最終更新　2024.6.10

Arduino実習

奈良教育大学　薮 哲郎

# 1. Arduinoとは

　フィジカルコンピューティング（physical computing）という概念が2005年頃に提唱された。physicalは「物理的な」という意味であり、現実世界とのやりとりを行うという意味を持つ。

　従来のコンピュータは入力として「キーボード、マウス」、出力として「ディスプレイ」を使う。フィジカルコンピューティング用のコンピュータは、それに加えて、電圧を入出力するための端子を持つ。その端子に電子回路を接続することにより、センサからの値を読み取り、アクチュエータを動作させる。

　現在の電子工作は、マイコンやシングルボードコンピュータに簡単な電子回路を接続することで構成されることが多い、そこで使われるコンピュータには様々なものがあるが、圧倒的にポピュラーなのがArduino, micro:bit, Raspberry Piの3種である。

　ArduinoはUNO R3というバージョンが2011年に発売され、長らく定番品となってきた。2023年にUNO R4というバージョンが発売され、性能が大幅に向上した。CPUが変更され、速度が上がりメモリが増えた。ただし、入出力インターフェースの端子構成は互換性がある。

　3種のフィジカルコンピューティングの特徴を以下にまとめる。

|  |  |  |
| --- | --- | --- |
| 呼称 | マイコン | シングルボードコンピュータ |
| 代表的な  製品 | Arduino  micro:bit | Raspberry Pi |
| 特徴 | * CPUの性能は低い * OSはない * マイコンの中で1個のプログラムが動作する * ネットワークを使うには別売のユニットが必要で、敷居が高い[[1]](#footnote-1)。 | * パソコンと同等の能力を持つ * OSを持つ。RaspbianというLinux系のOSが定番である * 多数のプログラムが同時に動作する * ネットワークに容易に接続できる |

　2009年頃からフィジカルコンピューティング用のコンピュータが普及しはじめた。様々な製品があるが、マイコンとしてはArduino、シングルボードコンピュータとしてはRaspberry Piがポピュラーである。本実験ではArduinoを使う。

　Arduinoは10年近くにわたって定番であったUNO R3が4000円弱であり、互換品は2000円～3000円程度である。回路図が公開されているので500円程度で自作できる。

　Amazonを見ると、Arduinoを含む電子工作学習用のキットが売っている。4000円～6000円程度で非常に多くのパーツ、ブレッドボード、ブレッドボード用ワイヤーが含まれており、お買い得である。2024.5の時点ではELEGOO, OSOYOOの製品がメジャーなようである。本実験では、2020.7頃に購入したSmraza社の製品を使う[[2]](#footnote-2)。

# 2. IDE（統合開発環境）のインストール

1. www.arduino.ccに接続しWindows 10版のArduino IDE 2.3.2をダウンロードする。メールアドレスを入れ、I confirm to have.... にチェックを入れる。Subscribe & Download ではなく、その下にあるJust Downloadを選択すると、寄付する必要はない。
2. ダウンロードしたarduino-ide\_2.3.2\_Windows\_64bit.exeをダブルクリックしてインストールする。どのユーザーにインストールしますか？　と聞いてくる。どちらを選んでもよい。インストールする場所が変わる[[3]](#footnote-3)。途中色々と許可を求めてくる。許可する。

## IDEの起動

1. 「File」→「Preferences」で設定を変更する。font sizeは14から大きくすると良い。Themeをdarkにすると目に優しい。Languageを日本語にすると、メニューが日本語になる。
2. 「ツール」→「ボード」→「Arduino AVR Borads」で「Arduino UNO」を選択する。
3. 「ツール」→「ポート」でArduino UNOが接続されているポート番号を選択する

## 動作確認

　Arduino UNOの13番ポートにはボード上のオレンジ色のLEDが接続されており、HIGHにすると点灯、LOWにすると消灯する。以下のプログラムを入力し「→」のアイコンをクリックすると、オレンジ色のLEDが点滅する。

void setup() {

pinMode(13, OUTPUT);

}

void loop() {

digitalWrite(13, HIGH);

delay(500);

digitalWrite(13, LOW);

delay(500);

}

# 3. Arduinoの概要

<http://denki.nara-edu.ac.jp/~yabu/soft/arduino.html>

に備忘録がある。

## ハードウェア

　アナログ入力：A0～A5の6個

　デジタル入出力：0～13の14個（～印の端子はPWM出力可能）

　その他の端子：5 V, 3.3 V, Gnd

[[4]](#footnote-4)

(注意) デジタル端子（0～13）とアナログ端子（A0～A5）を混同しないように注意する。特に0～6の番号はデジタルとアナログに共通する（ただし、アナログ端子はA0のようにAが付く）ので要注意である。アナログ端子（A0～A5）はアナログ入力専用であり、後述するpinMode()を指定する必要はない。

## ソフトウェア

　Arduino言語を用いる。Arduino言語はC言語とほぼ同じである。プログラムの構造は以下の通り。

void setup() {

// put your setup code here, to run once:

最初に1回だけ実行する命令を書く

}

void loop() {

// put your main code here, to run repeatedly:

このloopという関数が繰り返し呼ばれる。

}

## C言語の主な文法

コメント文 …… // はコメント文である。何を書いても無視される。

変数宣言 …… 変数を使う場合、宣言が必要である。変数はコンマで区切る。命令の末尾にセミコロン（;）を書く。

int a, b; // 整数型変数aとbを宣言　実数型変数はfloat

a = 10;

b = 20;

if 文 …… 条件式は ( ) で囲む。「等しい」は「==」、「等しくない」は「!=」。ifが成立したときに実行する命令、成立しないときに実行する命令は { } で囲む。

if (a == 10){

b = 5;

} else {

b = 10;

}

for 文 …… Basicは1から10まで変化させたいとき、For i = 1 to 10と書くが、Cではfor(i = 1; i <=10; i++){ } あるいは for(i = 1; i < 11; i++){ } と書く。「i++」は「i を1増やす」という意味であり、「i = i + 1」と同じである。

for(i = 0; i < 10; i++){

s = s + i;

}

## Arduino言語特有の命令

pinMode(13, OUTPUT); // pin 13 を出力用に設定

pinMode(12, INPUT); // pin 12 を入力用に設定

digitalWrite(13, HIGH); // pin 13 を HIGH に設定（5 V 出力）

digitalWrite(13, LOW); // LOW に設定（0 V 出力）

a = analogRead(A0); // A0端子の入力電圧を読み取る (0-1023)

d = digitalRead(2); // pin 2 の端子の電圧を読み取る (0 or 1)

delay(500); // 500 ms何もせずに待機する

## グローバル変数

　loopからリターンし、再度呼ばれるとき、前回loopが呼ばれたときの変数の値は保証されない。グローバル変数（常に値が保持し続けられる変数）は以下のように使用する。

int a; // a はグローバル変数　setup() の手前で宣言する

void setup() {

a = 0; // ここで初期化

Serial.begin(9600); // シリアル通信を9600 bpsで行う

}

void loop() {

a++; // a の値を1増やす

delay(1000);

Serial.println(a); // シリアル通信線に出力

}

　上記のサンプルプログラムではsetup()で a = 0 を実行している。C言語では変数の宣言時に初期化することもできる。宣言の部分で int a = 0; としてもよい。

　上記のプログラムはシリアル通信線に出力している。これをパソコンで受けて表示するには、「ツール」→「シリアルモニタ」としてシリアルモニタのウィンドウを表示する。シリアルモニタウィンドウにおいて、bpsをプログラム中で宣言した9600に合わせる必要がある。これを怠ると正しく表示されない。

## 原因不明のトラブル発生時の対処法

　プログラム、配線ともに正しいのに、動かないことがある。そのような場合は、以下の対処を試みる。

* Arduinoのリセットスイッチを押す
* Arduinoが接続されているUSBケーブルを一度抜いて、再度差す
* Arduino IDEを一度終了して、再度、起動する

# 4. 課題（既存のプログラムを動かす）

　以下の4つの課題を実行し、その動画を撮影してアップロードする。

　そのときに使ったプログラムをWordに貼り付け、提出する。プログラムリストは以下の設定で記述する。

・フォントはConsolas 10 pt

・行間固定値 13 pt

## Lチカ（第1課）

　1個のLEDを1秒間隔でチカチカさせる。動作チェックで使ったプログラムをそのまま使えば良い。LEDに直列に入れる電流制限抵抗は330 Ωを使う。電流制限抵抗は必ず入れること。出力端子をショートさせるとArduinoの出力端子のトランジスタが破壊される可能性がある。

## スイッチとLED（第5課）

　スイッチを押したらLEDが消える。スイッチと直列に必ず抵抗（図中 10 kΩの抵抗）を入れること。これを忘れて電源の5 Vをショートさせると、Arduinoに電力を供給しているパソコンのUSB端子がダメージを受ける可能性がある[[5]](#footnote-5)。LEDに直列に入れる電流制限抵抗も忘れないようにすること。

　サンプルプログラムは英語のコメントや変数宣言が書かれていて、難しそうに見えるが、以下と同一である。

void setup() {

pinMode(2, INPUT);

pinMode(13, OUTPUT);

}

void loop() {

int a;

a = digitalRead(2);

if (a == LOW){

digitalWrite(13, LOW);

} else {

digitalWrite(13, HIGH);

}

}

　第5課のブレッドボードのイラストにはLEDは描かれていない。忘れているか省略されていると思われる。LEDの接続は第1課のLチカのままでよい。こ

## シリアルモニタ（独自）

　端子A0のアナログ入力を読み取って、その値をシリアルモニタに出力する。

　10 kΩの可変抵抗を使用する。可変抵抗の3個の端子のうち、両端の端子をそれぞれ5 VとGndに接続する。可変抵抗のツマミをどちらかに回しきったとき、「中間端子」と「どちらかの端子」の間の抵抗は0 Ωである。この0 Ωの箇所を5 VとGndに接続すると、5 Vをショートさせることになる。そのようなことが絶対にないように注意すること。

　下記のプログラムは10個出力するたびに改行する。

　実行するときに「ツール」→「シリアルモニタ」でシリアルモニタを表示し、ボーレートを9600 baudに設定する。シリアルモニタの画面を自動スクロールするには、右端から3個目のアイコン（≫が90度回転したアイコン）をクリックし、アイコン周辺の色を濃い色にする。

int i = 0;

void setup() {

Serial.begin(9600);

}

void loop() {

int a;

a = analogRead(A0);

Serial.print(a);

Serial.print(" ");

if (i % 10 == 0){

Serial.println("");

}

i++;

delay(500);

}

## つまみ（第18課）

　サーボモータの端子は3個ある。3個の端子の意味は第17課のテキストの1ページ目に書いてある。以下の通り。

　赤 … 5 Vに接続

　茶 … Gndに接続

　橙 … Arduinoの9番ピンに接続

　サーボモータはブレッドボード用ワイヤーを利用して接続する。第17課のテキストp.4の写真を参考にすると良い。この写真では、サーボモータ側でない端子はArduinoに直接接続しているが、本実験をするときのサーボモータの接続は、第18課のテキストp.3 のイラストに従う。

#include <Servo.h> // サーボーモータを使うために Servo.h をインクルード

Servo myservo; // Servo クラスのオブジェクト myservo を作成

int potpin = 0; // ポテンショメータを接続するアナログ入力の端子番号 A0

int val; // グローバル変数として宣言しているが loop の中で宣言してもよい

void setup() {

myservo.attach(9); // サーボモータ用出力は D9

}

void loop() {

val = analogRead(potpin);

val = map(val, 0, 1023, 0, 180); // 0-1023 の val を 0-180 に変換する

myservo.write(val);

delay(15);

}

# 5. 課題(自分でプログラムを組む)

　以下の課題を解決する回路図（手書きかPowerPointなどで書く）とプログラムを示しなさい。また、動作させたときの動画を別途アップロードしなさい。

## 自動照明

　暗くなるとLEDが点灯する装置を作りなさい。

(ヒント)

　光センサとしてCdSを使う。CdSは明るさによって抵抗値が変わる。最初にテスタで明るいとき、暗いときの抵抗値を測定して記録する。

　CdSと抵抗を直列に接続した以下の回路を考える。Vの値は明るさの変化に対してどのように変化するか？ 　明るいとき、暗いときの電圧Vを測定し、そのときのanalogReadの値（0～1023）を計算で求め、しきい値を中間の値に設定する。

　電源電圧を測定する。その値が のとき、analogReadの値は次式で得られる。



　analogReadの値を確認するために、シリアルモニタを利用する方法もある。100 msおきくらいにシリアルモニタにanalogReadの値を出力するとよい。

## 温度を知らせる装置

　温度が高いとき赤色LEDが点灯し、低いとき青色LEDが点灯する装置を作りなさい。閾値を何度にするかは各自が決めてよい。

(ヒント)

　温度センサとしてサーミスタで以下の回路を組む。サーミスタは温度によって抵抗が変わる素子である。室温における抵抗値、手で温めたときの抵抗値をあらかじめ測定して記録する。

　光センサの時と同様に、以下の回路を組むと、電圧Vは温度によってどのように変わるか？



# 注意！

● テスタを抵抗測定モードの状態で回路に接続してはいけない

　テスタが抵抗測定モードのとき、接続してよいのは、単体の部品である。

　抵抗測定モードのとき、測定端子間は定電流源であり、電圧が出力されている。最悪のケースは、テスタの2つのプローブを電源端子（5 VとGND）に接続したときである。電源に電源を接続することになり、テスタが損傷する可能性がある。

● Arduinoの出力端子をショートさせない

　端子を出力モードに設定し、以下のことをすると、出力端子をショートさせたことになり、出力端子のトランジスタが破壊される可能性がある。

　・Gndに接続した状態で出力をHIGHにする

　・5 Vの端子に接続した状態で出力をLOWにする

　・2個の出力端子を接続し、片方をHIGH, もう片方をLOWにする

以上

1. Arduino UNO R4にはWifiつきの機種もある。しかし、1つのスレッドしか動作しないマイコンで、ネットワーク通信のプログラムを組むのは本質的に無理があると思われる。 [↑](#footnote-ref-1)
2. 2020.7の時点でELEGOO, Kuman, Smrazaの3つを購入して比較した。説明書が最も優れているSmrazaを選択した。 [↑](#footnote-ref-2)
3. 全てのユーザー用なら　C:\Program Files　現在のユーザーなら　C:\Users\user-name\Appdata\Local\Programs\Arduino IDE [↑](#footnote-ref-3)
4. 画像引用元　https://forum.arduino.cc/t/no-5v-from-arduino-uno-r3/1088407 [↑](#footnote-ref-4)
5. ArduinoのUSB端子にはMF-MSMF050-2というリセッタブルヒューズが入っている。このヒューズは0.5 A～1.0 Vの範囲でトリップし、0.15秒以内に回路を遮断する。このヒューズの働きにより、回路は速やかに遮断されるが、遮断されるまでに流れる大電流により、パソコンのUSB端子がダメージを受ける可能性があるように思われる。 [↑](#footnote-ref-5)