白板ソフトと Arduino の USB 通信について
2016/05/20

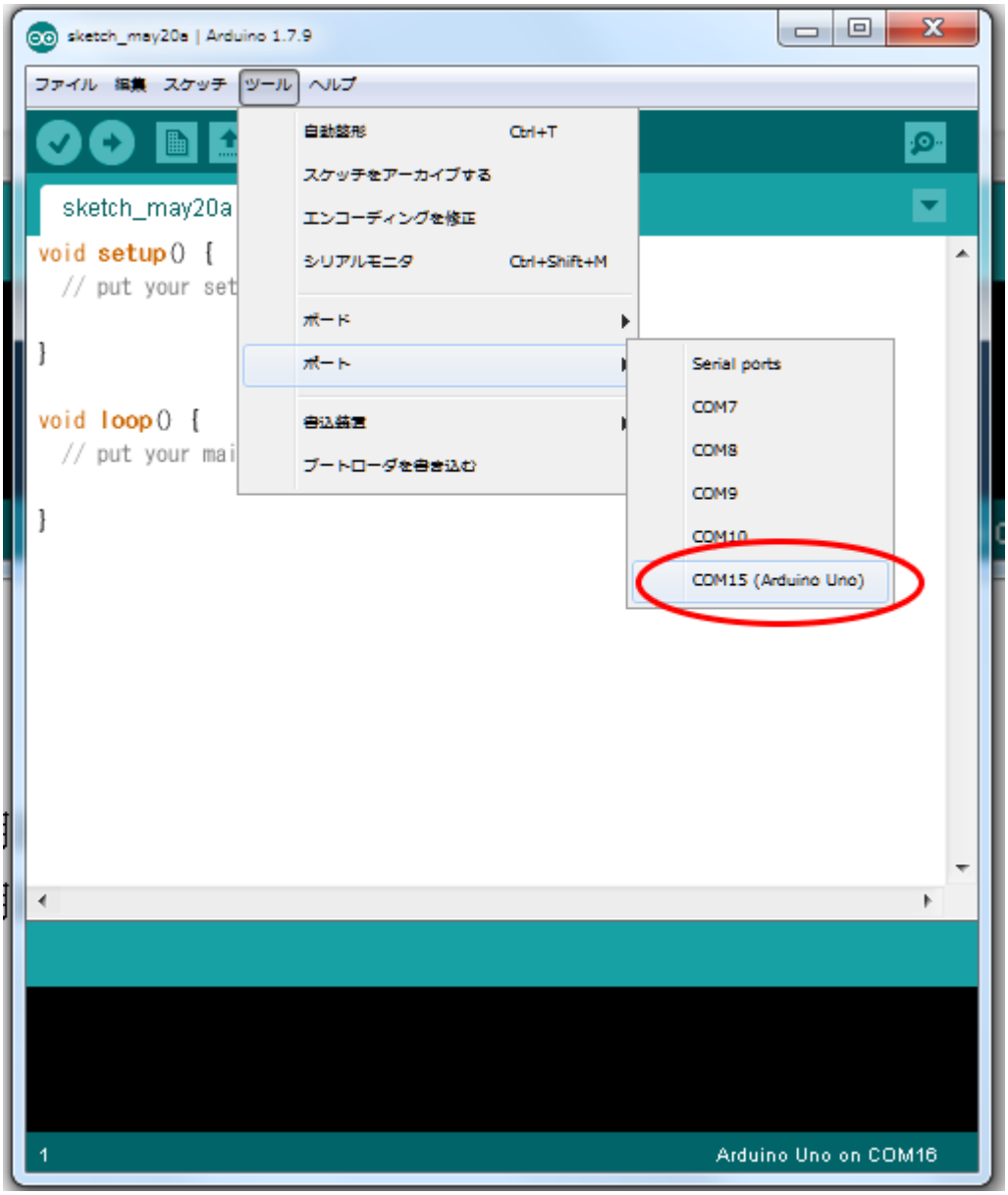
白板ソフトから USB ケーブルを使って、Arduino と通信する方法について説明します。

１）接続方法
２）USB 通信用部品の使い方
３）USB 通信用部品のカスタマイズ方法

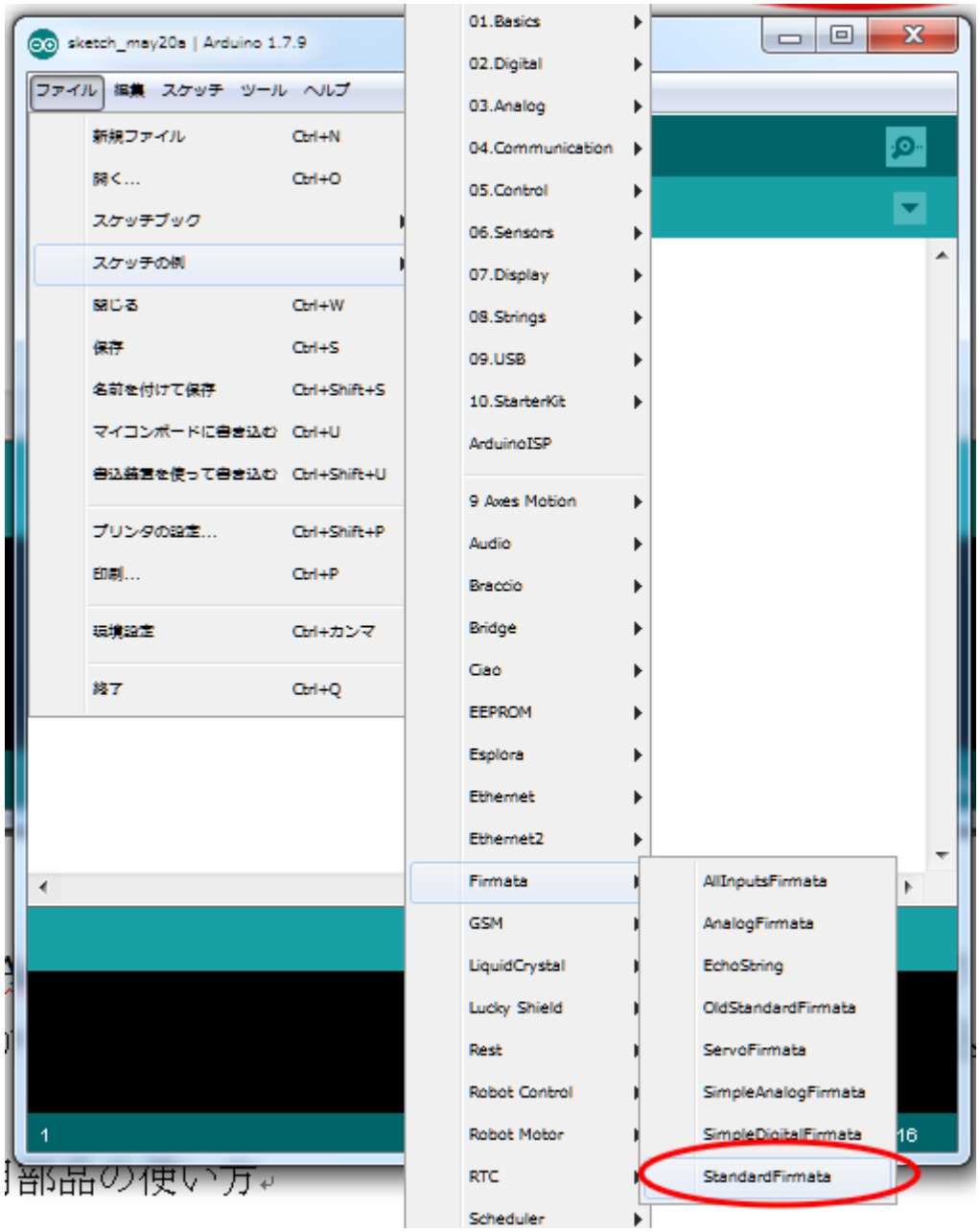
１）接続方法

WindowsPC に以下のソフトをダウンロードしてインストールまたは実行します。

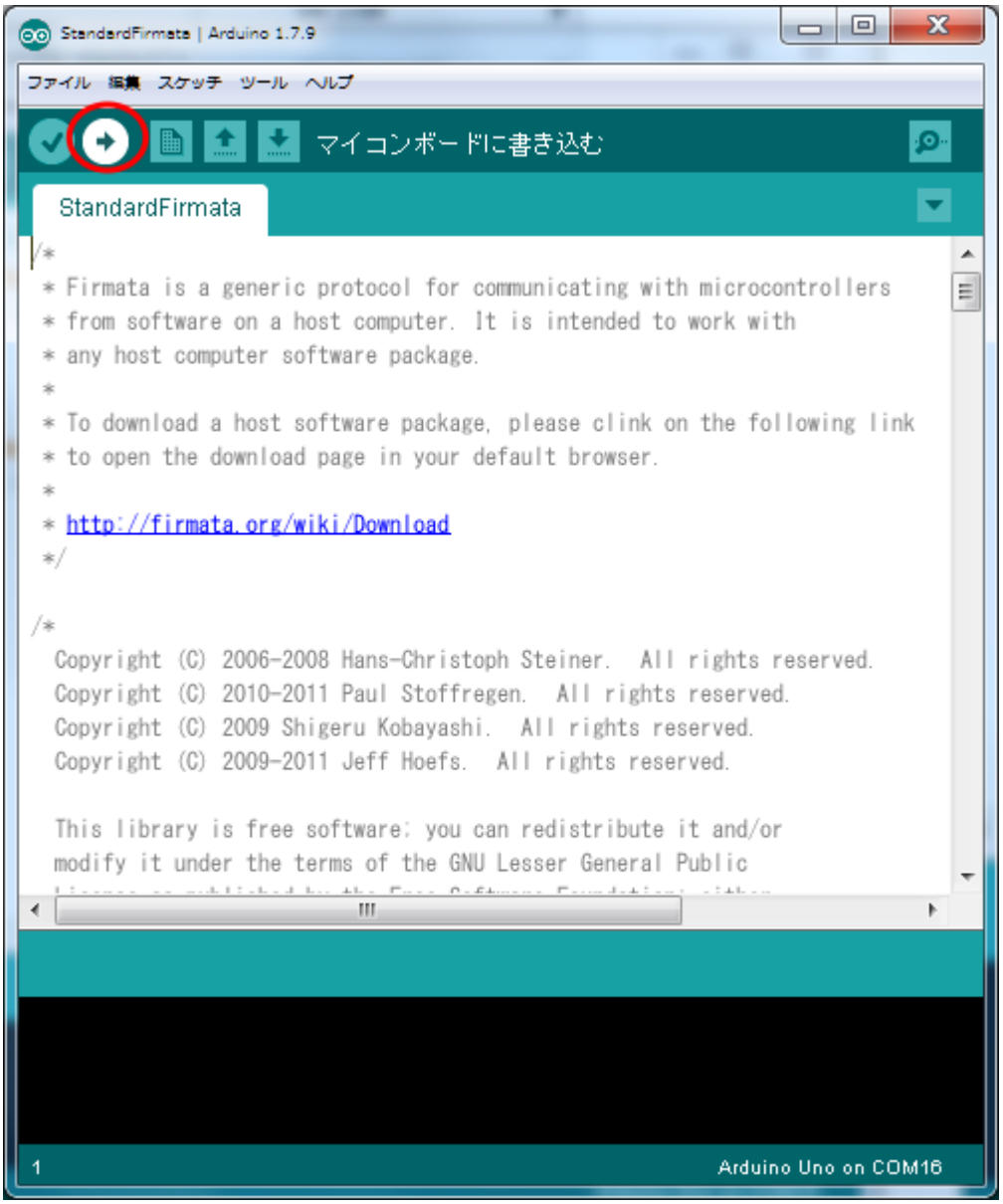
- 白板ソフト http://www.mbrain.com/wb/
- Arduino Software (windows installer) https://www.arduino.cc/en/Main/Software
  Arduino にスケッチ（プログラム）を書き込むのに使用します。
- Firmata のテストプログラム　http://firmata.org/wiki/Main_Page
  動作テストを行うために使用します。
- USB ドライバ
  互換ボードを使う場合には、Ardino Software に入っているドライバで USB が正しく認識されない場合があります。Arduino のボード上のチップに合わせた USB ドライバが必要になります。使っているボードの資料を見て、必要なドライバをインストールして下さい。



ソフトウェアのインストールが終わったら、USB ケーブルで Arduino を接続します。その後、Arduino Software を起動します。起動してしばらく待つと Arduino Software の画面が表示されます。ツール/ポートから、接続中の Arduino を選択します。接続中の Arduino が表示されない場合には、USB ケーブルや USB ドライバ（一度 USB ケーブルを抜いてから再度差し込んでドライバが起動しているか等）を確認して下さい。この COM ポートを使って通信します。COM ポートの番号は USB ケーブルを抜いてから差し込んだときに別の番号に変わる場合もあります。

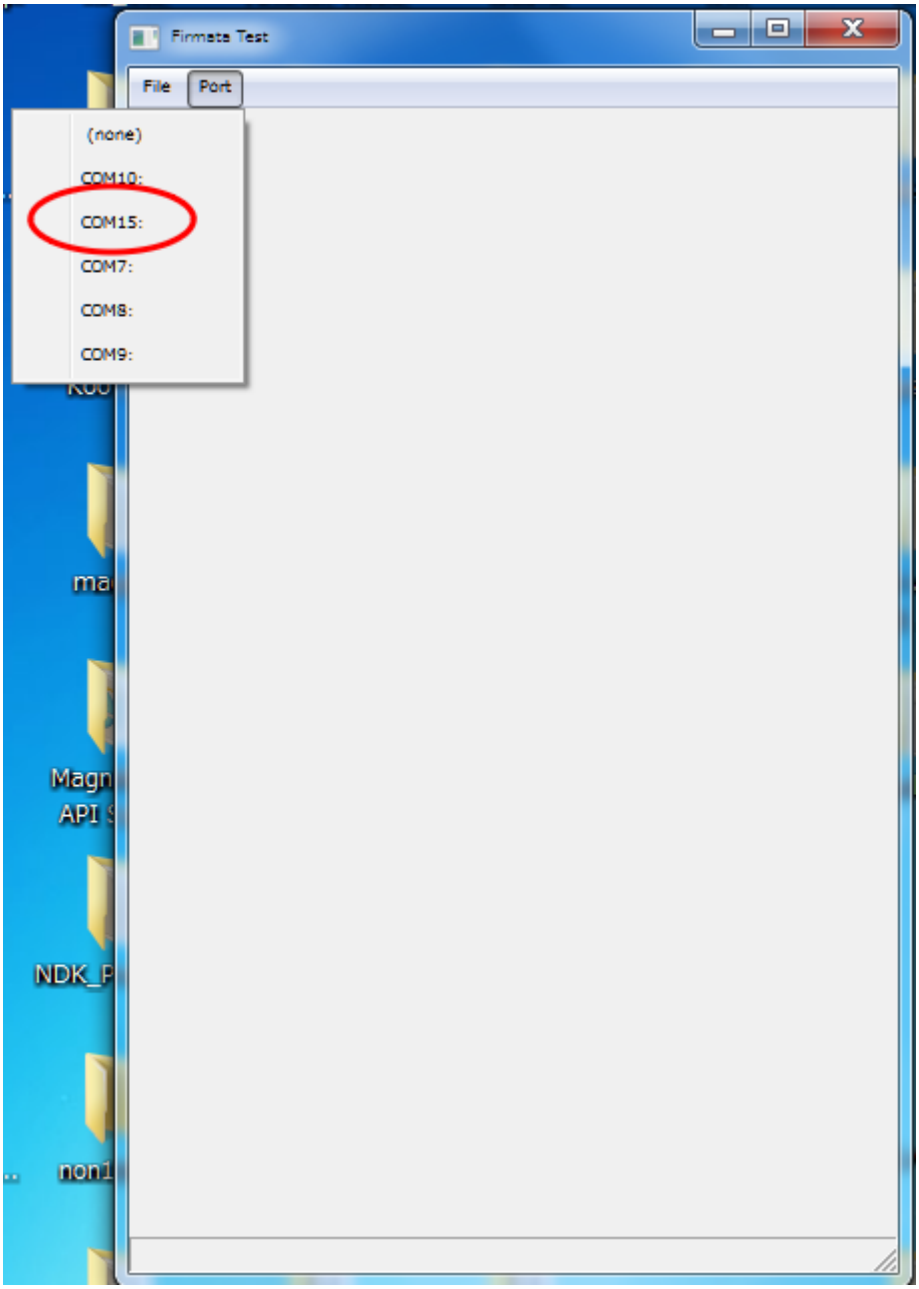


USB の接続で Ardino へスケッチ（プログラム）が送れるようになりました。次は USB 通信で Ardino を動かすためのスケッチ（プログラム）を Ardino に書き込みます。ファイル/スケッチの例/ Firmata/ StandardFirmata を選択してスケッチを呼び出します。

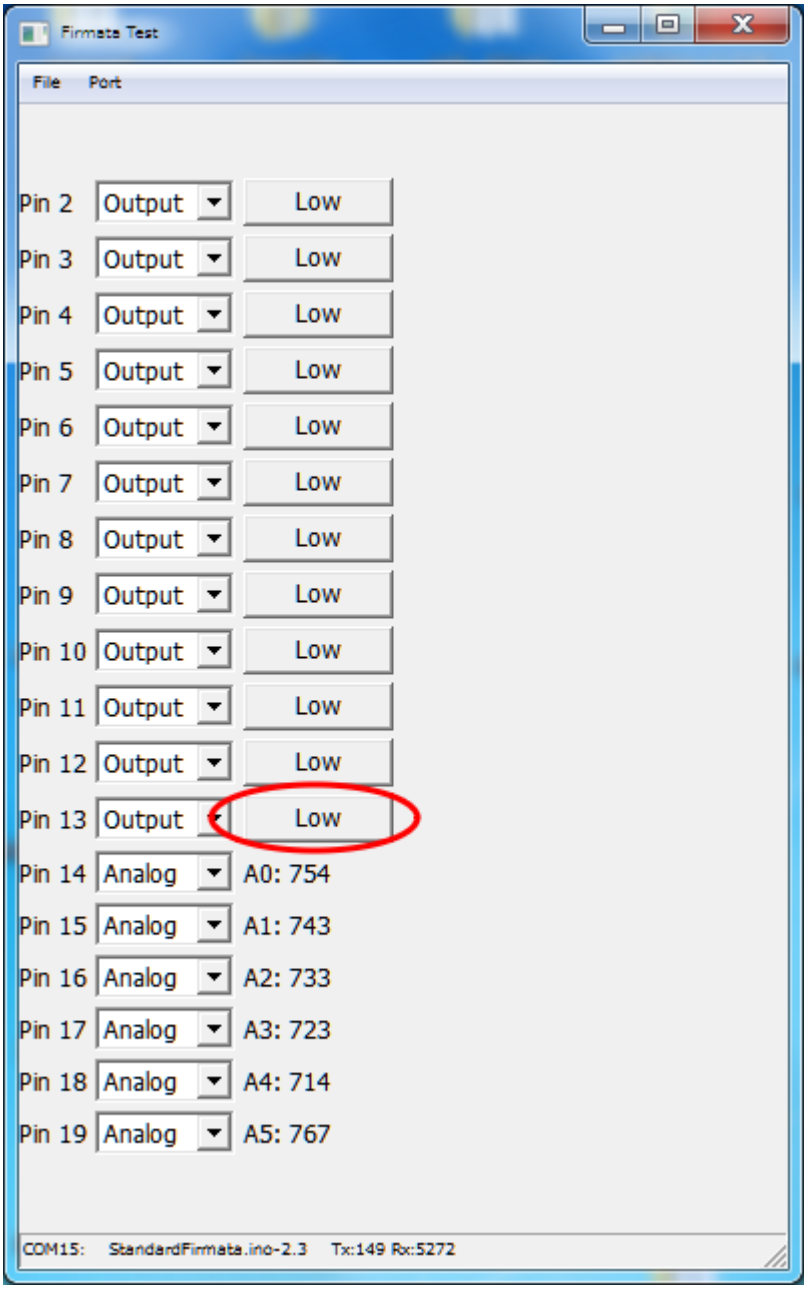


読み込んだスケッチをマイコン（Arduino）に書き込みます。左上の２番目の矢印のボタンを押すとコンパイルと書き込みが行われます。少し時間がかかる場合もあります。書き込みが終わると Firmata のスケッチが動きます。書き込みは一度だけで、次回からはボードの電源を入れるだけで Firmata のスケッチが動きます。開いている Arduino Software を閉じて下さい。

Firmata の動作テスト

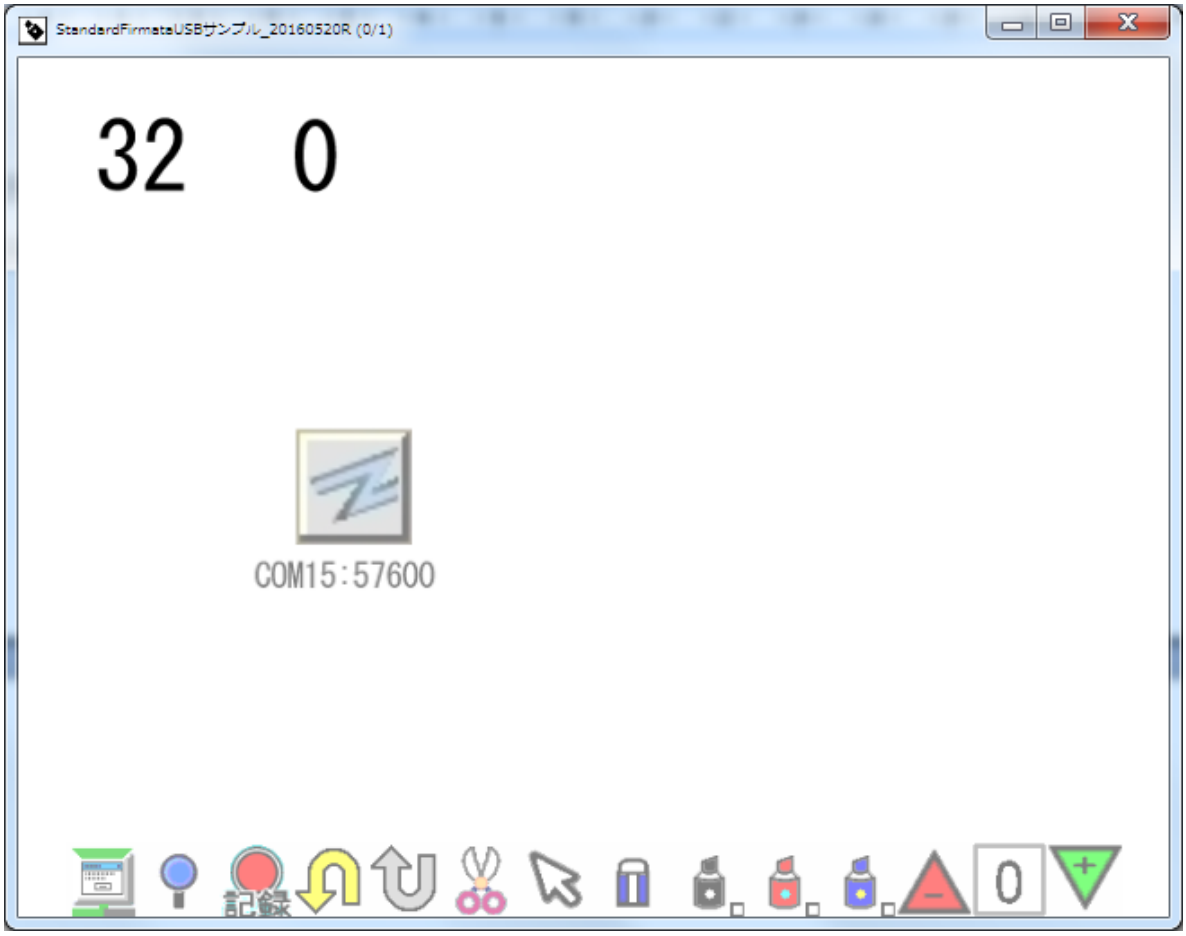


Firmata が起動したら、動作テストを行います。この動作テストはやらなくても良いのですがボードが正しく動いているかの確認のために行うことを推奨します。ダウンロードした Firmata test を起動します。起動したら Port を設定します。ここでの Port 番号は先ほど Arduino Software で使用した番号と同じ番号を設定して下さい。

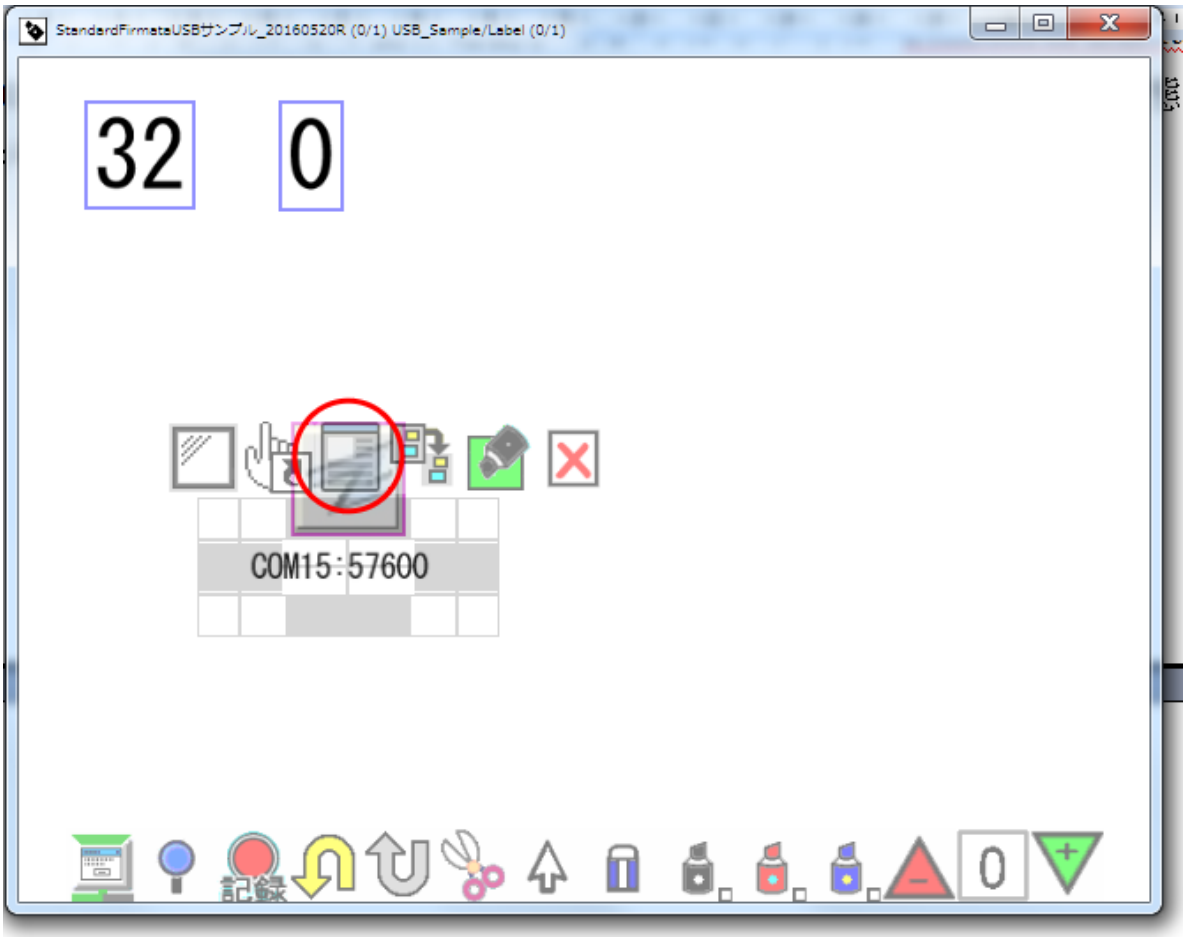


接続が行われるとテスト画面からピンの状態を設定する事が出来るようになります。Pin13 がボード上の LED と接続されているので、Output を H と L に切り換えてボード上の LED が ON/OFF することを確認して下さい。それ以外のピンも接続してテストする事も出来ます。動作が確認できたら Firmata test を閉じて下さい。Firmata test が開いたままだと白板ソフトからの USB 接続に失敗します。

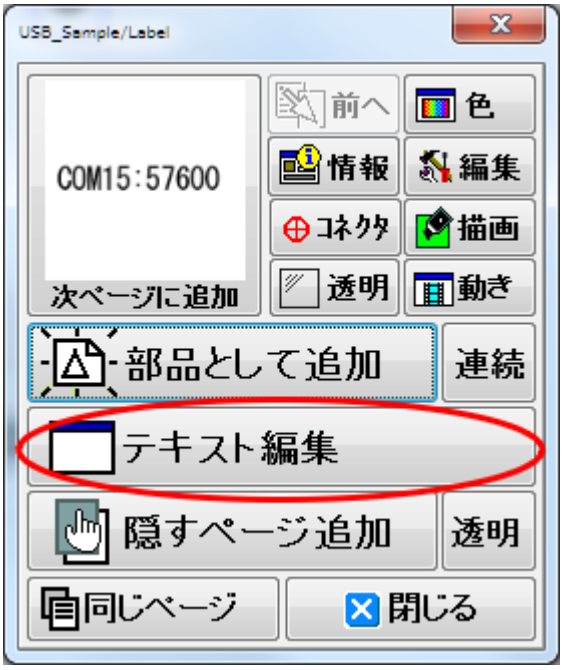
白板ソフトのサンプルでの動作確認を行います。



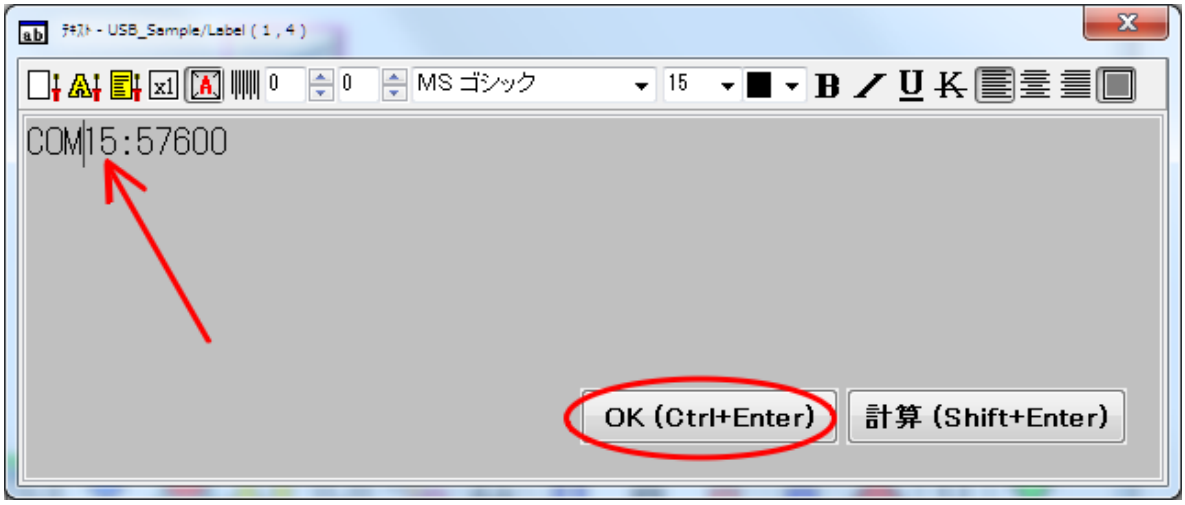
USB 通信用サンプルファイル http://www.mbrain.com/usb_sample.zip をダウンロードして開きます。zip ファイルを解凍しサンプルファイル StandardFirmataUSB_20160520R.jpeg を白板ソフトで開きます。四角の部品が USB 通信用の部品になります。この部品の種類は実行部品で USB 用の変数を含むことで USB 通信がおこなれます。最初は USB 通信開始前の状態です。



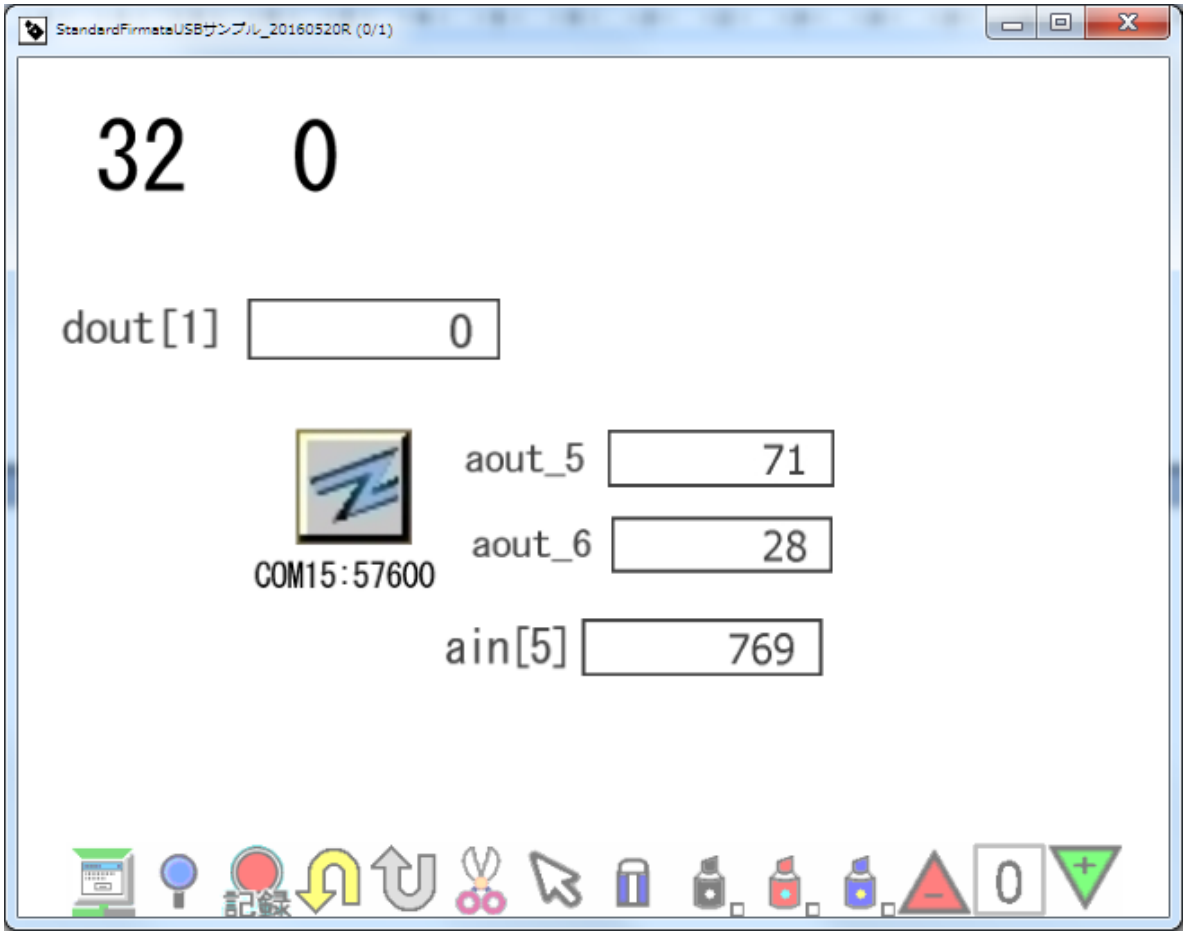
次に USB のポート番号に合わせて COMxx:57600 の xx 部分の数値を編集します。はさみを選択して部品上をクリック COM15:57600 の部分を２回クリックして選択し、表示されるポップアップボタンの真ん中の部品ダイアログボタンをクリックします。部品ダイアログが表示されます。



部品ダイアログのテキスト編集ボタンを押してテキストを編集します。

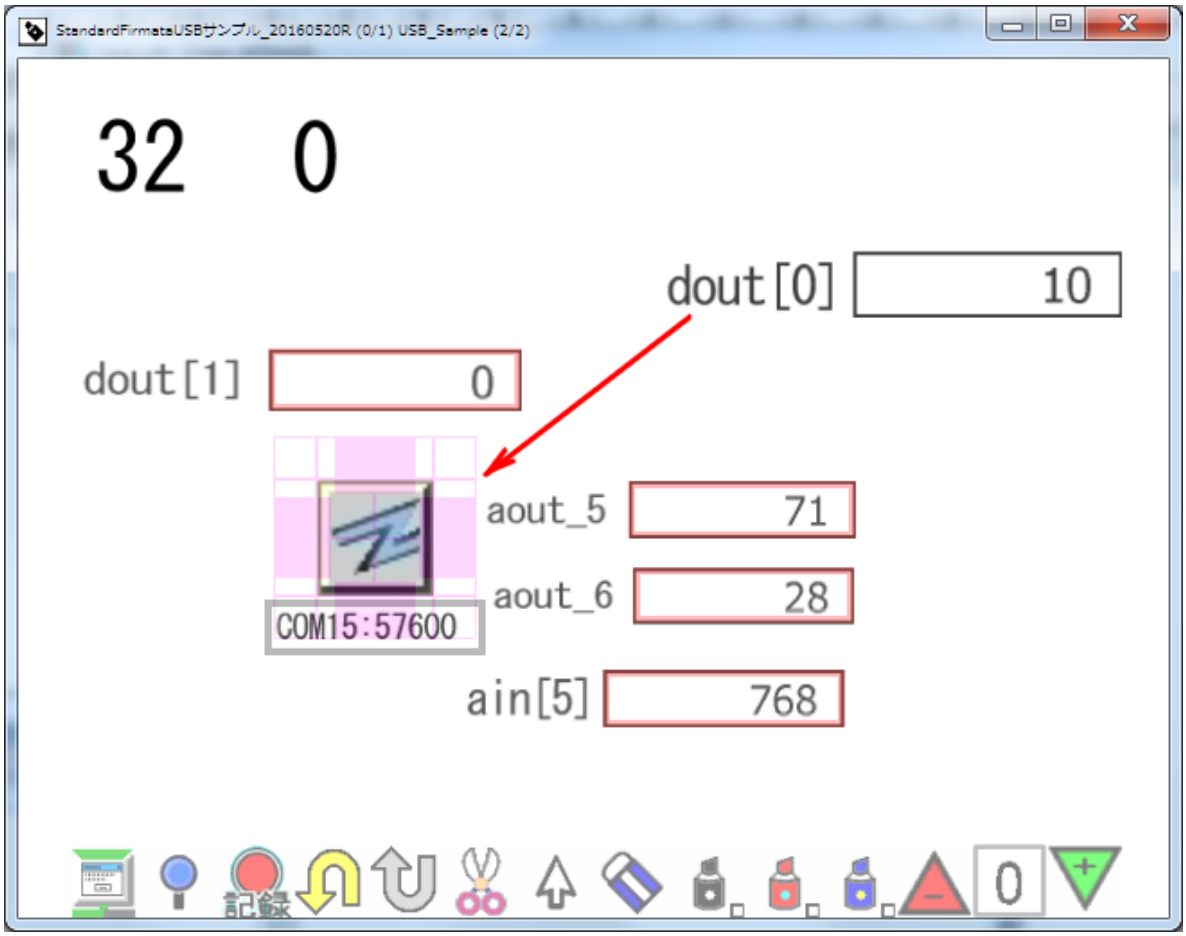


COM ポート番号を接続中の Arudino ボートに合わせた番号に変更してから OK ボタンをクリックします。これで接続先の COM ポートの設定が更新されました。



矢印を選択して、USB 通信用部品をクリックすることで通信が開始します。サンプルでは dout[1]がデジタル出力、Pin5,Pin6 をサーボ出力、ain[5]がアナログ入力５に設定しています。dout[1]に 32 の値をドラッグして pin13 の LED を ON、０の値をドラッグで OFF にして動作の確認が出来ます。また、サーボのコントロールを Pin5 と Pin6 に接続することでサーボモータを動作させることが出来ます。アナログ Pin5 に電圧を入れることでアナログ入力の動作の確認が出来ます。

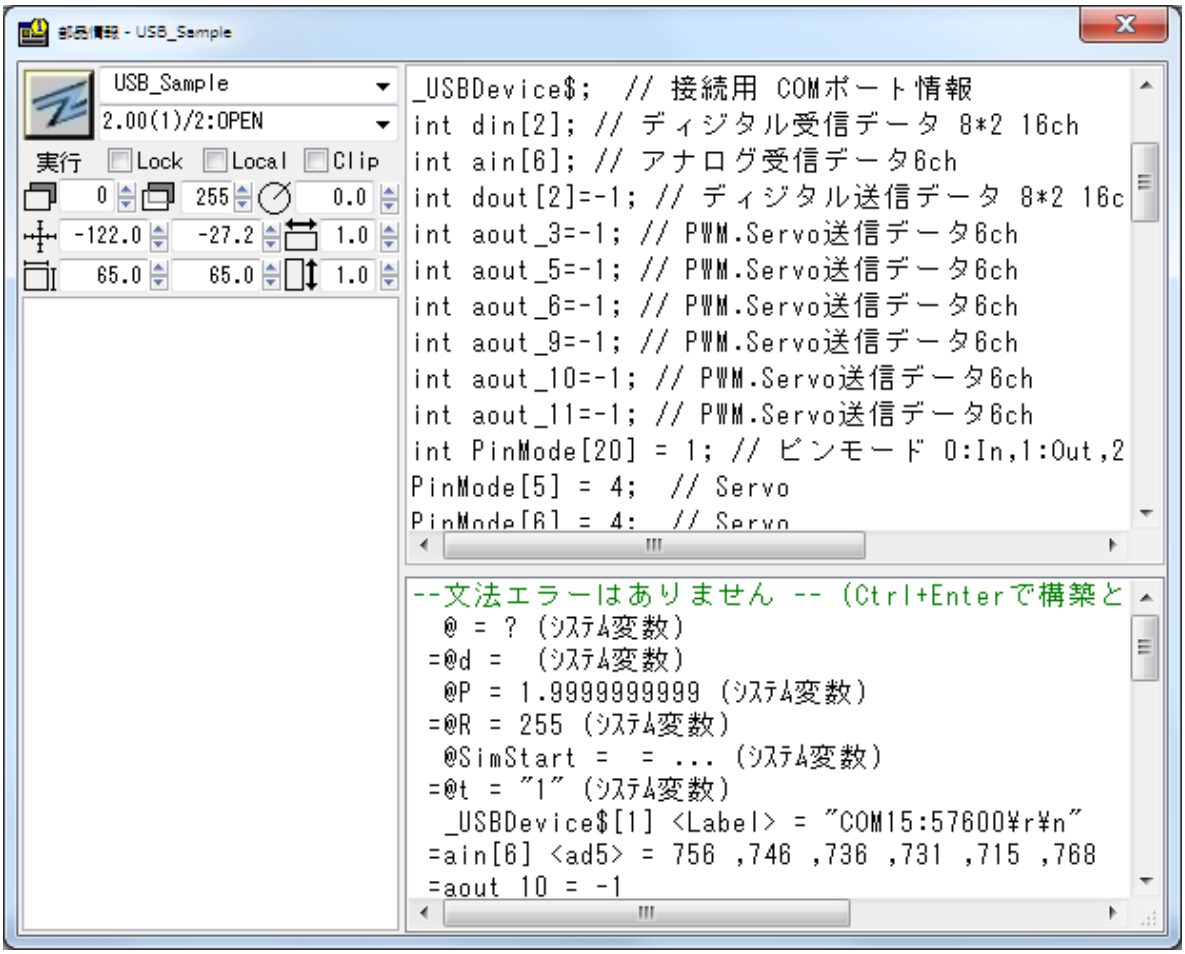
２）USB 通信用部品の使い方



USB 通信部品には、Firmata との通信プログラムとデータが入っています。参照したい変数を参照する部品を作成して部品間での通信を行います。参照したい変数名のテキストを作成して計算ボタンを押して変数参照の部品を作成してから、それを通信用部品の子部品として配置することで表を通して通信を行うことが出来ます。不要になった子部品は選択して削除出来ます。COM15:57600 のようにポートを設定している子部品は削除すると設定出来なくなるので削除しないで下さい。子部品として設定した表の値を変えることでサーボやディジタル出力を行うことが出来ます。アナログ入力に関しては、入力の変化に合わせて値が変化します。共に USB 通信部品内のローカル変数の値と接続されています。

３）USB 通信用部品のカスタマイズ方法

Firmata のピンのモードの変更も部品の変数の設定で行います。外部からの変数設定もしくは関係式を直接編集する事でも行えます。設定を変えた後は一度クリックして接続を切ってから再度接続して下さい。関係式の編集は通信用部品を選択してからポップアップボタンで部品ダイアログを表示して部品情報ウィンドウを開くことで行います。



右上が関係式（プログラム）部分で、その下がメッセージを変数になります。右上の式を修正してから、Ctrl+Enter キーで更新されます。ウィンドウを閉じた場合にも更新されます。

int PinMode[20] = 1;　はピンモードです。最初は 1 のデジタル出力になっていますので、モードを変更したいピンの値を変更します。サンプルでは Pin5,Pin6 がサーボ、Pin14 から 19 がアナログ入力になっています。これらの設定は Firmata のプロトコルに合わせた設定になっています。変数 din[2]はデジタル入力用、dout[2]はデジタル出力用変数で 1 データが 8bit の 2 要

素の配列になっています。ain[6]はアナログ入力用の変数です。aout_3 から aout_11 はアナログ出力用の変数です。_USBDevice$変数は、USB の通信ポートを示し、子部品のテキストから設定されます。_USBDevice$変数に値が設定されていることで部品（実行部品）の USB 通信が有効になり、@T で値を送信し@D で受信します。@D[0]が受信データ数で@D[1]移行が受信データになっています。以下に USB 通信部品の式を示します。

```
//
// USB 通信コード（StandrdFirmata）　2016/04/19
// ページ 0 で動作開始、ページ 1 で停止。
// @T による送信,@d の受信は実行部品でのみ有効
//
@R=(@P>=1)?255:128;
_USBDevice$;　// 接続用 COM ポート情報
int din[2]; // ディジタル受信データ 8*2 16ch
int ain[6]; // アナログ受信データ 6ch
int dout[2]=-1; // ディジタル送信データ 8*2 16ch
int aout_3=-1; // PWM.Servo 送信データ 6ch
int aout_5=-1; // PWM.Servo 送信データ 6ch
int aout_6=-1; // PWM.Servo 送信データ 6ch
int aout_9=-1; // PWM.Servo 送信データ 6ch
int aout_10=-1; // PWM.Servo 送信データ 6ch
int aout_11=-1; // PWM.Servo 送信データ 6ch
int PinMode[20] = 1; // ピンモード 0:In,1:Out,2;Analog,3:PWM,4:Servo
PinMode[5] = 4;　// Servo　ここを変更することでピンのモードを変えます。
PinMode[6] = 4;　// Servo　ここを変更することでピンのモードを変えます。
PinMode[14:19] = 2; // Analog　ここを変更することでピンのモードを変えます。

if((@d[0] >= 3)&&((@d[1] & 0xF0) == 0x90)){　// ディジタルデータ受信
  din[@d[1] & 0x0f] = (@d[2] & 0x7f) | ((@d[3] & 0x01) << 7),@d[0] -= 3;
}
if((@d[0] >= 3)&&((@d[1] & 0xF0) == 0xE0)){　// アナログデータ受信
  ain[@d[1] & 0x0f] = (@d[2] & 0x7f) | ((@d[3] & 0x7f) << 7),@d[0] -= 3;
}
```

```
if((@d[0] >= 1)&&((d[1] & 0xF0) != 0x90)&&((d[1] & 0xF0) != 0xE0)){
   @d[0] -= 1; //  それ以外のデータ
}
if((@d[0] >= 1)&&((@d[1] & 0x80) == 0)){ //  途中データが残った場合
   @d[0], @d[0] -= 1;
}
@SimStart?InitReq=0;     //  開始時に@P>=1 ならば初期化
@SimStart?InitReq=1;
if((@p >= 1)&&(InitReq)){ //  初期化データ送信
   @t=0xf4,@t=@,@t=PinMode;       // PinMode 送信
   @(16),@t=0xc0+@,@t=1,@t=0xd0+@,@t=1; // set reporting by pin
}
if(dout >= 0){ //  ディジタル出力
   @t=0x90+@,@t=dout&0x7f,@t=((dout&080)?1:0);
}
if(aout_3 >= 0){ // PWM,Servo アナログ出力
   @t=0xe3,@t=aout_3&0x7f,@t=((aout_3&0x3f80)>>7);
}
if(aout_5 >= 0){ // PWM,Servo アナログ出力
   @t=0xe5,@t=aout_5&0x7f,@t=((aout_5&0x3f80)>>7);
}
if(aout_6 >= 0){ // PWM,Servo アナログ出力
   @t=0xe6,@t=aout_6&0x7f,@t=((aout_6&0x3f80)>>7);
}
if(aout_9 >= 0){ // PWM,Servo アナログ出力
   @t=0xe9,@t=aout_9&0x7f,@t=((aout_9&0x3f80)>>7);
}
if(aout_10 >= 0){ // PWM,Servo アナログ出力
   @t=0xea,@t=aout_10&0x7f,@t=((aout_10&0x3f80)>>7);
}
if(aout_11 >= 0){ // PWM,Servo アナログ出力
   @t=0xeb,@t=aout_11&0x7f,@t=((aout_11&0x3f80)>>7);
}
```