ETC1.1　プログラミングマニュアル

V1.0

2021 年 10 月 28 日

FundyLab

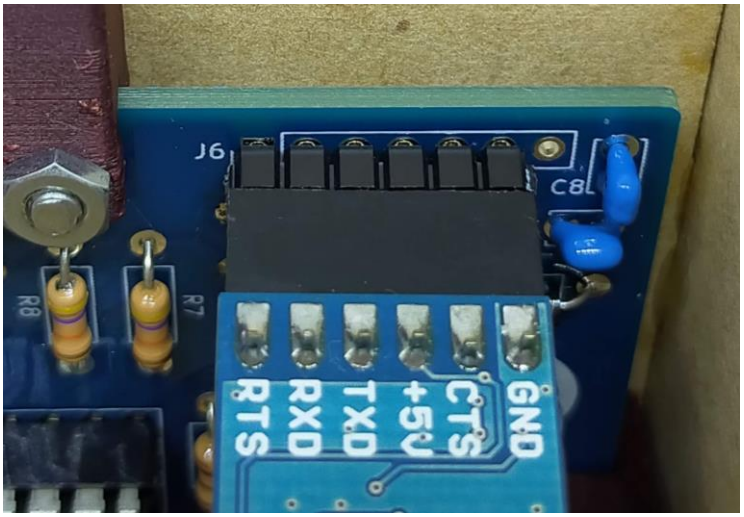
1.　必要部材

・電子宝箱キット本体（以下、本機と呼びます）
・ピンヘッダ 2.54mm ピッチ 6P
・USB シリアル変換器
　5V 又は 3.3V 電源出力可能タイプで、RTS 端子又は DTR 端子が付いており、
　ピンヘッダ出力のもの。以下の例では、そのまま本機に挿せます。
　例：FTDI USB シリアル変換アダプター　Rev.2（スイッチサイエンス様）
　　FT－232RQ　USBシリアル変換モジュール（秋月電子様）

2.　接続方法

本機を組み立て動作確認後、基板上のシルク J6 部分に、下図のように 6P ピンヘッダを差してはんだ付けします。J6 の Pin7 はオプションのリセットピンのため、空いていて構いません。

USB シリアル変換器のドライバーを PC にダウンロード・セットアップした後、上記例の USB シリアル変換器の場合は、下図のように差し込み、USB ケーブルで PC と接続します。別の USB シリアル変換器の場合、下の接続表の通り対応する端子を接続して下さい。



J6 のピン配置及び、USB シリアル変換器の接続表

本機　J6		USB シリアル変換側接続端子
ピン番号	機能	
Pin1	Arduino 書込リセットパルス入力	RTS 又は DTR
Pin2	TXD	RXD
Pin3	RXD	TXD
Pin4	Vcc 3-5V 入力	電源出力
Pin5	未接続	接続不要
Pin6	GND	GND
Pin7	$\overline{\text{RESET}}$	接続不要

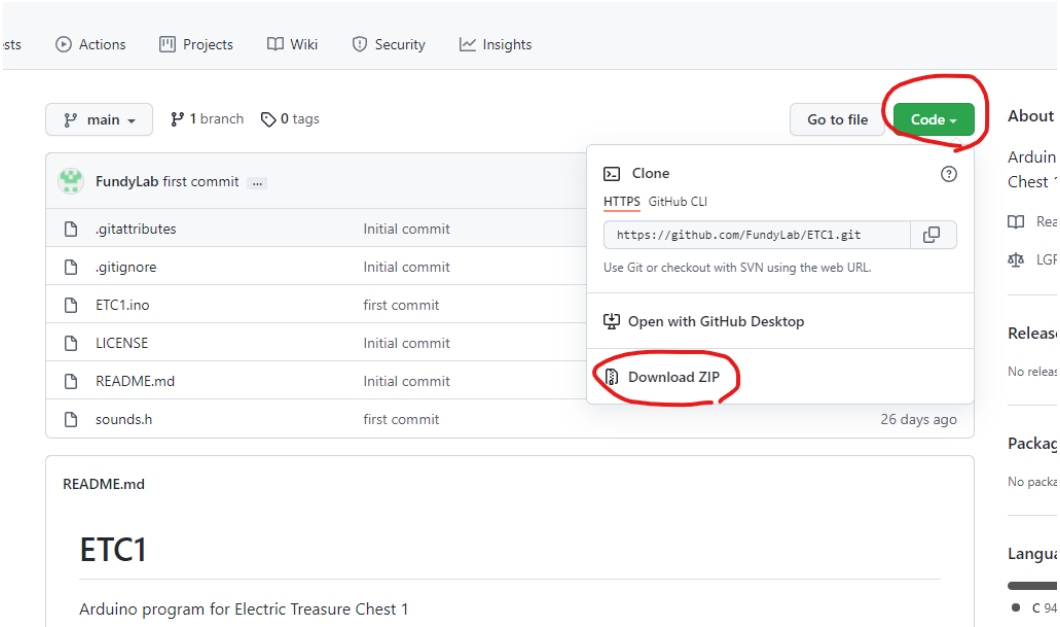
3.　Arduino のインストール

Google chrome 等のブラウザで、「arduino ide　インストール」と検索して、好きな記事を読んで使える状態にしておいて下さい。

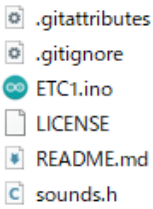
4.　ソースコードのダウンロード

ブラウザで下記 GitHub サイトへアクセスします。
https://github.com/FundyLab/ETC1

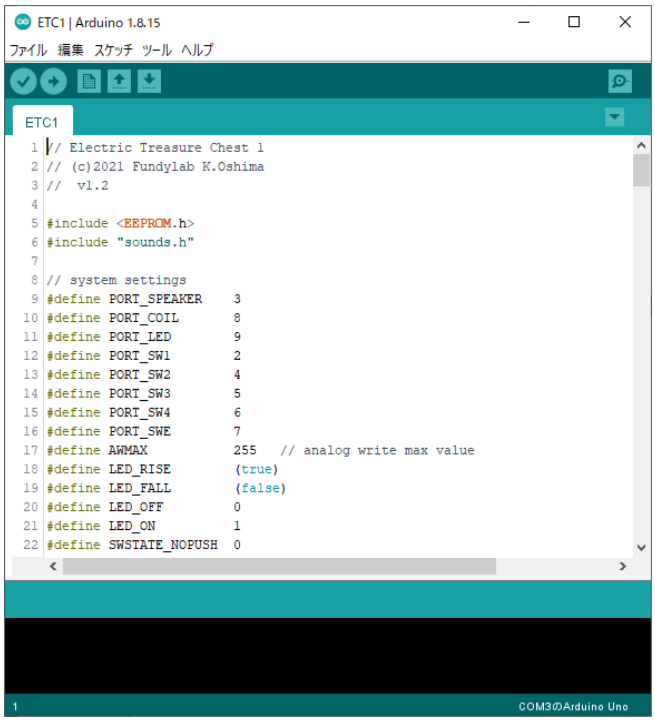
下図の赤丸部分右上の Code をクリック、Download ZIP をクリックしてソースコードを PC にダウンロードします。



ダウンロードした zip ファイルを任意のフォルダーで展開すると、2 階層下に下記ファイルが入っているので、そのフォルダ名を ETC1-main から ETC1 に変更し、ETC1.ino をダブルクリック。



すると Arduino IDE が起動します。

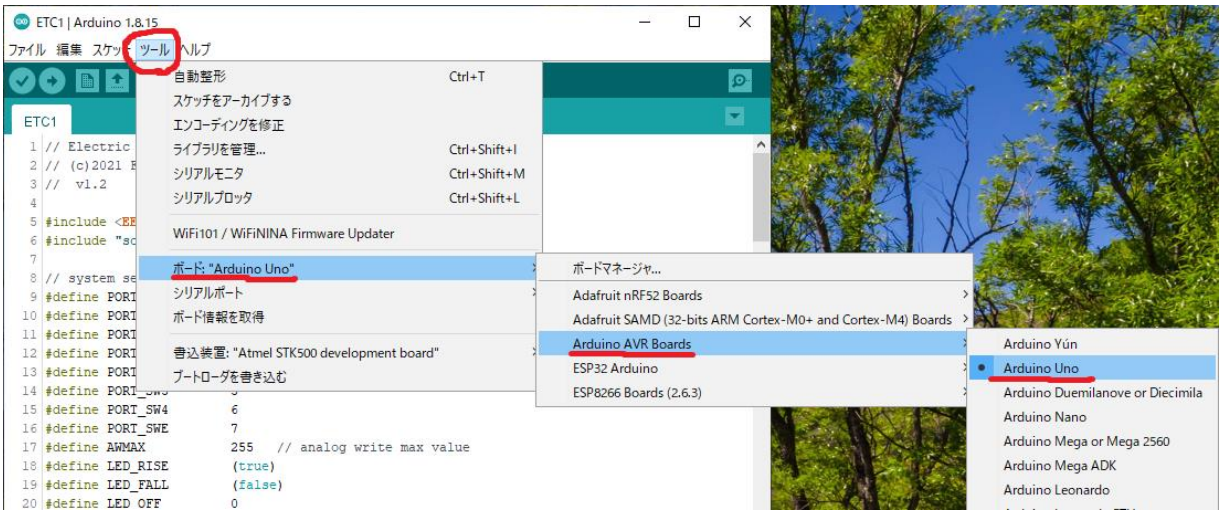


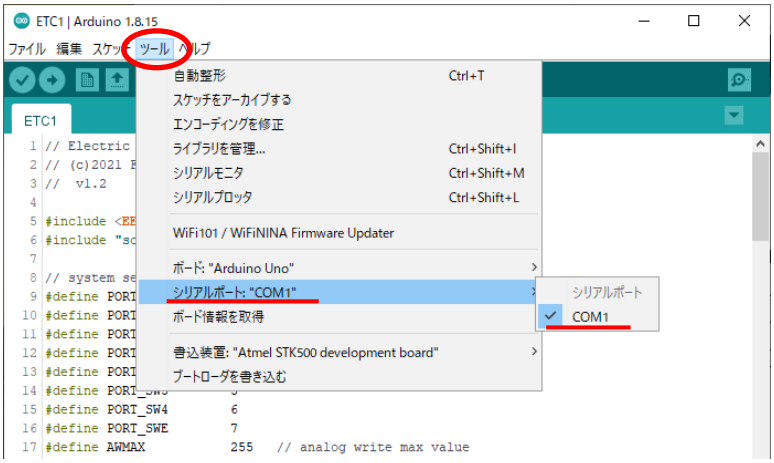
5. 書き込み方法

プログラミングする前に、4.でダウンロードしたソースコードがうまく書き込めるか先に確認しておきます。

下図の通り Arduino IDE の、ツール → ボード → Arduino AVR Boards → Arduino UNO を選択します。

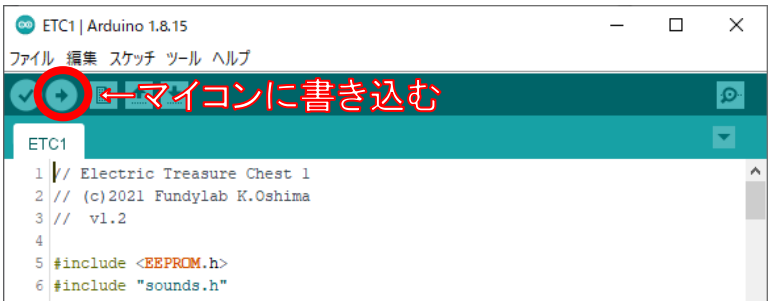
次に、その下の図の通り、ツール → シリアルポート で対応するシリアルポートを選択します。(図のシリアルポートの表示名は OS 環境やボードによって異なります。) シリアルポートに選択可能なものが無い場合は PC が USB シリアル変換器を認識できていない可能性があります。接続やドライバーなどをご確認下さい。





上記設定ができたら、宝箱の電源スイッチを下側（OFF 側）にして、PC から電源が供給されるようにします。

最後に、下図の「マイコンに書き込む」ボタンをクリックします。すると画面の下の方に文字が流れてきて、コンパイルが始まります。エラーがなければ橙色の文字が流れてきてそのままマイコンにコンパイルされたプログラムが書き込まれます。



6. プログラミング

6.1. パスワードの変更

6.1.1. 本体のみでパスワードを変更する方法

宝箱組み立て動画の最後の方にもありますが、USB シリアル変換器を繋がずに、電源が OFF の状態で Enter キーを押しながら電源を ON すると、パスワード変更モードに入ります。古いパスワードを入れて Enter、次に新しいパスワードを入れて Enter キーを押すと、新しいパスワードの保存が完了します。この方法は、パスワードの最大長さは 4 桁までに制限されます。

6.1.2. プログラミングでパスワードの最大桁数を増やす方法

まずソースコードの下図で、赤い下線部分の「4」は 4 桁という意味のため、パスワードの最大桁数を 8 桁にしたい場合、(4+1)の部分を(8+1)に書き換えます。

```
31 #define TIMLEDFALL      ((int)(3.92 * 100))  // rise time of led 100[ms]
32 // password length
33 #define PASWDMAX        (4+1)    // Maximum number of digits in password and Er
34
35 ////////////////////// Variable init
```

次に左下図の部分に初期パスワードを追加します。pwd[0]～pwd[3]の 4 行がパスワード書き込み値代入行です。5 行目の pwd[4]はパスワードの終わりを意味する行となります。これを右下の様に書き換えます。先程、パスワードの最大桁数を 8 桁にしたので、pwd[0]～pwd[8]の 9 行必要となります。「=」の右側には、SW1～SW4 のいずれかを入力してパスワードを作って下さい。最後の pwd[8] = の右側だけは、パスワードの終わりを意味する行にするため SWE と入力して下さい。

本プログラムは C 言語です。式の最後には「;」セミコロンを入力します。もちろん、全て半角で記載しないとエラーになります。特に、間違って全角のスペースをどこかに入力してしまうと、エラー箇所がわからなくなってしまうので注意が必要です。

```
58 void test_write_paswd(){
59   pwd[0] = SW4;
60   pwd[1] = SW1;
61   pwd[2] = SW2;
62   pwd[3] = SW3;
63   pwd[4] = SWE;
64   write_password_toEE();
65 }
```

```
58 void test_write_paswd(){
59   pwd[0] = SW4;
60   pwd[1] = SW1;
61   pwd[2] = SW2;
62   pwd[3] = SW3;
63   pwd[4] = SW1;
64   pwd[5] = SW3;
65   pwd[6] = SW2;
66   pwd[7] = SW4;
67   pwd[8] = SWE;
68   write_password_toEE();
69 }
```

最後に下図の行で、コメント行を示す赤下線部分の 2 個のスラッシュ「//」のみ一旦削除します。

```
242   // initial process
243   //test_write_paswd(); //test
244   read_password_fromEE();
245   if(!init_checkbutton()){
246     play_and_flash(SOUND_INIT);
247   }
248   Serial.println("Initialize done!");
249 }
```

5 章の通り「マイコンに書き込む」ボタンを押して一度書き込みます。エラーなく書き込めたら、一度 USB シリアル変換器を外して、電源を ON して下さい。すると起動時に 8 桁のパスワードが自動的にマイコン内部の EEPROM エリアに書き込まれ、そのパスワードでしか開かなくなりますので確認して下さい。

その状態で、先程上記 2 個のスラッシュ「//」を削除しましたが、もう一度 2 個のスラッシュを入力して、再度 USB シリアル変換器を取り付け、5 章の通り「マイコンに書き込む」ボタンを押して書き込みます。そうすることで、8 桁まで、本体だけでパスワードが変更できるようになります。

6.2. LED 点灯タイミングの変更

下図の赤下線の部分の数字を変えると、LED の点灯時間を変更できます。変更した後、USB シリアル変換器を取り付け、5 章の通り「マイコンに書き込む」ボタンを押して書き込みます。

```
27 #define NOOAF           5    // Number Of Open And Flash
28 // flash time
29 #define TIMLEDINIT      ((int)(3.92 * 500))  // rise time of led 500[ms]
30 #define TIMLEDRISE      ((int)(3.92 * 250))  // rise time of led 250[ms]
31 #define TIMLEDFALL      ((int)(3.92 * 100))  // rise time of led 100[ms]
32 // password length
33 #define PASWDMAX        (4+1)    // Maximum number of digits in password and E
```

6.3. サウンドの変更

ETC1.ino ファイルと同じフォルダに、sounds.h を入れたと思いますが、そのファイルに音のデータが入っています。

音データのフォーマットは、8kHz、モノラル、RAW です。ATMEGA328P マイコンは Flash 領域が 32kBytes のため、音データ全体で最大約 4 秒弱程度記録できます。

書き換えるための音データの作成方法は複雑ですので、Google 検索等で「arduino 単体で音を鳴らす」で検索してみてください。Audacity のフリーソフトや xxv コマンドを使って作成できます。xxv コマンドでそのまま使えるデ

ータに変換できなかった場合、Excel でフォーマットを修正し、下図の通り 16 進数ピリオド区切りのデータにします。
なお、データの最後に 0xff の値を追加して下さい。データの終わりを示します。

```
 8   const unsigned char sound_init[] PROGMEM = {
 9   0x7f, 0x7f, 0x80, 0x7f, 0x80, 0x7f, 0x7f, 0x7e, 0x7d, 0x7d, 0x
10   0x5e, 0x41, 0x68, 0x76, 0x58, 0x94, 0xe2, 0xa9, 0x80, 0x79, 0x
11   0x2b, 0x42, 0x65, 0x58, 0x8c, 0xec, 0xbe, 0x78, 0x61, 0x67, 0x
12   0x4d, 0x62, 0x48, 0x5d, 0xd5, 0xc2, 0x86, 0x80, 0x75, 0x6d, 0x
13   0x66, 0x65, 0x5a, 0xbe, 0xc8, 0x8c, 0x72, 0x6a, 0x7c, 0x8a, 0x
```