

# AIミニ四駆マニュアル

## 目次

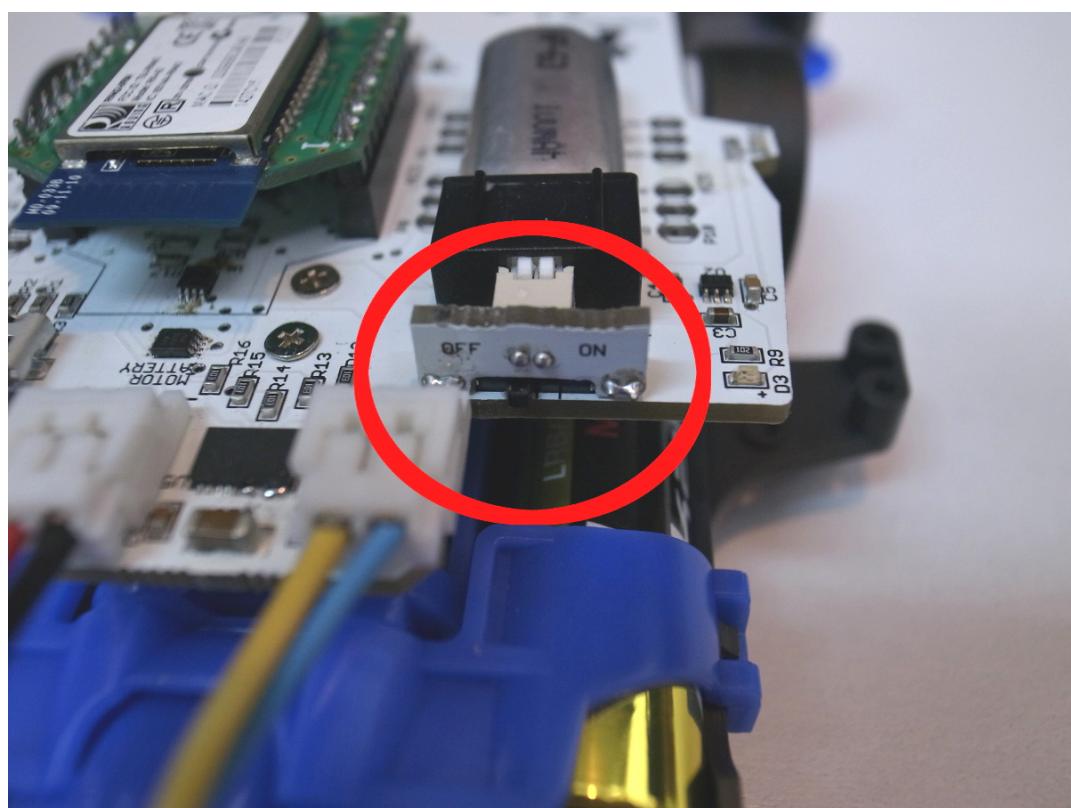
- AIミニ四駆の使い方
- processingサンプルプログラムの使い方
- Androidアプリの使い方

## AIミニ四駆の使い方

1. AIミニ四駆の入出力装置
2. AIミニ四駆の起動方法
3. AIミニ四駆の起動モード

### 1. AIミニ四駆の入出力装置

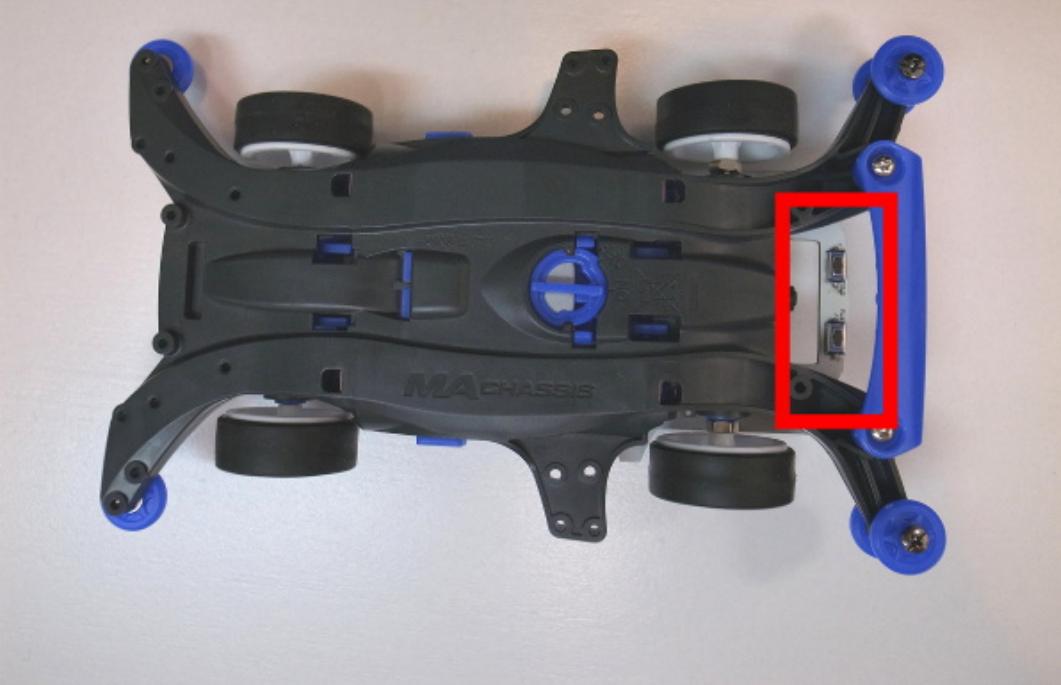
1. CPU用電源スイッチ (Fig.1-1)
2. モーター用電源スイッチ (Fig.1-2)
3. プッシュスイッチ×2 (Fig.1-3)(Fig.1-4)



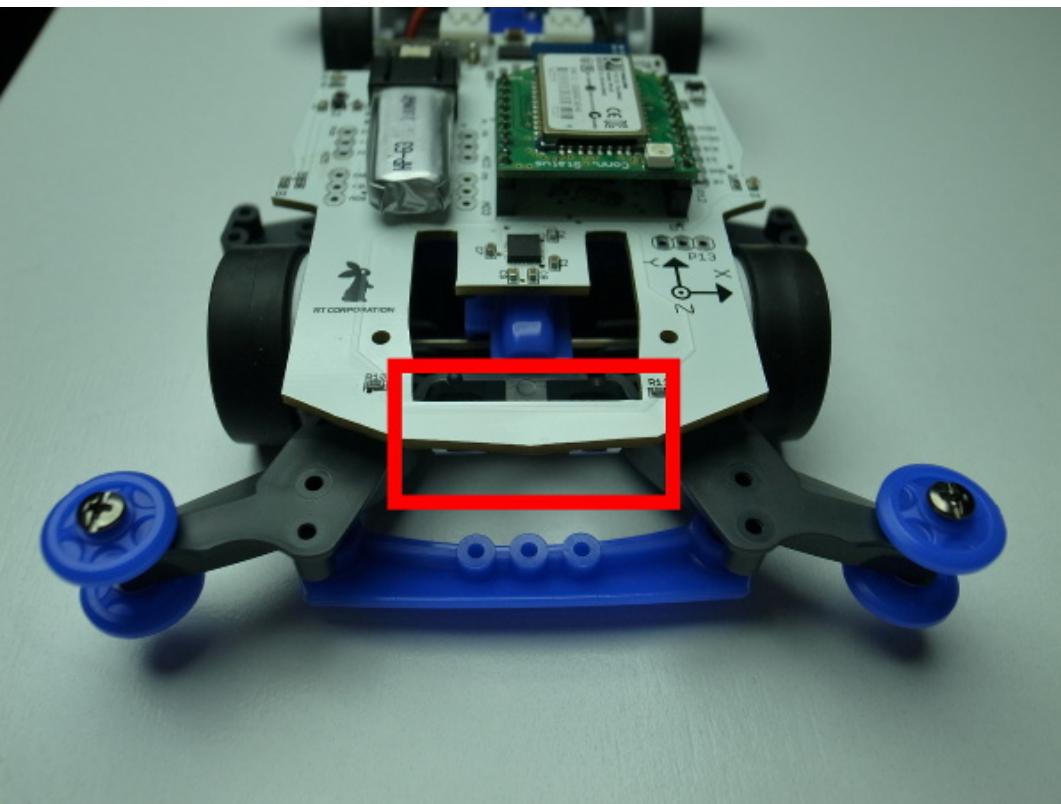
(Fig.1-1 CPU用電源スイッチ)



(Fig.1-2 モーター用電源スイッチ)



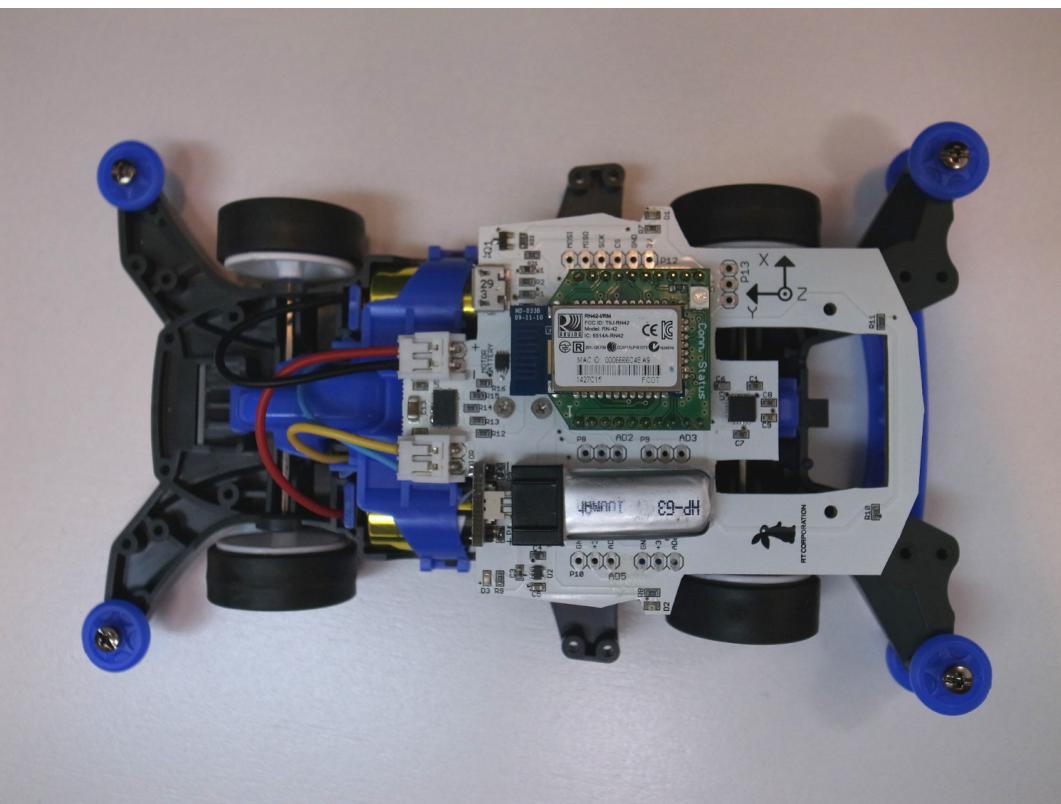
(Fig.1-3 プッシュスイッチ場所 (裏から) )



(Fig.1-4 プッシュスイッチ場所 (表から) )

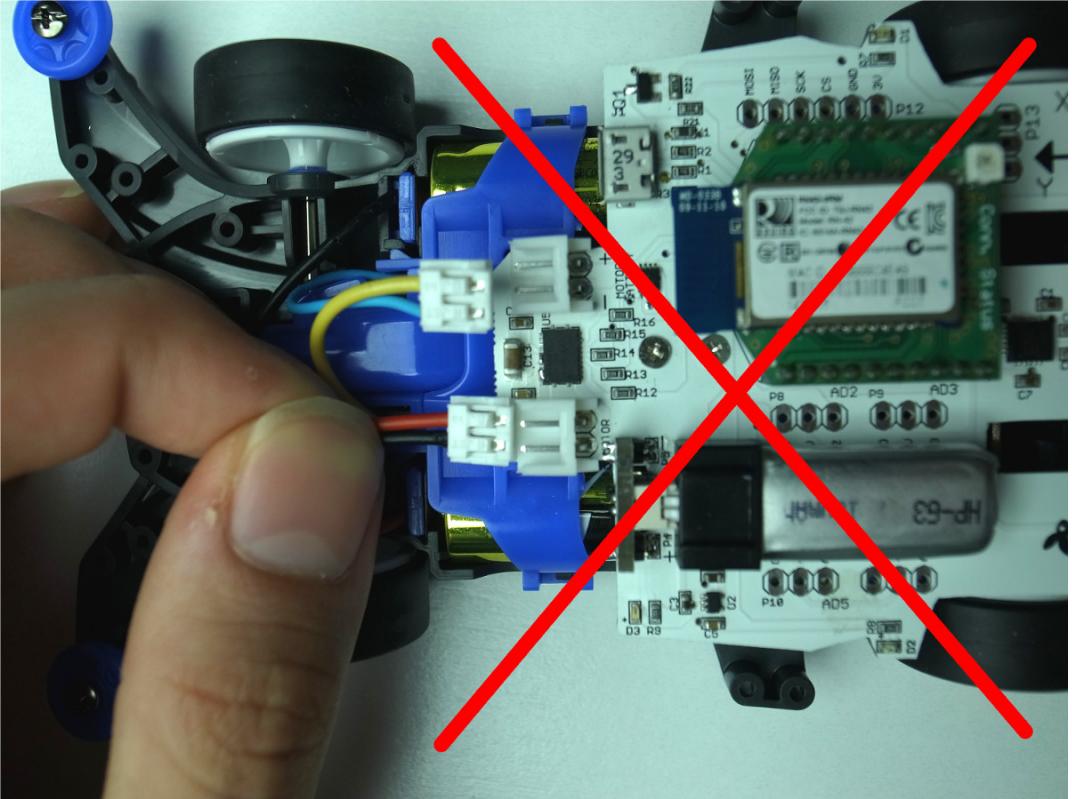
## 2. AIミニ四駆の起動方法

はじめに、AIミニ四駆の配線が図の通りになっているか確認をしてください。(Fig.1-5)



(Fig.1-5 通常の配線)

図の向きにおいてた時に、**右側が黒と白のケーブルが接続されていて、左側が青と黄色のケーブルが接続されているのが正しい配線です。図のように配線をすると壊れるので絶対に逆に配線しないでください。**(Fig.1-6)



(Fig.1-6 誤った配線)

AIミニ四駆には電源スイッチが二箇所あります。

一つが車体上部にあるCPU用電源スイッチです。

こちらはLiPoバッテリーの正面にスライドスイッチがあるので、それを「ON」または「+」側へスライドさせてください。

もう一つが、車体下部にある、モーター用電源スイッチです。

こちらは車体の裏側についています。「ON」の角度までスイッチをひねってください。

この二つの電源をつけることで、AIミニ四駆は起動します。

### 3. AIミニ四駆の起動モード

AIミニ四駆には（デフォルトで）三つの起動モードが存在します。これらはCPU用電源スイッチを起動する際に切り替えることが出来ます。

切り替えは車体後部についている、プッシュスイッチを用いて行います。

#### 1. DEMOモード

このモードは何もせずにCPU用電源スイッチを入れた時に起動します。

このモードは基本的に使用しないので説明を省きます。

#### 1. 通信モード

このモードは、車体後部左側にあるプッシュスイッチを押しながら、CPU用電源スイッチを入れた時に起動します。このモードはAIミニ四駆からデータが常に送信され、またすべてのコマンドを受け取り、動作するモードです。

各サンプルプログラムを使用する際には、このモードで起動してください。これ以外のモードを使用すると、一部機能が正常に動作しない可能性があります。

また、起動時はプッシュスイッチを緑色のLEDが二回点滅するまで必ず押し続けてください。

途中で手を離してしまうと、LEDテストモードになる可能性があるので、ご注意下さい。

#### 2. モーターテストモード

このモードは、車体後部右側にあるプッシュスイッチを押しながら、CPU用電源スイッチを入れた時に起動します。このモードで起動すると、モーターが回転します。モーターが正常に動作するかどうかのテストなどにご使用ください。

モーターは車体下部にあるスイッチをONにしないと動作しないので。モーターが回らない場合はスイッチを確認してください。

注) このマイコンのファーム書き換えにも使用します。

## processingサンプルプログラムの使い方

1. Processing の導入
2. サンプルプログラムのダウンロード
3. サンプルプログラムの実行
4. サンプルプログラムの解説

### 1. Processing の導入

Processing のダウンロードページからProcessingをダウンロードしてください。ダウンロードの際には、「No Donation」を選択したのちに、「Download」をクリックしてください。(Fig.2-1)(Fig.2-2)

[Cover](#)  
[Download](#)  
[Exhibition](#)  
[Reference](#)  
[Libraries](#)  
[Tools](#)  
[Environment](#)  
[Tutorials](#)  
[Examples](#)  
[Books](#)  
[Handbook](#)  
[Overview](#)  
[People](#)  
[Shop](#)

**Download Processing.** Please consider making a donation to the Processing Foundation before downloading the software.

Processing is open source, free software. All donations fund the [Processing Foundation](#), a nonprofit organization devoted to advancing the role of programming within the visual arts through developing Processing.

No Donation    \$10    \$25    \$50    \$100    \$ [ ]

ここをクリックしてください

[Donate & Download](#)

(Fig.2-1 ダウンロード画面1)



[Cover](#)  
[Download](#)  
[Exhibition](#)  
[Reference](#)  
[Libraries](#)  
[Tools](#)  
[Environment](#)  
[Tutorials](#)  
[Examples](#)  
[Books](#)  
[Handbook](#)  
[Overview](#)  
[People](#)  
[Shop](#)

**Download Processing.** Please consider making a donation to the Processing Foundation before downloading the software.

Processing is open source, free software. All donations fund the [Processing Foundation](#), a nonprofit organization devoted to advancing the role of programming within the visual arts through developing Processing.

No Donation    \$10    \$25    \$50    \$100    \$ [ ]

[Download](#)

クリックしてダウンロードページに移動

(Fig.2-2 ダウンロード画面2)

ダウンロード画面へ遷移すると、画面の一番上に最新のバージョンのダウンロード項目が表示されます。最新版を利用する場合には、この項目のうち、お使いのOSが表記されているのリンクを選択してください。(Fig.2-3)



[Cover](#)  
[Download](#)  
[Exhibition](#)  
[Reference](#)  
[Libraries](#)  
[Tools](#)  
[Environment](#)  
[Tutorials](#)  
[Examples](#)  
[Books](#)  
[Handbook](#)  
[Overview](#)  
[People](#)  
[Shop](#)

**Download Processing.** Processing is available for Linux, Mac OS X, and Windows. Select your choice to download the software below.



3.0 beta 4 (17 August 2015)

|                                |                              |                          |
|--------------------------------|------------------------------|--------------------------|
| <a href="#">Windows 64-bit</a> | <a href="#">Linux 64-bit</a> | <a href="#">Mac OS X</a> |
| <a href="#">Windows 32-bit</a> | <a href="#">Linux 32-bit</a> |                          |

お使いのOSを選択してください

Read about the [changes in 3.0](#). The [list of revisions](#) covers the differences between releases in detail.

[» Github](#)  
[» Report Bugs](#)  
[» Wiki](#)  
[» Supported Platforms](#)

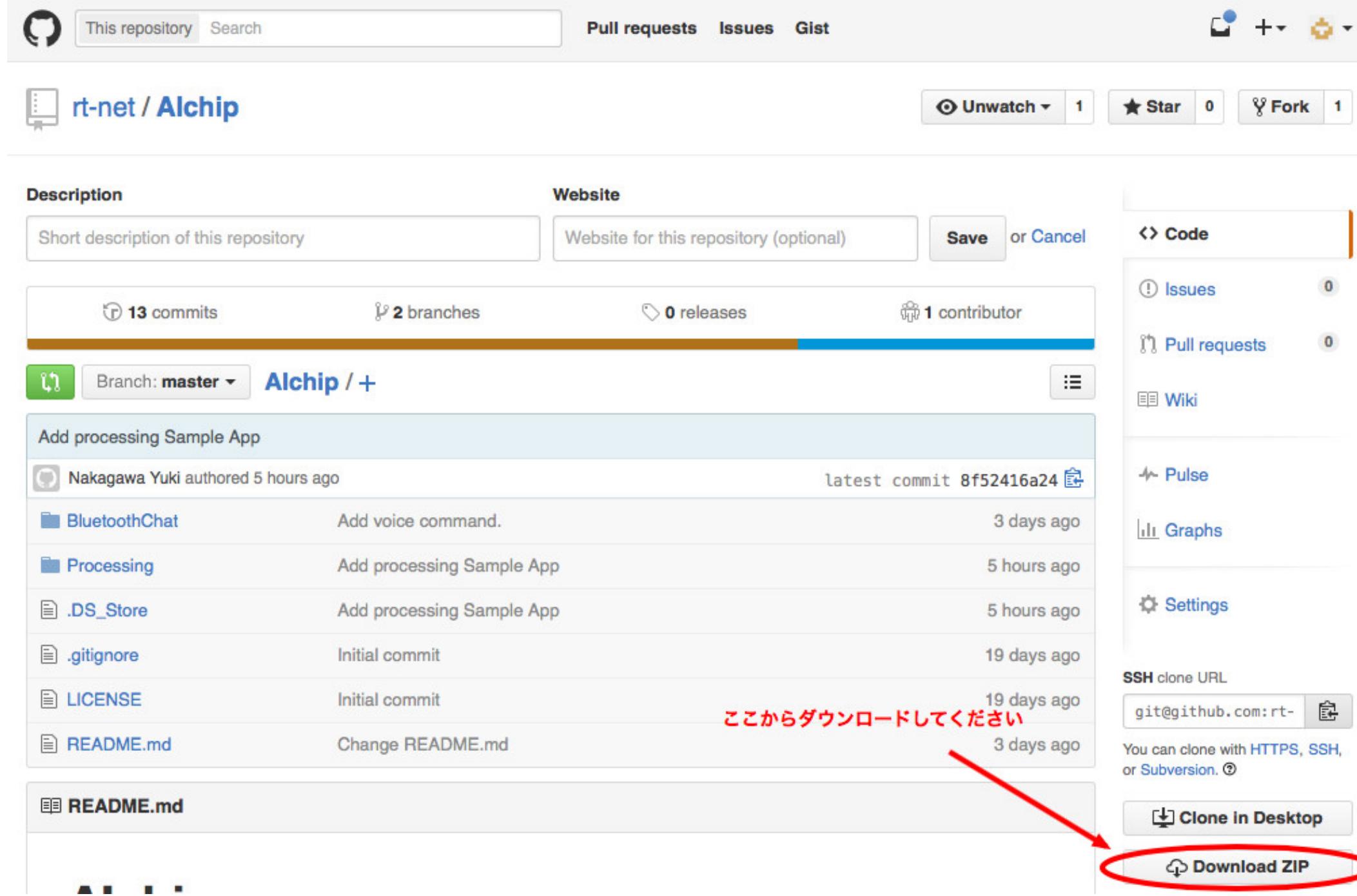
(Fig.2-3 ダウンロード画面3)

過去のバージョンをダウンロードしたい場合は、画面中部にある「Stable Release」の項目からダウンロードしてください。こちらでダウンロードする場合も、お使いのOSが表記されているリンクを選択してダウンロードしてください。  
 現在サンプルアプリケーションの動作が確認できているバージョンは **3.0 beta 4** 及び **2.2.1**です。Processingをダウンロードする際にはバージョンにご注意下さい。ダウンロードができたら、ソフトのインストールを行ってください。

## 2. サンプルプログラムのダウンロード

ProcessingのサンプルプログラムはGitHubの[AI-CHIPのページ](#)から行ってください。

ダウンロードする際には、画面右側の「Download ZIP」をクリックしてください。(Fig.2-4)



(Fig.2-4 githubのページ)

ZIPファイルがダウンロード出来たら、ダウンロードファイルを任意の場所に解凍してください。

## 3. サンプルプログラムの実行

まず、1. Processing の導入でインストールしたProcessingを起動してください。

processingを起動したら、2. サンプルプログラムのダウンロードで解凍したフォルダ内の「Processing」→「sample\_App」内にある「sample\_App.pde」を開いてください。

開いたら、パソコンとAIミニ四駆のBluetoothのペアリングと通信ポートの設定を行ってください。(Fig.2-5)



(Fig.2-5 サンプルプログラムのコード)

- Windows

コントロールパネルを開いて、ハードウェアとサウンドにある項目の「デバイスの追加」を選択します。(Fig.2-6)

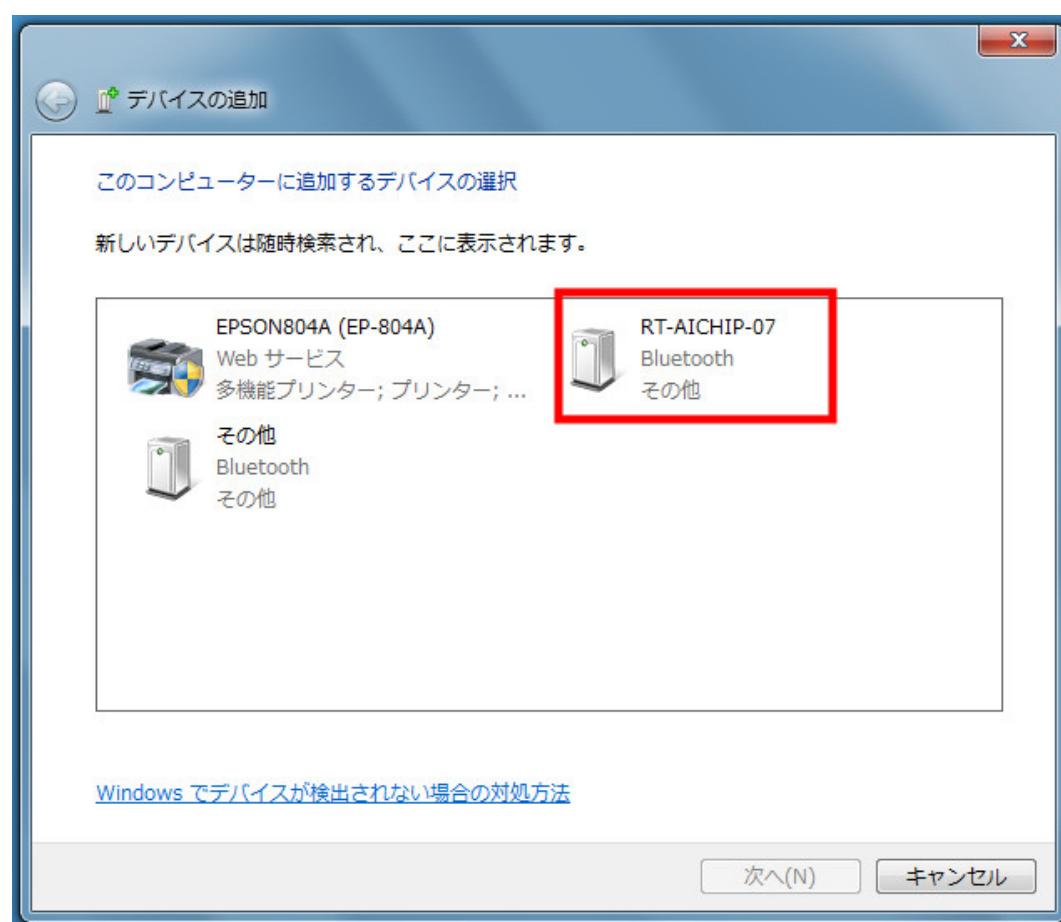


(Fig.2-6 コントロールパネルの画面)

デバイスの追加の画面が表示されたら、AIミニ四駆を通信モードで起動します。

起動すると、通常Bluetoothモジュールが赤く点滅します。

パソコン側で認識すると、「実際の表示名は何？」が表示されるので、選択して「次へ」を押してください。この時選択した名前は、後ほど使うので覚えておいてください。(Fig.2-7)



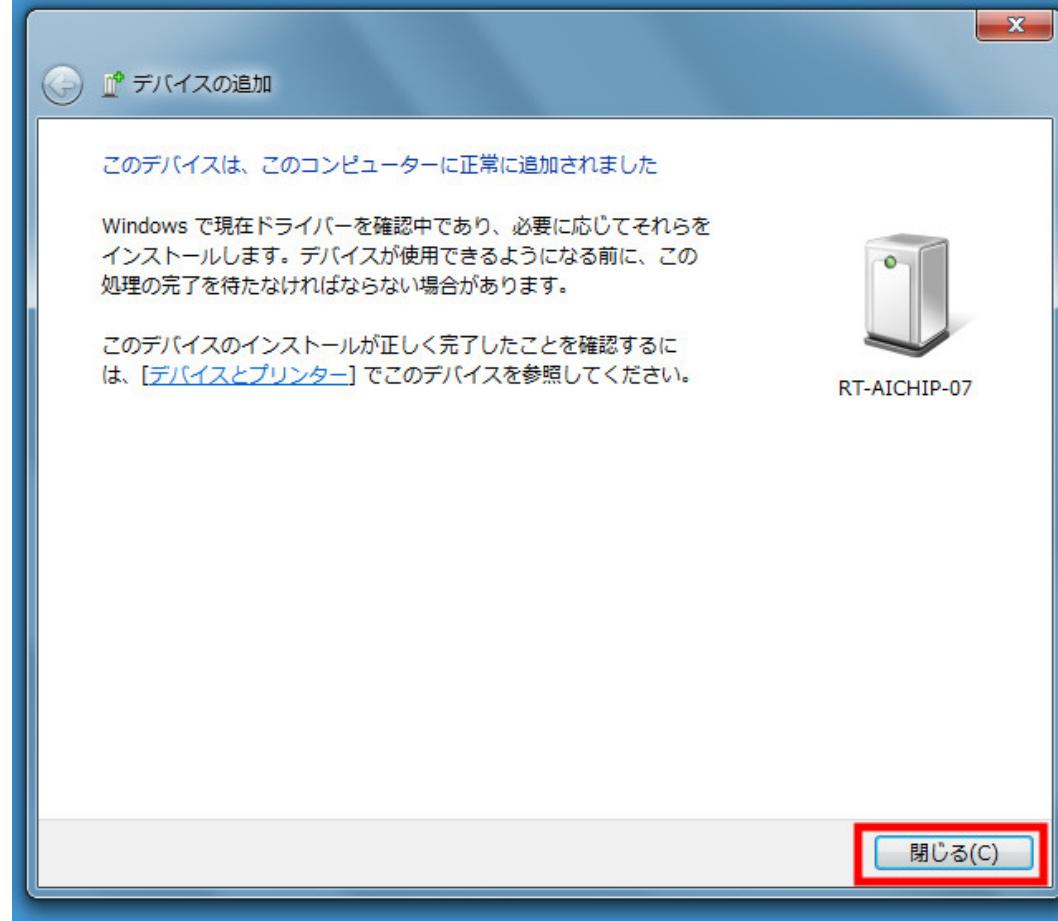
(Fig.2-7 デバイスの選択画面)

次に、ペアリングコードの比較のウィンドウが表示されるので、「次へ」を押してください。(Fig.2-8)



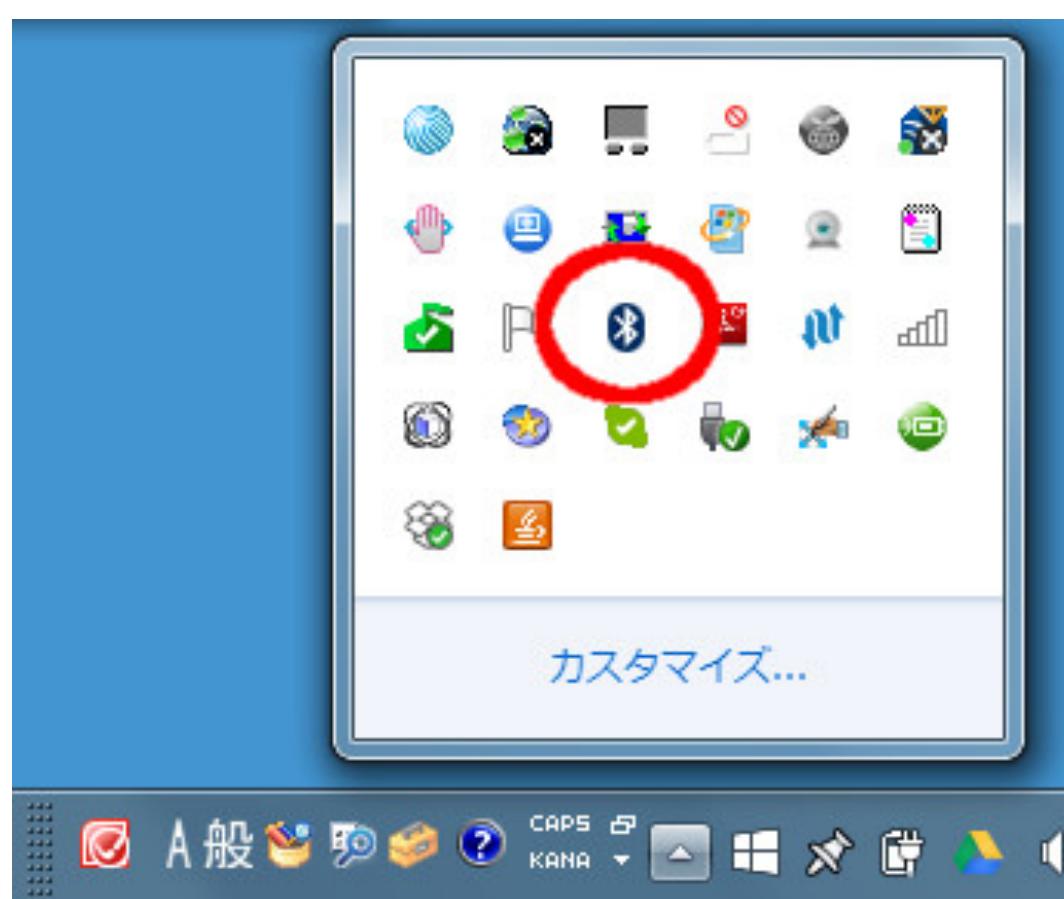
(Fig.2-8 ペアリングコード比較画面)

最後に次のような画面が表示されると接続が完了します。(Fig.2-9)



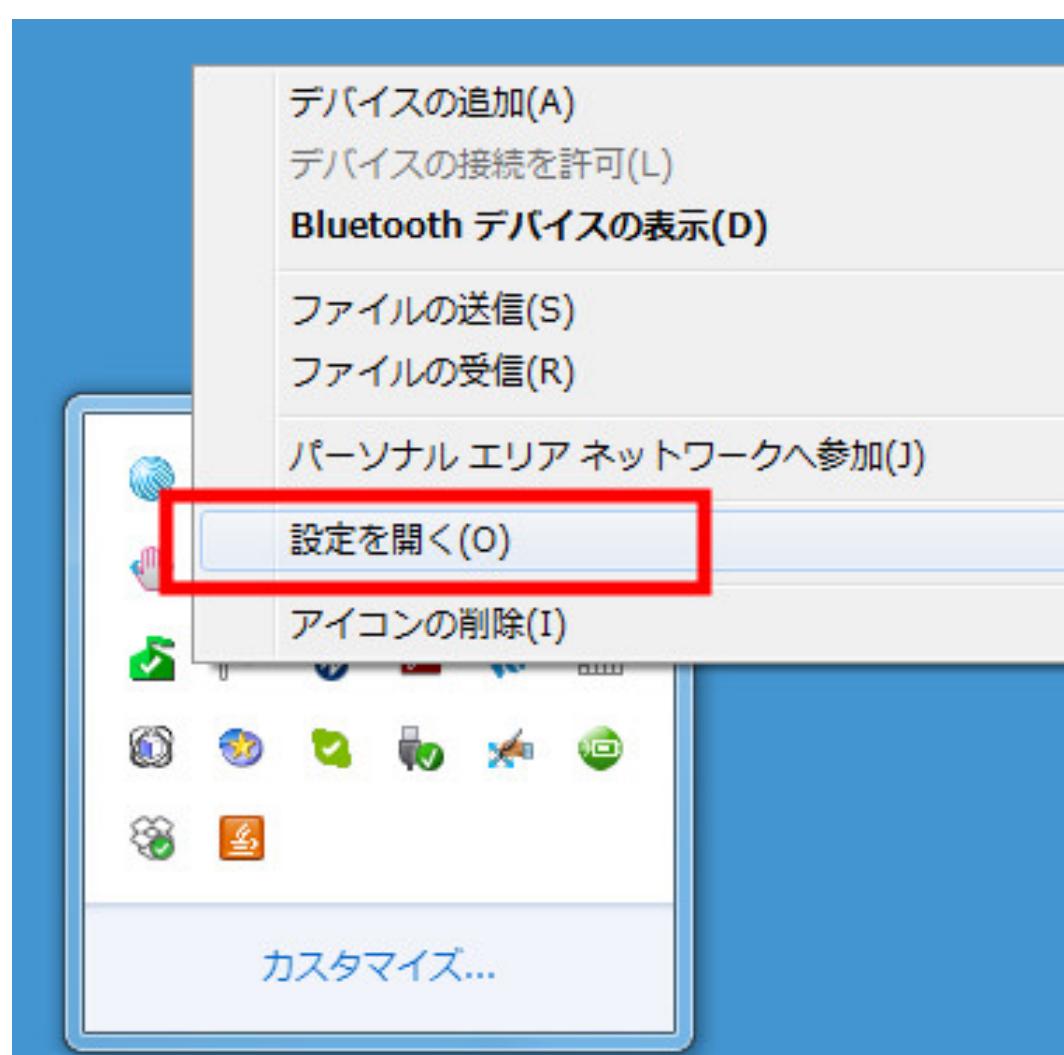
(Fig.2-9 接続完了画面)

接続が完了したら、接続されたポートの確認を行います。ポートの確認は、Bluetoothの設定画面から行います。まず、画面右下にあるタスクトレイの中の、Bluetoothアイコンを右クリックしてください。(Fig.2-10)



(Fig.2-10 Bluetoothアイコンの選択)

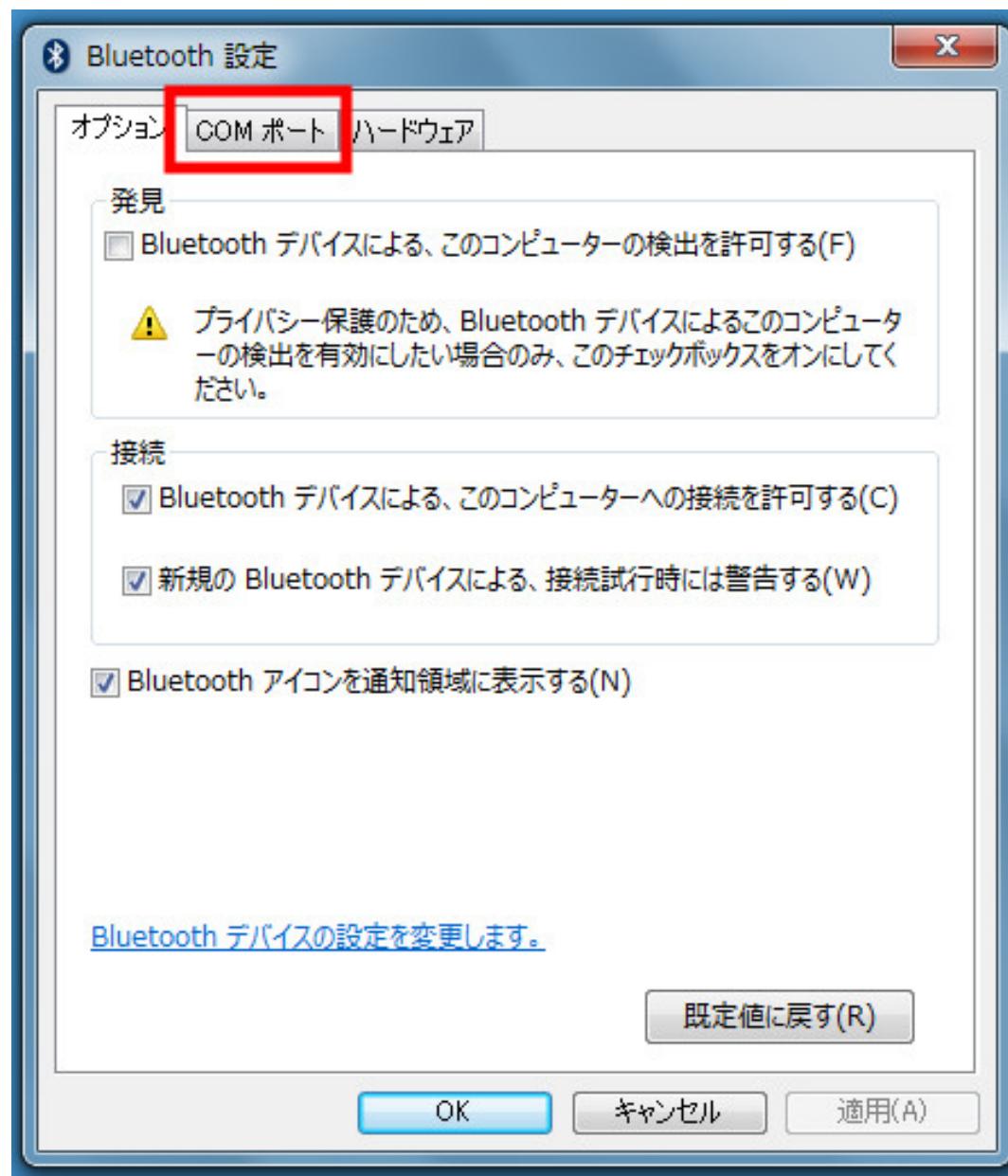
右クリックをするとメニューが開かれるので「設定を開く」を選択してください。(Fig.2-11)



(Fig.2-11 Bluetoothアイコンのメニュー)

「設定を開く」をクリックすると、次のような「Bluetooth」の設定画面が開かれます。開かれたら、タブの「COMポー

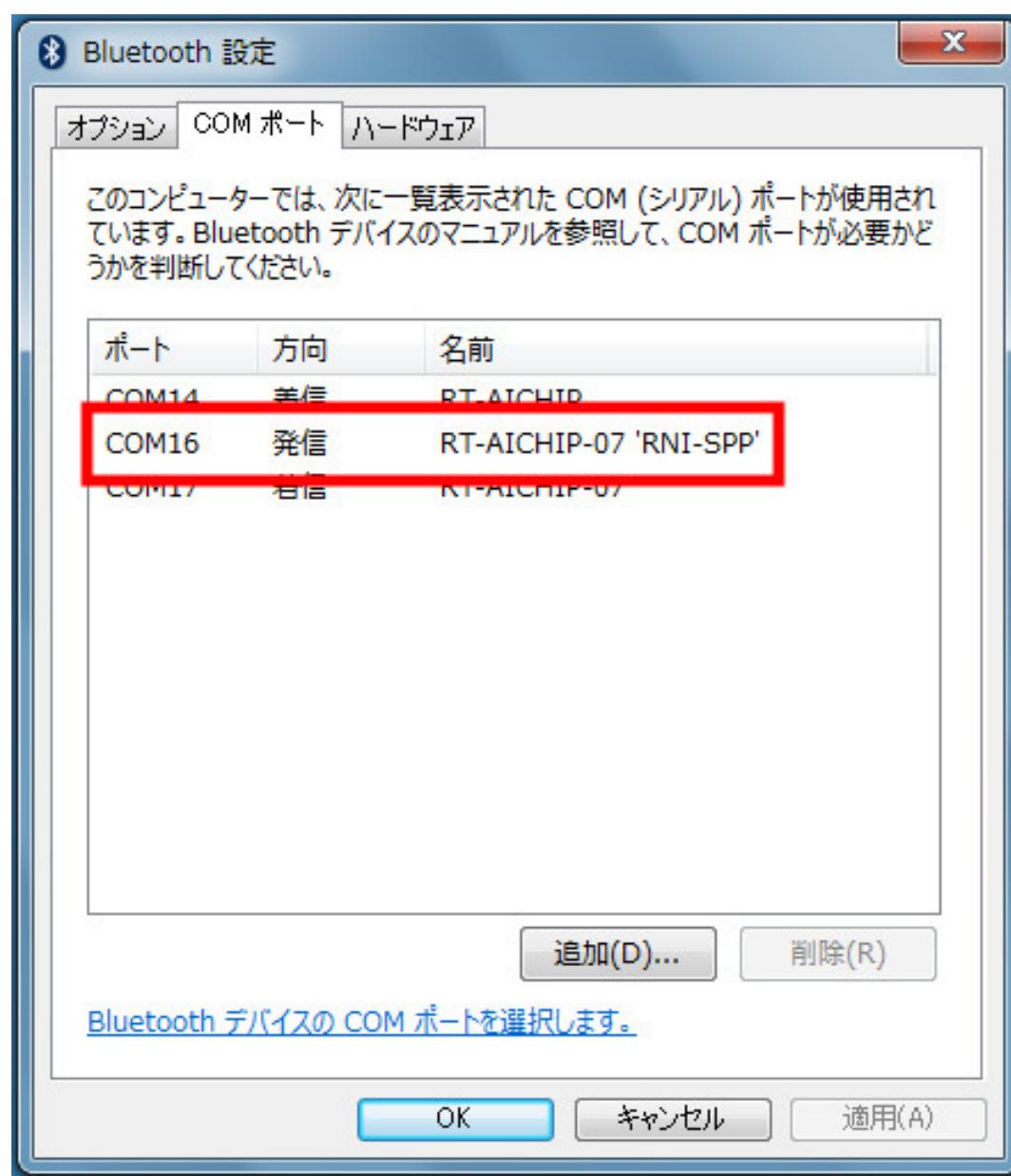
ト」をクリックしてください。(Fig.2-12)



(Fig.2-12 Bluetoothの設定画面)

「COMポート」をクリックすると、パソコンに接続されたBluetooth機器の一覧が表示されます。この中の、先ほど接続した名前と「'RNI-SPP」が書かれた欄を確認してください。

方向が「発信」となっているCOMポートを確認してください。(Fig.2-13)



(Fig.2-13 COMポート確認画面)

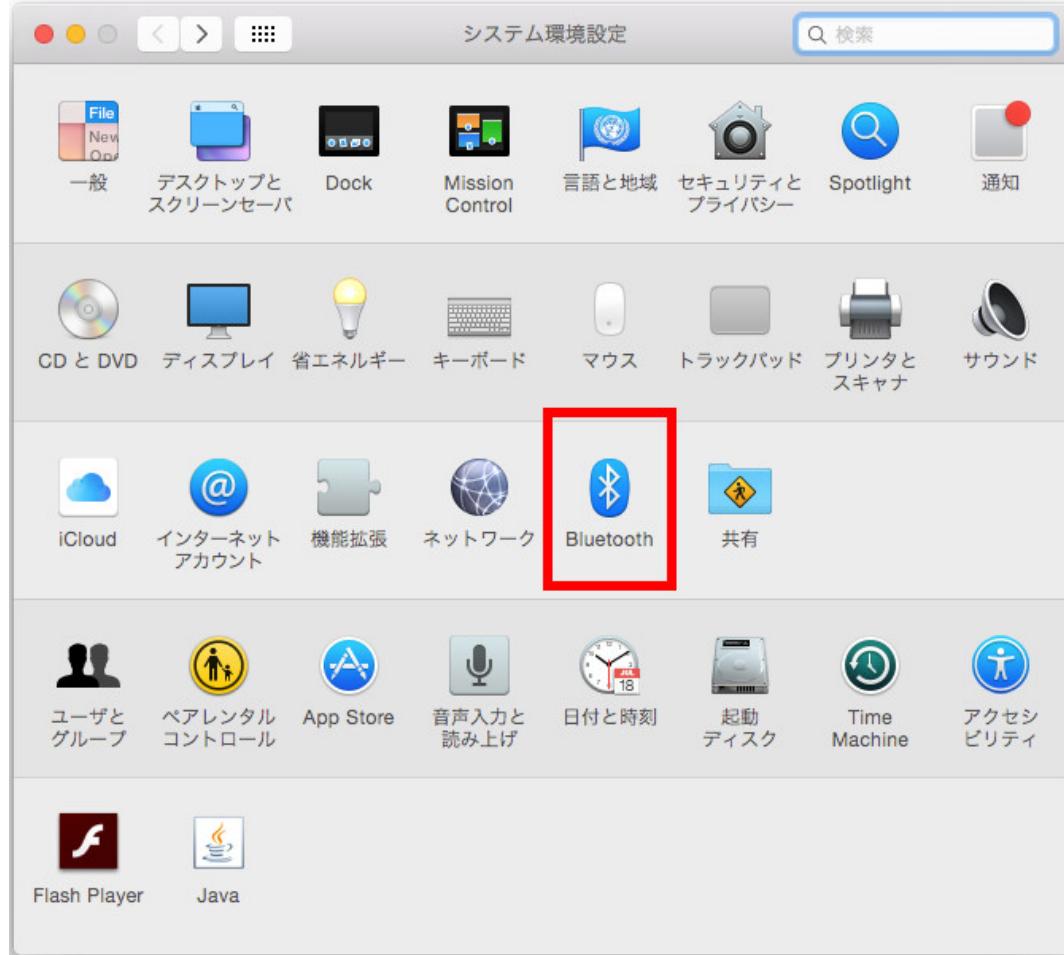
確認したポートはプログラムの14行目に書き込みます。デフォルトで「"COM5"」となっている部分を自分で確認したCOMポートに書き換えてください。

デフォルトでは14行目がコメントアウト（プログラムに反映されない状態に）されています。

14行目の先頭の「//」を削除し、13行目の先頭に「//」を書き足してください。

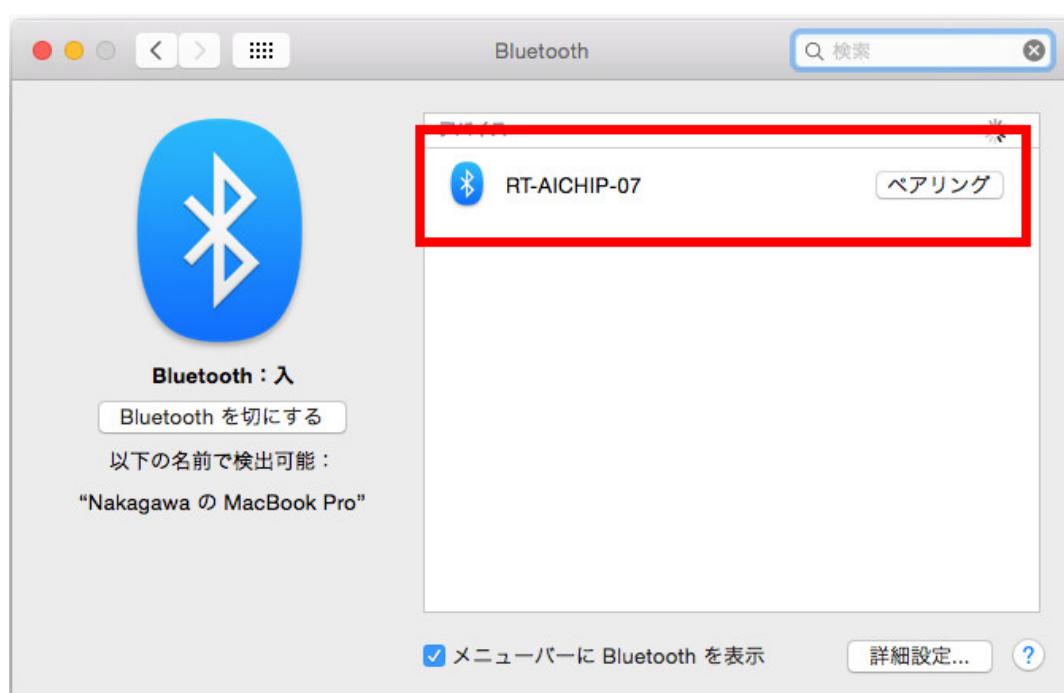
- Mac

システム環境設定のBluetoothを選択します。次に、AIミニ四駆の電源を、通信モードでつけます。(Fig.2-14)



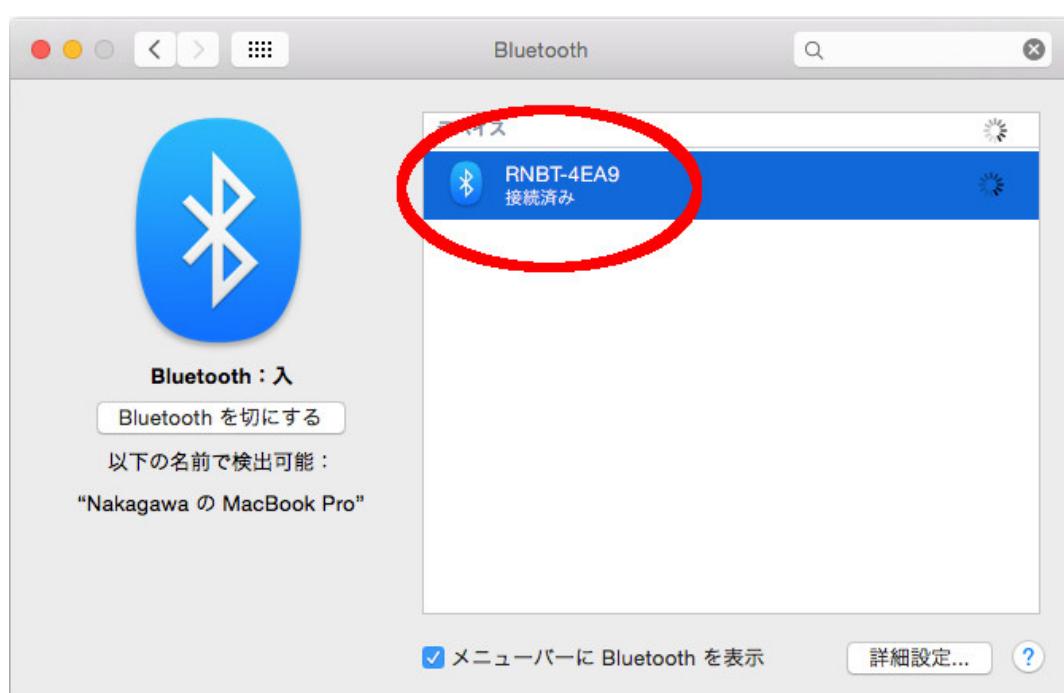
(Fig.2-14 システム環境設定画面)

その後、システム環境設定ウィンドウ内に表れるRNBT-xxxxを選択し、ペアリングを行います。(Fig.2-15)



(Fig.2-15 Buletooth接続画面)

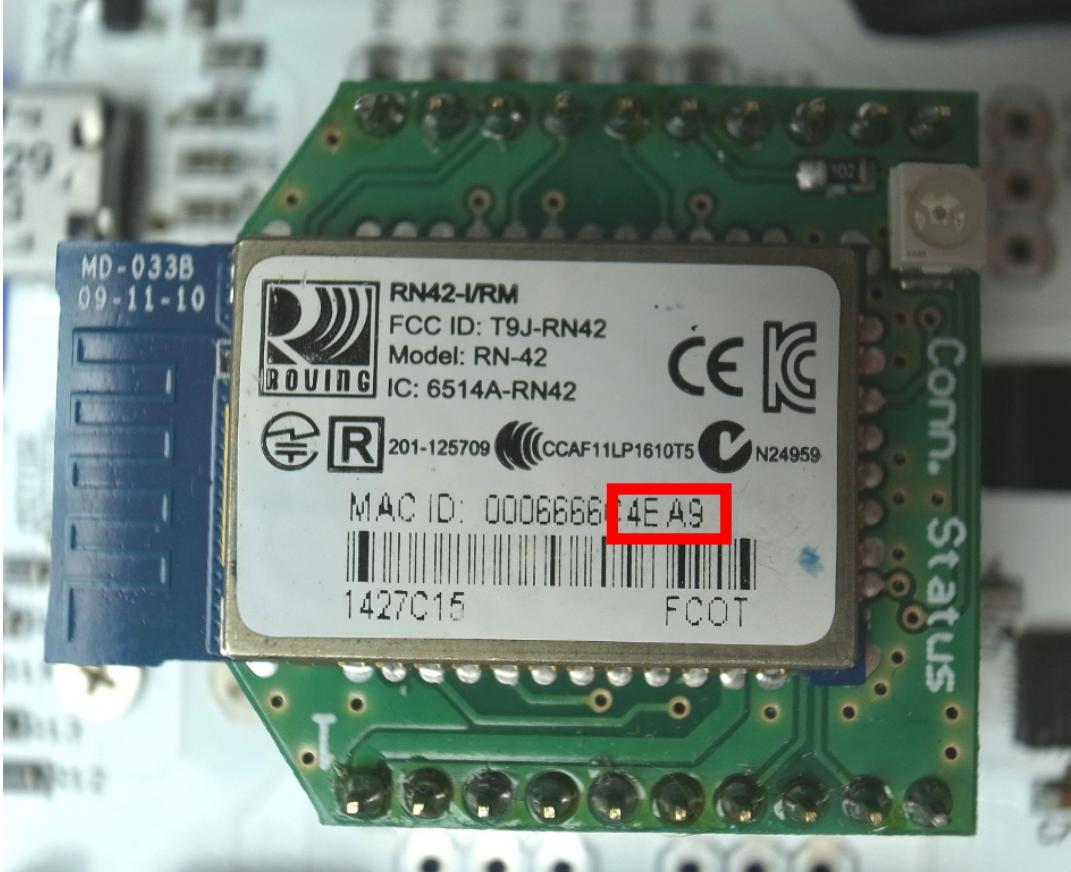
接続が完了すると、「接続済み」の表記になります。(Fig.2-16)



(Fig.2-16 Bluetooth接続確認)

この「xxxx」の部分は、BluetoothモジュールのMAC IDの下四桁になります。

各自確認して、プログラムを変更してください。(Fig.2-17)



(Fig.2-17 BluetoothモジュールのID確認)

ウィンドウのデバイス欄に表示されたデバイス名が違う場合は、13行目の設定を書き換えてください  
"/dev/tty.RNBT-xxxx-RNI-SPP"

ペアリングと通信ポートの設定出来たら、processingのプログラムを起動させてください。 (Fig.2-18)



(Fig.2-18 processingの実行ボタン)

起動するとウィンドウが一つ立ち上がります。ウィンドウ上でマウスクリックをすると、AIミニ四駆の緑のLEDが点灯します。また、エディタ下部にはログ出力が表示されます。ここにはAIミニ四駆からのデータが出力されます。データの詳細は以下をご覧ください。(Table.2-1)(Table.2-2)

(Table.2-1)

# 受信データのプロトコル1

| Byte | 内容                 | Byte | 内容                  |
|------|--------------------|------|---------------------|
| 0    | 0xff               | 11   | ACC Y 上位8bit (符号付)  |
| 1    | 0xff               | 12   | ACC Z 下位8bit (符号付)  |
| 2    | 0x52               | 13   | ACC Z 上位8bit (符号付)  |
| 3    | 0x54               | 14   | TEMP 下位8bit (符号付)   |
| 4    | 0x34               | 15   | TEMP 上位8bit (符号付)   |
| 5    | 0x57               | 16   | GYRO X 下位8bit (符号付) |
| 6    | 0x00               | 17   | GYRO X 上位8bit (符号付) |
| 7    | タイムスタンプ            | 18   | GYRO Y 下位8bit (符号付) |
| 8    | ACC X 下位8bit (符号付) | 19   | GYRO Y 上位8bit (符号付) |
| 9    | ACC X 上位8bit (符号付) | 20   | GYRO Z 下位8bit (符号付) |
| 10   | ACC Y 下位8bit (符号付) | 21   | GYRO Z 上位8bit (符号付) |

23

(Table.2-2)

# 受信データのプロトコル2

| Byte | 内容                 | Byte | 内容                   |
|------|--------------------|------|----------------------|
| 22   | MAG X 下位8bit (符号付) | 33   | isCurve (符号なし)       |
| 23   | MAG X 上位8bit (符号付) | 34   | isSlope (符号なし)       |
| 24   | MAG Y 下位8bit (符号付) | 35   | 経過時間 0byte (符号なし)    |
| 25   | MAG Y 上位8bit (符号付) | 36   | 経過時間 1byte (符号なし)    |
| 26   | MAG Z 下位8bit (符号付) | 37   | 経過時間 2byte (符号なし)    |
| 27   | MAG Z 上位8bit (符号付) | 38   | 経過時間 3byte (符号なし)    |
| 28   | 角度 下位8bit (符号付)    | 39   | Lipo電圧 下位8bit (符号なし) |
| 29   | 角度 上位8bit (符号付)    | 40   | Lipo電圧 上位8bit (符号なし) |
| 30   | duty 下位8bit (符号付)  | 41   | モーター電圧 下位8bit (符号なし) |
| 31   | duty 上位8bit (符号付)  | 42   | モーター電圧 上位8bit (符号なし) |
| 32   | isStop (符号なし)      |      |                      |

24

## 4. サンプルプログラムの解説

- サンプルプログラムの機能
- サンプルの流れ
- コマンドの生成方法

### サンプルプログラムの機能

プログラムを実行すると、ウィンドウが一つ開きます。そのウィンドウ上をクリックすると、AIミニ四駆の緑のLEDが点灯します。

エディタ下部の、ターミナルにAIミニ四駆からの出力データが表示されます。このデータは (Table.2-1) と (Table.2-2) を参考に解析してください。

また、キーボードのArrowkeyの右と左を押すとAIミニ四駆のモーターが動きます。

サンプルプログラムは以上の3つの機能を有しています。

### サンプルの流れ

はじめに次の行が実行されます

```
//シリアル通信ライブラリを取り込む
import processing.serial.*;
//ポートのインスタンス
Serial port;
//シリアルポートから取得したデータ(Byte)
int inByte;
float m_duty = 0;
```

ここではプログラムの実行に必要なライブラリの取り込みや、Bluetoothの通信に必要なインスタンスの生成及びグローバル変

数の定義を行っています。

この行はプログラム開始時に一回だけ実行されます。

また、この処理が終了すると、各関数が初めて実行されるようになります。

```
void setup(){
//画面の設定
size(100,100);
//シリアルポート設定（Bluetoothのポート）
port=new Serial(this,"/dev/tty.RNBT-4EA9-RNI-SPP",115200); //Mac
//port=new Serial(this,"COM5",115200); //Windows
port.write(command0(0));
}
```

setup関数は、プログラムが開始すると初めに一度だけ実行される関数です。

この関数では、実行時に出てくるウィンドウのサイズの設定と、通信ポートの設定を行っています。

必要に応じて設定内容の変更を行って下さい。詳しい設定の方法は**3. サンプルプログラムの実行**に記載されています。

うまく動作しない場合はそちらを確認してください。

```
void draw(){ background(51); }
```

draw関数はループして実行されます。

background関数は表示されるウィンドウの背景の色を変更しているだけです。

```
void serialEvent(Serial p){
// 設定したシリアルポートからデータを読み取り
inByte = port.read();
println(hex(inByte));
}
```

serialEvent関数はAIミニ四駆からデータが、送られるたびに実行されます。

データはprintln関数により、ターミナルに表示されます。

```
void keyPressed() {
if (key == CODED) { // コード化されているキーが押された
if (keyCode == LEFT) {
if(m_duty > -1){
m_duty -= 0.1;
//println(m_duty);
}
port.write(command0(m_duty));
}else if(keyCode == RIGHT){
if(m_duty < 1){
m_duty += 0.1;
//println(m_duty);
}
port.write(command0(m_duty));
}
}
}
```

keyPressed関数はキーボードが押されるたびに実行されます。今回はArrowkeyの左と右が押された時のみ有効になっています。

コマンドはcommand0関数を用いて生成されます。

生成されたコマンドはport.write関数を用いて、AIミニ四駆に送信されます。

**コマンドの生成方法については後述します。**

```
/  
* id 0: dutyの変更コマンド
```

```
* @param duty dutyを-1.0から1.0で指定
```

```
* 負の値はモーターを逆の方向に回す
```

```
* @return 10byteのコマンド配列
```

```
/
```

```
byte[] command0(float duty){
```

```
byte[] command = new byte[10];
```

```
int int_duty;
```

```
int duty_L;
```

```
int duty_H;
```

```
int_duty = (int)(duty * 32767.0);
```

```
if(int_duty<0)
```

```
{
```

```
int_duty += 65535;
```

```
}
```

```
duty_L = int_duty & 0x000000ff ;
```

```
duty_H = (int_duty & 0x0000ff00)>>8;
```

```
//ヘッダー
```

```
command[0] = 99;
```

```
command[1] = 109;
```

```
command[2] = 100;
```

```
//id
```

```
command[3] = 0;
```

```
//値
```

```
command[4] = byte(duty_L);
```

```
command[5] = byte(duty_H);
```

```
//ダミー
```

```
command[6] = 0;
```

```
command[7] = 0;
```

```
command[8] = 0;
```

```
command[9] = 0;
```

```
return command;
```

```
}
```

このコマンドはkeyPressed内で実行されます。

引数のdutyによってAIミニ四駆のモーターのパワーを変更するようなコマンドを生成します。

```
void mousePressed(){
```

```
byte[] command = new byte[10];
```

```
//ヘッダー
```

```
command[0] = 99;
```

```
command[1] = 109;
```

```
command[2] = 100;
```

```
//id
```

```
command[3] = 1;
```

```
//値
```

```
command[4] = byte(1);
```

```
//ダミー
```

```
command[5] = 0;
```

```
command[6] = 0;
```

```
command[7] = 0;
```

```
command[8] = 0;
```

```
command[9] = 0;
```

```
port.write(command);
```

```
}
```

mousePressed関数はマウスがクリックされると実行されます。

今回は緑のLEDが点灯するようにコマンドを生成したのちに、AIミニ四駆に送信しています。

```
void mouseReleased(){
byte[] command = new byte[10];
//ヘッダー
command[0] = 99;
command[1] = 109;
command[2] = 100;
//id
command[3] = 1;
//値
command[4] = byte(0);
//ダミー
command[5] = 0;
command[6] = 0;
command[7] = 0;
command[8] = 0;
command[9] = 0;
port.write(command);
}
```

mouseRelease関数はマウスのクリックが解除されると実行されます。

今回は緑のLEDが消灯するようにコマンドを生成したのちに、AIミニ四駆に送信しています。

## コマンドの生成方法

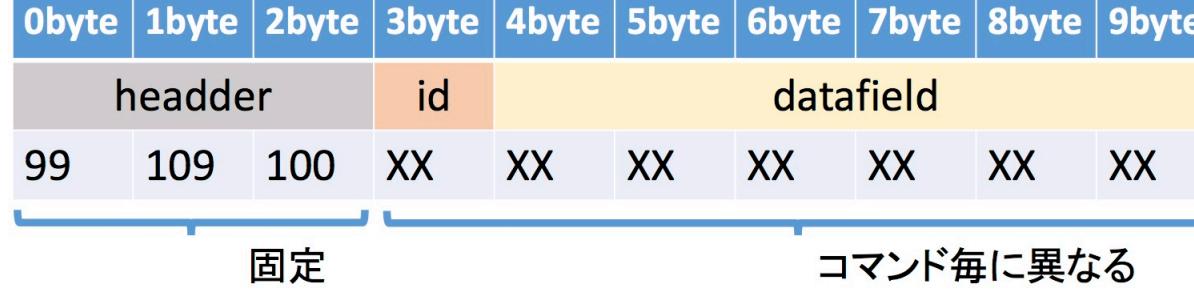
AIミニ四駆への送信コマンドは要素数10のbyte配列を使用します。 (固定長) (ref.2-1)

ヘッダーの始め3byteは必ず固定で、「99」「109」「100」を代入してください。4byte目はコマンドidです。この数字を変更することで、AIミニ四駆への命令の種類を変えることができます。

5byte目以降はデータフィールドです。データフィールドは全て利用するわけではありません。必要な要素には「0」を代入してください。詳細は、同ディレクトリ内に存在する「AIミニ四駆ハッカソン」を参照ください。

## 送信コマンドのプロトコル

- 長さ10byteで以下のフォーマット



- 意味のある10byteコマンドをAICHIP側が受信すると対応した動作を実行
- 送信コマンドはid 0,1, ..., 5の5種類
- BluetoothChatのbyte列送信関数をもちいて10byteのコマンドを送る

13

(ref.2-1 コマンド送信プロトコルの説明)

- id = 0** モーターコントロール用コマンド

モーターを制御するには、送信コマンドの4byte目の「id」を「0」にする必要があります。 (ref.2-2)

データフィールドは5byte目と6byte目にdutyの値を入れます。その他のデータフィールドには「0」を代入します。dutyの値は16bitの符号付整数です。上位bitが5byte目、下位bitが6byte目です。配列の順番に気をつけてください。

# モーターduty設定コマンド

| 0byte  | 1byte | 2byte | 3byte | 4byte     | 5byte | 6byte | 7byte | 8byte | 9byte |
|--------|-------|-------|-------|-----------|-------|-------|-------|-------|-------|
| header |       | id    |       | datafield |       |       |       |       |       |
| 99     | 109   | 100   | 0     | XX        | XX    | 0     | 0     | 0     | 0     |

16bit符号付整数 ←ここを与えたいdutyに応じて変更

アクセルの吹かし具合



速度ではなく加速度を操作していることに注意

14

(ref.2-2 モーター制御コマンド)

- モーターのduty設定例

- duty=50%のコマンド例

```

/*
int_duty = (int)(0.5 * 32767.0);
int_duty = 16384 正確には16383.5ですが、切り上げました。
16384を16進数に変換すると0x4000になります。
よって duty_L = 00, duty_H = 40 となります。
*/
//ヘッダー
command[0] = 99;
command[1] = 109;
command[2] = 100;
//id
command[3] = 0;
//値
command[4] = byte(00);
command[5] = byte(40);
//ダミー
command[6] = 0;
command[7] = 0;
command[8] = 0;
command[9] = 0;

```

- duty=20%のコマンド例

```

/*
int_duty = (int)(0.2 * 32767.0);
int_duty = 6553 正確には6553.4ですが、切り下げました。
16384を16進数に変換すると0x1999になります。
よって duty_L = 99, duty_H = 19 となります。
*/
//ヘッダー
command[0] = 99;
command[1] = 109;
command[2] = 100;
//id
command[3] = 0;
//値
command[4] = byte(99);

```

```

command[5] = byte(19);
//ダミー

command[6] = 0;
command[7] = 0;
command[8] = 0;
command[9] = 0;

```

- **duty=-50%**のコマンド例

```

/*
int_duty = (int)(-0.5 * 32767.0);
int_duty = -16384 正確には-16383.5ですが、切り下げました。
int_dutyが負の値なので65535を加えます。加えた値は次のようにになります。
int_duty = (-16384 + 65535) = 49151
49151を16進数に変換すると0x0000BFFFになります。
よって duty_L = FF, duty_H = BF となります。
*/
//ヘッダー
command[0] = 99;
command[1] = 109;
command[2] = 100;
//id
command[3] = 0;
//値
command[4] = byte(FF);
command[5] = byte(BF);
//ダミー
command[6] = 0;
command[7] = 0;
command[8] = 0;
command[9] = 0;

```

- **id=1** 緑のLED制御コマンド

LEDを制御するには、送信コマンドの4byte目の「id」を「1」にする必要があります。(ref.2-3)

データフィールドは5byte目に「0」を入れると、緑のLEDが消灯します。「1」を入れると緑のLEDが点灯します。  
その他のデータフィールドは「0」を代入します。

## LEDの点灯制御コマンド

- 右(緑)LEDの制御コマンド

| 0byte  | 1byte | 2byte | 3byte     | 4byte | 5byte | 6byte | 7byte | 8byte | 9byte |
|--------|-------|-------|-----------|-------|-------|-------|-------|-------|-------|
| header |       | id    | datafield |       |       |       |       |       |       |
| 99     | 109   | 100   | 1         | XX    | 0     | 0     | 0     | 0     | 0     |

- 左(赤)LEDの制御コマンド 1:LED点灯 0:LED消灯

| 0byte  | 1byte | 2byte | 3byte     | 4byte | 5byte | 6byte | 7byte | 8byte | 9byte |
|--------|-------|-------|-----------|-------|-------|-------|-------|-------|-------|
| header |       | id    | datafield |       |       |       |       |       |       |
| 99     | 109   | 100   | 2         | XX    | 0     | 0     | 0     | 0     | 0     |

1:LED点灯 0:LED消灯

16

(ref.2-3) LED制御コマンド

## Androidアプリの使い方

1. Androidアプリのインストールから導入まで
2. Andoridサンプルアプリの使い方

### 3. Androidアプリに付属しているライブラリの使いかた

## 1. Androidアプリのインストールから導入まで

初めにAndroid開発環境である「Android Studio」を、[AndroidStudioのダウンロードページ](#)からダウンロードしてください。リンクのページの下部に各OSごとのインストールパッケージが置いてあるので、そこからダウンロードしてください。(Fig.3-1)

### All Android Studio Packages

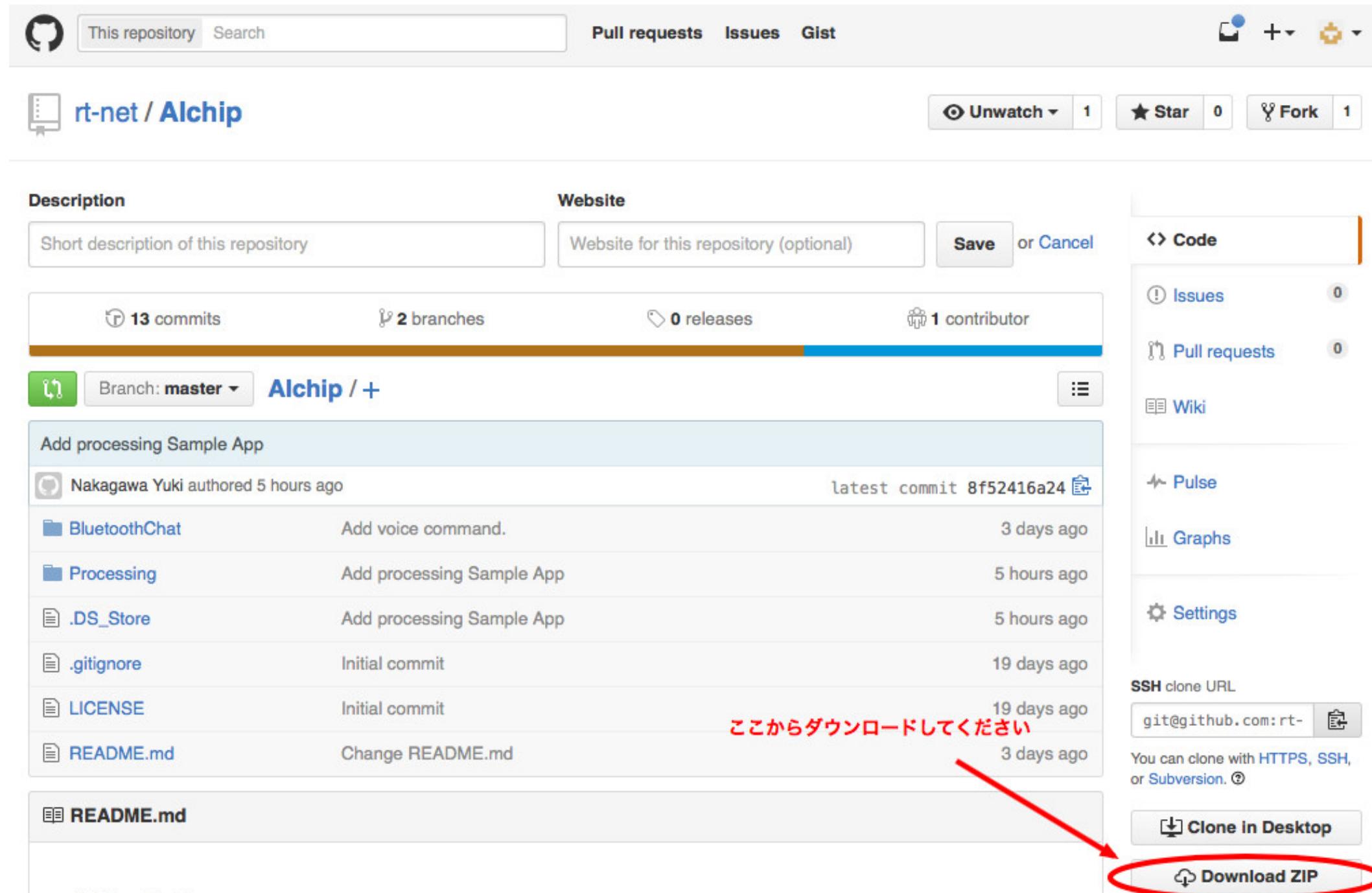
Select a specific Android Studio package for your platform. Also see the [Android Studio release notes](#).

| Platform | Package   | Size             | SHA-1 Checksum                           |
|----------|---|------------------|--|
| Windows  | <a href="#">android-studio-bundle-141.2178183-windows.exe</a><br>(Recommended)        | 1136982712 bytes | c7d39c529dd434489da9d086ff689d34dc791526 |
|          | <a href="#">android-studio-ide-141.2178183-windows.exe</a><br>(No SDK tools included) | 321810248 bytes  | b5d1aaa000729c03a3cf980add79d1b93121c56d |
|          | <a href="#">android-studio-ide-141.2178183-windows.zip</a>                            | 344424713 bytes  | 3134f226b5f3c3f74d4fc2d9cff03a4458f01d69 |
| Mac OS X | <a href="#">android-studio-ide-141.2178183-mac.dmg</a>                                | 368335367 bytes  | 75b67eb15a34a152a40e7189484ab0ebc375b877 |
| Linux    | <a href="#">android-studio-ide-141.2178183-linux.zip</a>                              | 352010593 bytes  | cf780413f8c8223eb348bd27c19a9c04b75eaeb2 |

(Fig.3-1 AndroidStudioのダウンロードページ)

次にダウンロードしたファイルから、「Android Studio」をインストールしてください。

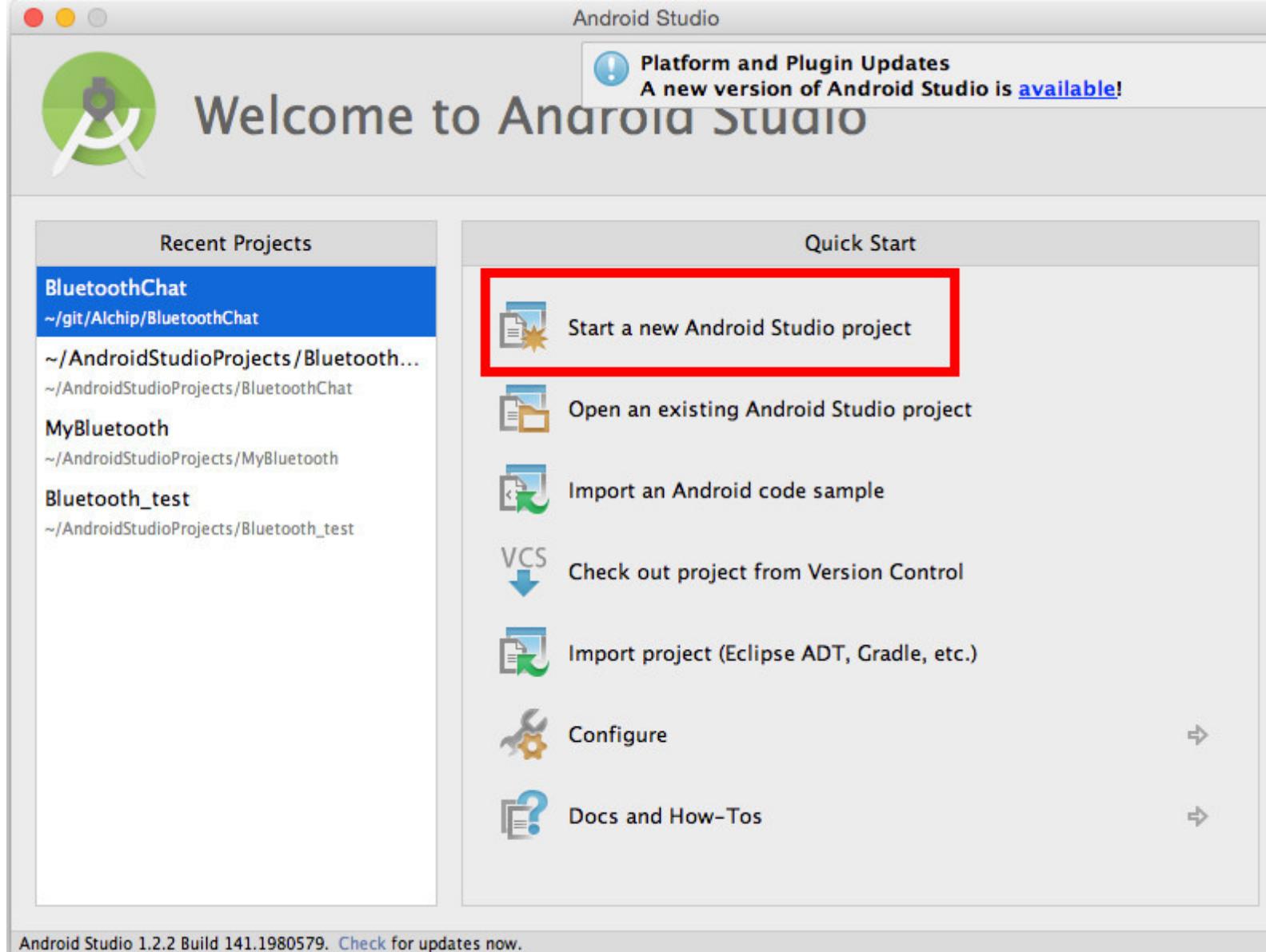
インストールが終了したら、次は[AIミニ四駆のgithubのページ](#)から、プロジェクトをダウンロードします。ダウンロードする際には、画面右側の「Download ZIP」をクリックしてください。(Fig.3-2)



(Fig.3-2 githubのページ)

ダウンロードが完了したら、ダウンロードしたZIPファイルを任意の場所に解凍してください。

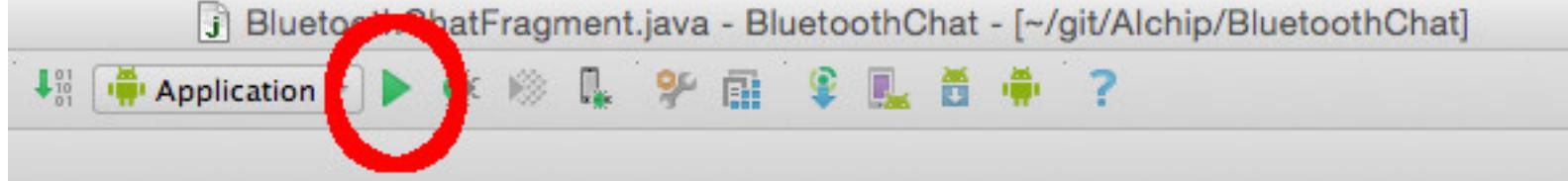
ここまで完了したら、「Android Studio」を起動してください。起動すると、次のような画面が開かれると思うので、「Start a new Android project」を選択してください。(Fig.3-3)



(Fig.3- )

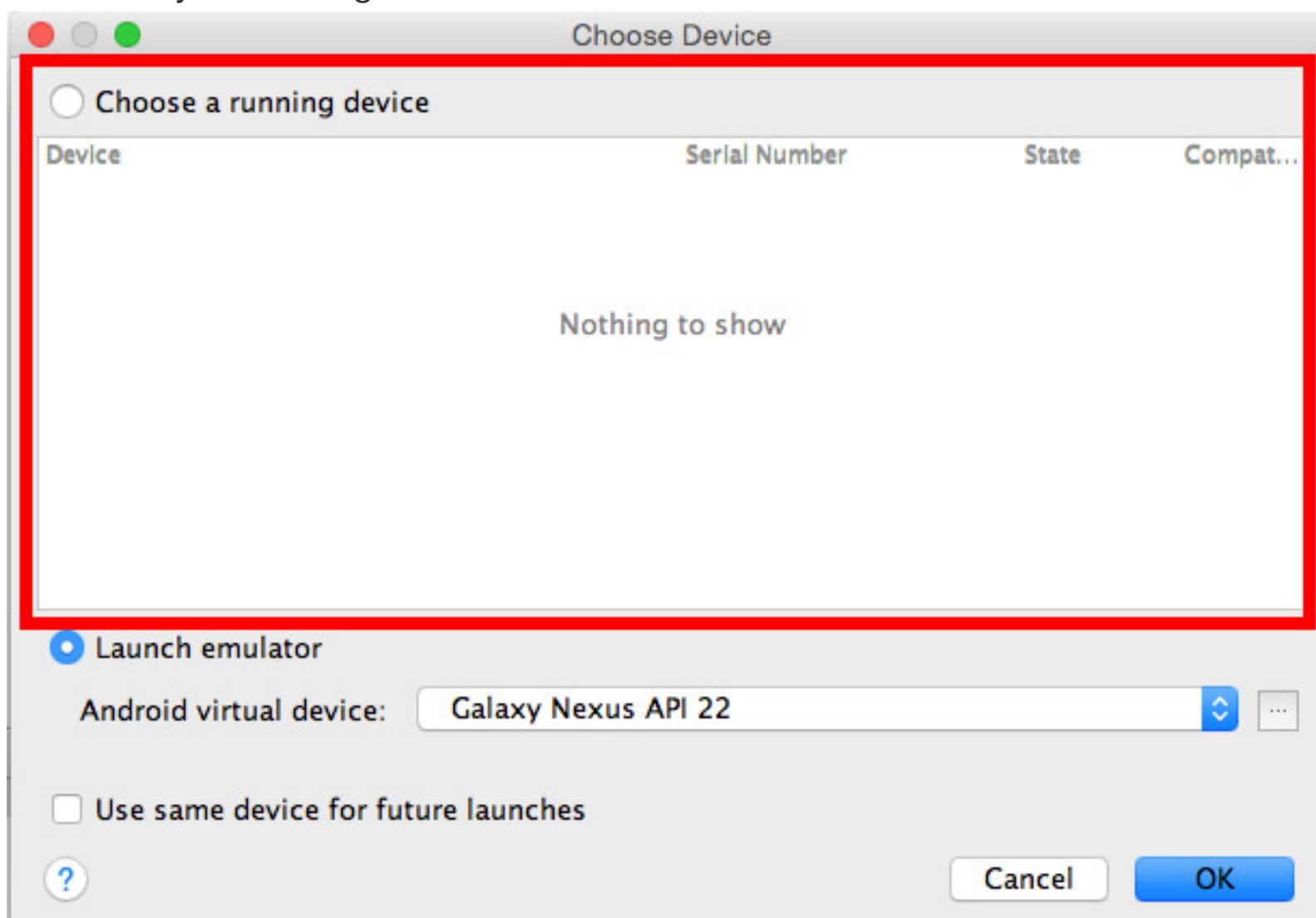
選択したら、先ほど解凍した「AI-CHIP」ディレクトリ内にある「BluetoothChat」を選択して、開いてください。このプロジェクトが、AIミニ四駆のAndroid端末ようサンプルアプリのプロジェクトです。  
次に、サンプルアプリをAndroid端末に転送する手順です。お手持ちのAndroid端末を「USBデバックモード」をONにしてPCと接続してください。

接続ができたら、Android Studio上部にある緑の三角アイコンをクリックしてください。(Fig.3- )



(Fig.3- )

「Choose your running device」にお手持ちのデバイスが表示されていれば書き込み可能です。(Fig.3- )

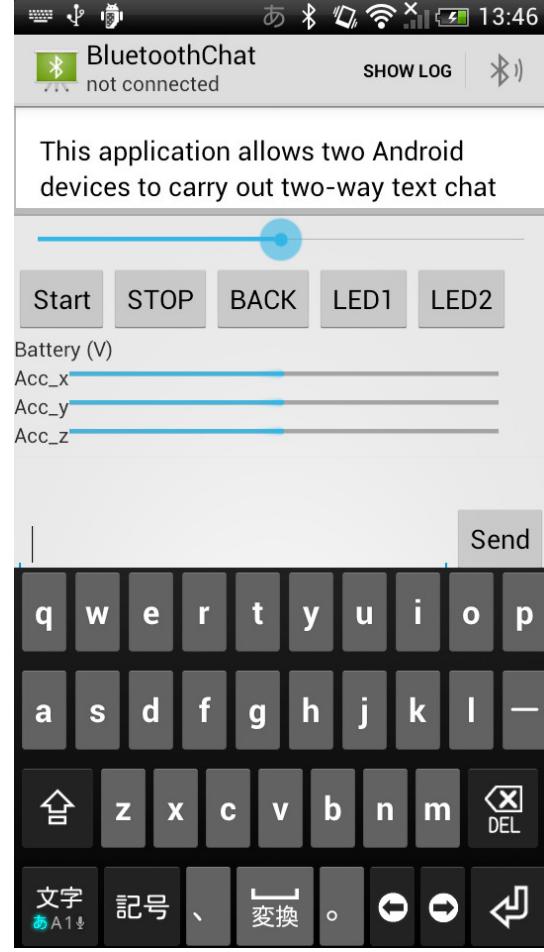


(Fig.3- )

「OK」を選択して、書き込みをしてください。お手持ちのデバイスに「BluetoothChat」のアプリがインストールされれば成功です。

## 2. Andoroidサンプルアプリの使い方

画面上にはスイッチが五つとプログレスバーが三つと、シークバーが配置されており、それぞれがAIミニ四駆と情報の送受信が可能です。(Fig.3- )



(Fig.3- )

AIミニ四駆と通信をするためには、画面右上のBluetoothアイコンをおして、ペアリングをしてください。

接続ができると、画面上の「non connection」が接続したデバイス名になります。

画面中央に配置された、左三つのスイッチを押すとAIミニ四駆のモーターが動きます。

右側二つのスイッチを押すとLEDが点灯します。

シークバーをスライドさせると、AIミニ四駆の速度が変動します。

画面上のプログレスバーは、それぞれx,y,z,軸方向のAIミニ四駆の加速度を示します。

Battery(V)の部分にはリアルタイムでAIミニ四駆のモーター電圧が表示されます。

文字を何も打たずに「send」ボタンを押すと、音声入力モードになり、特定のキーワードに反応してAIミニ四駆が動きます。

(Table.3-1)

特定の文字を打ち込んで「send」ボタンを押すと、AIミニ四駆が動きます。(Table.3-2)

(Table.3-1 音声入力で反応するキーワード)

| 音声入力      | AIミニ四駆の動作 |
|-----------|-----------|
| 動け、行け、Go! | 前進100%    |
| ゆっくり      | 前進50%     |
| ストップ、止まれ  | 停止        |
| 戻れ、バック    | 後進100%    |

(Table.3-1 テキスト入力で反応するキーワード)

| テキスト入力 | AIミニ四駆の動作   |
|--------|-------------|
| 0      | 停止          |
| set    | ジャイロセンサー初期化 |
| get    | ジャイロセンサー値取得 |
| gon    | 緑色LED点灯     |
| goff   | 緑色LED消灯     |
| gf     | 緑色LED点滅     |
| ron    | 赤色LED点灯     |
| roff   | 赤色LED消灯     |
| rf     | 赤色LED点滅     |

### 3. Androidアプリに付属しているライブラリの使いかた

「BluetoothChatService」から拡張された、AI-CHIP.javaがAIミニ四駆のライブラリです。

AIミニ四駆からのデータのget関数と、AIミニ四駆へのset関数が用意されています。

Androidアプリ内で関数を呼び出して利用してください。

詳しくはAI-CHIP.javaを参照してください。