.io/jp/start/lora/,
- LoRa Arduino 開発シールド AL-050 （株式会社ソラコム）
  https://dev.soracom.io/jp/start/lora_hw_al-050/
- STM32L0 LoRa Discovery Kit （株式会社ソラコム）
  https://dev.soracom.io/jp/start/lora_hw_stm32l0/
- SORACOM LoRaWAN デバイス設定（株式会社ソラコム）
  https://dev.soracom.io/jp/start/lora_uc_device/
- SORACOM-LoRaWAN, Arduino library for SORACOM LoRaWAN shield（株式会社ソラコム）
  https://github.com/soracom/SORACOM-LoRaWAN
- AL-050 User's Manual V1.5（株式会社エイビット）
- B-L072Z-LRWAN1 Discovery kit for LoRaWAN™, Sigfox™, and LPWAN protocols with STM32L0, Data brief, January 2018 DB3090 Rev 2 （STMicroelectronics）
- UM2115 Discovery kit for LoRaWAN™, Sigfox™, and LPWAN protocols with STM32L0, User manual, January 2018 UM2115 Rev 4 （STMicroelectronics）
- AN4967 Examples of AT commands on I-CUBE-LRWAN, Application note, December 2017 DocID030084 Rev 3

著者について

国野　亘（くにの・わたる） ボクにもわかる電子工作　管理人　https://bokunimo.net/

関西生まれ．言葉の異なる関東や欧米などさまざまな地域で暮らすも，近年は住みよい関西圏に生息し続けている哺乳類・サル目・ヒト属・関西人．おもにホビー向けのワイヤレス応用システムの研究開発を行い，その成果を書籍やウェブサイトで公開している．

権利情報

履歴

2018 年 4 月～5 月	調査・執筆期間
2018 年 5 月 13 日	執筆完了
2023 年 5 月 6 日	Web 公開