



# Arduino入門

2010年度仰木研究会  
電子教材開発プロジェクト

# Arduinoとは？



Fig1. Arduino

- 簡単な入出力を備えた基盤
- Processing言語を実装した開発環境
- オープン(ソース and ハードウェア)

# 開発環境 : Arduino



The screenshot shows the Arduino IDE interface. The title bar reads "Arduino - 0011 Alpha". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for running, stopping, saving, and other functions. The main text area displays the "Blink" sketch code. The code is as follows:

```
/*
 * Blink
 *
 * The basic Arduino example. Turns on an LED on for one second,
 * then off for one second, and so on... We use pin 13 because,
 * depending on your Arduino board, it has either a built-in LED
 * or a built-in resistor so that you need only an LED.
 *
 * http://www.arduino.cc/en/Tutorial/Blink
 */

int ledPin = 13;           // LED connected to digital pin 13

void setup()               // run once, when the sketch starts
{
  pinMode(ledPin, OUTPUT); // sets the digital pin as output
}

void loop()                // run over and over again
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);                // waits for a second
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);                // waits for a second
}
```

At the bottom of the IDE, a status bar shows "Done compiling." and "Binary sketch size: 1098 bytes (of a 14336 byte maximum)". The page number "22" is visible in the bottom left corner.

- Javaアプリケーション
- フリーソフト

Fig2. Arduino Software 実行画面

Download : <http://arduino.cc/en/Main/Software>

# Hardware

- マイクロコントローラ
- 用途に応じた多様な基盤

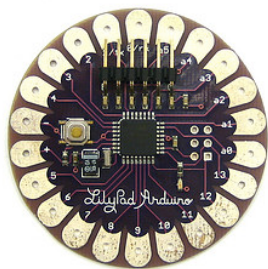
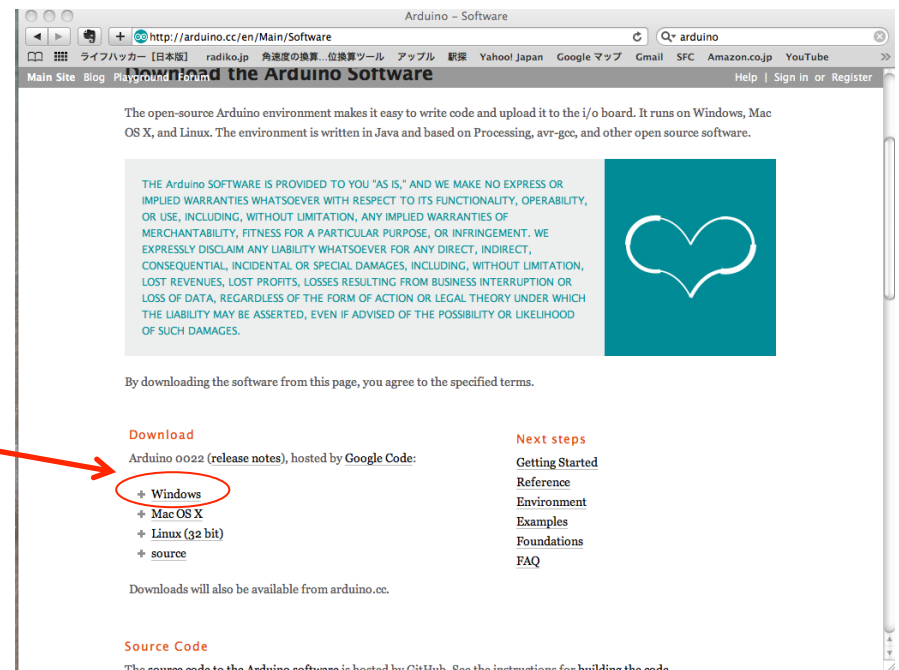


Fig3. Arduino Hardware

# 入門: arduinoをインストールしよう (Windows編)

- ① <http://www.arduino.cc/> へアクセス！
- ② 「Download」 をクリック
- ③ Download にある Windows をクリック

ここをクリック



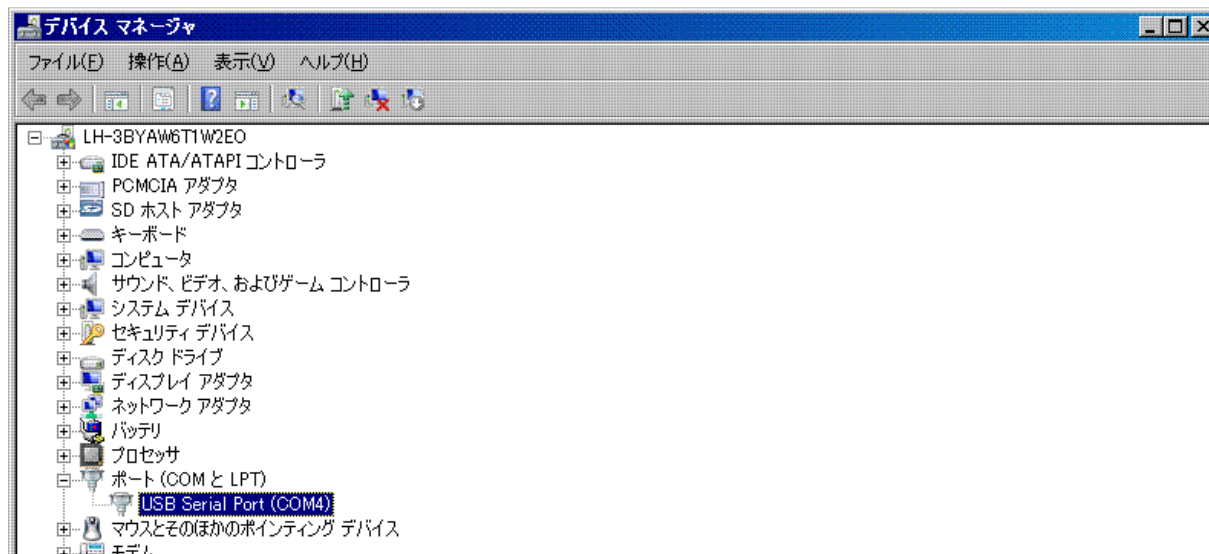
- ④ 解凍してから好きなところ(通常は「C: ¥ Program Files」フォルダの中)に移動

# 入門 arduinoをインストールしよう (Windows編)

- ⑤ arduino とパソコンを USB ケーブルでつなぐと「新しいハードウェアが見つかりました」というウィンドウが現れるので、「ドライバ、ソフトウェアを検索してインストールします(推奨)」を選択し、案内に沿ってインストールする。

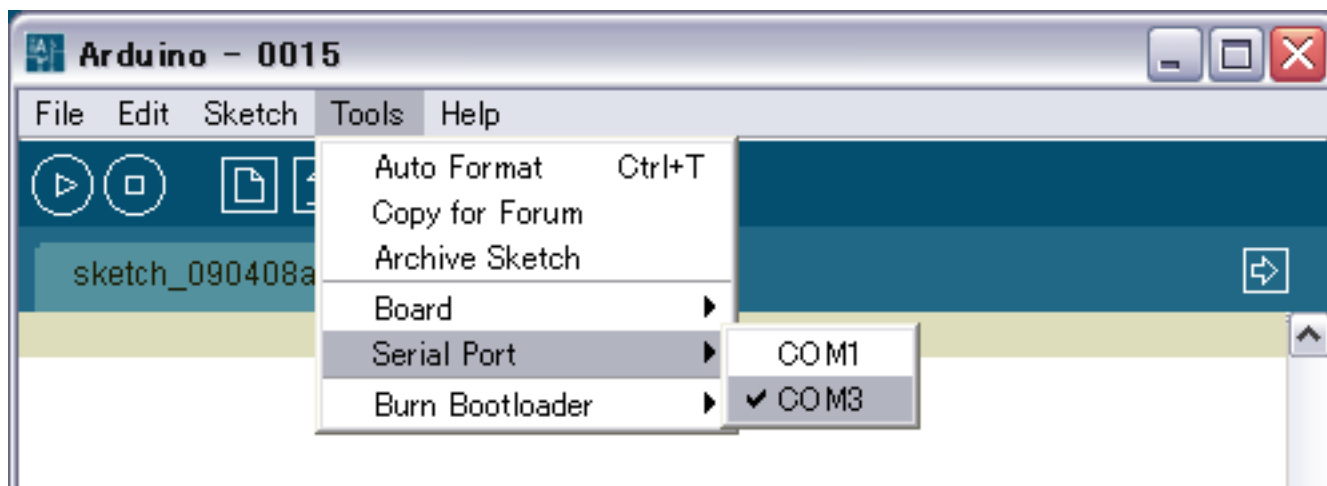
注: エラーが出たときはダウンロードしたarduinoフォルダの中に「drivers」フォルダがあるので、そのフォルダから検索できるように変更する。

- ⑥ 「スタート」→「コントロールパネル」→「ハードウェアとサウンド」→「デバイス マネージャー」の「ポート (COMとLPT)」→「USB Serial Port (COM●)」を見る。(COM●を覚えておく)



# 入門 arduinoをインストールしよう (Windows編)

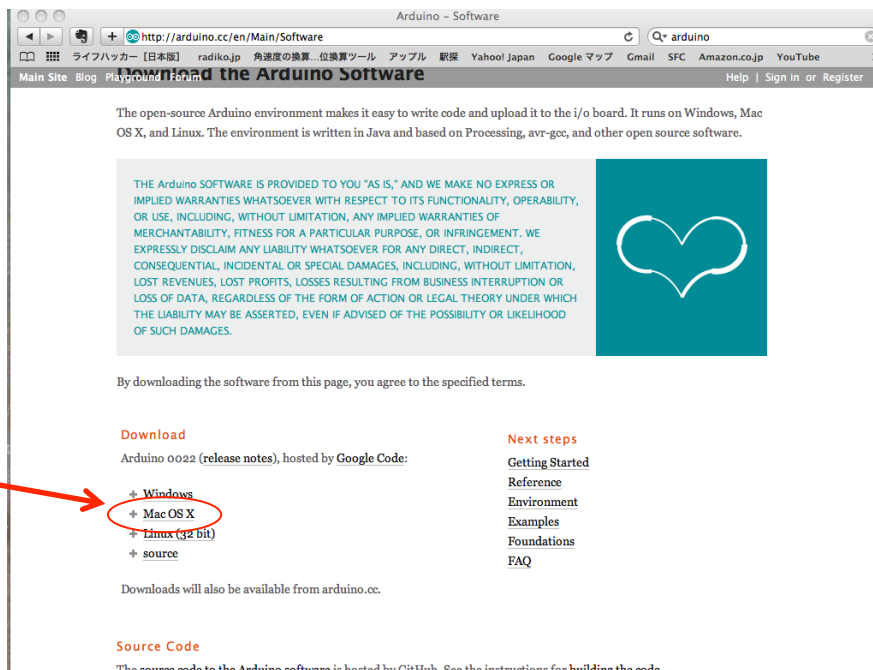
- ⑦arduino とパソコンをUSBケーブルでつないでarduinoソフトウェアを起動する。
- ⑧「Tool」メニューから「Board」を開き、「arduino Duemilanove or Nano w/Atmega328」を選択。同じく「Tool」メニューから「Serial Port」を開き⑥で調べた「COM●」を選択。これにより、パソコンとarduinoが接続された。



# 入門 arduinoをインストールしよう (Mac編)

- ① <http://www.arduino.cc/> へアクセス！
- ② 「Download」 をクリック
- ③ Download にある「Mac OS X」をクリックしてダウンロード(時間がかかります)

ここをクリック

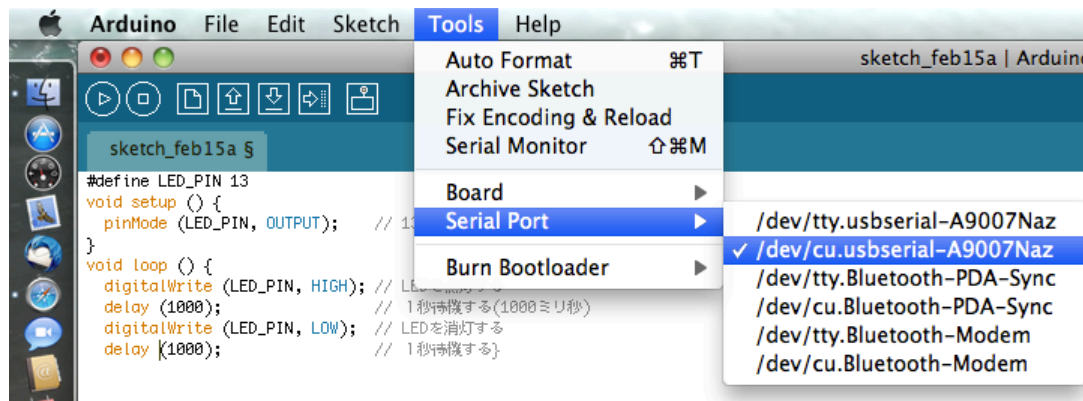


- ④ 解凍してから好きなところ(通常は「アプリケーション」フォルダの中)に移動

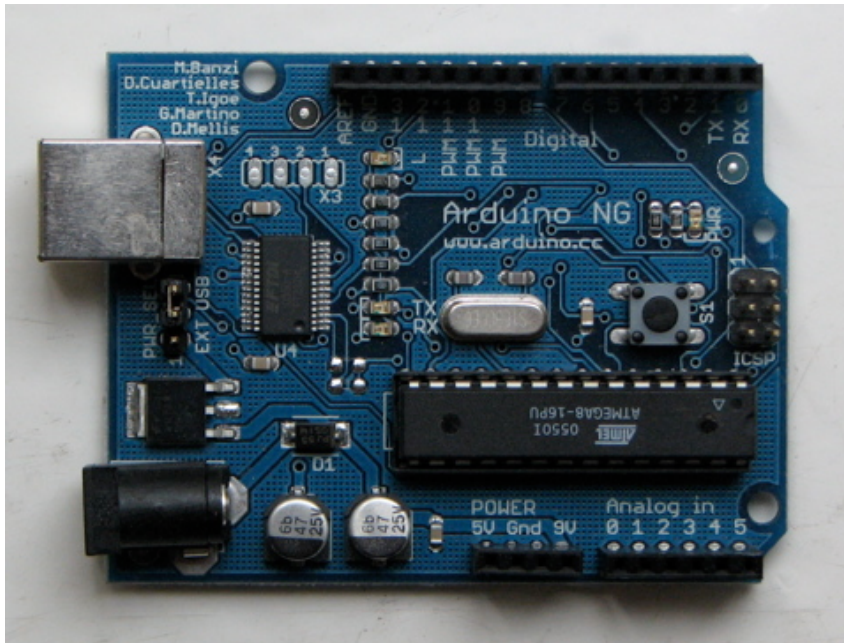


# 入門 arduinoをインストールしよう (Mac編)

- ⑤ ダウンロードした「arduino.dmp」フォルダの中の「FTDIUSBSerialDriver\_x\_x\_x.dmg」ファイルをダブルクリックする。(xは数字になっている)
- ⑥ そのあとは画面の説明にしたがってドライバをインストールしていきます。
- ⑦ arduino とパソコンをUSBケーブルでつないでarduinoソフトウェアを起動する。
- ⑧ 「Tool」メニューから「Board」を開き、「arduino Duemilanove or Nano w/Atmega328」を選択。同じく「Tool」メニューから「Serial Port」を開き、「/dev/cu.usbserial-」で始まる項目を選択。これにより、パソコンとarduinoが接続された。



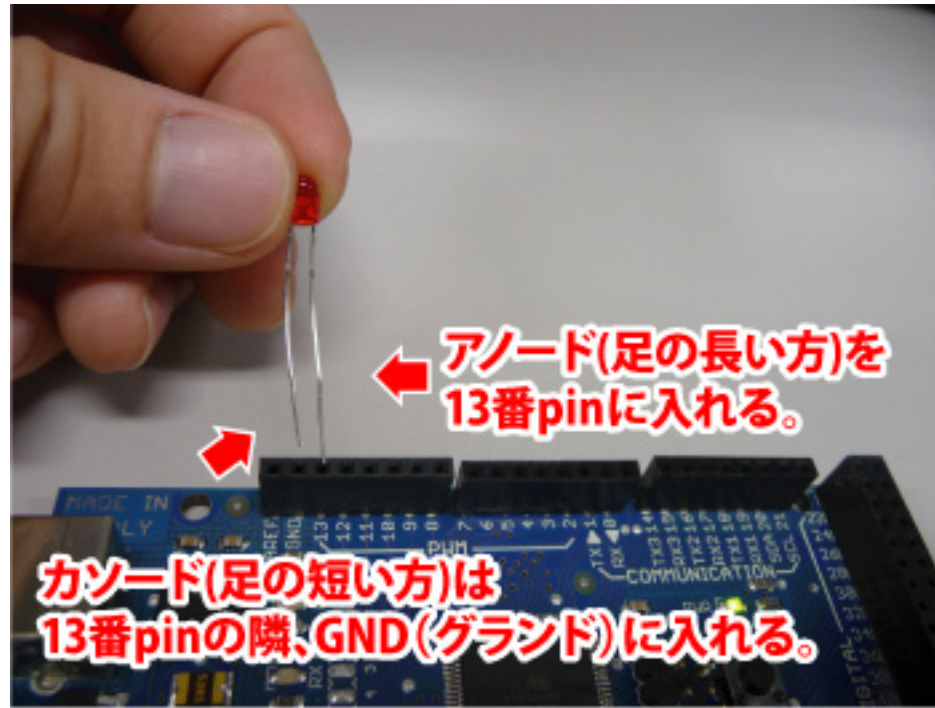
# 入門 LEDを光らせる



- Arduino Softwareでプログラミング
- サンプルソースコード
- まずは基本的なLEDを光らせてみよう

# 入門 LEDを光らせる

- ① LEDライトには2つの線がありますが、長いほう(アノードという)を arduino の 13番ピンに挿し込み、短いほう(カソードという)を隣のGNDに挿し込む。



# 入門 LEDを光らせる

- ② arduino とパソコンを USB ケーブルでつなぐ。
- ③ arduino ソフトを起動させて、以下のプログラムを書き込む

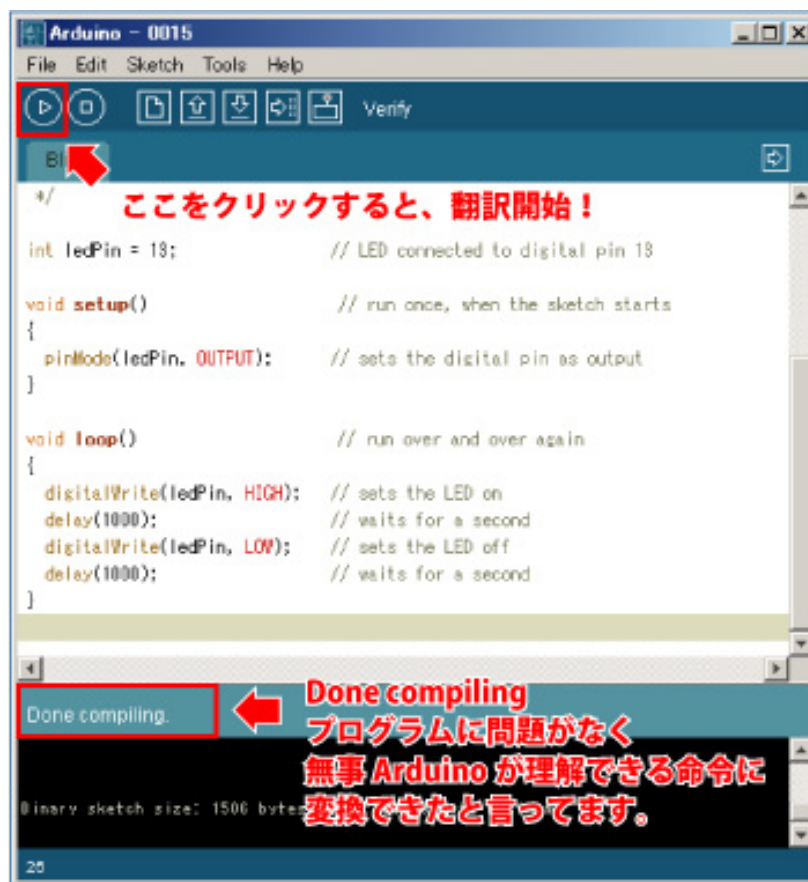
```
#define LED_PIN 13

void setup () {
    pinMode (LED_PIN, OUTPUT);    // 13番ピンをデジタル出力に設定する
}

void loop () {
    digitalWrite (LED_PIN, HIGH); // LEDを点灯する
    delay (1000);                 // 1秒待機する(1000ミリ秒)
    digitalWrite (LED_PIN, LOW);  // LEDを消灯する
    delay (1000);                 // 1秒待機する
}
```

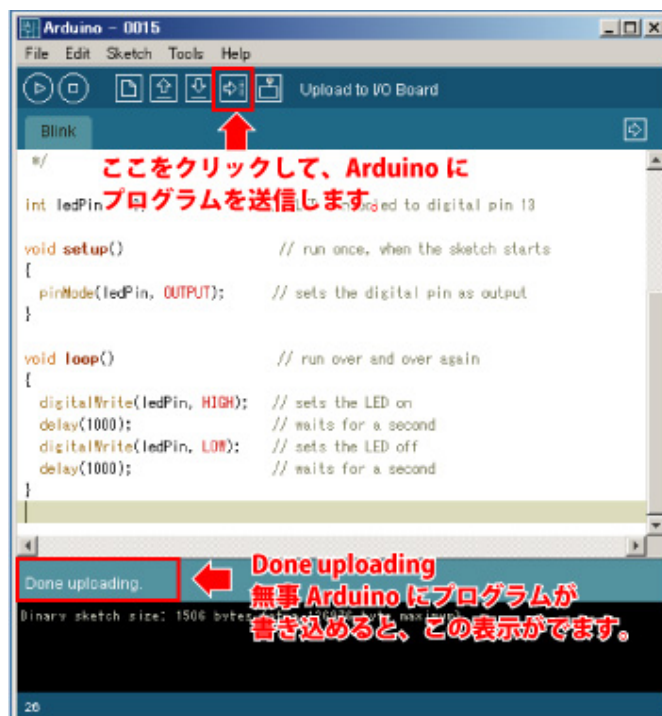
# 入門 LEDを光らせる

- ④ 図のように左上の三角ボタン「verify」を押して、下に「Done compiling」が表示されたらプログラムに問題なし。他の文字が出たらどこかプログラムが間違っている。



# 入門 LEDを光らせる

- ⑤ 図のように上の「upload」ボタンを押してarduino にプログラムを送信。  
「Done uploading」が表示されたら LEDが光り始める。



注: 「Done uploading」が表示されなかった場合、「Tool」の「Board」か「Serial Port」が間違っている可能性がある。

# コードの解説

```
#define LED_PIN 13
```

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

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

定義する場所

この場合13番ピンを「LED\_PIN」という名に定義

初期化情報を書き込む  
最初に1度だけ読み込まれる情報

情報をずっと繰り返し続ける場所

この場合

LEDライトを付ける

1秒待つ

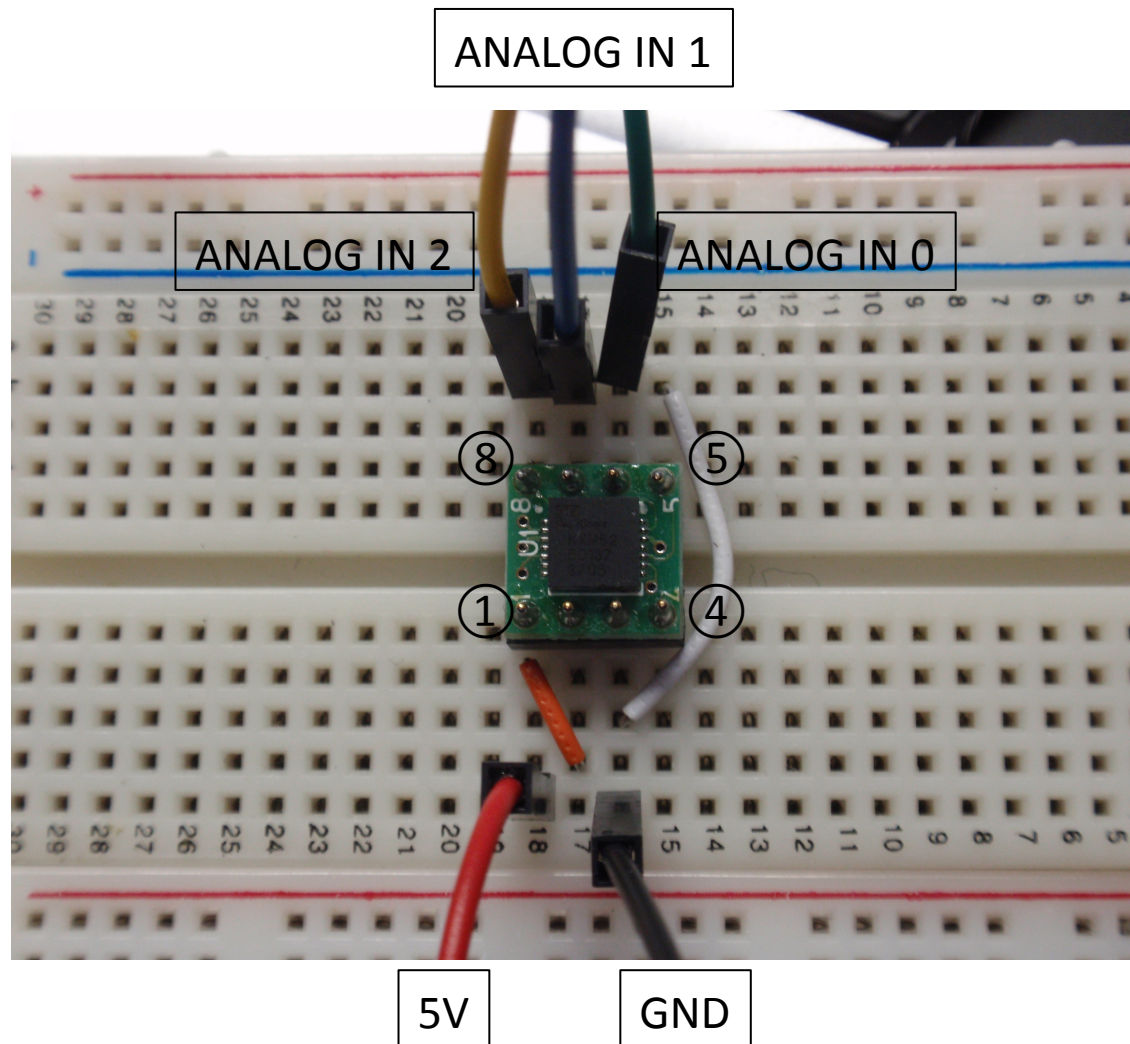
LEDライトを消す

1秒待つ

これを繰り返す

# 加速度を計測しよう

Arduinoのそれぞれにピンを挿していこう





# 加速度を計測しよう

```
int AccX;
```

```
void setup(){  
  Serial.begin(9600);  
}
```

```
void loop(){  
  AccX = analogRead(0);  
  
  Serial.print("AccX=");  
  Serial.print(AccX);  
  Serial.print("\t");  
  
  delay(1000);  
}
```

int型に宣言する

通信のデータ転送レートを指定

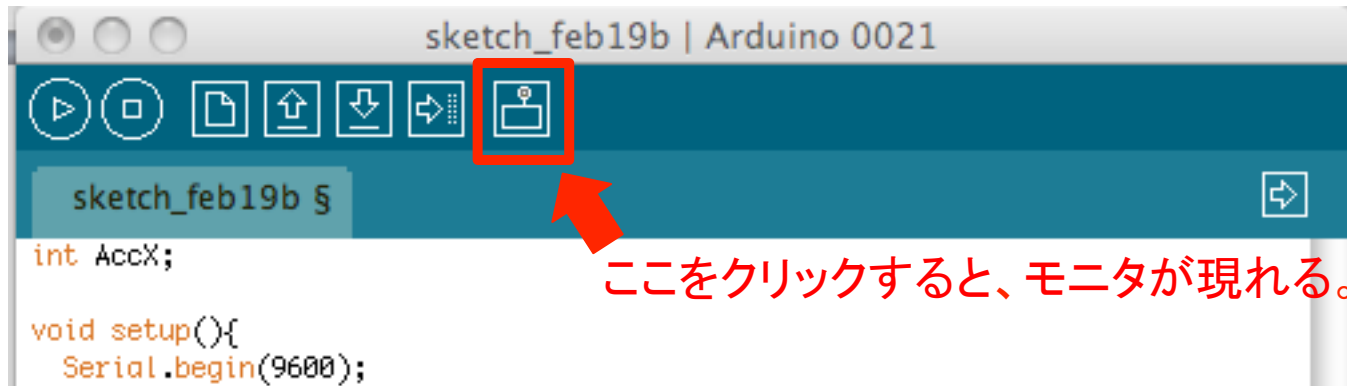
「0」番ピンから得られた値をAccXとする

Serial.printは文字を表示させる。  
Serial.print("〇〇〇〇");はそのまま  
「〇〇〇〇」で表示される。  
Serial.print(AccX);のように""がないと、  
定義した値が表示される。  
("/t")は改行を意味する。

\* Accとはaccelerationのこと. X,Y,Zは軸

# 加速度を計測しよう

「verify」ボタンと「upload」ボタンを押して正常にアップロードできたら  
☒のボタンを押してみる。



\* 表示される値は生データであり、実際の値とは異なる。  
実際の値にするためにはキャリブレーションを行う必要である。

\* Arduino1には、アナログ計測用のチャンネルが6つ用意されている  
(10bit/6ch). そこでさらに追加して他の物理量, 例えば角速度など  
を測定することができる。自分でプログラムを追加して測れるよう  
にしよう。

# 加速度を計測しよう(コード解説)

```
int AccX;
```

```
void setup(){  
  Serial.begin(9600);  
}
```

```
void loop(){  
  AccX = analogRead(0);  
  
  Serial.print("AccX=");  
  Serial.print(AccX);  
  Serial.print("\t");  
  
  delay(1000);  
}
```

センサーの値(analogRead(0))を  
獲得しAccXに代入

「AccX=」という文字を表示

先ほど獲得した値(AccX)を表示

改行をする

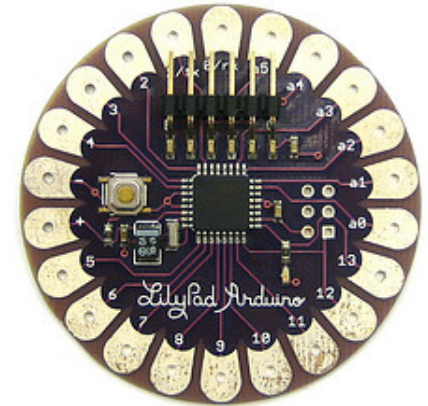
1秒待つ

これを繰り返す

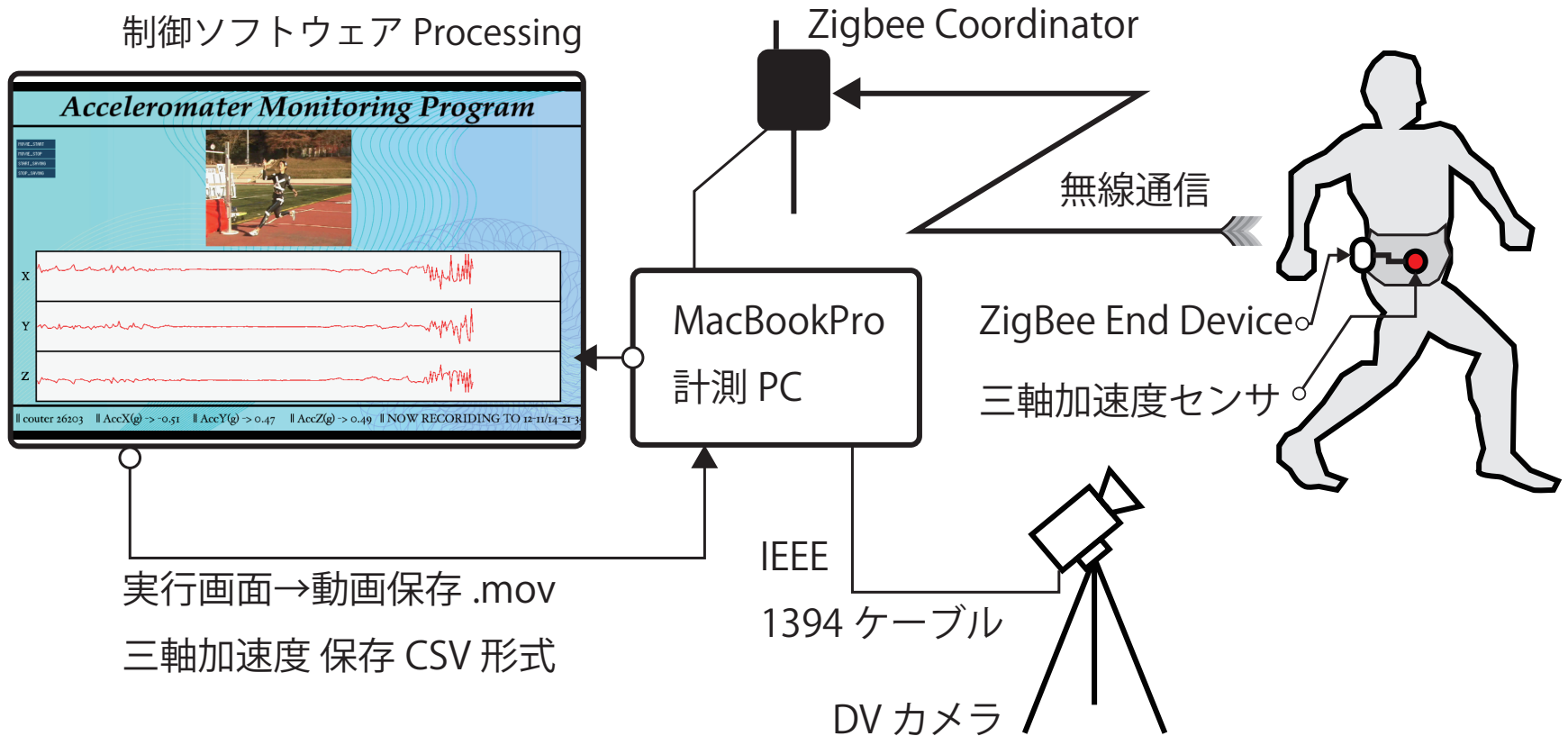


## 応用編

# ベルト式無線通信加速度センサ

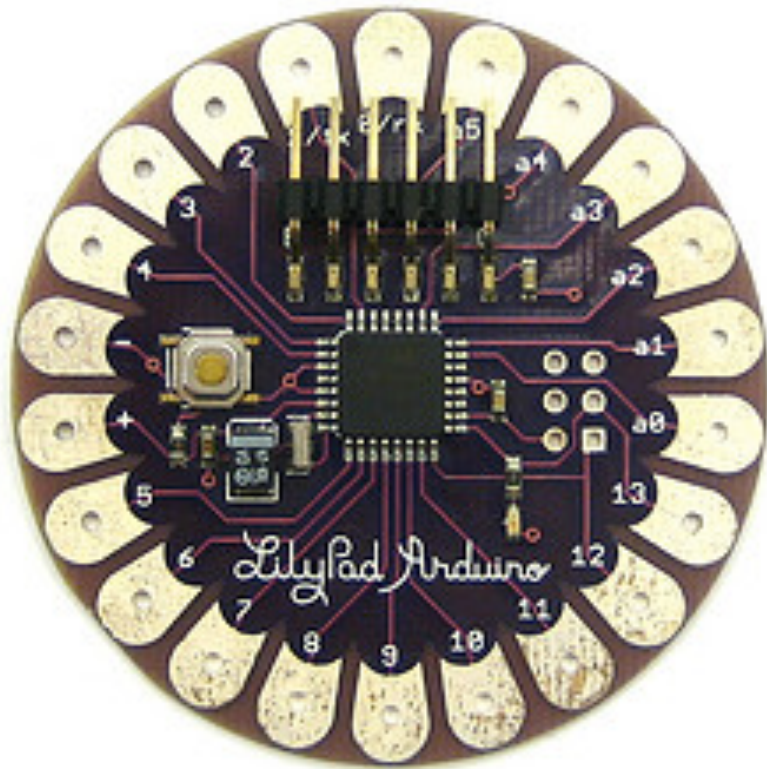


# 概要



Hardware 

# LilyPad Arduino



- 服に縫いつけて着ることができるマイコン
- 導電性の糸を使用することも可能

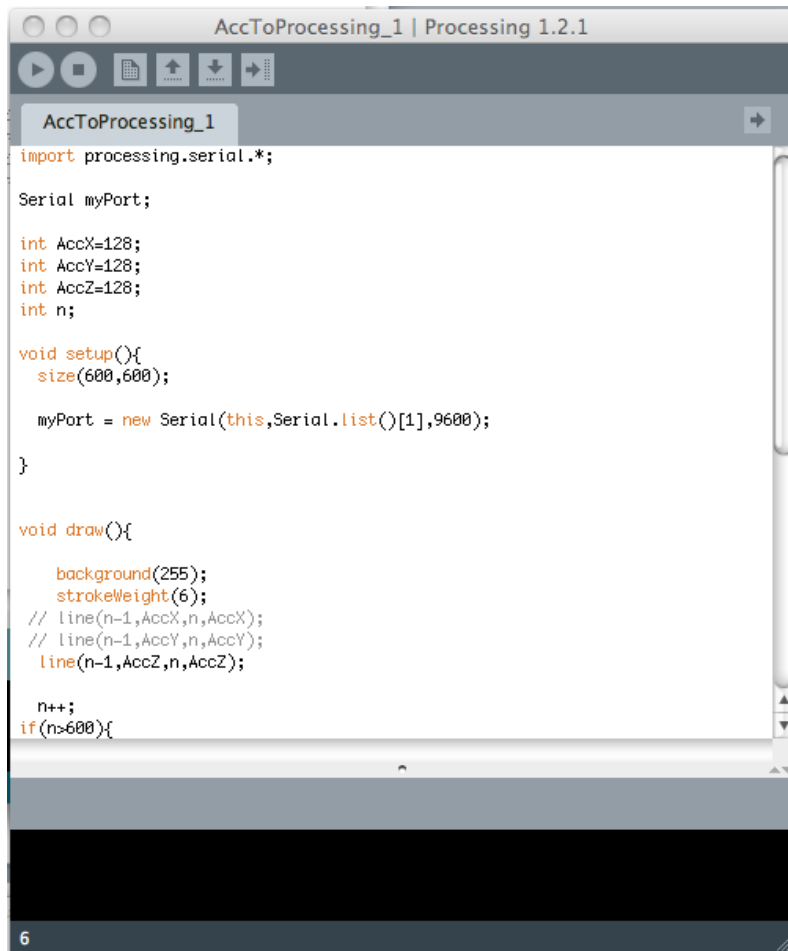
# 無線通信 ZigBee



- 短距離無線通信規格のひとつ
- 低速で転送距離が短い、安価で消費電力が少ない。



# 制御ソフトウェア Processing



```
AccToProcessing_1 | Processing 1.2.1
import processing.serial.*;

Serial myPort;

int AccX=128;
int AccY=128;
int AccZ=128;
int n;

void setup(){
  size(600,600);

  myPort = new Serial(this,Serial.list()[1],9600);
}

void draw(){

  background(255);
  strokeWeight(6);
  // line(n-1,AccX,n,AccX);
  // line(n-1,AccY,n,AccY);
  line(n-1,AccZ,n,AccZ);

  n++;
  if(n>600){
```

- Javaを簡略化したグラフィックに特化した言語
- arduinoとの相互性があり、容易に操作することができる。

# Processing をインストールしよう

手順は arduino とほぼ同じ

- ① <http://www.processing.org/> にアクセス
- ② 「Download」から自分の OS にあったものをダウンロード
- ③ ダウンロードしたフォルダを解凍し好きなところへ移動

Windows の人は OS を選ぶときに Java の有無に合わせて選択  
(わからなければ Java ありを選択)

# 電源モジュール

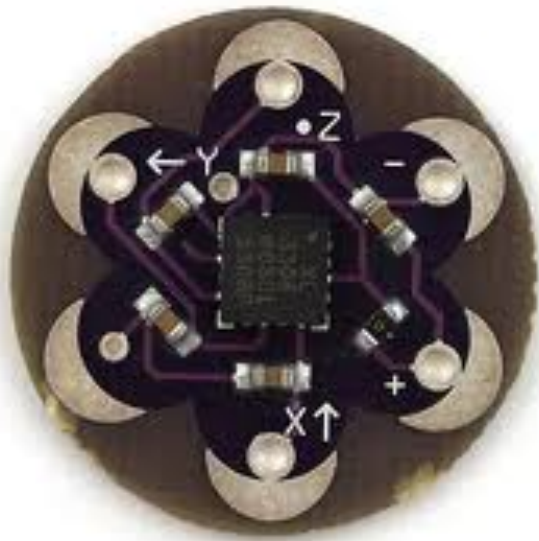
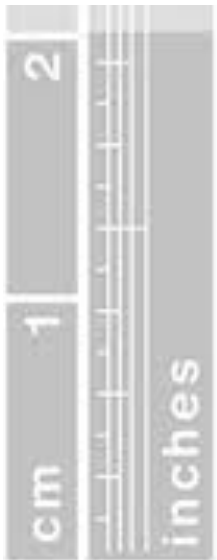


- LilyPadの駆動電圧は、2.7~5.5Vと決まっている
- ここでは、リチウムポリマー電池(写真下:3.7V/1000mAh)を使用
- Arduinoに安定した電圧を供給するため、DC-DCコンバーター(写真上:LiPower)を使用。
- LiPowerから5Vが取り出せるので、これをLilyPadへ

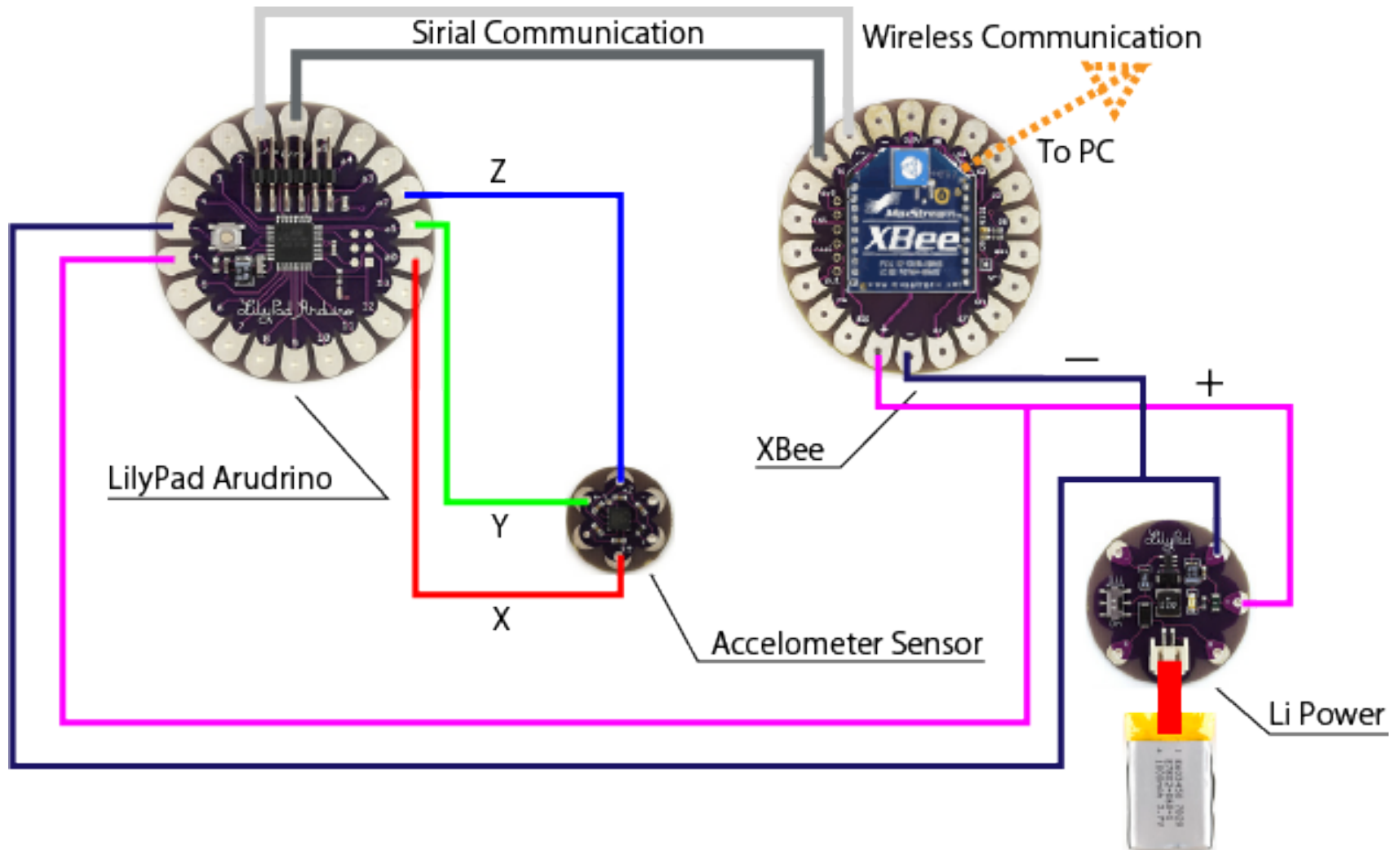


# 三軸加速度センサ

- LilyPad加速度センサ
- ADXL335使用(±3G)
- V+, GND
- X,Y,Z



# 回路図



Software 

# 通信

Arduinoとシリアル通信をし、得られた数値を画面に表示しよう。

- サンプルコード (参考: <http://www.processing.org/reference/libraries/serial/index.html>)

```
import processing.serial.*;
```

```
Serial myPort;
```

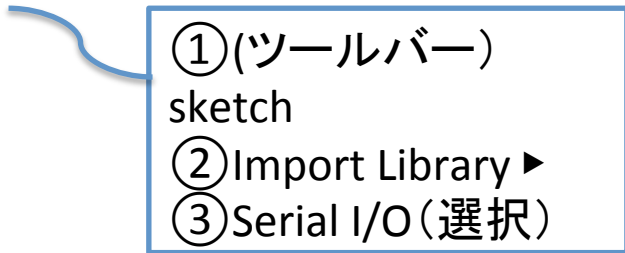
```
String StringData;
```

```
void setup(){
```

```
  size(400,400);
```

```
  background(255);
```

```
//Libraryをimportする
```

- 
- ① (ツールバー) sketch
  - ② Import Library ▶
  - ③ Serial I/O (選択)

```
//画面のサイズを決定する
```

```
//背景色を決定する
```

```
myPort = new Serial(this, "/dev/tty.usbserial-A9007NaE", 9600);
```

```
myPort.clear();  
myPort.bufferUntil(10);  
}  
void draw(){  
}
```

```
void serialEvent(Serial p){  
StringData=myPort.readStringUntil(10);  
StringData=trim(StringData);
```

```
String[] List=split(StringData,",");    //","ごとにデータをとってくる
```

```
print(int(List[0]));    //list[0]を表示  
print(",");           //// , を表示  
print(int(List[1]));   //list[1]を表示  
print(",");           // , を表示  
println(int(List[2])); //list[2]を表示
```

```
}
```

(Macの場合)

①ターミナルを起動

②ls /dev/tty.\* と入力

③"oo..."をコピーして緑字のところに貼付け



# グラフの描画

CSVデータをX軸Y軸Z軸のグラフにしよう。

- サンプルコード(参考:<http://processing.org/reference/try.html>)

```
BufferedReader reader;
String line;
int oldX=0;           //定数を設定する
int oldY=0;
int AccX=0;
int AccY=0;
int GraphPosition=0;

void setup(){
  size(500,400);      //画面のサイズを決定する
  background(255);
  frameRate(25);
  reader = createReader("test.csv"); //“test.csv”という名前のCSVファイルを読み込む
}
```

```
void draw(){
  try {
    line = reader.readLine();
  } catch (IOException e) {
    e.printStackTrace();
    line = null;
  }
  if (line == null) {
    noLoop();
  } else {
    String[] pieces = split(line, ",");           //","ごとにデータを区切る

    AccX =int(pieces[2])*4+250;                   //3番目の数字をAccXと定義し、大きさを調整する
    AccY =int(pieces[3])*5-750;                  //4番目の数字をAccYと定義し、大きさを調整する

    if(GraphPosition<500){
      GraphPosition++;
    }else{background(255);
      GraphPosition=0;
    }
  }
}
```

```
strokeWeight(3.0);           //グラフの太さを決定する
stroke(255,0,0);             //グラフの色を設定する
line(GraphPosition,oldX, GraphPosition+1,AccX); //線を描く
stroke(0,0,255);
line( GraphPosition,oldY, GraphPosition+1,AccY);

oldX=AccX;                   //データを更新する
oldY=AccY;

}
}
```

これができたら、シリアル通信で  
とれたデータをグラフにしてみよう！

# 動画の描画

DVカメラで撮影した動画を画面に描画しよう。

- サンプルコード(参考:[http://processing.org/reference/libraries/video/Capture\\_read\\_.html](http://processing.org/reference/libraries/video/Capture_read_.html))

```
import processing.video.*;
```

//Libraryをimportする

```
Capture myCapture;
```

```
void setup(){
```

```
  frameRate(25);
```

```
  size(320,240);
```

//画面のサイズを決定する

```
  myCapture = new Capture(this,320,240); //キャプチャ画面ののサイズを決定する
```

```
}
```

- 
- ① (ツールバー) sketch
  - ② Import Library ▶
  - ③ Video (選択)

```
void draw(){
  if(myCapture.available()){
    myCapture.read();
  }
  image(myCapture,0,0);
}
```

//キャプチャ画面の場所を指定する  
(画面の左上の座標を指定)

# 保存

実行画面を動画保存しよう。

- サンプルコード(参考:<http://processing.org/reference/libraries/video/MovieMaker.html>)

```
import processing.video.*;           //Library(video)をimportする
Capture myCapture;                   //動画表示の設定をする
MovieMaker MovieFile;               //保存ファイルの設定をする
String title;

void setup(){
  frameRate(25);
  size(320,240);                      //画面のサイズを決定する
  myCapture = new Capture(this,320,240); //キャプチャ画面ののサイズを決定する
}
```

```

void draw(){
  if(myCapture.available()){
    myCapture.read();
  }
  image(myCapture,0,0); //キャプチャ画面の場所を指定する
                        (画面の左上の座標を指定)

  if(mousePressed == true){ //画面上をクリックしたら実行される
    title = month()+"-"+day()+"/"+hour()+"-"+minute()+"-"+second();
                        //ファイルのタイトルを付ける(月-日▶時-分-秒)
    MovieFile = new MovieMaker(this,width,height,title+".mov",15,
      MovieMaker.VIDEO, MovieMaker.LOSSLESS); //ファイルを作成する
    MovieFile.addFrame(); //ファイルに書き込む
  }
}

```

カメラで撮影した動画を画面に描画し、その画面をクリックすることで動画を保存できる。  
 保存された動画は書類▶Processing▶保存したsketchbook▶  
 (ファイルを作成した月-日)▶(ファイルを作成した時-分-秒.mov)  
 という名前のファイルに動画が保存されている。

# 全プログラム

- arduinoとprocessingのコードが長いいため、身体運動解析履修者の皆さんは、SFC-SFSから、資料をダウンロードしてください
- その他の方は、仰木研究会のWebサイトからダウンロード
  - <http://web.sfc.keio.ac.jp/~ohgi/>