Wi-SUN Enhanced HAN
plus
B-route Dual stack

J11 18 台以上のセンサーデータアップロード
サンプルスクリプト操作説明書

2020 年 1 月 27 日
第 1.0 版

ISB CORPORATION

注意事項

1. 本書に記載されている内容は、本書発行時点のものであり、予告なく変更する可能性があります。
2. 本書に記載されている情報は、正確を期するために慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本書に記載されている情報の誤りに起因する損害がお客様に生じた場合におきましても、当社は一切その責任を負いません。
3. 本書に記載された技術情報の使用に関連して発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は一切その責任を負いません。当社は本書に基づき、当社または第三者の特許権、著作権その他知的財産権に基づくいかなる権利も許諾するものではありません。
4. 本書の全部または一部を当社の事前承諾を得ずに転載または複製することを固く禁じます。
5. 本書に記載されたサンプルプログラムの使用または使用不能によってお客様に生じるいかなる直接的、間接的損害（情報、データ、プログラム等の無体財産の損失、利益の喪失、中断などによる損害を含むがこれらに限られません）に関して、当社は一切の責任を負いません。

ISB CORPORATION

目次

ISB CORPORATION

1　はじめに

1.1　概要

本書では、Enhanced HAN(※以降 HAN と表記) において 18 台以上のセンサーからデータアップロードが可能なサンプルスクリプトの動作と使用方法について記載します。

1.2　構成

本サンプルスクリプトは下記の構成で動作します。

なお、エンドデバイスから送信されるデータは暗号化されていない平文で送信される点に注意ください

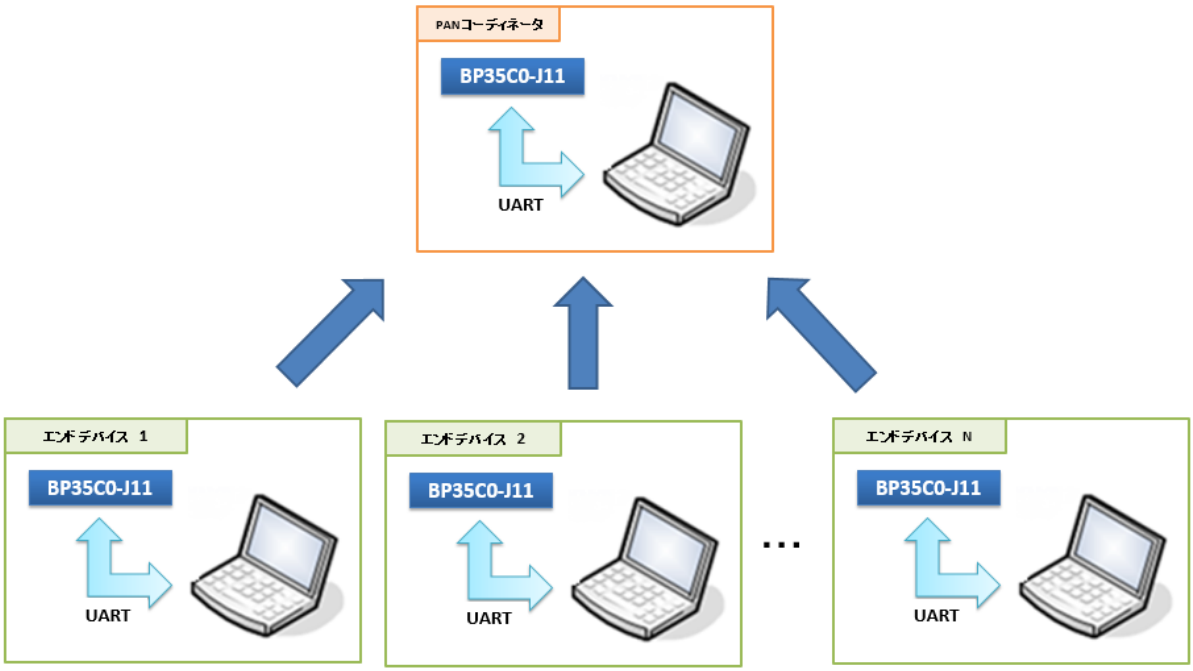

図 1　構成図

図 1 における機材詳細を以下に示します。

表 1　使用機材詳細

機材名	備考
PC	スクリプトを実行するための PC 推奨：Ubuntu 18.04 LTS desktop 64-bit
BP35C0-J11	Wi-SUN 無線モジュール スクリプトによって、下記それぞれの動作を行う ・HAN PAN コーディネータ ・HAN エンドデバイス

1.3 用語

表 2 用語一覧

用語	定義
Enhanced HAN	Enhanced Home Area Network の略。 HEMS と家電間の Wi-SUN 通信プロファイル

1.4 参照資料

表 3 参照資料一覧

番号	ドキュメント名
1	J11 UART_IF コマンド仕様書 第 1.1 版
2	J11 プロトコルスタック機能説明書 第 1.0 版

2　動作説明

HAN デバイスの接続数は、スマートメーター（B ルート）に接続しない場合、最大 17 台です。
本サンプルスクリプトでは、デバイスリストに登録されるエンドデバイスの台数を 17 台以下に保つことで、PAN コーディネータが 18 台以上のセンサーデータを受け取ることを可能としています。

なお、PANA 認証を行わないため、エンドデバイスから送信される無線データは暗号化されていない平文で送信されます。セキュアなデータを送信する場合、アプリケーションにて暗号化する必要があります。

2.1　起動から接続まで

PAN コーディネータは、HAN 運用状態に移行したらセンサーデータ通信用の UDP ポートを開いた後、エンドデバイスの接続を待ち受けます。

エンドデバイスは HAN 動作開始要求コマンドを発行し、HAN 運用状態（＝MAC 接続状態）まで遷移させます。

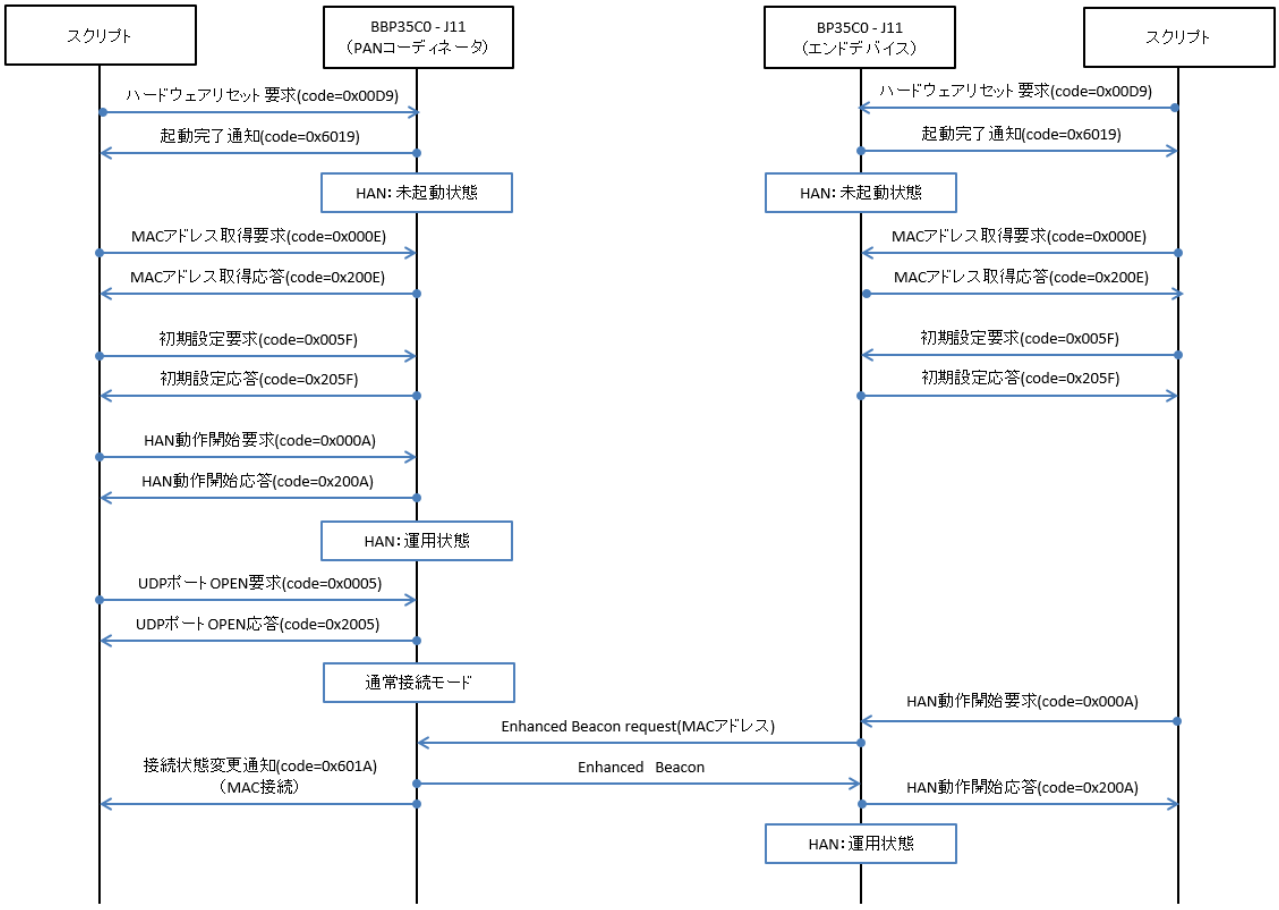

図 2　起動～接続シーケンス

※シーケンス内の「〇〇要求」「〇〇応答」「〇〇通知」の〇〇はコマンド名、括弧内の code はコマンドコードです。各コマンドの詳細については「UART IF コマンド仕様書」を参照してください。

2.2　データ送信

エンドデバイスは HAN 動作開始要求に成功したらデータ送信を実行します。送信成功したら HAN 動作終了要求を実行して HAN 未起動状態に遷移させます。その後、一定時間 DeepSleep した後、HAN 動作開始要求から繰り返します。

PAN コーディネータはデータ受信通知を受けたら HAN デバイスリスト削除要求を実行し、デバイスリストからデータ送信元のエンドデバイスを削除します。

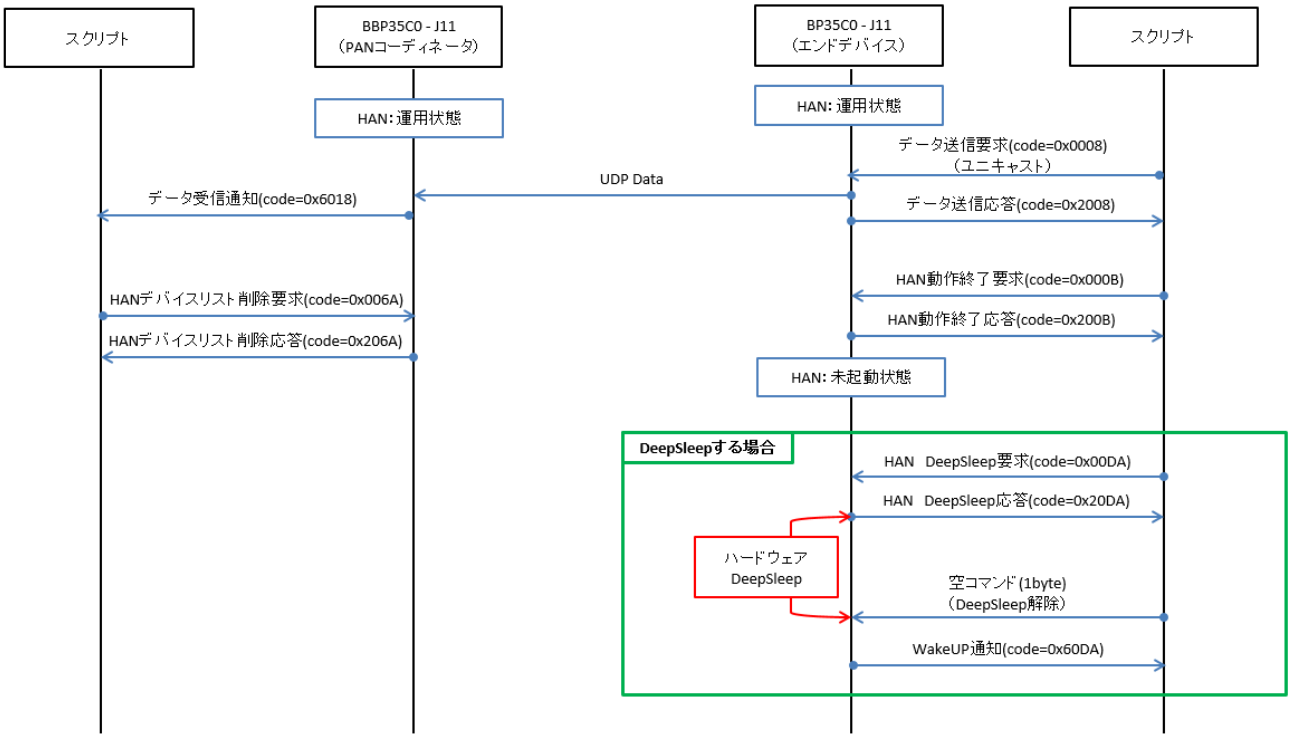

図 3　データ送信シーケンス

※シーケンス内の「○○要求」「○○応答」「○○通知」の○○はコマンド名、括弧内の code はコマンドコードです。各コマンドの詳細については「UART IF コマンド仕様書」を参照してください。

3　実行環境

サンプルスクリプトを実行するにあたって、PC には以下の環境が必要です。

A) python3.5 以降がインストール済み
　また、後述する python パッケージがインストール済み

B) Wi-SUN 無線モジュール（BP35C0-J11）のシリアルデバイスが認識済み

確認済み環境

- Ubuntu18.04 LTS desktop 64-bit
- Python 3.6.8

3.1　必要な Python パッケージ

本サンプルスクリプトでは python パッケージ「pySerial」を利用します。
実行前にパッケージをインストールしてください。

インストール手順

1. pip のインストール（インストール済みであれば不要です）

```
$ sudo apt-get install python3-pip
```

2. パッケージのインストール

```
$ sudo python3 -m pip install pyserial
```

4　ファイル説明

本サンプルスクリプトでは PAN コーディネータ 1 台、エンドデバイス 3 台分のファイルを提供しています。

4 台目以降のエンドデバイスを追加する手順については 6.5 章 を参照してください。

表 4　ファイル一覧

ファイル名	説明
pc_main.py	HAN PAN コーディネータ　メインモジュール
pc_config.json	HAN PAN コーディネータ　設定ファイル
ed_main_1.py	HAN エンドデバイス①　メインモジュール
ed_config_1.json	HAN エンドデバイス①　設定ファイル
ed_main_2.py	HAN エンドデバイス②　メインモジュール
ed_config_2.json	HAN エンドデバイス②　設定ファイル
ed_main_3.py	HAN エンドデバイス③　メインモジュール
ed_config_3.json	HAN エンドデバイス③　設定ファイル
isb_wsn_serial.py	Wi-SUN スタック シリアル通信制御モジュール

5　設定値

5.1　PAN コーディネータ

PAN コーディネータのメインモジュール（pc_main.py）は、設定値を pc_config.json から読み込みます。

```
{
    "SerialInterface": {
        "port": "/dev/ttyUSB0"
    },
    "InitialSetting": {
        "actionMode": 1,
        "sleepSupport": false,
        "channel": 5,
        "txpower": 0,
        "panid": "0xCAFE"
    }
}
```

図 4　pc_config.json 設定例

各設定の説明は以下のとおりです。

表 5　PAN コーディネータ 設定値

設定名	説明
SerialInterface.port	シリアル通信デバイスファイル デバイスファイルのフルパス
InitialSetting.actionMode	動作モード 1：PAN コーディネータ(HAN) 2：コーディネータ(HAN) 3：エンドデバイス(HAN) 5：Dual（B ルート＆HAN）
InitialSetting.sleepSupport	HAN Sleep 機能設定 false：無効 true：有効
InitialSetting.channel	チャネル番号(10 進数) 4～17：922.5～927.7MHz
InitialSetting.txpower	送信電力 0：20mW 1：10mW 2：1mW
InitialSetting.panid	PAN ID（16 進文字列） “0x0000”～”0xFFFE”

5.2　エンドデバイス

エンドデバイス①、②、③のメインモジュール（ed_main_N.py）は、設定値を ed_config_N.json から読み込みます。※「N」は 1～3 の数字です

```
{
    "SerialInterface": {
        "port": "/dev/ttyUSB1"
    },
    "InitialSetting": {
        "actionMode": 3,
        "sleepSupport": false,
        "channel": 5,
        "txpower": 0
    },
    "HAN": {
        "pancAddr": "001D129100003DA7"
    }
}
```

図 5　ed_config_N.json 設定例

各設定の説明は以下のとおりです。

表 6　エンドデバイス 設定値

設定名	説明
SerialInterface.port	シリアル通信デバイスファイル 表 5 PAN コーディネータ 設定値 と同じ
InitialSetting.actionMode	動作モード 表 5 PAN コーディネータ 設定値 と同じ
InitialSetting.sleepSupport	HAN Sleep 機能設定 表 5 PAN コーディネータ 設定値 と同じ
InitialSetting.cannel	チャネル番号 表 5 PAN コーディネータ 設定値 と同じ
InitialSetting.txpower	送信電力 表 5 PAN コーディネータ 設定値 と同じ
InitialSetting.pancAddr	PAN コーディネータの MAC アドレス 8byte の 16 進文字列

6　実行手順

サンプルスクリプトを実行する前に必ず設定ファイル（*.json）の値を実行環境、および使用機器に合わせて変更してください。

具体的には以下の 2 つです。

・シリアル通信デバイスファイル（＝SerialInterface.port）

・PAN コーディネータの MAC アドレス（＝InitialSetting.pancAddr）

サンプルスクリプトの実行手順は以下の通りです。

1. PAN コーディネータを起動する
   詳細は「6.1 PAN コーディネータを起動する」を参照

2. エンドデバイス①を起動する
   詳細は「6.2 エンドデバイス①を起動する」を参照

3. 2 台目のエンドデバイス②を起動する
   詳細は「6.3 エンドデバイス②を起動する」を参照

4. 3 台目のエンドデバイス③を起動する
   詳細は「6.4 エンドデバイス③を起動する」を参照

5. 4 台目以降（N 台目）のエンドデバイスを起動する
   詳細は「6.5 N 台目のエンドデバイスを起動する」を参照

6. PAN コーディネータにて受信データを確認する
   詳細は「6.6 PAN コーディネータにて受信データを確認する」を参照

6.1　PAN コーディネータを起動する

pc_main.py を実行します。

実行コマンド：

```
$ sudo python3 pc_main.py
```

PAN コーディネータとして起動後、エンドデバイスからの HAN 接続を待ち受けます。

PAN コーディネータ起動後の画面を以下に示します。

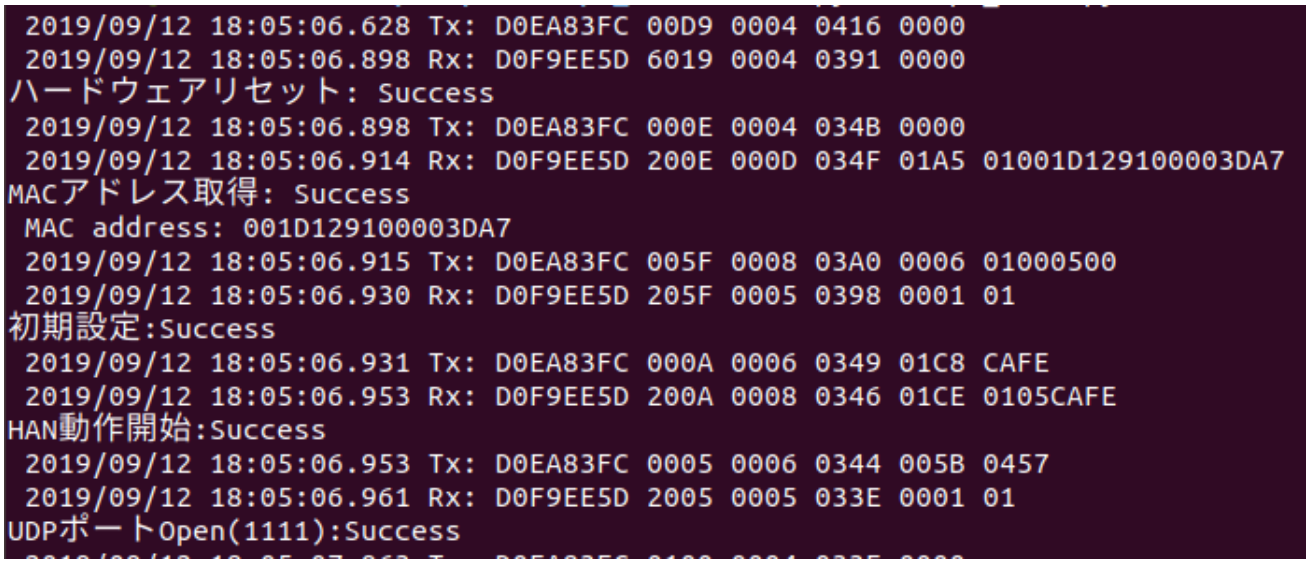

図 6　pc_main.py 実行画面（起動）

6.2　エンドデバイス①を起動する

ed_main_1.py を実行します。

実行コマンド：

```
$ sudo python3 ed_main_1.py
```

エンドデバイスとして起動後、PAN コーディネータとの HAN 接続を行います。
エンドデバイス起動後の画面を以下に示します。

```
 2019/09/12 18:05:30.062 Tx: D0EA83FC 00D9 0004 0416 0000
 2019/09/12 18:05:30.334 Rx: D0F9EE5D 6019 0004 0391 0000
ハードウェアリセット: Success
 2019/09/12 18:05:30.334 Tx: D0EA83FC 000E 0004 034B 0000
 2019/09/12 18:05:30.349 Rx: D0F9EE5D 200E 000D 034F 01AC 01001D129100019D2
MACアドレス取得: Success
 MAC address: 001D1291000019D2
 2019/09/12 18:05:30.349 Tx: D0EA83FC 005F 0008 03A0 0008 03000500
 2019/09/12 18:05:30.365 Rx: D0F9EE5D 205F 0005 0398 0001 01
初期設定:Success
 2019/09/12 18:05:30.365 Tx: D0EA83FC 000A 000C 034F 01A4 001D129100003DA7
 2019/09/12 18:05:32.951 Rx: D0F9EE5D 200A 0011 034F 0450 0105CAFE001D129100003DA7DE
HAN動作開始:Success
 MAC Connected to '001D129100003DA7', rssi=-34dBm
```

図 7　ed_main_1.py 実行画面（起動、HAN 接続）

エンドデバイスは HAN 接続後に PAN コーディネータに対してデータ送信を行います。
データ送信時の画面を以下に示します。

```
 2019/09/12 18:05:32.951 Tx: D0EA83FC 0008 001D 035E 0479 FE80000000000000021D129100003DA7045704570003123456
 2019/09/12 18:05:32.989 Rx: D0F9EE5D 2008 0009 0345 009D 0100123456
UDPデータ送信:Success
 2019/09/12 18:05:32.989 Tx: D0EA83FC 000B 0004 0348 0000
 2019/09/12 18:05:33.003 Rx: D0F9EE5D 200B 0005 0344 0001 01
HAN 動作終了:Success
```

図 8　ed_main_1.py 実行画面（データ送信）

6.3　エンドデバイス②を起動する

ed_main_2.py を実行します。

実行コマンド：

```
$ sudo python3 ed_main_2.py
```

エンドデバイスとして起動後、PAN コーディネータとの HAN 接続を行います。
エンドデバイス起動後の画面を以下に示します。

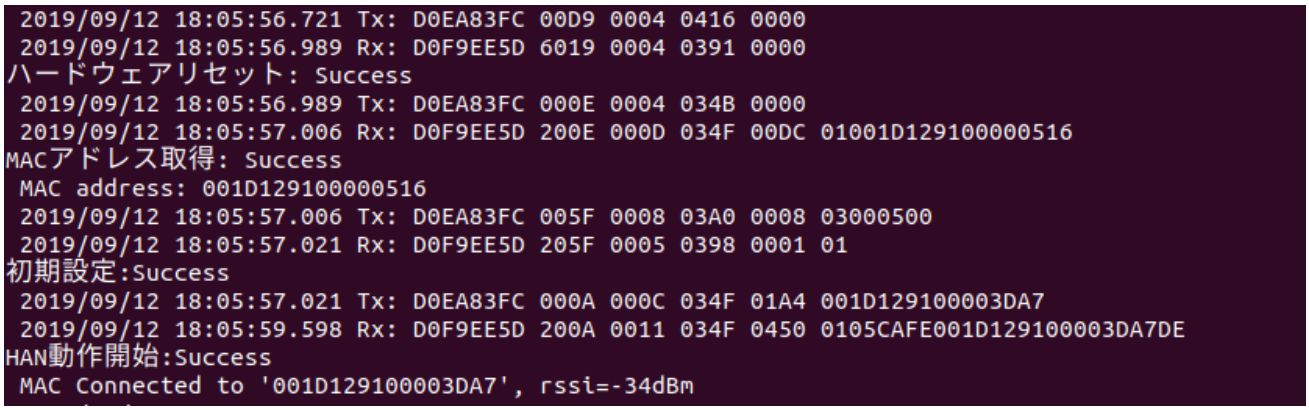

図 9　ed_main_2.py 実行画面（起動、HAN 接続）

エンドデバイスは HAN 接続後に PAN コーディネータに対してデータ送信を行います。
データ送信時の画面を以下に示します。

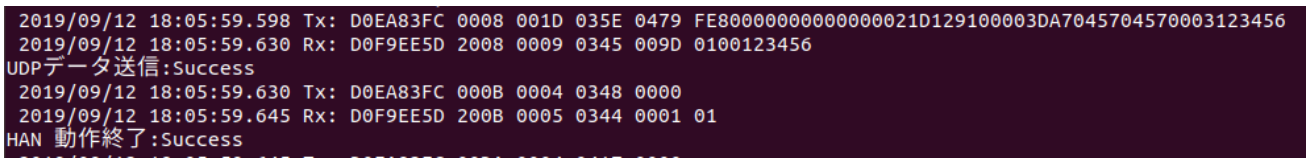

図 10　ed_main_2.py 実行画面（データ送信）

6.4　エンドデバイス③を起動する

ed_main_3.py を実行します。

実行コマンド：

```
$ sudo python3 ed_main_3.py
```

エンドデバイスとして起動後、PAN コーディネータとの HAN 接続を行います。
エンドデバイス起動後の画面を以下に示します。

```
 2019/09/12 18:06:25.641 Tx: D0EA83FC 00D9 0004 0416 0000
 2019/09/12 18:06:25.910 Rx: D0F9EE5D 6019 0004 0391 0000
ハードウェアリセット: Success
 2019/09/12 18:06:25.911 Tx: D0EA83FC 000E 0004 034B 0000
 2019/09/12 18:06:25.926 Rx: D0F9EE5D 200E 000D 034F 00CF 01001D129100000509
MACアドレス取得: Success
 MAC address: 001D129100000509
 2019/09/12 18:06:25.926 Tx: D0EA83FC 005F 0008 03A0 0008 03000500
 2019/09/12 18:06:25.950 Rx: D0F9EE5D 205F 0005 0398 0001 01
初期設定:Success
 2019/09/12 18:06:25.950 Tx: D0EA83FC 000A 000C 034F 01A4 001D129100003DA7
 2019/09/12 18:06:28.533 Rx: D0F9EE5D 200A 0011 034F 0450 0105CAFE001D129100003DA7DE
HAN動作開始:Success
 MAC Connected to '001D129100003DA7', rssi=-34dBm
```

図 11　ed_main_3.py 実行画面（起動、HAN 接続）

エンドデバイスは HAN 接続後に PAN コーディネータに対してデータ送信を行います。
データ送信時の画面を以下に示します。

```
 2019/09/12 18:06:28.533 Tx: D0EA83FC 0008 001D 035E 0479 FE8000000000000021D129100003DA7045704570003123456
 2019/09/12 18:06:28.565 Rx: D0F9EE5D 2008 0009 0345 009D 0100123456
UDPデータ送信:Success
 2019/09/12 18:06:28.565 Tx: D0EA83FC 000B 0004 0348 0000
 2019/09/12 18:06:28.581 Rx: D0F9EE5D 200B 0005 0344 0001 01
HAN 動作終了:Success
```

図 12　ed_main_3.py 実行画面（データ送信）

6.5　N 台目のエンドデバイスを起動する

4 台目以降のエンドデバイスを追加し、起動する手順を以下に記載します。

1. ed_config_1.json をコピーし、別ファイルで「ed_config_4.json」を作成
   ※作成するファイル名は任意でかまいません

2. ed_main_1.py をコピーし、別ファイルで「ed_main_4.py」を作成
   ※作成するファイル名は任意でかまいません

3. 手順 2 で作成したファイルを開き、下記の箇所で指定している設定ファイル名を手順 1 で作成したファイル名に変更
   変更前：

```
def load_config():
    """設定を json ファイルから読み込む
    """
    f = open("ed_config_1.json", 'r')
```

   変更後：

```
def load_config():
    """設定を json ファイルから読み込む
    """
    f = open("ed_config_4.json", 'r')
```

4. ed_config_4.json の SerialInterface.port を実行環境に合わせて編集する

5. 手順 2 で作成した python スクリプトを実行する

```
$ sudo python3 ed_main_4.py
```

6.6　PAN コーディネータにて受信データを確認する

PAN コーディネータはエンドデバイスからのデータを受信した後、送信元のエンドデバイスを HAN デバイスリストから削除します。その後、再度エンドデバイスからの HAN 接続を待ち受けます。

エンドデバイスからのデータを受信した時の画面を以下に示します。

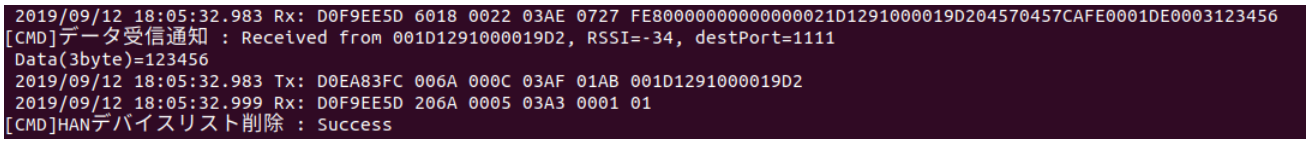

図 13　pc_main.py 実行画面（データ受信）

6.7　実行画面における設定値

本章の実行画面において、サンプルスクリプト実行時の設定値は以下の通りです。

表 7　実行画面における共通設定値一覧

設定名	値
HAN Sleep 機能設定	無効
チャネル番号	5 (922.9MHz)
送信電力	20mW
PAN-ID	0xCAFE(51966)

表 8　実行画面での MAC アドレス一覧

動作モード	MAC アドレス
PAN コーディネータ	01:00:1D:12:91:00:00:3D:A7
エンドデバイス①	01:00:1D:12:91:00:00:19:D2
エンドデバイス②	01:00:1D:12:91:00:00:05:16
エンドデバイス③	01:00:1D:12:91:00:00:05:09

改訂履歴

日付	版数	改訂内容
2020 年 1 月 27 日	第 1.0 版	初版