平成 28 年度前期

情報学実験Ⅰ　テーマ８　Arduino による組込み制御

平成 28 年 6 月 2 日

1　実験目的

マイクロコンピュータ（microcomputer; マイコン）は，入力信号に応じて出力信号を演算する小型の装置である．ロボットでもセンサーからの入力信号に応じたアクチュエータへの出力信号を計算する際に用いられる．マイコンの演算内容はパソコン上のプログラミングで定義でき，シリアル通信等でマイコンへ書き込むことができる．センサー，アクチュエータ，マイコン，そのプログラミングをまとめて，組込システム（embedded system）と呼ぶ．組込システムによる演算は，デザイナーやアーティストからはフィジカルコンピューティングとも呼ばれる．組込システムは，低価格な部品で小型軽量に実装でき，さらにインターネット接続も容易に実現可能なため，モノのインターネット(Internet of Things: IoT）の基本技術である．

本実験の目的は，Arduino を用いて組込システムの設計と実装を行い，組込システムによるデジタル入出力，アナログ入力，パルス幅変調（Pulse Width Modulated: PWM）出力，パルス周波数変調（Pulse Frequency Modulated: PFM）出力を理解することである．なお，Arduino とは，マイコンと入出力ピン等を一つにまとめた Arduino ボードと，パソコン上で組込プログラミングを行うための Arduino 統合開発環境（Integrated Development Environment: IDE）との総称である．

Massimo Banzi「Arduino をはじめよう第 2 版」船田巧訳，オライリー・ジャパン, 2012
は，Arduino の開発者の一人が著した入門書である．本実験で扱うような導入的な操作，および，本実験を超えた内容の他に，Arduino 開発の理念・方針が書かれているので，一読を勧める．

2　使用器具

- Arduino Uno
- USB ケーブル（A-B タイプ）
- 小型ブレッドボード
- ジャンプワイヤ　150mm 10 本
- 抵抗器（100Ω，150Ω，10kΩ）
- 9V 乾電池
- DC プラグ付電池スナップ
- LED（赤色、緑色）
- 圧電スピーカー
- タクトスイッチ
- 傾きスイッチ（傾斜センサー）
- ボリューム（可変抵抗）
- パーソナルコンピュータ（諸元は情報科学研究教育センターHP を参照のこと）

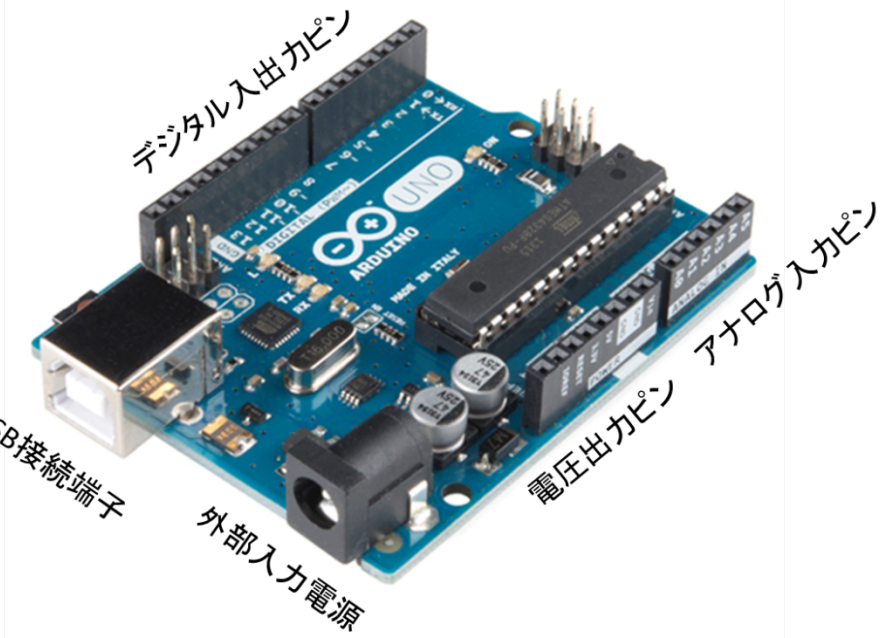

図 1: Arduino Uno ボードの外観

図 1 に Arduino ボードの外観を示す．デジタル入出力ピンは，0 番から 13 番までありデジタル入力とデジタル出力に用いる．一部のピンは PWM 信号の出力が可能である．電圧出力ピンには 3.3V, 5V, GND, Vin があり，それぞれの電圧を出力する．Vin からは外部入力電源の電圧がそのまま出力される．アナログ入力ピンは 0 番から 5 番までありアナログ入力に用いる．USB 接続端子では電源の供給も受けるが，パソコンからマイコンへ処理内容を書きこむ際に用いる．

3　設計と実装

以下の項目を順に設計し実装することにより，デジタルおよびアナログの入出力方法を理解する．

3.1　デジタル出力（LED 点滅）
3.2　PWM 出力（LED の明るさ制御）
3.3　PFM 出力（圧電スピーカー）
3.4　デジタル入力（タクトスイッチ，傾きスイッチ）
3.5　アナログ入力（ボリューム）

マイコンの演算内容は Arduino IDE 上のプログラミングにより定義する．なお，プログラミング言語は，Processing を基にした独自の Arduino 言語であり，C 言語にも似ている．Arduino がプログラマに限らずデザイナーやアーティストからも好んで用いられている経緯から，Arduino 言語によるプログラムはスケッチと呼ばれる．スケッチの中では，起動時の処理を定義する `setup` 関数と継続的な処理を定義する `loop` 関数を編集する．

ハードウェア実装とパソコン上でのスケッチ作成を終えた後，USB ケーブルで両者を接続し Arduino 内のマイコンにスケッチの内容を書き込む．一度書き込むと，電源が供給される限りマイコンに書き込まれた処理が実行されるので，USB ケーブルを外して 9V 電池から電源を供給し，処理を実行する．

Arduino IDE のダウンロードとインストール

本実験の予習に際して，下記の URL から最新版をダウンロードし，インストーラ画面の指示に従いインストールせよ．

http://arduino.cc/en/main/software

実験当日にスケッチ作成を始めても恐らく時間内に【実験 8】まで終わらないので，事前にスケッチを作成し検証・デバグしておくこと．

演習室での Arduino IDE の起動

デスクトップのコンピュータから U:ドライブ→Arduino フォルダ→arduino 1.6.4 フォルダを開き，arduino.exe を起動する．図 2 のような IDE 画面が出てくる．この IDE を用いてスケッチを作成しマイコンへの書き込みを行う．

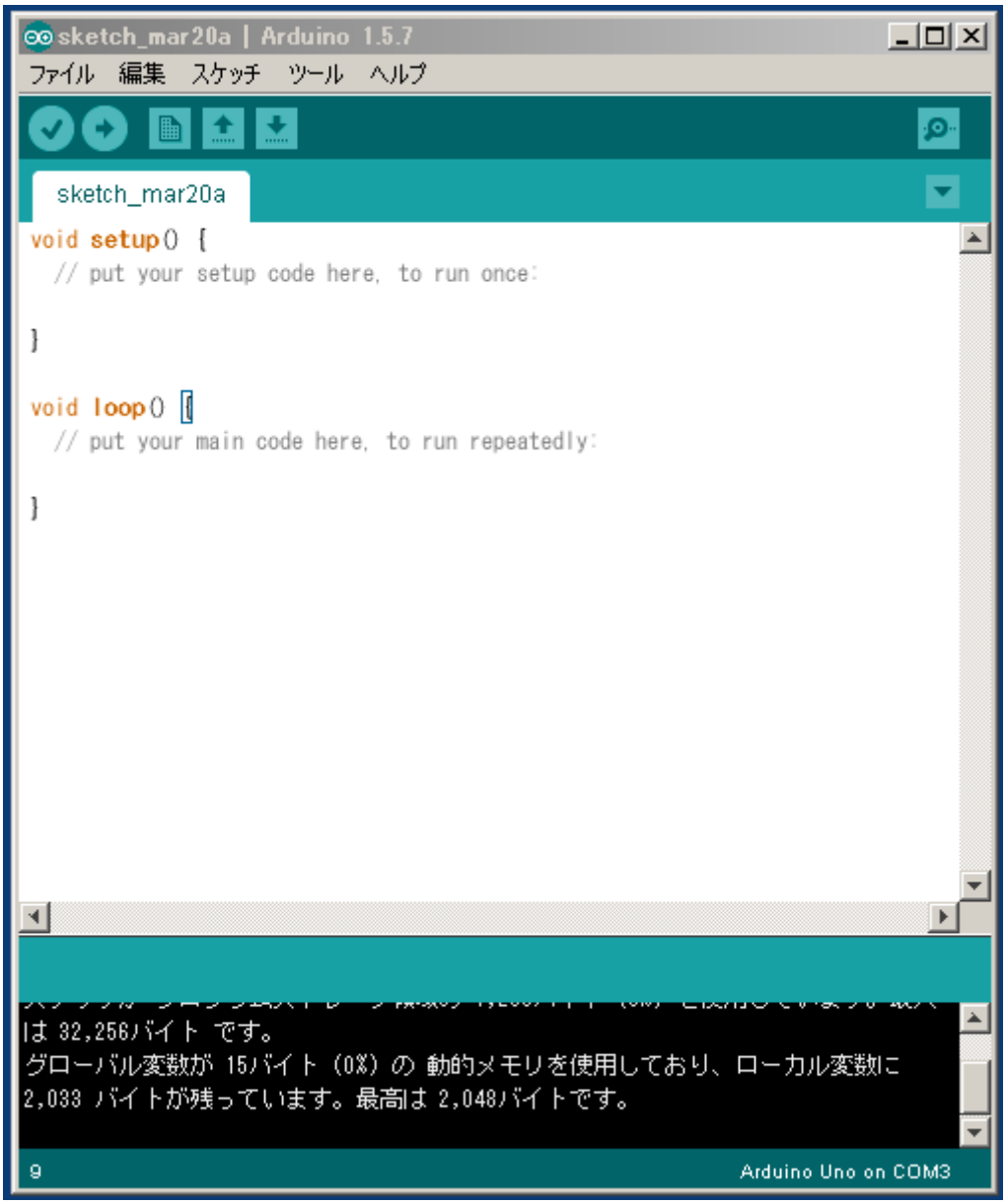

図 2: Arduino IDE の画面

3.1　デジタル出力（LED 点滅）

LED（Light Emitting Diode; 発光ダイオード）は一定の電流がかかると発光する．本実験では，デジタル出力を制御して LED を点滅させる．

ハードウェア実装

図 3 に LED 点滅の回路図を示す．小型ブレッドボードとジャンプワイヤを利用して，Arduino のデジタル入出力の 13 番端子，LED，抵抗，Arduino の GND 端子を直列に接続する．LED には向きがあるので，アノード（＋電位側，長い）とカソード（－電位側，短い）を間違えないよう注意せよ．抵抗は，回路に流れる電流を制限するために挿入されている．回路に流れる電流を 20mA とする場合，抵抗値をいくらにすればよいか考えよ．ダイオードの電圧降下は 2.0V、電源電圧は 5.0V である．

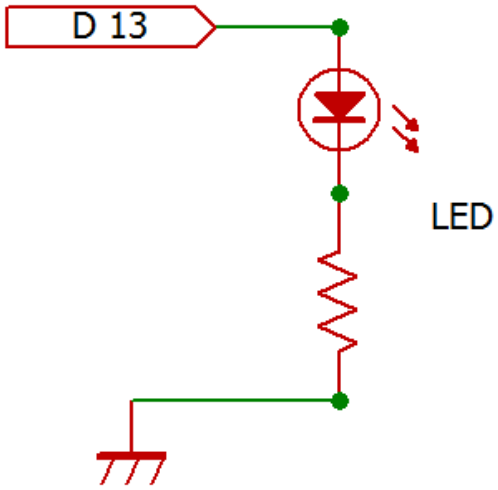

図 3: LED 点滅の回路図

スケッチ作成

スケッチ 1 を作成する．ファイル名は実験日の日付とアルファベットで自動的につけられるが，自身で管理しやすい名前に変更してもよい．スケッチの中では，スラッシュふたつ「//」に続く行末までの文は，コメントとしてマイコン書き込み時には無視される．また，C 言語と同様に「/*」と「*/」に挟まれた部分もコメントとなる．スケッチには適宜コメントを書き添えるとよい．

スケッチ 1 では，まず，LED へつなぐ端子が 13 番ピンであることを宣言し定義する．次に，`setup` 関数で，ピンを出力モードに指定する．繰り返しの処理を定義する `loop` 関数では，`digitalWrite` 関数を用いて出力ピンの電位を `HIGH` (+5V) と `LOW` (0V) に切り替える。`delay` 関数は引数に指定したミリ秒間，待機する関数である．つまり，スケッチ 1 の `loop` 関数では，出力ピンの電位を `HIGH` に切り替えて 1000 ミリ秒待機し，次に出力ピンの電位を `LOW` に切り替えて 1000 ミリ秒待機する，という処理を定義している．

スケッチを書き終えたら「検証」ボタンを押してバグがないか検証する．バグがある場合には，「コ

ンパイル時にエラーが発生しました。」というエラーメッセージの上に，バグが含まれる関数と行番号が表示されるので，それを参考にデバグする．

```
/* LED 点滅
@author expInfo
*/
int led = 13; // LED のピン
void setup(){
    pinMode(led, OUTPUT);
    // ピンを出力モードに指定
}
void loop(){
    digitalWrite(led, HIGH);
    // LED 点灯
    delay(1000); // 1 秒待機
    digitalWrite(led, LOW);
    // LED 消灯
    delay(1000); // 1 秒待機
}
```

スケッチ 1: LED 点滅

マイコンへの書き込み

USB ケーブルで Arduino ボードとパソコンを接続する．IDE の「ツール」→「ポート」メニューから接続しているポートを選択する．「マイコンボードに書き込む」ボタンを押すと，マイコンへの書き込みが開始される．書き込みに成功すると，ボード上の TX, RX インジケーターが点滅した後，スケッチで定義した処理が実行される．

電池での実行

電池プラグに 9V 電池を接続すると，マイコンに電源が供給されスケッチで定義した処理が実行される．Arduino ボードから USB ケーブルを取り外し，9V 電池を電池スナップで接続して，動作を確認せよ．パソコンから切り離して動作できることを示す写真の撮影を忘れないこと．

【実験 1】スケッチ 1 の待機時間をいろいろ変え，2 通りほどの結果を表にまとめよ．

※ Arduino ボードへ USB ケーブルと電池スナップの両方から同時に給電するとマイコンが壊れる．電池スナップを取り外してから USB ケーブルをつなぐこと．再び電池を使うときも USB ケーブルを取り外してから電池スナップをつなぐこと．

（課題）抵抗を 100Ω，150Ωにした場合それぞれについて LED 点灯時の LED 回路の電流と電力を答えよ．

3.2　PWM 出力（LED の明るさ制御）

LED の明るさを制御する実験を行う．単位時間に光が点滅する頻度を増やしていくと 50Hz 程度の周波数で連続的に点灯しているように見える．図 4 に示すように，同じ周波数で点灯の時間の比率（デューティー比，duty ratio）を変化させると，デューティー比が大きいほど明るく見える．このように，矩形波のパルスの幅を変化させて制御量を変化させることをパルス幅変調（Pulse Width Modulation: PWM）と呼ぶ．

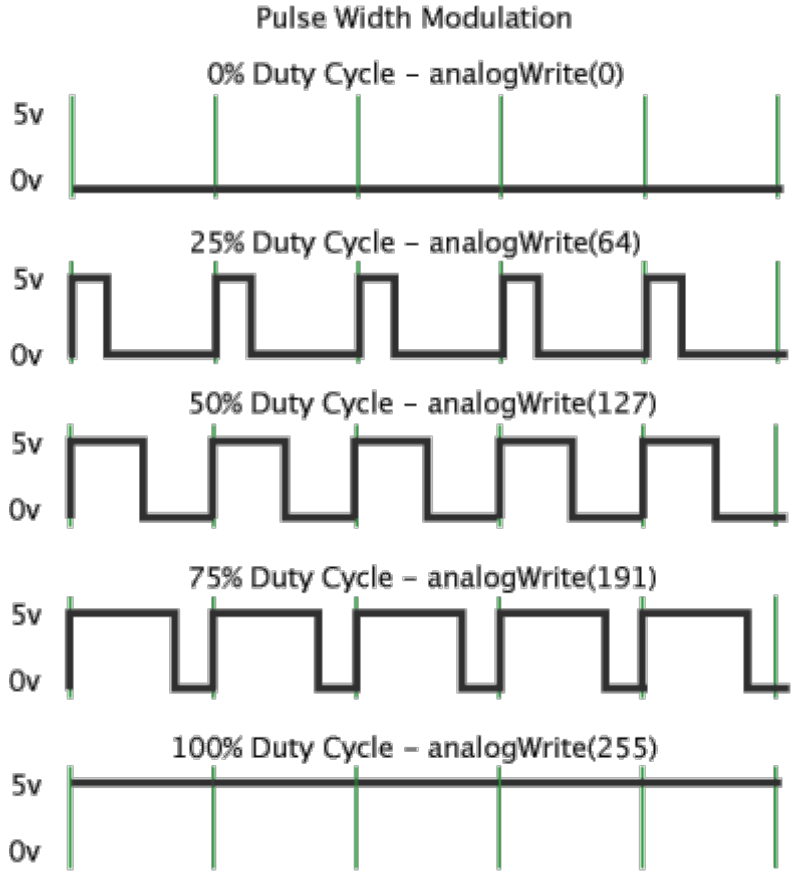

図 4: パルス幅変調

ハードウェア実装

3.1 で組んだ回路と並列に，デジタル入出力の 12 番端子，緑色 LED，150Ω抵抗，GND 端子の順に組んだ直列回路を追加する（図 5）．

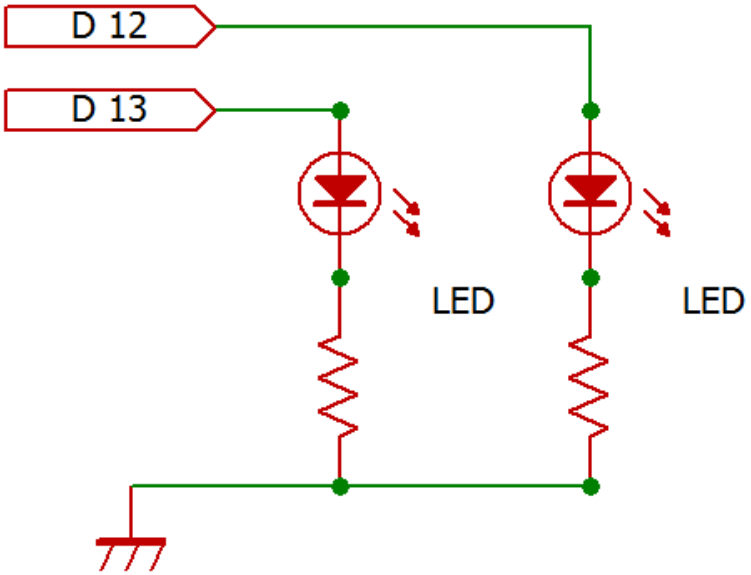

図 5: LED 明るさ制御の回路図

スケッチ作成

スケッチ 2 のように 12 番ピン用のコードも記述する．さらに LED 点滅の周波数を表す変数 `freq` とオンの時間の割合を表す変数 `dutyRatio` を宣言し定義する．さらに，赤色 LED を点灯する時間 `timeOn` と消灯する時間 `timeOff` を宣言する．`setup` 関数では緑色 LED のピンも出力モードに設定する．`loop` 関数では， `timeOn` と `timeOff` を計算し，計算した時間で赤色 LED を点滅させる．緑色 LED は，明るさの比較のため，赤色が点灯の時には消灯，赤色が消灯の時には点灯するように記述する．

```
/* LED 明るさ制御
@author expInfo
*/
int ledRed = 13; // 赤色 LED のピン
int ledGreen = 12; // 緑色 LED のピン
float freq = 50; // 点滅の周波数[Hz]
float dutyRatio = 0.1; // 点灯の割合
int timeOn, timeOff; // 点灯・消灯時間
void setup(){
    pinMode(ledRed, OUTPUT);
    pinMode(ledGreen, OUTPUT);
}
void loop(){
    timeOn = dutyRatio*1000/freq; // 点灯時間
    timeOff = 1000/freq - timeOn; // 消灯時間
    digitalWrite(ledRed, HIGH); // 赤点灯
    digitalWrite(ledGreen, LOW); // 緑消灯
    delay(timeOn); // 待機
    digitalWrite(ledRed, LOW); // 赤消灯
    digitalWrite(ledGreen, HIGH); // 緑点灯
    delay(timeOff); // 待機
}
```

スケッチ 2: LED 明るさ制御

マイコンへの書き込みと電池での実行

3.1 と同様に実行せよ。以降はこの指示を省略するが，各処理の動作をパソコンから切り離して確認すること.

【実験 2】dutyRatio を 0.1, 0.5, 0.9 に変えた場合の LED の明るさを調べよ. また, freq を 0.2Hz, 2Hz, 50Hz に変えた場合の動作を観察せよ.

※ 10Hz 程度で点滅する輝度が高くコントラストが強い光を見ると，体調異常（光感受性障害）を起こす恐れがある．3～30Hz の周波数帯では絶対に実行しないこと.

(課題)デューティー比を変えた場合, 赤色 LED 回路と緑色 LED 回路それぞれの平均電力を求めよ.

ハードウェア実装（PWM～端子を使う）

3.2 で作成した回路では 13 番端子と 12 番端子から電位を取っていた．これらを 11 番端子と 10 番端子に接続し直す（図 6）．数字の前に「～」が印字されている端子は，以下のように analogWrite 関

数を用いて PWM 信号を出力できる．

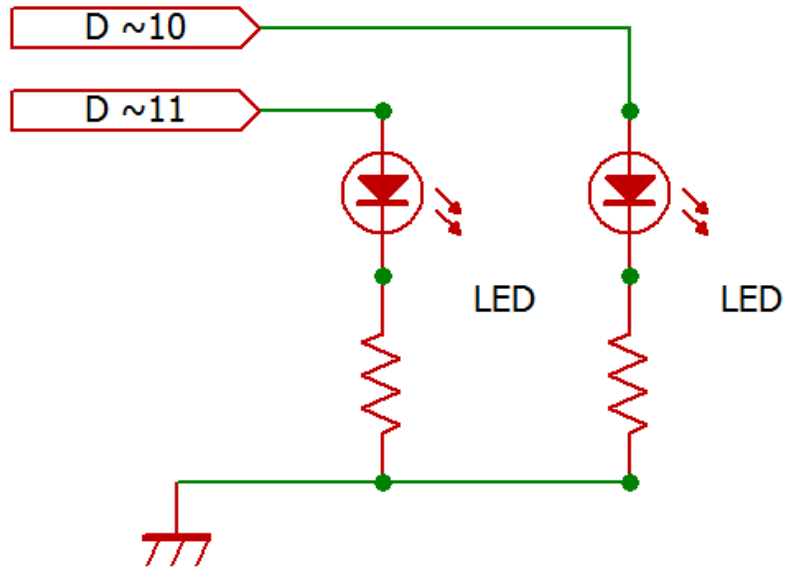

図 6: LED 明るさ制御の回路図（PWM～端子）

スケッチ作成（PWM～端子を使う）

スケッチ 3 に PWM 信号による明るさ制御のコードを示す．analogWrite 関数は，関数名にアナログとあるが，実際は前述のようにオンオフを繰り返すデジタル信号を出力する．ピン番号と出力の強さを引数に持ち，出力の強さは 0～255 の整数で表される．範囲外の数値が引数に指定された場合は 256 で割った余りが出力される．

```
/* LED 明るさ制御 (PWM~端子を使う)
@author expInfo
*/
int ledRed = 11; // 赤色 LED のピン
int ledGreen = 10; // 緑色 LED のピン
int red = 26; // 赤色の強さ
void setup(){
    pinMode(ledRed, OUTPUT);
    pinMode(ledGreen, OUTPUT);
}
void loop(){
    analogWrite(ledRed, red); // PWM 出力
    analogWrite(ledGreen, 255-red); // PWM 出力
}
```

スケッチ 3: LED 明るさ制御（PWM～端子）

スケッチ作成（明るさを変える）

次に，スケッチ 4 のように変数 deltaRed を導入し，LED の明るさを動的に変化させる．

```
/* LED 明るさの動的変化
@author expInfo
*/
int ledRed = 11; // 赤色 LED のピン
int ledGreen = 10; // 緑色 LED のピン
int red = 0; // 赤色の強さ
int deltaRed = 1; // red の変化量
void setup(){
    pinMode(ledRed, OUTPUT);
    pinMode(ledGreen, OUTPUT);
}
void loop(){
    red += deltaRed; // red を変化させる
    analogWrite(ledRed, red);
    analogWrite(ledGreen, 255-red);
    delay(10);
}
```

スケッチ 4: LED 明るさの動的な変化

【実験 3】スケッチ 4 では，赤色 LED の明るさが 255 から 0 に急激に変化し同時に緑色 LED の明るさが 0 から 255 に急激に変化する瞬間がある．赤色 LED が 0 から 255 まで徐々に明るく（同時に緑色が 255 から 0 に暗く）なると今度は 0 に向かって徐々に暗く（緑色は 255 に向かって明るく）なる動作を繰り返すようなスケッチを作成し，動作を確認せよ．

なお，Arduino 言語では C 言語と同様に if 文や数値が等しいかの判定「==」，論理和「||」，数値の代入「=」（例：deltaRed = -deltaRed）が使える．

＜発展課題＞ analogWrite 関数で出力される矩形波の周波数が何 Hz か書籍などで調べよ．

＜発展課題＞ PWM 信号を，電圧の強弱で出力値を表す「本当のアナログ」信号（Amplitude Modulated: AM）に変換するには，どのような素子を用いればよいか考えよ．

3.3 PFM 出力（圧電スピーカー）

圧電スピーカーでメロディを奏でる実験を行う。圧電スピーカーは，電圧に応じて膜の張力を変化させる．スピーカーにかかる電圧を高い周波数でオンオフさせると，高い周波数で空気が振動し，人間の耳には高い音が聞こえる．反対に，電圧を低い周波数でオンオフさせると低い音が聞こえる．このように，矩形波パルスの周波数で信号を変化させることをパルス周波数変調（Pulse Frequency

Modulation: PFM）と呼ぶ.

ハードウェア実装

回路から緑色 LED の回路を取り除き，さらに，赤色 LED を圧電スピーカーに置き換える（図 7）.

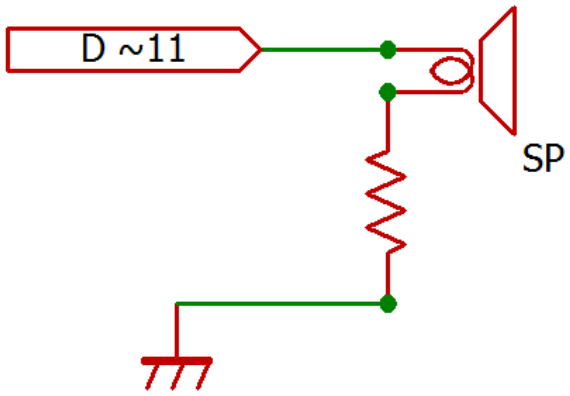

図 7: メロディ演奏の回路図

スケッチ作成

スケッチ 5 はメロディを演奏する．tone 関数はピン番号と周波数を引数に持つ．tone 関数による矩形波は，次に tone 関数が呼び出されるか，矩形波をなくす noTone 関数が呼び出されるまで出力を続ける.

```
/* ドレミ演奏
@author expInfo
*/
int spkr = 11; // スピーカーのピン
float freqC = 261.6; // ドの周波数[Hz]
float freqD = 293.7; // レの周波数[Hz]
float freqE = 329.6; // ミの周波数[Hz]
int time = 200;
void setup(){
    pinMode(spkr, OUTPUT);
}
void loop(){
    tone(spkr, freqC); // ドを鳴らす
    delay(time); // 待機
    tone(spkr, freqD); // レを鳴らす
    delay(time); // 待機
    tone(spkr, freqE); // ミを鳴らす
    delay(time); // 待機
    noTone(spkr); // 音を消す
    delay(time); // 待機
}
```

スケッチ 5: メロディ演奏

【実験 4】変数 state などを導入して低いドレミと高いドレミを交互に鳴らすようなスケッチを作成し，動作を確認せよ．なお，周波数を 2 倍にすると 1 オクターブ上がり，半分にすると 1 オクターブ下がる．

（課題） PFM 信号の図を，PWM 信号の図 4 のように，横軸を時間にして描け．

＜発展課題＞ 配列を用いて広い音域を使った長いメロディを鳴らせ．音階と周波数については書籍やインターネットで調べよ．スケッチ 6 に，freqC などを配列として持たせ for ループを用いるコードを示す．また，setup 関数と loop 関数の中でそれぞれサブ関数を呼出す．

＜発展課題＞ tone 関数で出力される矩形波のデューティー比はいくつか調べよ．

```
/* 配列とサブ関数を用いるドーレミレ演奏
@author expInfo
*/
int spkr = 11; // スピーカーのピン
float freq [] = {261.6, 293.7, 329.6};
// ドレミの周波数[Hz]用配列
float melody [4]; // メロディ用配列
int time [5]; // 時間用配列
void setup(){
    pinMode(spkr, OUTPUT);
    initMelody(); // サブ関数を呼出す
}
void initMelody(){
    melody[0] = freq[0]; time[0] = 400;
    melody[1] = freq[1]; time[1] = 200;
    melody[2] = freq[2]; time[2] = 200;
    melody[3] = freq[1]; time[3] = 200;
    time[4] = 200;
}
void loop(){
    playMelody(); // サブ関数を呼出す
}
void playMelody(){
    for(int i=0; i<4; i++){
        tone(spkr, melody[i]); // 鳴らす
        delay(time[i]); // 待機
    }
    noTone(spkr); // 音を消す
    delay(time[4]); // 待機
}
```

スケッチ 6: 配列とサブ関数を用いるメロディ演奏

3.4　デジタル入力（タクトスイッチ，傾きスイッチ）

デジタル信号を入力する実験を行う．3.3 までは，周りの物理環境やユーザの意図に関わらず定義した通りに処理を実行するスケッチを作成した．外部からマイコンへ信号を入力できれば，物理環境やユーザの意図に応じた処理を実行できる．

ハードウェア実装

図 8 にデジタル入出力の回路図を示す．電圧出力の 5V 端子，タクトスイッチ，10kΩ抵抗，GND 端子を直列につなぎ，タクトスイッチと 10kΩ抵抗との接点をデジタル入出力の 7 番端子へ接続せよ．タクトスイッチは，上（ボタン）側から見て平行に出ている端子どうしが常時つながっており，ボタンを押すと，内部の金属板が下に押され 4 つの端子すべてがつながる．この回路と独立に，デジタル入出力の 13 番端子，LED，150Ω抵抗，GND 端子を直列につなぐ．

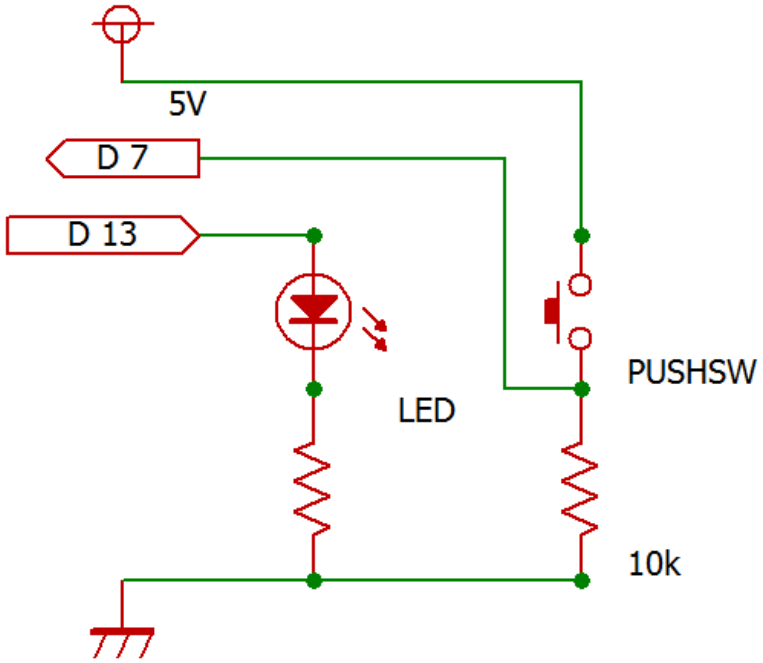

図 8: デジタル入出力の回路図

スケッチ作成

スケッチ 7 はデジタル入出力を行う．`setup` 関数では，入力ピンの `pinMode` を `INPUT` に指定する．Arduino にはシリアルポートに関する関数が用意されており，その中の `Serial.begin` 関数によりパソコンとマイコンの間のシリアル通信を開始する．引数はデータ転送レートを bps（ビット／秒）で示しており，ここでは 9600bps とする．`loop` 関数では，`digitalRead` 関数により入力ピンの入力値を読み取る．`Serial.println` 関数は，引数の値をシリアルモニターに表示する．シリアルモニターは「ツール」メニューから表示できる．また，`digitalWrite` 関数では，C 言語と同様の三項演算子（`[論理式]?[真の場合の値]:[偽の場合の値]`）を用いて入力値を反転させ，出力する．

```
/* デジタル入出力
@author expInfo
*/
int pinIn = 7; // 入力ピン
int led = 13; // LED ピン
int val = LOW; // 入力値用の変数
void setup(){
    pinMode(pinIn, INPUT); // 入力モード
    pinMode(led, OUTPUT);
    Serial.begin(9600); // シリアル通信開始
}
void loop(){
    val = digitalRead(pinIn); // 入力
    digitalWrite(led, val==HIGH?LOW:HIGH);
    // 入力値を反転して LED に出力
    Serial.println(val);
    // シリアルモニターに表示
    delay(10); // 待機
}
```

スケッチ 7: デジタル入出力

【実験 5】タクトスイッチの代わりに傾きスイッチを用いて，動作とシリアルモニターの表示を確認せよ．電池で動作させるときはシリアル通信がされないので，シリアルモニター表示は確認しなくてよい. 傾きスイッチは, 少し振ってみるとカタカタ音がする. これは内部にある金属の玉の音である. 傾きスイッチをブレッドボードに差し込み，ブレッドボードごと傾けると，金属の玉が 4 本の端子すべてをつなげ電位が等しくなる.

スケッチ作成（切り替えスイッチ）

スケッチ 7 では，スイッチを押している間ずっと LED が消えるが，スイッチを離すと LED が点いてしまう．消灯を続けるには，スイッチをずっと押していなければならない．スケッチ 8 は，家庭や教室の電灯と同じように，スイッチが押された時，点灯から消灯に，消灯から点灯に切り替える．変数 valPrev に前回のデジタル入力値を代入し，前回が LOW で今回が HIGH なら LED の state を反転する.

```
/* 切り替えスイッチ
@author expInfo
*/
int pinIn = 7; // 入力ピン
int led = 13; // LED ピン
int val = LOW; // 入力値
int valPrev = LOW; // 前回の入力値
int state = LOW; // LED の状態
void setup(){
    pinMode(pinIn, INPUT); // 入力モード
    pinMode(led, OUTPUT); // 出力モード
}
void loop(){
    val = digitalRead(pinIn); // 入力
    if(valPrev==LOW && val==HIGH){
        state = state==HIGH?LOW:HIGH; // 状態を反転
        digitalWrite(led, state); // 出力
        delay(100); // 動作を安定させるため少し待機
    }
    valPrev = val;
}
```

スケッチ 8: 切り替えスイッチ

【実験 6】デジタル入力を利用し，スイッチを押すとメロディが鳴り始め，再びスイッチを押すとメロディが鳴りやむような回路とスケッチを作成せよ．スイッチにタクトスイッチと傾きスイッチを用いる両方の場合で動作を確認せよ．

3.5　アナログ入力（ボリューム）

可変抵抗を内蔵するボリュームつまみを使い，アナログ入力の実験を行う．物理環境から観測する変数の多くは連続値である．連続値変数は量子化（quantize）され電気信号として扱われる．Arduino のアナログ入力端子では 0V～+5V の電圧が 10 ビットで量子化される．

ハードウェア実装

ボリュームの三つの端子に 5V，A0，GND をそれぞれ接続する．また，この回路とは独立に，3.2 で組んだ赤色 LED と緑色 LED の回路を作成する（図 9）．

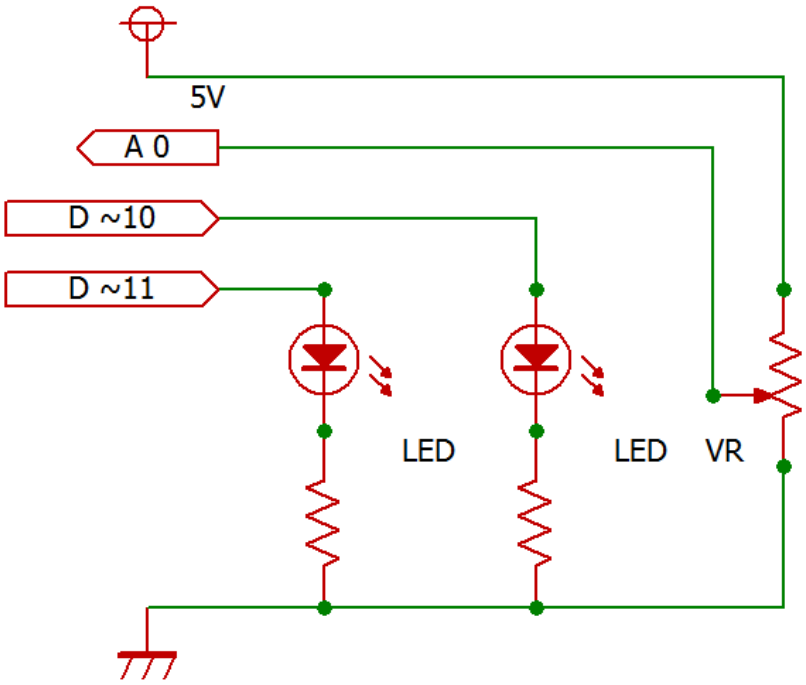

図 9: アナログ入力 PWM 出力の回路図

スケッチ作成

スケッチ 9 にアナログ入力に応じて PWM 出力するコードを示す．ボリュームを回し赤色 LED が明るくなると緑色 LED が暗くなり，反対に回すと逆になる．ただし，ボリュームを片方の端からもう片方の端まで回した時，明暗が 4 回繰り返される．

```
/* アナログ入力
@author expInfo
*/
int volume = 0; // アナログ入力ピン
int ledRed = 11; // 赤色 LED のピン
int ledGreen = 10; // 緑色 LED のピン
int red; // アナログ入力値
void setup(){
    // アナログ端子は自動的に入力とみなされる
    pinMode(ledRed, OUTPUT); // 出力モード
    pinMode(ledGreen, OUTPUT); // 出力モード
}
void loop(){
    red = analogRead(volume); // アナログ入力
    analogWrite(ledRed, red); // PWM 出力
    analogWrite(ledGreen, 255-red); // PWM 出力
}
```

スケッチ 9: アナログ入力を PWM 出力

【実験 7】スケッチ 9 でボリュームを端から端まで回した時に赤色が明→暗，緑色が暗→明と 4 回繰り返される原因を考えよ．（ヒント：前述のように analogWrite 関数は，8 ビット量子化した値 0～255 を出力する．）その上で，ボリュームを端から端まで回した時に 1 回だけ徐々に変化するようなスケッチを作成し，動作を確認せよ．

【実験 8】どちらかの LED を圧電スピーカーに置き換えた回路を作成せよ．ボリュームを回して LED が明から暗へ変化すると同時にスピーカーの音が高音（高いド）から低音（低いド）へ連続的に変化し，反対に回すと逆に変化するようなスケッチを作成し，動作を確認せよ．

4　レポート報告事項

実験日時などの基本事項，実験目的，実験方法，実験結果，考察，結論のように章立てし，【実験 1】から【実験 8】までについて報告せよ．特に，各処理をパソコンから切り離して動作確認したことを，写真等を使って報告せよ．（課題）＜発展課題＞については考察に含めて論ぜよ．レポートの本文中にスケッチ全文を載せる必要はない．本文には，スケッチのうち特筆すべき箇所だけを抜き出して記述せよ．