AirChecker（C3/USB) 利用ガイド（サンプルプログラム説明書）

Ver.1.0.0.b

CLEALINK TECHNOLOGY Co., Ltd.

2024-05-01

目次

1 概要

AirChecker（C3/USB) は、LoRa 通信での伝送電波の強度（RSSI 値）を、３桁の液晶表示により簡単に目視できる機器です。
LoRa(E220-900T22S(JP) を使った Air 伝送での通信を行うため、送信機１個、受信機１個を１セットとして動作します。
本機は MCU として ESP32C3-WROOM-02 を搭載しており、出荷時設定の状態で、PC の USB ポートあるいは、モバイルバッテリ、USB 充電アダプタ等にセットするだけで、独自に動作するようになっています。
出荷時設定では、同じファームウェアが書き込まれていますので、２個をそれぞれ電源に接続すると、送信機（送信側）になるか受信機（受信側）になるかを自動判定するネゴシエーションを実行、その後自動的に送受信動作を開始し、それぞれの受信時の RSSI 値を３桁液晶に表示します。

送受信動作は、
　送信側の送信後、受信側は受信した RSSI 値の表示。一定時間受信がなければ赤色 LED の点灯。
　受信側は受信後、受信通知を送信側に送信。
　送信側は受信通知を受信後に RSSI 値を表示し、一定時間受信がなければ赤色 LED の点灯。
を継続して行ないます。

サンプルプログラムについては、ArduinoIDE の開発環境をセットアップし、タイマーのライブラリをセットするだけで、コンパイル＆実行することができます。
また、サンプルプログラムを自由に改編して、実験装置の制作、LoRa の学習などに利用できます。

2 AirChecker のハードウェア構成

```
MCU : ESP32-C3-WROOM-02
LoRa: E220-900T22S(JP)
LCD : 3digit-LCD
LED : 6-LED,4-color
2 Shift-Register
USB : typeA
```

3 Arduino 開発環境のセットアップ

ArduinoIDE のセットアップの方法については、ネット検索で多数ヒットしますので、参考にしてください。

① Arduino IDE をインストールし、　日本語化までを実行

② AirChecker をＵＳＢポートに差し込む

③ Arduino のツールメニューの 『ボード』 をクリックし、プルダウンメニューで ”ESP32C3 DEV MODULE” を選択します。

④ 同じくツールメニューで ポートを確認し、ESP32-C3 が表示されたポート COMxx(mmmm ESP32-C3) を選択します。

⑤ファイルメニューの基本設定を選択し、『追加のボードマネージャーの URL』に

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_dev_index.json

を記入（URL をカット＆ペースト）し “OK” をクリック。

⑥ツールメニューのボード -> ボードマネージャーを開き、ボードマネージャの検索に”ESP32” を入力し、 表示された “esp32 by Espressif Systems” の バージョンの選択で、2.0.5 (推奨) を選択し、インストールボタンをクリック。 （少し時間がかかります）

（ボードマネージャーは逐次更新されていますが、現状では 2.0.11 までのバージョンで動作確認をしていますが、2.0.6 以降については、シリアルモニタにタイマー割り込みのエラーが表示されます。）

⑦ ツールメニューのライブラリを管理で、ライブラリマネージャーで、 ”ESP32C3_Timer_interrupt” をインストールしてください。

タイマー割り込みについての詳細はこちらを参考に https://github.com/khoih-prog/ESP32_C3_TimerInterrupt

これで、AirChecker のためのセットアップが完了です。

（プログラムを改編し、デバッグのためにシリアルモニタを使用する場合には、
Arduino のツールメニューで、USB CDC On Boot:“Enabled” を選択してください。）

4 サンプルプログラムのファイル構成

ダウンロードした airchecker_verx.x.x.zip をそのまま解凍すると airchecker フォルダが生成され、以下のファイルが展開されます。

```
airchecker.h                --- ハードウェアの設定他、コンパイル時に設定できる値、条件コンパイルの設定
airchecker.ino              --- メインプログラム
airchecker.h                --- 条件コンパイル、各種I/O定義などの設定用ヘッダファイル
esp_e220900t22s_jp_lib_3.h  --- E220-900T22S 用ヘッダファイル
esp_e220900t22s_jp_lib_3.ino --- E220-900T22S 用初期設定、送受信など
LCD.ino                     --- ３桁液晶表示ソフトウェアモジュール
LED.ino                     --- LED点灯用ソフトウェアモジュール
LoRa.ino                    --- LoRa通信用ソフトウェアモジュール
timer.ino                   --- タイマー用ソフトウェアモジュール
```

```
10    CONFIG.h                          --- E220-900T22s(JP)のコンフィグレーション設定ファイル
```

5 Arduino　の起動と操作

エクスプローラで表示された上記のファイルで、airchecker.ino ファイルをダブルクリックすると、ArduinoIDE が起動し、すべてのファイルが読み込まれます。

『マイコンボードに書き込み』ボタンをクリックするとコンパイル～AirChecker への書き込みが実行され、airchecker が動作を開始します。

5.1 プログラムの変更など

E220-900T22S(JP) が動作する主な出荷時設定は、ＢＷ：１２５、ＳＦ：９、固定送信モード、周波数チャンネル０などです。詳細は、データシートを参照しながら、CONFIG. ｈを確認してください。

CONFIG.h は、データシートのレジスタ記載順に、各種パラメータを設定できるようになっています。アドレス、チャンネル、暗号化キーなどの直接していする数値以外は、#define の選択によって指定できます。
(#define の二重定義や設定漏れがあった場合、コンパイル時にエラーが生じます。)
(WoR の設定については、WoR 機能を使用するとき（Mode1,Mode2 設定時）に有効になります。)

5.2 条件コンパイルの　#define 定義について

コンパイル時に設定できる値、条件コンパイルについて説明します。この設定は、airchecker.h に集約されています。

```
1  ○ プログラムのバージョン記載
2    #define VERSION_NO 100          // means 1.0.0
3
4    起動時に３桁液晶に表示されます。
5
6  ○ Private Air伝送の機器確認のための設定
7    #define MAGIC_KEY 0x54          // 本 Air伝送確認のためのマジックキー.
8    #define UNIQUE_ID 123           // 本 Air伝送確認のための固有ID.
9
10 ○ 自機番号
11   #define MACHINE_NO 1            // Machine No
12
13   上記バージョン表示のあとに表示されます。
14
```

```
15  ○ 起動時に送信側か受信側かの自動判定(Negotiation) を行うかの設定
16    #define STARTUP_NEGOTIATION 1
17
18  ○ 送信側動作か受信側動作を指定
19    #define SENDER_or_RECEIVER 1    // 1 : Sender / 0 : Receiver
20
21    サンプルプログラムではairchecker起動時に送信側、受信側を判定するため
        の
22    Negotiationを行ないますので、この定義は上記 STARTUP_NEGOTIATION が'0'
23    のときに有効になります。
24    このとき　送信側(1)か受信側(0)かを設定し、コンパイル～デバイスへの書
        き
25    込みが必要です。
26
27  ○ 3桁液晶表示を上下逆転するか否かの指定
28    #define LCD_UpsideDown 0        // 液晶表示を上下逆にするか否かの設
        定。
29
30    PCなどのUSBポートの差し込み位置に対応。
31    実装されている3桁液晶は上下反転表示が可能なため、この機能を実装して
        います。
32
33  ○ Arduinoのシリアルモニタを使用するか否かの指定
34    #define USE_SerialPrint 0       // Use or not Serial.print() to
        display Monitor
35                                      or TerminalAp on PC
36    '0' の場合、Serial.printf()等のコンパイル自体を行ないません。
37
38    ソースプログラムを変更して、試行、デバッグなどを行う場合に 1（有効）
        にします。
39
40    モバイルバッテリー、USB電源アダプタにセットする場合と、Arduinoで
41    シリアルモニタを使用しない場合には、0（無効）にしてください。
42    Serial.printf()実行時にwait状態になります。
43
44  ○ 受信可能なデータが存在する場合に、RSSI値表示のみを行うか否か
45    #define RSSI_MEASUREMENT_ONLY false // Air受信時のRSSI表示のみを行う
        場合　true
```

6 airchecker 起動後の動作

起動後の動作は、下記の表示の順に移行します。

①３桁液晶の表示テスト

②全 LED ４回点滅テスト

③ Version 表示

②自機番号表示～送信側、受信側自動判定の Negotiation 実行

③以後 RSSI 表示　受信時の RSSI 値が表示されます。(データ受信がない場合には、最後の受信時の RSSI 値が表示されたままになります。)

6.1 ３桁液晶表示について

表示される数値は受信感度を示す RSSI 値（Received Signal Strength Indication）で、３桁の数値で表示されます。
RSSI 値とは電波の強さを示す値で、単位はｄＢｍ（デシベルミリワット）です。数値の範囲は、０～－１４０ｄＢm程度までの１０進数で表示されますが、液晶が３桁表示になりますので、‘－’（マイナス）は省略します。この数値は大きい方（マイナスが小さい方）ほど電波は強いということになります。安定して通信（Air 伝送）ができる目安は、０～－１２０ｄＢm程度までです。

6.2 LED 点灯について

LED は図のように USB コネクタ側に２個（上から黄色、緑)、アンテナ端子側に４個（上から、黄色、青、赤、緑）の６個が実装されています。

LED 点灯が意味する状態は、以下になります。(サンプルプログラムでの設定)

```
1 アンテナ端子側　緑　：　送信機として動作している場合に点灯
2 アンテナ端子側　赤　：　受信データがない場合に点灯
3 アンテナ端子側　青　：　受信機として動作している場合に点灯
4 アンテナ端子側　黄　：　受信データの先頭の ＭＡＧＩＣ　ＮＯがairchecker
    のものと異なる場合に点灯。
5 USBコネクタ側　緑　：　E220-900T22S(JP)のAUXの状態（LOWの場合に点灯）
6 USBコネクタ側　黄　：　リザーブ（bluetoothLE接続状態などで使用）
```

7 ESP32C3-WROOM-02 の Wi-Fi、Bluetooth を使用する場合の注意事項について

本製品に使用している、無線マイコンモジュール ESP32-C3-WROOM-02 は、モジュール表面に刻印はありませんが、工事設計認証（技術適合認証）の取得済み品です。詳細は、以下 (ハイパ－リンク) で確認ください。

技術基準適合証明等を受けた機器の情報

また、本機器の初期状態で書き込まれているファームウェア（サンプルプログラム）は、このモジュールの無線出力を行っておりません。ユーザーによって Wi-FI, Bluetooth を実装する場合は、本

認証番号を示した上で、ご活用ください。（なお、本 E220-900T22S(JP) 920MHz LoRa 通信モジュールは、認証番号はモジュール刻印済みです）

本サンプルプログラムで、Wi-Fi、Bluetooth を利用する場合に、認証番号の表示を行なう関数も用意しました。上記ファイル中の　LCD.ino　の、最下部にある Line で、#define USING_WIFI_BLUETOOTH true
とし、適切な箇所（起動時など）で　Set_SEGMENT_r(); を実行してください。”r201-220555”　を３桁液晶に横スクロール表示します。