PWM 信号受け取って B3M サーボを動かす

1．目的

メルカリで安く買った B3M-SC-1170-A を VS-RC003HV で動かす

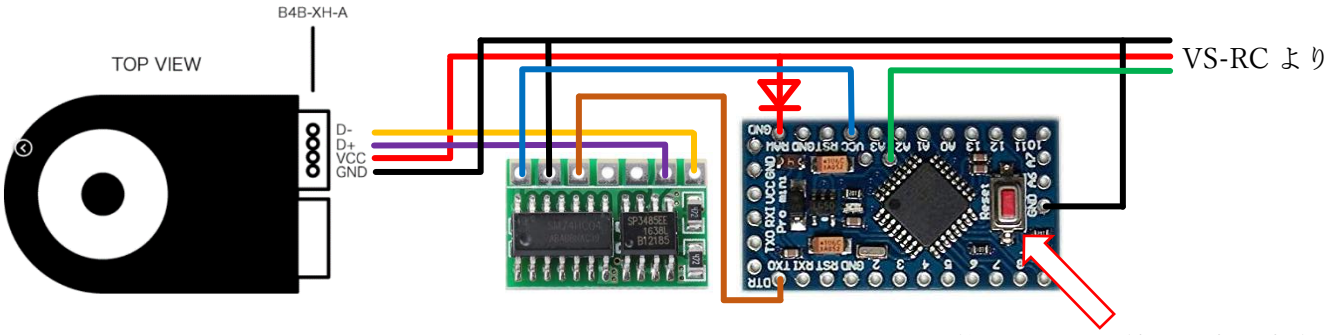
2．製作物の概要

PWM 信号を適当なマイコンで受け，都合のいい値にスケーリング。その後 UART で信号を吐き出し RS485 に変換する基板を通して B3M に信号を送信する。リターンは一切受けない。

3．制御に使用したモノ

・VS-RC003HV
　→ロボット用マイコンボード。ROBO-ONE で一部人気。
・Arduino Pro Mini 5v 16MHz
　→みんな大好き Arduino。なぜか 3.3v のやつだとうまくいかなかった。
・RS485 トランシーバモジュール
　→UART を RS485 に変換するやつ。たぶんなんでもいい。当方は Amazon で¥369 と当時一番安かった「トランシーバモジュール ミニトランシーバモジュール ワイヤレス Bluetooth イーサネットトランシーバモジュール R411B01 3.3V 5V UART シリアル to RS485 SP3485(3V3)」を購入したが，いまは¥1599 となぜか値段が約 5 倍に上がっているのでお勧めできない。

4．構成



黒・・・GND
赤・・・Vcc。今回は Lipo 3 S を VS-RC 経由で接続。Arduino にはダイオード経由で給電。
　　※ダイオードは Vcc を一方通行にするのが目的なので何でもよい。１Ｎ４１４８とか。
緑・・・PWM 信号。
青・・・5v。Arduino のレギュレータから RS485 変換基板へ電源を給電。
茶・・・UART の TX。
黄・・・RS485 の D―。
紫・・・RS485 の D＋。

※画像の引用元は最終ページ下。

5．Arduino のスケッチ（次ページ）

他人に見せる前提で書いていないので，ご了承ください。

```
#define PWM_IN_PIN 18 //WPM を入力するピン
#define d_band 70 //PWM の揺れを無視する幅
#define GAIN 9.0 //PWM 信号とサーボの移動角度の比を調整

int last = -30000; //最後に送った位置情報を格納

void setup() {
  Serial.begin(115200, SERIAL_8N1); //ハードウェアシリアル通信の開始
  pinMode(PWM_IN_PIN, INPUT); //PWM ピンを入力ピンに設定
  move_b3m(0, -32768); //とりあえず B3M に脱力信号を送信
  delay(1000); // 1 秒待つ。1 ミリ秒まで短縮してよい。
}

void loop() {
  int target = (pulseIn(PWM_IN_PIN, HIGH, 30000) - 1500.0) * GAIN;
    //PWM 信号を入力し、都合の良い値に調整する。
  if (target >= last - d_band && target <= last + d_band )target = last;
    //PWM 信号の揺れを無視する範囲内に収まってれば前の値をそのまま上書き。
  move_b3m(0, target); //B3M に信号送信
  last = target; //最後に送った値を格納
}

void move_b3m(byte id, int target) {
  byte move_tx[9] = {0x09, 0x04, 0x00, 0x00, 0x05, 0x00, 0x2A, 0x01, 0x3D};
    //位置情報用の配列の宣言とデフォルトの値の格納
  byte power_tx[9] = {0x09, 0x04, 0x00, 0x00, 0x00, 0x00, 0x28, 0x01, 0x36};
    //トルク情報用の配列の宣言とデフォルトの値の格納
  int sum;
    //チェックサム用の変数の宣言

  move_tx[3] = id; //ID 情報を位置情報用配列に代入
  power_tx[3] = id; //ID 情報をトルク情報用配列に代入

  if (target < -13499 || target > 15000 ) { //要求された位置情報が範囲外なら
    power_tx[4] = 0x02; //トルク情報を脱力に設定
    sum = 0; //チェックサム用の変数を初期化
    for (byte i = 0; i <= 7; i++) sum += power_tx[i]; //チェックサム用に配列内の総和を計算。
    power_tx[8] = (byte)(sum & 0x00FF); //トルク情報配列のチェックサムを代入。
    Serial.write(power_tx, 9); //信号送信
    delayMicroseconds(1500); //1500 マイクロ秒待つ
  }
  else {
    power_tx[4] = 0x00; //トルク情報にトルクオンを設定
    sum = 0;   //チェックサム用の変数を初期化
    for (byte i = 0; i <= 7; i++) sum += power_tx[i]; //チェックサム用に配列内の総和を計算。
    power_tx[8] = (byte)(sum & 0x00FF); //トルク情報配列のチェックサムを代入。
    Serial.write(power_tx, 9); //信号送信
    delayMicroseconds(1500); //1500 マイクロ秒待つ
    move_tx[4] = (byte)(target & 0x00FF); //位置情報下位ビットを配列に代入
    move_tx[5] = (byte)((target & 0xFF00) >> 8); //位置情報上位ビットを配列に代入
    sum = 0;   //チェックサム用の変数を初期化
    for (byte i = 0; i <= 7; i++) sum += move_tx[i]; //チェックサム用に配列内の総和を計算。
    move_tx[8] = (byte)(sum & 0x00FF); //位置情報配列のチェックサムを代入。
    Serial.write(move_tx, 9); //信号送信
    delayMicroseconds(1500); //1500 マイクロ秒待つ
  }
}
/*
サーボの画像：https://kondo-robot.com/faq/b3msb1040a-spec
Arduino の画像：https://www.amazon.co.jp/waves-Arduino-%E4%BA%92%E6%8F%9B%E3%83%9C%E3%83%BC%E3%83%89-16MHz-2%E5%80%8B%E3%82%BB%E3%83%83%E3%83%88/dp/B01CX2APN4
RS485 変換基板の画像：
https://www.amazon.co.jp/%E3%83%88%E3%83%A9%E3%83%B3%E3%82%B7%E3%83%BC%E3%83%90%E3%83%A2%E3%82%B8%E3%83%A5%E3%83%BC%E3%83%AB-%E3%83%9F%E3%83%8B%E3%83%88%E3%83%A9%E3%83%B3%E3%82%B7%E3%83%BC%E3%83%90%E3%83%A2%E3%82%B8%E3%83%A5%E3%83%BC%E3%83%AB-%E3%83%AF%E3%82%A4%E3%83%A4%E3%83%AC%E3%82%B9Bluetooth%E3%82%A4%E3%83%BC%E3%82%B5%E3%83%8D%E3%83%83%E3%83%88%E3%83%88%E3%83%A9%E3%83%B3%E3%82%B7%E3%83%BC%E3%83%90%E3%83%A2%E3%82%B8%E3%83%A5%E3%83%BC%E3%83%AB-R411B01-UART%E3%82%B7%E3%83%AA%E3%82%A2%E3%83%ABto/dp/B07RQRJDRS/
*/
```