モータードライバーシールド
スタートマニュアル

はじめに

モータードライバーシールド(SU-1201)は、Arduinoによって動作させることを前提としています。このマニュアルでは、モータードライバーシールドを動かすときの具体的な設定や、サンプルプログラムについて解説しています。

●対応するArduino　2012年9月1日現在

Arduino基板:　Duemilanove、 UNO(R3)

Arduino-IDE:　バージョン1.0.1

※Arduinoを使用するにはUSBポートのあるパソコンとUSBケーブルが必要です。

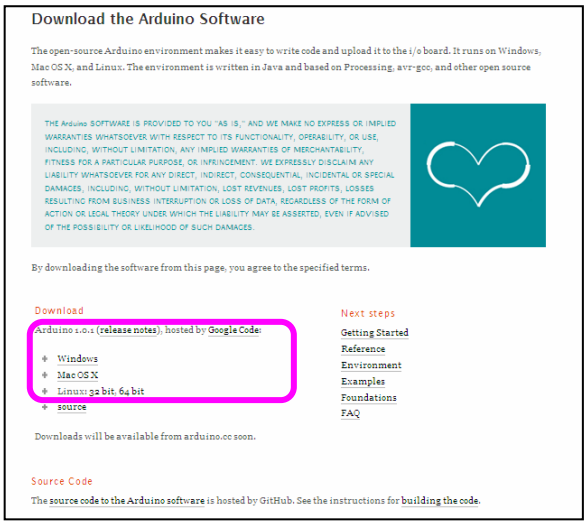
Arduinoプログラム開発環境の準備

Arduinoの開発環境を使って、プログラムを作成し、Arduino基板への書き込みを行います。この開発環境のことを「Arduino-IDE」と言います。

Arduino-IDEの入手と起動

1.まず、Arduinoのホームページにアクセスします。http://www.arduino.cc/ (英語のサイトです。)

2.ダウンロード画面から、お使いのOSに対応したソフトを選択し、適当なフォルダーに保存します。

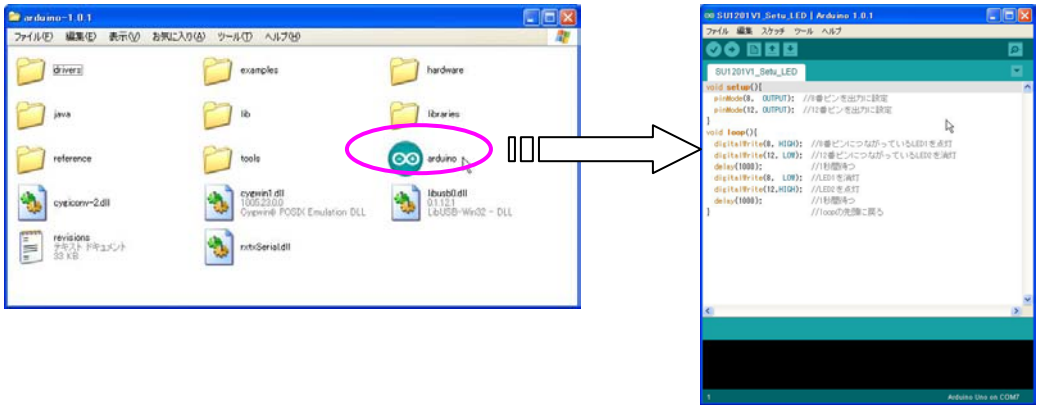


※画面は変わる可能性があります。

3.ダウンロード後、保存したフォルダー内に、arduino-X.X.X-XXX が作成されます。このファイルは圧縮されているので、解凍ソフトで解凍してください。
※XXX は使用する OS やソフトのバージョンによって異なります。

4.ファイルを解凍すると、arduino-1.0.1 のように、バージョン番号が付いたフォルダーが作成されます。
5.先ほど解凍したフォルダー内にある、arduino のアイコンをダブルクリックすると、Arduino-IDE が起動し画面が表れます。

（arduino フォルダーの中）　　（IDE 画面）

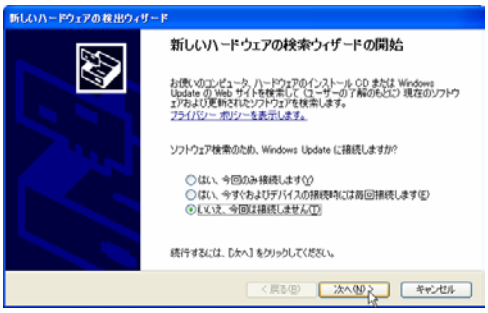


ドライバーのインストールとシリアルポート（COM ポート）の確認　　Windows の場合

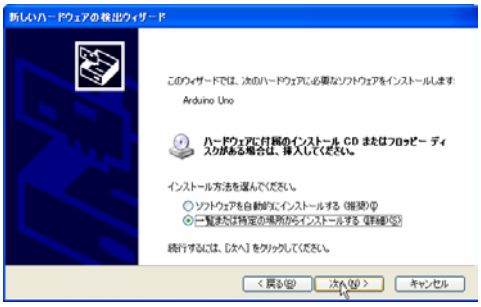
次に、Arduino 基板とパソコンが通信するために必要なドライバーをインストールします。

1.Arduino 基板とパソコンを USB ケーブルで接続します。
2.WindowsXP の場合：

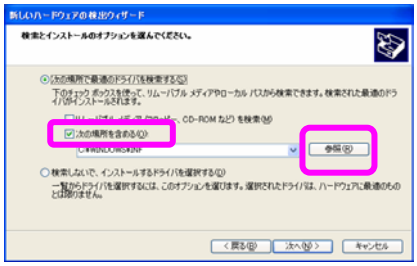
接続後に「新しいハードウェアの検出」画面が表示されますので、
「いいえ。今回は接続しません。」を選択して次へ進みます。

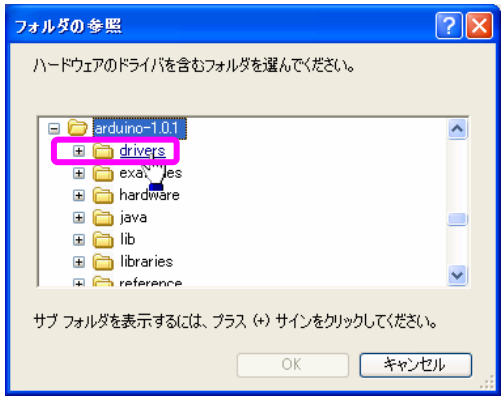


「一覧または特定の場所からインストールする。」を選択して次へ進みます。



「次の場所を含める。」で、参照ボタンをクリックして、先ほど解凍したフォルダーの中にある「drivers」を選択して、次へ進みます。



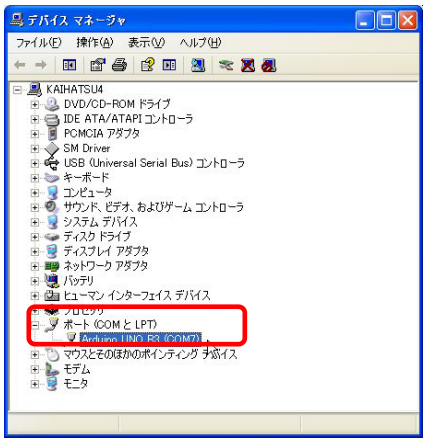


ドライバーのインストールが始まります。自動的に完了します。
完了後に、もう一度ドライバーのインストールを求められる場合がありますので、
1 回目と同じように進めて完了させます。

WindowsVista/7 の場合：

「デバイスドライバーソフトウェアをインストールしています。」と表示され、しばらく待つと、パソコンが自動でドライバーを検索しインストールが完了します。
もしも自動でインストールができない場合は手動でインストールする必要があります。
＊トラブルシューティング「ドライバーを手動でインストールする。」をお読みください。

3.次に Arduino 基板が、どのシリアルポート（COM ポート）に接続されたか確認します。
WindowsXP では、マイコンピュータのアイコンを右クリックして「プロパティー」を選択し、「ハードウェア」のタブの中にある「デバイスマネージャー」をクリックします。
WindowsVista/7 では、「コントロールパネル」→「システムとセキュリティ」→「デバイスマネージャー」と進みます。
デバイスの一覧が表示されますので、「ポート（COM と LPT）」という項目の下を確認します。



Arduino 基板が接続されたポートが表示されています。
図では Arduino 基板が「COM7」に接続されていることが分かります。この COM の番号を覚えておきます。

4.Arduino-IDE 画面の「ツール」メニューから、「マイコンボード」を選択し、パソコンに接続した Arduino 基板を選択します。

5.次に Arduino-IDE 画面の「ツール」メニューから、「シリアルポート」を選択し、先ほど確認した COM ポートの番号と同じものを選択します。

これで、Arduino を使用する準備が整います。

ドライバーのインストールとシリアルポート（COM ポート）の確認　　Mac OS X の場合

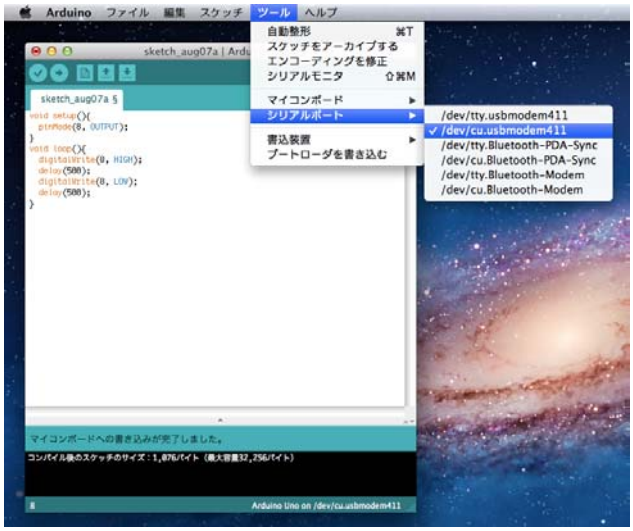
Mac OS X の場合、ダウンロードが完了すると自動的にドライバーがインストールされますので、表示されるメッセージに従い必要に応じて管理者パスワードの入力や再起動を行ってください。

Mac OS X でシリアルポートを選択する。

1.Arduino 基板をパソコンと接続します。

2.Arduino-IDE を起動し、画面の「ツール」メニューから、「マイコンボード」を選択し、パソコンに接続した Arduino 基板を選択します。

3.次に Arduino-IDE 画面の「ツール」メニューから、「シリアルポート」を選択し、
「/dev/cu.usbmodem-」 または「/dev/tty.usbmodem-」
ではじまる項目を選んでください。



これで、Arduino を使用する準備が整います。

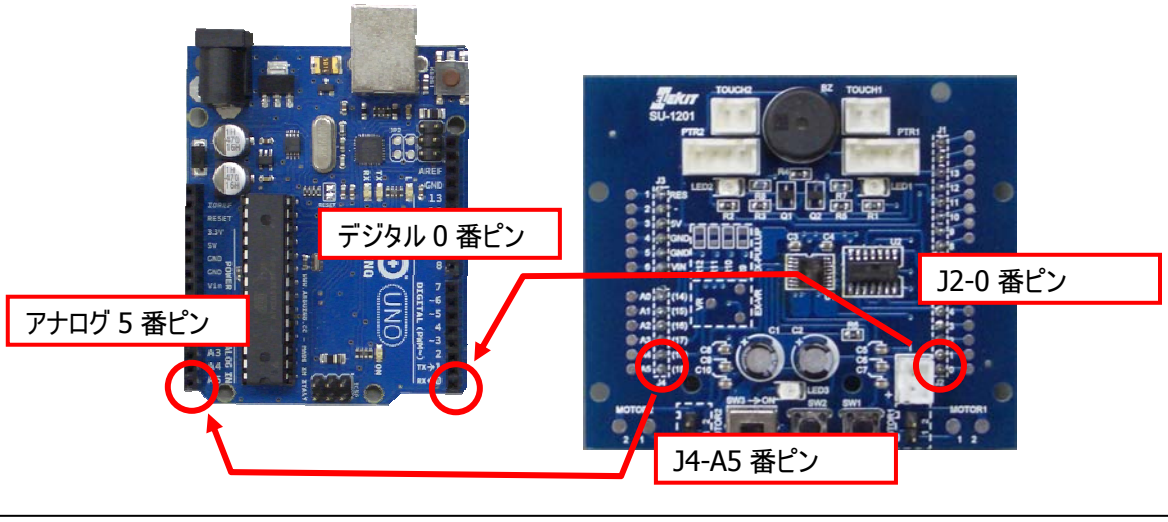
モータードライバーシールドの取り付け

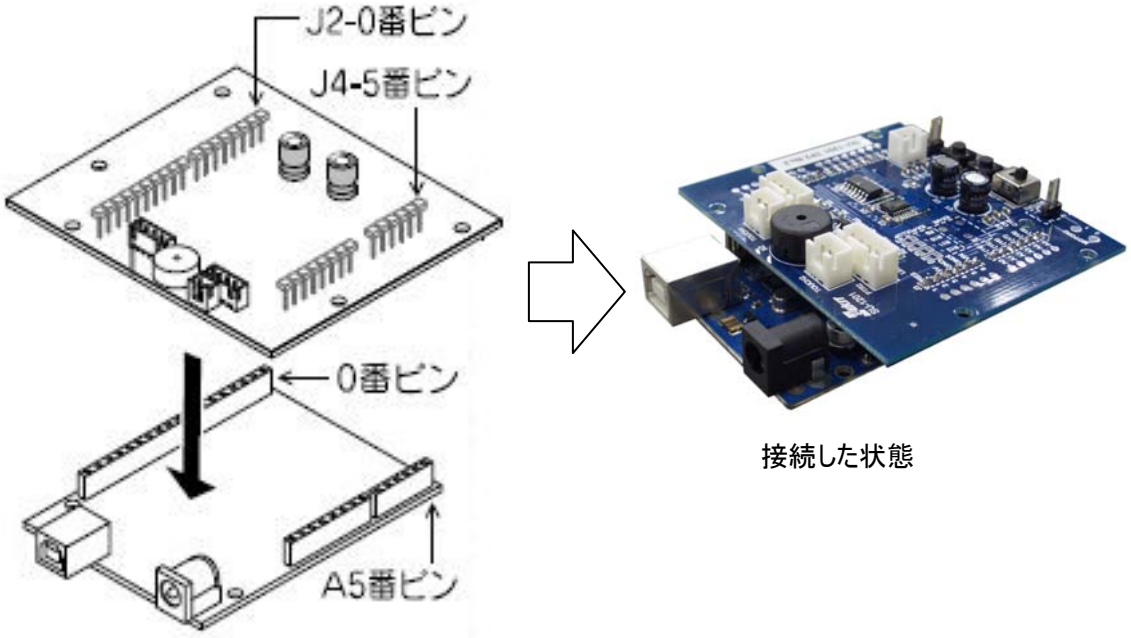
モータードライバーシールド（以下「シールド基板」といいます。）を Arduino 基板に取り付けます。

気を付けよう

シールド基板を Arduino 基板へ取り付けたり取り外したりするときは、パソコンと接続しているケーブルを外してください。ケーブルでパソコンとつながっているときは、Arduino 基板に電流が流れていますので、もしもシールド基板を誤って取り付けたり、Arduino 基板の上に取り落としたりすると、最悪の場合ショートして、基板が壊れる場合があります。

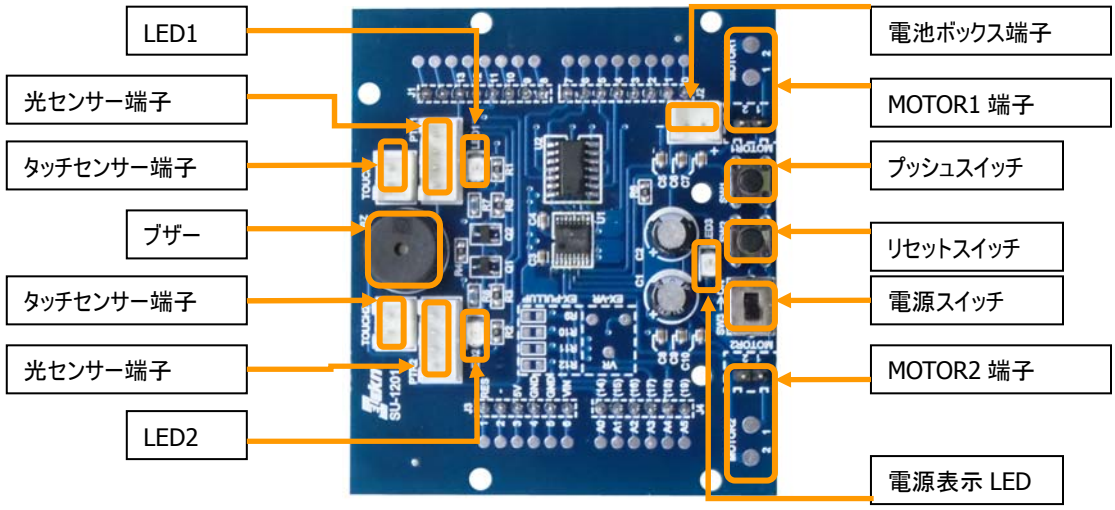
シールド基板の J4-A5 番ピンと、Arduino 基板のアナログ A5 番ピン。
シールド基板の J2-0 番ピンと、Arduino 基板のデジタル 0 番ピン。
を合わせて奥までしっかり差し込みます。





接続した状態

モータードライバーシールドの各部の名称



MOTOR1 端子 MOTOR2 端子	モーターを接続する端子です。モーターのコードをはんだ付けして使用します。 モーターのコネクター形状が合っていれば、ピン端子を使うこともできます。
電源スイッチ	電池ボックスからの電源を ON/OFF します。※「電源について」をお読みください。
リセットスイッチ	Arduino 基板をリセットします。 （Arduino 基板上のリセットスイッチと同じ働きをします。）
プッシュスイッチ	Arduino 基板のデジタルピン 3 に接続されています。
電池ボックス接続端子	電池ボックスを接続する端子です。 重要 電池ボックスを使う場合は、Arduino 基板の DC ジャックに電源を接続しないでください。基板が破損する可能性があります。
LED1	Arduino 基板のデジタルピン 8 で制御する LED です。
LED2	Arduino 基板のデジタルピン 12 で制御する LED です。
電源表示 LED	モータードライバーIC に電源が供給されているときに点灯します。
ブザー	Arduino 基板のデジタルピン 11 で制御するブザーです。
光センサー端子 タッチセンサー端子	当社キロボ（MR-9132）のセンサーを接続するときに便利なコネクターです。

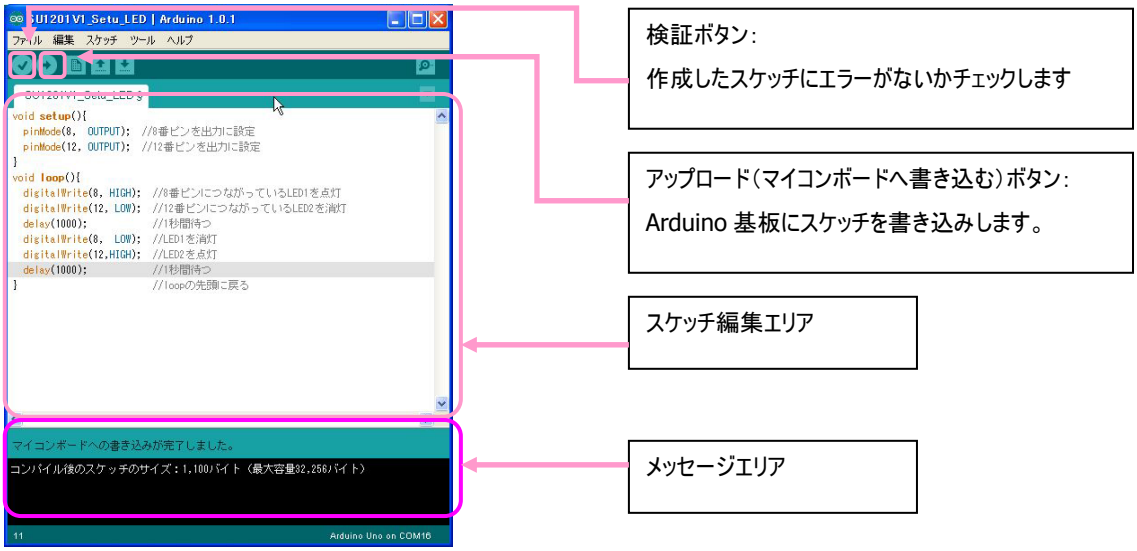
※ 電源について

Arduino 基板とパソコンを USB ケーブルで接続しているときは（例えばスケッチをアップロードするときは）、USB 端子から電源が供給されます。その状態でモーターが動くと、モーターに大きな電流が流れますので、USB 端子を破損する可能性があります。

パソコンと USB ケーブルで接続中に、モーターを動かす必要がないときは、電源スイッチを OFF にしておくことをお勧めします。シールド基板の電源スイッチが OFF の位置であってもスケッチのアップロードやブザー、LED、プッシュスイッチは動作します。

パソコンと接続していないときは、電源スイッチを ON にするとスケッチを実行します。

Arduino-IDE の基本画面

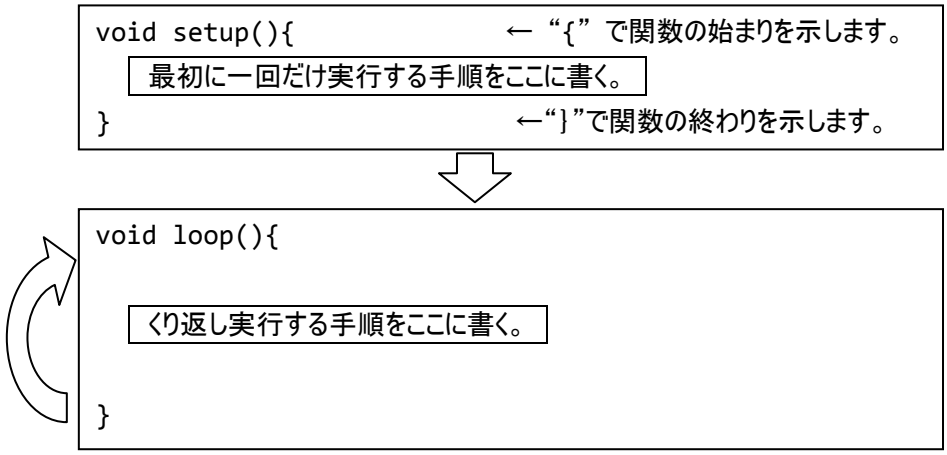


Arduino スケッチの基本

Arduino-IDE で作成したプログラムのことを「スケッチ」といいます。作成したスケッチをアップロードすると、Arduino 基板のマイコンの中に書き込まれます。書き込まれたスケッチは、基板の電源を ON/OFF しても消えません。書き込まれたスケッチは、基板の電源を ON 後、またはリセット後に、自動で実行されます。

Arduino のスケッチには、「setup」と「loop」という 2 つの関数が必要です。
関数とは、Arduino 基板上のマイコンに、何をどんな順番で行わせるかを記述した手順書です。
電源 ON 後に、「setup」の中の手順を一回だけ実行し、続けて、「loop」の中の手順を、電源を切るまでくり返し実行するということを覚えてください。

スケッチを実行するイメージ



void setup()、void loop() って何？と思うかもしれませんが、これは Arduino-IDE で、setup 関数と loop 関数を使うときの決められた形です。
Arduino-IDE の関数や命令の詳細については書籍などで学習を行ってください。

サンプルスケッチ1 ： LED を点滅させる

ここでは、「シールド基板の LED1 を 1 秒間隔で点滅させるスケッチ。」を考えます。

知っておくこと:

・シールド基板上の LED1は、Arduino のデジタル 8 番ピンにつながっています。

・8 番ピンを H にすると（5V を出力すると）LED が点灯します。L にすると（0V を出力すると）消灯します。

考え方:

LED を点滅させるためには、Arduino のマイコンから 5V を出力したり、0V を出力したりします。

まずマイコンの 8 番ピンを「出力」に設定しなければいけません。

スケッチの作成:

LED を点灯させたら、1 秒間そのまま光らせておく。

1 秒経過したら、LED を消灯させ、1 秒間そのまま消しておく。

この動作を繰り返す。

というスケッチを作成すれば良さそうだとイメージして、実際にスケッチを作成します。

ほとんどの場合、最初に一回だけ設定したいことは、「setup 関数」の中に書きます。希望の動作を実現するためには、どのピンを利用するのか、そのピンは出力にするのか入力するのかなどを設定します。

今回の LED を点滅させたい場合の setup の中味は図 1 のようになります。

```
void setup() {
  pinMode(8, OUTPUT);
}
```

図 1

まず、マイコンに、「8 番ピンを出力にしなさい。」と命令しなければいけません。

そのための命令が pinMode です。

pinMode（8, OUTPUT）と書くことで、8 番ピンを出力に設定します。

一つの命令が終わったら、そのことを示す「;」（セミコロン）を書きます。

これで、マイコンの 8 番ピンの設定は終わります。次にくり返し動作させたい部分を作成します。

くり返し動作させたいことは、「loop 関数」の中に書きます。

LED を点滅させたい場合の loop の中のスケッチは図 2 のようになります。

```
void loop() {
  digitalWrite(8, HIGH);
  delay(1000);
  digitalWrite(8, LOW);
  delay(1000);
}
```

図 2

解説:

まず、マイコンに「8 番ピンから、H(5V)を出力しなさい。」と命令しなければいけません。
そのための命令が digitalWrite です。
digitalWrite(8, HIGH); と書くことで、マイコンの 8 番ピンから H が出力されます。
8 番ピンが H になるとそこにつながっているシールド基板上の LED1 が点灯することになります。

次に、LED1 を点灯させておく時間を命令します。
delay(1000); と書くと、マイコンは、今の状態そのままで、1000 ミリ秒つまり 1 秒待ちます。
マイコンが 1 秒数えてから次の命令に進むようなイメージです。

次に、マイコンに「8 番ピンから、L(0V)を出力しなさい。」と命令します。
digitalWrite(8, LOW); と書くことで、マイコンの 8 番ピンから L が出力されます。
8 番ピンが L になるとそこにつながっているシールド基板上の LED1 が消灯することになります。
消灯した状態を 1 秒間続けたいので、先ほどと同じ delay 命令を使います。
delay(1000); と書くと、消灯した状態で 1 秒数えて次に進みます。

次に命令はありませんので、マイコンが自動的に loop の一番先頭に戻り、loop の中を繰り返します。

スケッチのアップロード

作成したスケッチを Arduino 基板上のマイコンに書き込むことをアップロードと言います。
サンプル1のスケッチを作成して、実際にアップロードをやってみましょう。

1. 図 3 のスケッチを間違えないように、全く同じように、作成してください。

気を付けよう

文字や記号は全て「半角英数」を使います。全角文字ではスケッチに異常があると判断されてしまい、マイコンに書き込むことができずエラーと表示されます。特にスペース(空白)を全角で入力しないように注意しましょう。全角スペースは見て分かりづらく、発見の難しいエラーになってしまいます。
※ 後述する「コメント」の中だけは全角文字を使うことができます。

```
void setup() {
  pinMode(8, OUTPUT);
}
void loop() {
  digitalWrite(8, HIGH);
  delay(1000);
  degitalWrite(8, LOW);
  delay(1000);
}
```

図 3

2. スケッチの入力が終わったら、シールド基板の電源スイッチを OFF にして、USB ケーブルで Arduino 基板とパソコンを接続し、パソコンの画面上の「アップロードボタン」をクリックします。
   しばらく待つと、メッセージ表示エリアに、「マイコンボードへの書き込みが完了しました。」とメッセージが表示されます。
   もしも違うメッセージが表示された場合は、入力したスケッチに間違いがないか再確認します。
   { 、 }、; を忘れていないか、文字は半角英数になっているかをチェックしてみましょう。

3. アップロード完了後（マイコンボードへの書き込みが完了後）、シールド基板の LED1 が約 1 秒間隔で点滅していることを確認できればアップロードは成功です。

気を付けよう

USB ケーブルでパソコンと接続している状態では、シールド基板の電源スイッチが OFF であってもスケッチは実行されます。

実験しておこう

今は、delay 命令を使って、点灯する時間を 1 秒、消灯する時間を 1 秒としています。
この秒数をいろいろな値に変える実験をして、アップロードして、光り方の変化を確認しましょう。
例えば、点灯時間を 50 ミリ秒、消灯時間を 50 ミリ秒にしてみる。
　　点灯時間を 1 ミリ秒、消灯時間を 20 ミリ秒にしてみる。
と変えてみて光り方の変化を確認してみましょう。

スケッチの実行を止める

スケッチは電源を切るまでくり返し実行されます。スケッチの実行を止めるには電源を切ります。

①USB ケーブルでパソコンと接続している場合。
シールド基板の電源スイッチを OFF にして、USB ケーブルを抜きます。

パソコンと USB ケーブルで接続しているときは、USB から電源が供給されていますので、シールド基板の電源スイッチを OFF してもスケッチの実行は止まりません。

②USB ケーブルでパソコンと接続していない場合。（例えば乾電池で動かしている場合。）
シールド基板の電源スイッチを OFF にします。

サンプルスケッチ２ ： ブザーを鳴らす

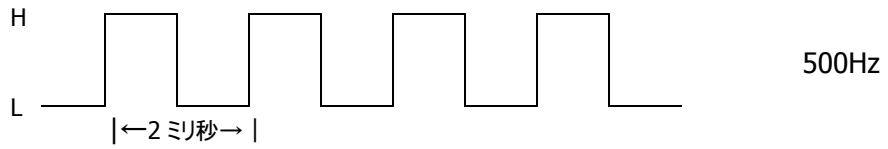
ここでは、「シールド基板上のブザーを、0.5 秒鳴らして 0.5 秒休む。これを繰り返す。」を考えます。

知っておくこと:

- シールド基板上のブザーは、Arduino のデジタル 11 番ピンにつながっています。
- 一般的にブザーを鳴らすためには、ブザーに数百 Hz～数kHz の周波数の電圧(※1)を、鳴らしたい時間加えます。逆に鳴らさないときは、ブザーの端子を 0V にします。
- 数百 Hz の周波数の電圧を出力するときに便利な、PWM 機能(※2)を使います。Arduino の PWM の周波数は約 500Hz または 1kHz です。（基板の種類によって違います。）
-

※1 周波数について

数百 Hz～数 KHz の周波数とは、1 秒間に何回 H→L→H→L→・・・と電圧が切り替わるかを表します。H になったときから、次に H になるまでの時間周期で周波数が決まります。例えば、H から H までの時間が 2 ミリ秒(=0.002 秒)であれば、
1 秒間÷0.002 秒=500 となり、周波数は 500Hz ということになります。



考え方:

ブザーを鳴らすための信号を出力するので、11 番ピンを出力に設定します。
PWM 機能を使って、11 番ピンから H→L→H・・・と出力してブザーを鳴らします。

スケッチの作成:

まずブザーを鳴らした状態にします。その状態で 0.5 秒間待ちます。
次に 11 番ピンを L（0V）にして、ブザーを止めます。ブザーを止めた状態で、0.5 秒待ちます。
この動作を繰り返す。
という動作をイメージしてスケッチを作成します。

```
void setup() {
  pinMode(11, OUTPUT);     //ピン 11 を出力にする
}
void loop() {
  analogWrite(11, 128);    //ブザーを鳴らす
  delay(500);               //0.5 秒待つ
  analogWrite(11, 0);       //ブザーを止める
  delay(500);              //0.5 秒待つ
}
//スケッチはここまで。loop の先頭に戻り繰り返す
```

サンプル1と同じ考え方です。
setup 関数の中で
11 番ピンを出力に設定します。

図 4

解説:

loop 関数の中では、PWM 機能を使うための命令 analogWrite を使っています。
analogWrite(11, 128); と書くと、11 番ピンを PWM で動作させることができ、11 番ピンから自動的に H,L の信号がくり返し出力され、ブザーが鳴り始めます。

次にブザーを鳴らす時間を決めます。ブザーが鳴っている状態で 0.5 秒間待ちたいので、delay(500); と書きます。この命令で 500 ミリ秒つまり 0.5 秒間ブザーを鳴らした状態を続けます。

次にブザーを止める命令です。analogWrite(11, 0); と書くと 11 番ピンからは H が出力されなくなります。つまりブザーから音が出なくなります。

次に、ブザーが鳴っていない状態で 0.5 秒間待ちたいので、delay(500);と書きます。
この命令の後は loop の先頭に戻り、ブザーの動作を繰り返します。

サンプル1と同じように、スケッチを作成して、アップロードを行い、動作を確認しておきましょう。

「コメント」 を記入しよう

図 4 をよく見ると、 // という記号の後に、スケッチの動作が日本語で記入されています。
この // に続けて書かれた文字はコメントといって、Arduino から無視されます。
スケッチの説明や、その命令が何をしているのかなどの短いメモを記入するときに使います。

※2 PWM 機能について

(1) Arduino 基板は PWM 機能が使えるピンと使えないピンがあります。シールド基板では Arduino 基板の 9 番、10 番、11 番ピンの PWM 機能を使っています。

(2)ここでは PWM 機能の概要や詳細については説明しませんが、モーターの制御にも PWM 機能を使用しますので、以下の動作のイメージを覚えておきましょう。

PWM 機能を使うときは、 analogWrite(ピンの番号,0～255 の範囲の値); と書きます。
ここで設定する 0～255 の範囲の値のことを「デューティ値」と言います。
デューティ値の設定でどのように出力が変わるのか図にしていますので、イメージを覚えておきましょう。

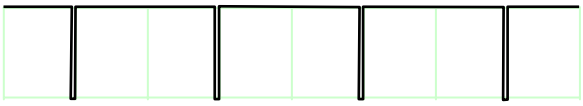
図 5 analogWrite(ピン番号, 1); のとき

デューティ値＝1

図 6 analogWrite(ピン番号, 128); のとき

デューティ値＝128

図 7 analogWrite(ピン番号, 254); のとき



デューティ値＝254

図 8 analogWrite(ピン番号, 0); のとき

デューティ値＝0

図 9 analogWrite(ピン番号, 255); のとき

デューティ値＝255

サンプルスケッチ3 ： モーターを動かす

ここでは、「 モーター1 正転（速く） → モーター1 ブレーキ →
モーター1 反転（ゆっくり） → モーター1 ブレーキ →
モーター2 正転（速く） → モーター2 オフ 」
という動作を順番に、それぞれ 1 秒ずつ行う。」を考えます。

知っておくこと：

- Arduino 基板の 4 番、5 番、9 番ピンで Motor-1 の動作を制御します。
- Arduino 基板の 6 番、7 番、10 番ピンで Motor-2 の動作を制御します。
- ブレーキは、モーターの端子に電気的にブレーキを掛けるのでピタッと止まります、オフはブレーキを掛けずに止まるので少し空回りする感じで止まります。
- スピードの制御には、9 番ピンと 10 番ピンの PWM 機能を使用します。
- Arduino のピンの設定と、モーターの回転方向は以下の表の通りになっています。

モーター1		
4 番ピン	5 番ピン	動作
LOW	LOW	ブレーキ
LOW	HIGH	正転
HIGH	LOW	反転
HIGH	HIGH	オフ

モーター2		
6 番ピン	7 番ピン	動作
LOW	LOW	ブレーキ
LOW	HIGH	正転
HIGH	LOW	反転
HIGH	HIGH	オフ

※ PWM ピン（9 番、10 番）が LOW のときは、常にブレーキ動作になります。

気を付けよう

パソコンの USB から電源を供給している場合は、モーターを動かすことは避けてください。
パソコンに接続したままモーターを動かしたい場合は、Arduino 基板の DC ジャックやシールド基板の電池ボックス接続端子に電源を接続してください。
シールド基板に使用できるモーターは、モーター1/2 合わせて 1.4A 以内です。

考え方：

最初にモーターを制御するためのピンを出力に設定します。
次に実際の動作について考えます
モーターの回転方向やブレーキ、オフは、表に従ってピンの出力を H や L に設定すれば OK です。
モーターのスピード制御にはブザーのサンプルで使った PWM 機能=analogWrite 命令を使用します。
analogWrite 命令のデューティ値の設定でスピードが変わります。最大の 255 を設定すると、モーターのスピードが最大に、デューティ値を小さくしていくと、モーターのスピードもだんだん遅くなります。

気を付けよう

デューティ値をどんどん小さくすると、モーターが遅く動ける限界以下になり動かなくなります。使用するモーターの限界は実験して確かめましょう。

スケッチの作成：

```
void setup(){
  pinMode(4, OUTPUT);      //モーター1用のピンの出力設定
  pinMode(5, OUTPUT);      //モーター1用のピンの出力設定
  pinMode(9, OUTPUT);      //モーター1用のピンの出力設定
  pinMode(6, OUTPUT);      //モーター2 用のピンの出力設定
  pinMode(7, OUTPUT);      //モーター2 用のピンの出力設定
  pinMode(10, OUTPUT);     //モーター2 用のピンの出力設定
}
void loop() {
  analogWrite(9, 255);     //PWM 値を最大に設定
  digitalWrite(4, LOW);    //モーター1 を正転に設定
  digitalWrite(5, HIGH);   //モーター1 を正転に設定
  delay(1000);             //1 秒の動作待ち時間

  analogWrite(9, 0);       //PWM 値を 0 でブレーキ
  delay(1000);             //1 秒の動作待ち時間

  analogWrite(9, 50);      //PWM 値を 50 でゆっくり回転
  digitalWrite(4, HIGH);   //モーター1 を反転に設定
  digitalWrite(5, LOW);    //モーター1 を反転に設定
  delay(1000);             //1 秒の動作待ち時間

  analogWrite(9, 255);     //PWM 値を最大に設定
  digitalWrite(4, HIGH);   //モーター1 をオフに設定
  digitalWrite(5, HIGH);   //モーター1 をオフに設定
  delay(1000);             //1 秒の動作待ち時間

  analogWrite(10, 255);    //PWM 値を最大に設定
  digitalWrite(6, LOW);    //モーター2 を正転に設定
  digitalWrite(7, HIGH);   //モーター2 を正転に設定
  delay(1000);             //1 秒の動作待ち時間

  analogWrite(10, 0);      //PWM 値を 0 でブレーキ
  delay(1000);             //1 秒の動作待ち時間
}
```

モーターを制御するピンの
初期設定です。

モーター1 を、
最大スピードで正転させます。

モーター1 にブレーキを掛けて
停止します。

モーター1 を、
ゆっくり反転させます。

モーター1 をオフします。

モーター2 を、
最大スピードで正転させます。

モーター2 にブレーキを掛けて
停止します。

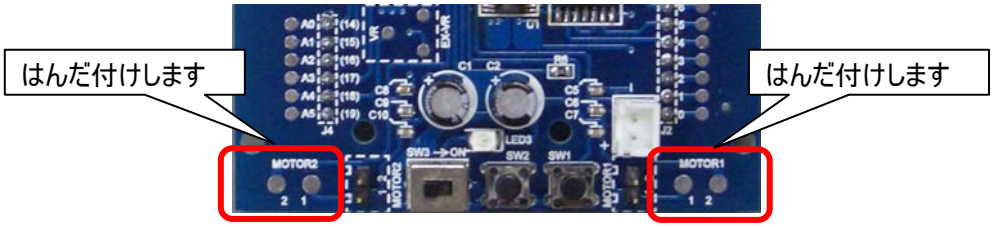
解説:

loop 関数の中では、digitalWrite と analogWrite を使って、モーターの動作を制御しています。
例えばモーター1 を速く正転させるときには、
9 番ピンの PWM のデューティ値を最大の 255 にして、4 番ピンから L を、5 番ピンから H を出力します。
ブレーキを掛けるときは、
9 番ピンの PWM のデューティ値を最小の 0 にします。4 番ピン、5 番ピンは関係しません。
モーターをオフにするときは、
9 番ピンの PWM のデューティ値を最大の 255 にして、4 番ピンと 5 番ピンから H を出力します。
あとはそれぞれの状態を 1 秒続けるように、delay(1000);を使っています。

モーターの準備

モーターを使う場合は、USB を電源として使うことを避けた方がよいので、Arduino 基板の DC ジャックやシールド基板の電池ボックス接続端子に電源を接続します。
このサンプルでは、モーター1、2接続端子にモーターを接続しているものとしています。

モーターの配線は以下の場所にはんだ付けしてください。



※当社製品の KIROBO(MR-9132)のモーターや電池ボックスをお持ちの場合は、モーターのコネクタをモーター1、2 接続端子に、電池ボックスのコネクタを電池ボックス接続端子に差し込みます。

スケッチの実行手順

モーターを動かすスケッチを作成した場合は、以下のような手順で、接続～アップロード～実行を行うと良いでしょう。

1. シールド基板の電源スイッチを OFF にして、USB ケーブルでパソコンと接続します。
2. (作成したスケッチを)アップロードして、完了のメッセージを確認します。
3. USB ケーブルを抜きます。
4. シールド基板の電源スイッチを ON にしてスケッチを実行します。

手順に従い、モーターを動かすサンプルも、アップロードを行い、動作を確認しておきましょう。

サンプルスケッチ4 ： プッシュスイッチを使う

ここでは、シールド基板上のプッシュスイッチ「SW1」を使った動作例を紹介します。SW1 を使う場合は、「入力」の設定や、SW1が今どんな状態なのかを検出する命令を知ることが必要です。
例として、「SW1 が押されている間は LED1を点灯。押されていないときは LED1 を消灯。」を考えます。

知っておくこと:

- シールド基板上の SW1 は Arduino のデジタル 3 番ピンにつながっています。
- SW1 を使う場合は、「プルアップ機能」(※3)を使う必要があります。
- プルアップ機能を使うと、SW1 が押されていないときは、3 番ピンは H になっていて、SW1 が押されている間は L になります。

考え方:

SW1 がつながっている 3 番ピンを「入力」設定します。
LED1 を使うので、8 番ピンを「出力」に設定します。

スケッチの作成:

もしも 3 番ピンが H ならば、LED1 を点灯。3 番ピンが L ならば LED1 を消灯。
と考えればよいことになりますので、実際のスケッチを作成すると次のようになります。

```
int x;                          //このスケッチで「x」という変数を使いますという宣言
void setup() {
  pinMode(3, INPUT_PULLUP);     //3 番ピンを入力にして、プルアップ機能を使う。
  pinMode(8, OUTPUT);           //8 番ピンを出力にする。
}
void loop() {
  x = digitalRead(3);           //3 番ピンの状態をチェックして、x に記憶する。
  if(x == LOW){                 //もしも x が LOW なら、(つまり SW1 が押されていれば、)
    digitalWrite(8, HIGH);      //LED1 を点灯。
  } else {                      //そうでなかったら(x が HIGH つまり SW1 が押されていなければ。)
    digitalWrite(8, LOW);       //LED を消灯する。
  }
}
```

解説:

まず「3 番ピンを入力にしなさい。」と命令します。そのための命令は、pinMode(3, INPUT); と書けば良さそうですが、シールド基板で SW1 を使うときは「プルアップ機能」を使う必要がありますので、
pinMode(12, INPUT_PULLUP); と書きます。（「 _ 」 アンダーバーを忘れないように注意します。）
LED1を使うので 8 番ピンを出力に設定します
次にくり返し動作させる部分です。
まずは、3 番ピンの状態を知る必要あります。そのための命令が digitalRead です。
digitalRead(3); と書くことで、3 番ピンが現在 H なのか L なのかを知ることができます。
そして、その 3 番ピンが H なのか L なのかの情報は後で使いたいので、「x」という変数の中に入れておくことにします。 スケッチで書くと、 x = digitalRead(3); となります。

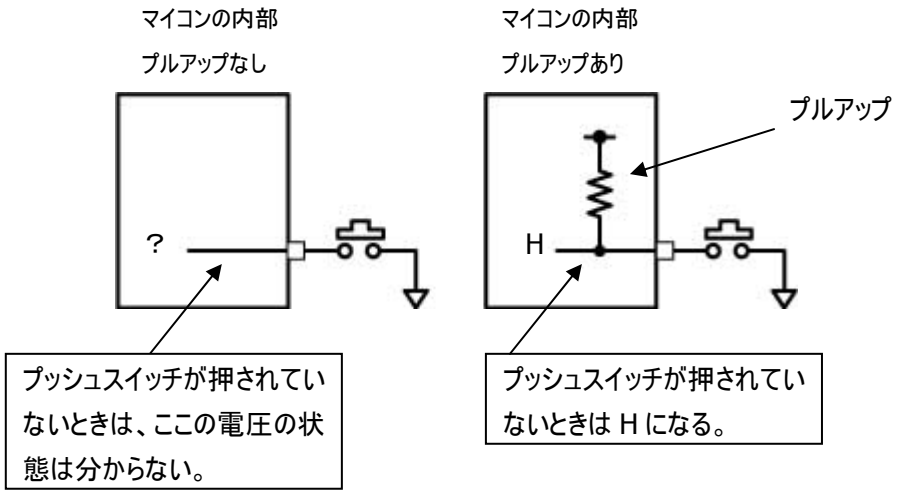
3 番ピンの状態が分かったら、次はその状態によって動作を分けます。
「もしも〇〇〇ならば、×××。そうでなければ、△△△をする。」とう動作をさせたいので、「if 文」を使っています。if 文は、

```
if( 条件 ){
    条件に一致したときの動作
}else{
    条件に一致しなかったときの動作
}
```

という使い方をします。
この「条件」に、「x が LOW ならば」という条件を書きます。それが x == LOW の部分です。
(=は 2 個使いますので、作成時に気を付けましょう。)
このサンプルも、スケッチを作成して、アップロードを行い、動作を確認しておきましょう。

※3 プルアップ機能について

まず、プルアップ機能を使用しないときのことを考えます。
本マニュアル 20 ページの回路図も見ながら動作をイメージしてください。
シールド基板の SW1 と Arduino 基板の 3 番ピンがつながっていて、SW1 を押すと、3 番ピンが LOW になります。それでは、SW1 が押されていないときは、3 番ピンはどうなるのでしょうか？SW1 が押されていないときは、3 番ピンは電気的にどこにもつながっていません。つまり、どんな状態になるのか分からないのです。もちろん Arduino(マイコン)にも分かりません。だから digitalRead 命令で 3 番ピンの状態を確認しても、H なのか L なのかはそのとき次第ということになってしまいます。
そこで、これを解決するための機能が「プルアップ」です。マイコンの内部で、3 番ピンを数十 kΩの抵抗を経由して電源につなぐ機能です。こうすることで、普段は(SW1 が押されていないときは)3 番ピンは H になり、SW1が押されると、L になります。



Arduino でのプログラミングについて

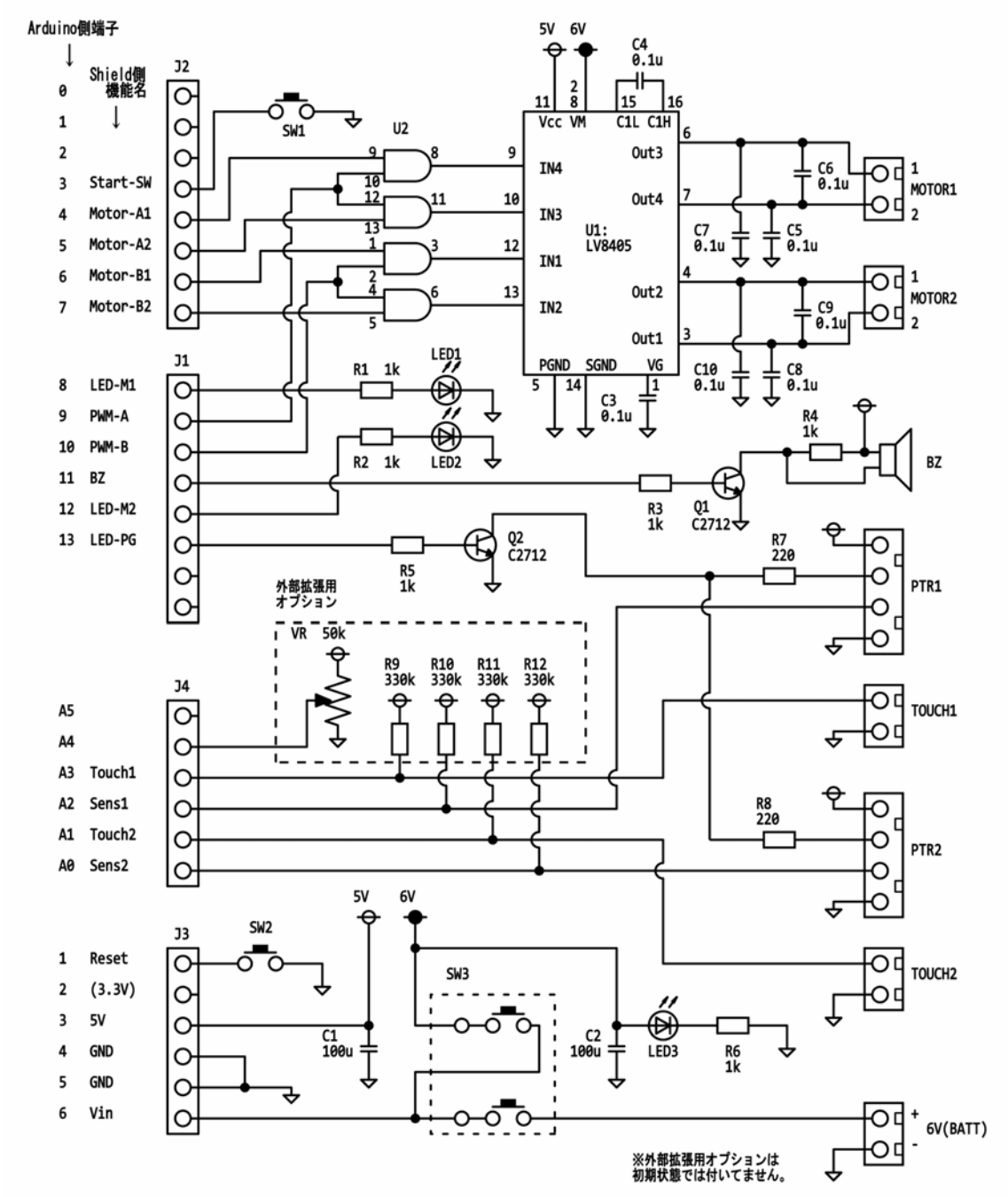
もっと本格的に Arduino のプログラミングを学ぶと、もっと複雑な動きをプログラムしたり、自分だけの関数が作ることができたり、もっとスマートで見て分かりやすいスケッチを作成できるようになります。
Arduino のプログラミングの情報は、http：／／www. arduino. ccや、インターネット、書籍などで手に入れて、学習を行ってください。

シールド基板と Arduino 基板の接続について

ここでは、Arduino 基板でシールド基板を制御するのに必要な接続を表にしています。
※　シールド基板内部回路については回路図でご確認ください。

Arduino 側 ピン No.	シールド側 ピン名	シールド側機能	
0		（使用していません）	
1		（使用していません）	
2		（使用していません）	
3	SW1	押しスイッチ	スケッチ側でプルアップを記述して使用します。
4	Motor-A1	Motor1 制御信号 A1	回転方向については表を参照ください。
5	Motor-A2	Motor1 制御信号 A2	
6	Motor-B1	Motor2 制御信号 B1	
7	Motor-B2	Motor2 制御信号 B2	
8	LED-M1	LED1	8 番ピンに HIGH を設定すると LED1 が ON します。
9	PWM-A	Motor1 用 PWM 信号	PWM 機能でモーターの回転速度を設定します。
10	PWM-B	Motor2 用 PWM 信号	
11	BZ	BZ	数百 Hz～数 kHz の信号でブザーが鳴動します。
12	LED-M2	LED2	12 番ピンに HIGH を設定すると LED2 が ON します。
13	LED-PG	光センサー基板上 LED 制御信号	当社キロボ(MR-9132)の光センサー基板用です。
A0	Sens2	光センサー2 入力端子	当社キロボ(MR-9132)のセンサー類接続用です。
A1	Touch2	タッチセンサー2 入力端子	
A2	Sens1	光センサー1 入力端子	
A3	Touch1	タッチセンサー1 入力端子	
A4		（使用していません）	
A5		（使用していません）	
Reset	SW2	リセットスイッチ	SW2 は Arduino の Reset スイッチと同じ働きをします。

回路図

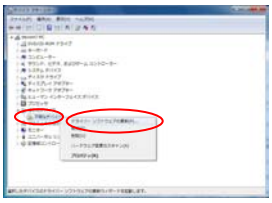


トラブルシューティング

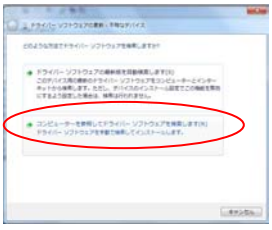
ドライバーを手動でインストールする

（1） コントロールパネル → システムのセキュリティ → デバイスマネージャー と選んで、デバイスマネージャーの画面を開きます。

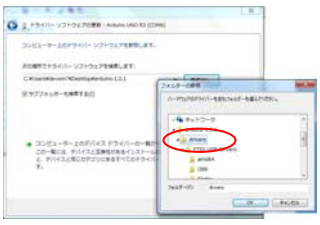
（2） デバイスマネージャー画面の中に、「不明なデバイス」という表示がされていますので、その文字の上で右クリックして、「ドライバーソフトウェアの更新・・・」を選びます。



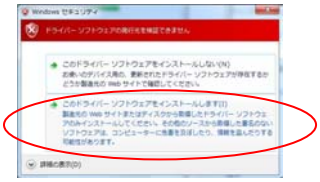
（3）「ドライバーソフトウェアの更新」画面から、「コンピューターを参照してドライバーソフトウェアを検索します。」を選びます。



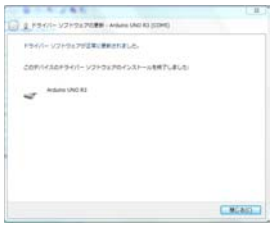
（4）「次の場所でドライバーソフトウェアを検索します。」→「参照」を選び、最初に保存した Arduino のフォルダーの中から Drivers を選びます。



(5) Windows セキュリティの画面が表れますので、「このドライバーソフトウェアをインストールする。」を選びます。



(6) しばらく待つと、ドライバーのインストール完了画面が表示されます。



(7) もう一度デバイスマネージャーを表示して、Arduino 基板が、ポート（COM と LPT）に登録されていることを確認してください。