システム設計CAD　マイコンプログラミング編

1. 設計物とその目的

このプロジェクトでは、ブレッドボードを用いてLED、スライドスイッチ、プッシュスイッチ、そして可変抵抗といった電子部品からなる入出力回路を自己構築し、それらをArduinoのマイクロコントローラに接続します。この手作業による回路設計と構築は、電子工学の基本的な理解とスキルを深めるための重要なステップです。

次に、接続したこれらのデバイス群に対して、特定の課題に対応するためのプログラムを書きます。これはソフトウェアとハードウェアが適切に連携することの必要性を実感する良い機会となります。また、この過程で、コードが物理的なデバイスとどのように相互作用するか、具体的な視覚的表現を通じて理解することができます。

しかし、電子回路を設計と構築する際には、特定の注意点があります。誤った接続や設定を行うと、PCやArduino自体に損傷を与える可能性があるためです。そのため、PCへの接続や電源供給を行う前に、必ず短絡のチェックを行います。具体的には、Vin-GND、5V-GND、3.3V-GNDといった異なる電圧レベル間の抵抗値を測定します。

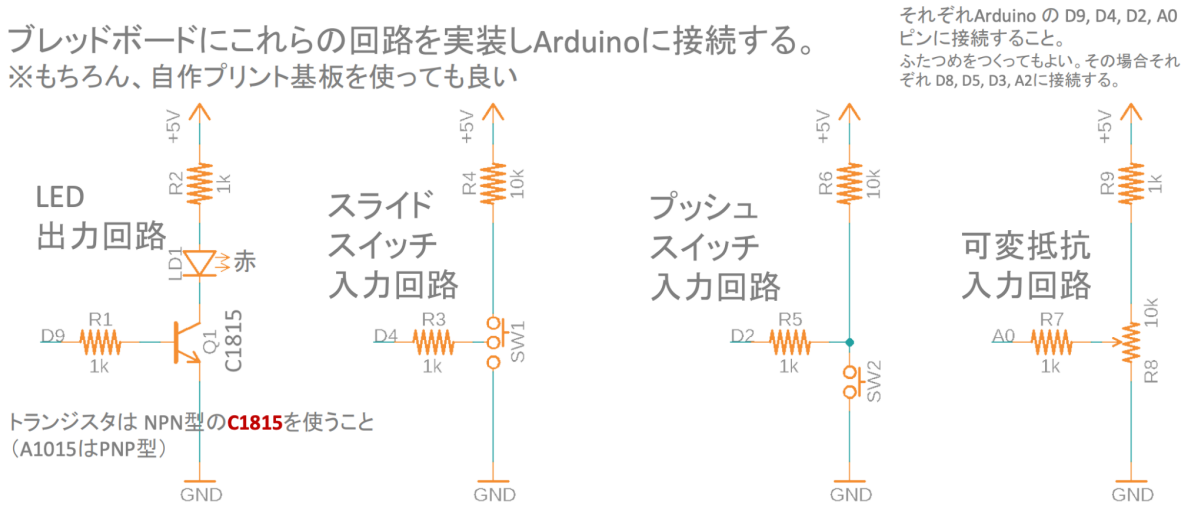
また、電子工学の実践においては、視覚的な確認も非常に重要です。特に、5Vの接続および抵抗、そしてGNDの接続は、直接触れて確認することが推奨されます。これはハードウェアに特有の実践的な手順であり、回路の物理的な状態を理解する上で重要なスキルとなります。

2. 設計概説

本プロジェクトの設計目的、使用した素材やツール、そして設計の過程とその結果について詳細に記述します。

プロジェクトの主な目的は、電子部品を使用して独自の入出力回路を設計し、それをArduinoと接続することで、実際に動作する物理的なシステムを構築することにありました。これにより、電子工学の基本的な概念と、ソフトウェアとハードウェアがどのように相互作用するかについての理解を深めることを目指しています。

この目的を達成するために、ブレッドボード、LED、スライドスイッチ、プッシュスイッチ、可変抵抗、そしてArduinoを含む多くの素材とツールを使用しました。ブレッドボードは回路を物理的に組み立てるための基盤で、各部品はブレッドボード上で必要に応じて組み合わせ、接続されます。Arduinoはこれらの部品を制御し、互いに連携するための中心的な役割を果たしました。



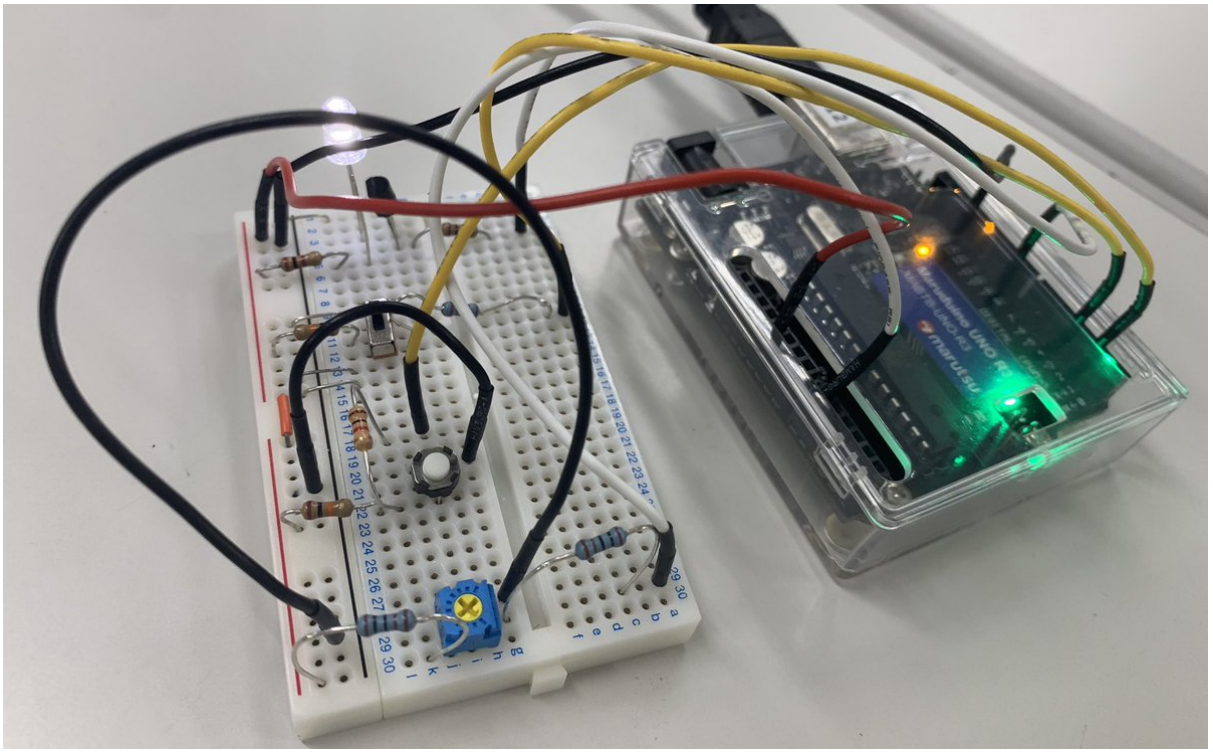
設計過程では、初めに回路の全体的な設計図を作成しました。そして、その設計に基づいて各部品をブレッドボード上に配置し、適切に配線しました。この過程で、短絡を避けるために各接続点の電圧レベルを確認し、回路の安全性を保証しました。

次に、Arduinoに対して、各部品が期待する動作を行うためのプログラムを書きました。それにより、ユーザーの入力に基づいてLEDの点灯や消灯、スライドスイッチやプッシュスイッチの状態変更などを制御することができます。課題2のところまで自分で取り組むことができました。私は課題**2**についてレポートを書きます。

最終的な設計結果として、ユーザーからの物理的な入力を受け取り、それに応じてLEDの状態を変化させるシステムが実現しました。これは、ハードウェアとソフトウェアが複雑に絡み合う実世界のシステムを設計し、構築する経験を通じて、私たちが目指した学習目標を達成した具体的な証です。

以上が本プロジェクトの「設計概説」になります。設計目的の達成に向けた方法論から具体的な手順、そしてその結果まで、詳細に踏み込んだ説明を通じて、私たちがどのようにして理論から実践へと移行したかを明示することができました。

3. 設計結果



課題2は以下の通りです。pulseIn() & analogWrite()

使用する関数
- pinMode()
- digitalRead()
- pulseIn()
- analogWrite()
- delay()

要件
- プッシュスイッチを1度押すとLEDの明るさが変わる
- 押してから放すまでの時間によりLEDの明るさが変わる
    - 0秒では0%
    - 1秒につき10%程度上昇
    - 10秒以上では100%
- 60秒でタイムアウトして消灯

以下のような仕様でプログラムを書きました。----------
```
#define BUTTON_PIN 2
#define LED_PIN 9
#define TIMEOUT 60000 // 60 seconds in milliseconds

void setup() {
  pinMode(BUTTON_PIN, INPUT);
  pinMode(LED_PIN, OUTPUT);
```

```
}

void loop() {
  unsigned long pulseWidth = pulseIn(BUTTON_PIN, HIGH, TIMEOUT);

  if (pulseWidth > 0) {
    int brightness = pulseWidth / 1000; // Convert pulse width to seconds
    brightness = min(brightness, 10); // Limit to 10 seconds
    brightness = map(brightness, 0, 10, 0, 255); // Convert to brightness scale (0 -
255)

    analogWrite(LED_PIN, brightness);
  } else {
    analogWrite(LED_PIN, 0); // Turn off LED after timeout
  }
}
```
―----------

結果として正常にコンパイルして実行することはできましたが、LEDは最初から光ったままの状態になってしまいました。つまり意図した挙動を完全に達成することはできませんでした。田中先生と話しながらデバック手法を検討していましたが時間切れとなってしまいました。

なぜできなかったのかについて考えられる限り書いていこうと思います。

チェック項目

1. プルダウン抵抗:
    - ボタンが接続されているピンにプルダウン抵抗が接続されている。

理由:この抵抗がないと、ボタンが押されていないときのピンの状態が不確定になり、誤った値を読み取る可能性がある。

2. コードの挙動:
    - `Serial.begin()`と`Serial.println()`を使用してデバッグ

理由:これにより、`pulseWidth`の値が期待どおりになっているか、また`brightness`が正しく計算されているかを確認できる。

以下のように`setup()`と`loop()`内にデバッグ情報を出力するコードを追加してデバッグをしました。

```
void setup() {
  pinMode(BUTTON_PIN, INPUT);
  pinMode(LED_PIN, OUTPUT);
  Serial.begin(9600);  // Start serial communication at 9600 baud rate
```

```
}

void loop() {
  unsigned long pulseWidth = pulseIn(BUTTON_PIN, HIGH, TIMEOUT);
  Serial.print("Pulse width: ");
  Serial.println(pulseWidth);

  if (pulseWidth > 0) {
    int brightness = pulseWidth / 1000; // Convert pulse width to seconds
    brightness = min(brightness, 10); // Limit to 10 seconds
    brightness = map(brightness, 0, 10, 0, 255); // Convert to brightness scale (0 -
255)
    Serial.print("Brightness: ");
    Serial.println(brightness);
    analogWrite(LED_PIN, brightness);
  } else {
    analogWrite(LED_PIN, 0); // Turn off LED after timeout
  }
}
```

3. アナログ通信とシリアル通信:

- LEDをPWM対応のピンに接続していることを確認する

理由：すべてのデジタルピンがPWMをサポートしているわけではない。たとえば、Arduino Unoではピン3、5、6、9、10、11がPWMに対応している。

4.**Arduino**の電気配線

- ボタンの接続: ボタンは、通常、デジタル入力ピンとGNDピンの間に接続する。ボタンが押されていないときにピンがHIGHまたはLOWにプルアップまたはプルダウンされていることを確認する。

- **LED**の接続: LEDの正極（長いリード）がPWM対応のデジタルピンに接続され、負極（短いリード）がGNDに接続されていることを確認する。

- 抵抗の使用: LEDとボタンに適切な抵抗が使用されていることを確認する。LEDには通常、カレントリミティング抵抗が必要で、ボタンはプルアップまたはプルダウン抵抗が必要

以上のようなことを丁寧に確認していけばできたと思いました。

4. まとめ

設計の概観

この演習を通じて、Arduinoを使用したハードウェアとソフトウェアの基本的な操作を学習しました。具体的には、スライドスイッチ、プッシュスイッチ、LED、および可変抵抗の制御について学びました。ArduinoのデジタルI/O関数、PWMを用いたアナログ出力、デジタルおよびアナログの読み取り、そしてデバッグのためのシリアル通信の使用方法について理解を深めました。

学んだこと

以下の主要なテーマについて理解を深めました:

1. **I/O**: Arduinoのデジタル入出力関数（`pinMode()`, `digitalRead()`, `digitalWrite()`）、アナログ入出力（`analogRead()`, `analogWrite()`）、およびパルス幅測定関数（`pulseIn()`）の使用方法を学習しました。
2. デバッグ: Arduinoのシリアル通信機能を用いたデバッグ方法について学びました。これにより、実行中のプログラムの状態を確認し、問題の解析と解決が可能となりました。
3. ハードウェア: LED、スイッチ、および可変抵抗の接続方法と動作原理を学びました。これらのコンポーネントは一般的な電子プロジェクトでよく使用されるため、その理解は重要です。

これからの展望

この演習を通じて獲得した知識とスキルは、より複雑なArduinoプロジェクトの基盤となります。例えば、センサーデータの読み取りと処理、モーターの制御、通信機能（Bluetooth、Wi-Fi）の利用など、様々な応用が可能です。

さらに、Arduino以外のマイクロコントローラ（Raspberry Pi、ESP32など）や、より高度なプログラミング技術（割り込み、マルチスレッドなど）の学習にも進むことができます。