組込システムⅠ
第６回 課題

提出日 2025/05/29

学籍番号 21T2166D
名前 渡辺 大樹

1 課題

本課題では、Raspberry Pi 4B を用いてハードウェア PWM 制御により LED の輝度を変化させ、その光を CdS 素子で受光する。CdS 素子の抵抗値変化を MCP3002 ADC を介して測定し、PWM のデューティ比と CdS 素子の抵抗値の関係をグラフで示す。また、PWM の具体的な用途について調査する。

2 使用部品

- Raspberry Pi 4B
- 白色 LED
- 抵抗 330Ω (LED 用電流制限抵抗)
- CdS 光センサ (光可変抵抗器)
- 抵抗 6.8kΩ (本レポートの計算では、CdS 素子と分圧回路を構成する抵抗として使用)
- MCP3002 (2 チャンネル 10bit A/D コンバータ)
- ブレッドボード、ジャンパワイヤ

3 回路の説明

本実験で使用した回路の構成を図 3 に示す。

- **LED 制御回路**: Raspberry Pi の GPIO19 (物理ピン 35) をハードウェア PWM 出力として使用し、白色 LED の輝度を制御する。LED には 330Ω の抵抗が電流制限のために直列に接続されている。PWM 信号の周波数は Python スクリプト内で 10kHz に設定されている。
- **光センサ回路と AD 変換**: CdS 素子と 6.8kΩ の抵抗を用いて分圧回路を構成する。LED からの光量に応じて CdS 素子の抵抗値が変化し、それによって分圧点の電圧 (V_{CDS}) が変動する。このアナログ電圧を MCP3002 A/D コンバータの CH0 に入力する。MCP3002 は 10bit の逐次比較型 ADC であり、SPI 通信プロトコルを介して Raspberry Pi と接続される。Raspberry Pi の SPI0 バス (GPIO10/MOSI, GPIO9/MISO, GPIO11/SCLK, GPIO8/CE0) が MCP3002 の DIN, DOUT, CLK, CS 端子にそれぞれ接続される。MCP3002 は入力されたアナログ電圧を 0 から 1023 の範囲のデジタル値に変換する。

回路全体の電源は Raspberry Pi の 3.3V ピンから供給される。図 3 では、CdS 素子と 6.8kΩ 抵抗の接続において、V_{CDS} が 6.8kΩ 抵抗にかかる電圧として示されている。この場合の CdS 抵抗値の計算式と実験データの整合性については考察で述べる。

4 アルゴリズムの説明

本実験の制御は Python スクリプト (ソースコード 1) により行われる。

1. **初期化**:
   - pigpio ライブラリを初期化し、GPIO19 を出力モードに設定する。
   - spidev ライブラリを初期化し、SPI デバイス (0,0) を開き、ビット長 (8bit/word) と最大速度 (10kHz) を設定する。
   - MCP3002 からデータを読み取るための SPI コマンドの各部 (start bit, single-ended mode, channel select, MSB first) を定義する。
2. **PWM デューティ比の変更と ADC 値の測定ループ**:
   - PWM のデューティ比に対応する変数 i を 0 から 99 まで 1 ずつ増加させるループを実行する。

- duty_func(i) 関数は i * 1000 を返す。pi.hardware_PWM(PIN, Hz, duty_func(i)) により、GPIO19 から周波数 10kHz、デューティサイクル i * 1000 (pigpio ライブラリのデューティサイクル範囲は 0-1,000,000) の PWM 信号を出力する。これは、i が示すパーセント値の 0.1 倍のデューティ比 (例: i=50 なら 5% デューティ) に相当する。本レポートのグラフ横軸はスクリプト内の i の値をそのまま「デューティ比 (%)」として表記する。
- 各デューティ比において、`mcp3002(ch0)` 関数を 100 回呼び出して CH0 から AD 変換値を取得し、その合計を求める。各読み取りの間には 1ms の遅延 (`time.sleep(0.001)`) を挿入する。
- 100 回の測定値の平均を算出し、そのデューティ比における平均 ADC 値としてコンソールに出力する。

3. **終了処理**: ループ完了後、または KeyboardInterrupt 発生時に、GPIO19 を入力モードに戻し、pigpio ライブラリを停止する。

ソースコード 1　実験用 Python スクリプト (Python/BuiltIn/exam6-4.py)

```
import pigpio
import time
import spidev

PIN = 19
Hz = 10000
pi = pigpio.pi()
pi.set_mode( PIN, pigpio.OUTPUT )

spi = spidev.SpiDev()
spi.open(0, 0)
spi.bits_per_word = 8
spi.max_speed_hz = 10000

start = 0b01000000
sgl  = 0b00100000
ch0  = 0b00000000
ch1  = 0b00010000
msbf = 0b00001000

def mcp3002(ch):
   rcv = spi.xfer2([(start + sgl + ch + msbf ), 0x00 ] )
   ad = ((( rcv[0] & 0x03 ) << 8 ) + rcv[1] )
   return ad

def duty_func(per):
   return per * 1000

try:
   for i in range(100):
      pi.hardware_PWM( PIN, Hz, duty_func(i) )
      sum_adc = 0
      for j in range(100):
         sum_adc += mcp3002(ch0)
         time.sleep(0.001)

      cds_adc = sum_adc / 100
      print(f"Duty␣{i}␣%␣cds␣R:␣{cds_adc}")

   pi.set_mode(PIN, pigpio.INPUT)
   pi.stop()

except KeyboardInterrupt:
   pass

pi.set_mode(PIN, pigpio.INPUT)
pi.stop()
```

注*:* 上記ソースコードリストでは、*Python* の組み込み関数 ***sum*** との衝突を避けるため、変数名を ***sum_adc*** に変更して示している。また、***print*** 文中の ***"cds R:"*** は生の平均 *AD* 変換値を指す。

5　結果

PWM デューティ比 (Python スクリプト内の変数 i) を 0% から 99% まで変化させた際の平均 ADC 値と、それから算出した CdS 素子の抵抗値を表 1 に抜粋し、全データポイントを用いたグラフを図 1 に示す。CdS 素子の抵抗値 R_{CDS} は以下の式を用いて算出した。

$$R_{CDS}[\mathrm{k\Omega}] = 6.8 \times \frac{\text{平均 ADC 値}}{1023 - \text{平均 ADC 値}}$$

表 1　PWM デューティ比と平均 ADC 値、算出 CdS 抵抗値 (抜粋)

デューティ比 (%)	平均 ADC 値	算出 CdS 抵抗値 (kΩ)
0	1022.65	19867.29
10	484.68	6.12
20	357.91	3.66
30	296.27	2.77
40	259.13	2.31
50	232.82	2.00
60	213.67	1.80
70	198.11	1.63
80	185.84	1.51
90	175.62	1.41
99	167.74	1.33

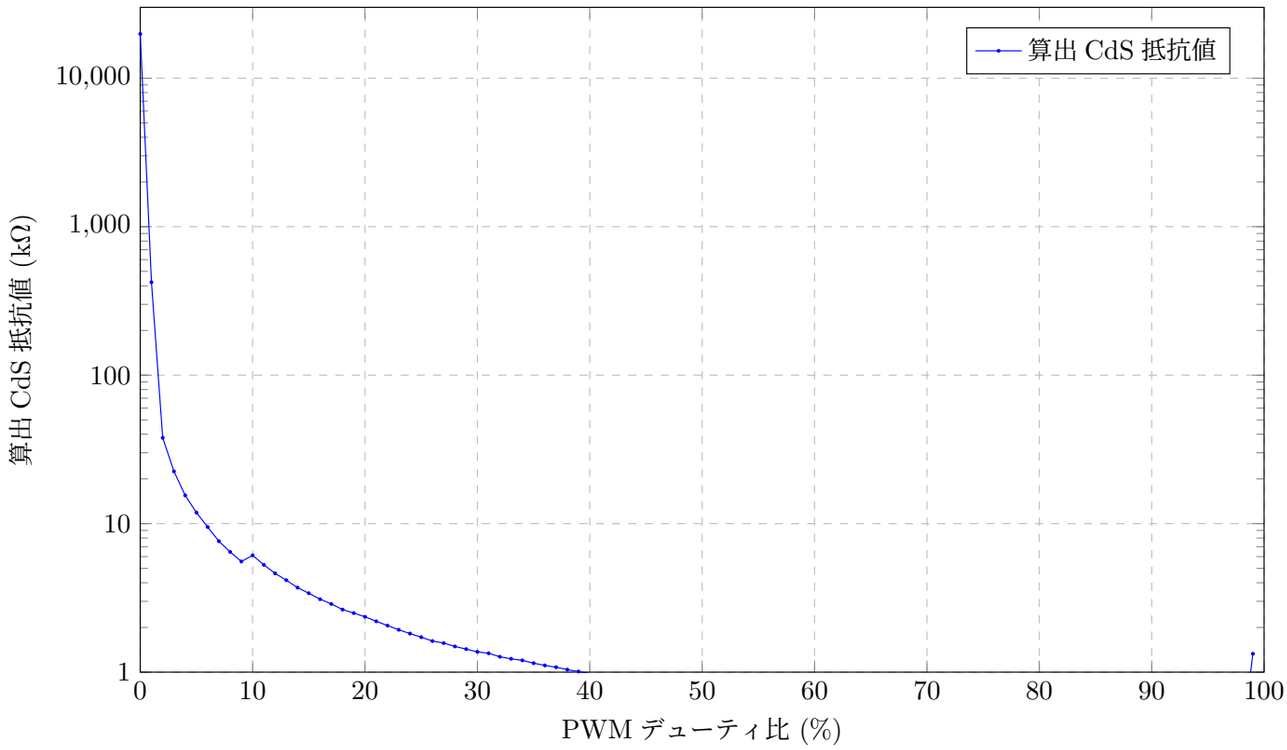

図 1　PWM デューティ比と算出 CdS 抵抗値の関係 (全 100 点プロット)

注: グラフの Y 軸は対数スケール。デューティ比 *0%* の時の抵抗値 *(約 19.87 MΩ)* は他の値と比較して著しく高いため、プロット範囲とスケールを調整している。

6　考察

6.1　実験結果について

図 1 に示すように、PWM デューティ比が増加する (LED が明るくなる) につれて、CdS 素子の算出抵抗値は概ね減少する傾向が見られた。これは、光量が増えると抵抗値が減少するという CdS 素子の一般的な特性と一致する。デューティ比 0% (LED 消灯時) では抵抗値が約 19.87MΩ と非常に高くなり、デューティ比 99% (LED 最も明るい時) では約 0.24kΩ まで低下した。ただし、グラフのいくつかの箇所 (例: デューティ比 9% から 10% にかけて、53%

から 54% にかけて、90% から 91% にかけてなど) で、抵抗値が期待される単調減少からわずかに逸脱する部分が見られる。これは、測定時の微小な外乱光の変化、LED と CdS 素子の位置関係の微妙な変動、電源ノイズ、あるいは CdS 素子自体の応答特性や個体差などが影響した可能性が考えられる。特にデューティ比 99% の ADC 値 (167.74) が 90% の ADC 値 (175.62) よりも低く、結果として抵抗値が 0.243kΩ となり、90% の時の 0.290kΩ よりも低くなっている。これは期待される傾向と一致する。

6.2 CdS 抵抗値の計算式と回路構成について

本レポートで CdS 素子の抵抗値 R_{CDS} の算出に用いた式は $R_{CDS} = R_0 \times \frac{ADC}{1023-ADC}$ (ここで $R_0 = 6.8\text{k}\Omega$) である。この式は、基準電圧 V_{ref} と抵抗 R_0 と CdS 素子 R_{CDS} が直列に接続され、R_0 がプルアップ抵抗として機能し、ADC が R_{CDS} (GND との間) の電圧を測定する回路構成 ($V_{ADC} = V_{ref} \times \frac{R_{CDS}}{R_0+R_{CDS}}$) から導出される。この構成では、 R_{CDS} が低い (明るい) ほど V_{ADC} は低く (ADC 値も低く) なる。実験データでは、デューティ比が高い (明るい) ほど ADC 値が低くなっているため、この計算式と前提回路構成が実験結果と整合する。

一方、提供された回路図 (図 3) では、CdS 素子が 3.3V と V_{CDS} の間に、6.8kΩ 抵抗が V_{CDS} と GND の間に接続されている (プルダウン構成)。この場合、V_{CDS} は 6.8kΩ 抵抗にかかる電圧となり、$V_{CDS} = V_{ref} \times \frac{R_0}{R_{CDS}+R_0}$ となる。ここから導出される式は $R_{CDS} = R_0 \times \frac{1023-ADC}{ADC}$ である。この式を実験データに適用すると、明るいほど抵抗値が高くなるという物理的に逆の結果が得られる。したがって、実際の実験回路は図 3 の CdS 素子部分とは異なる接続 (例: 6.8kΩ がプルアップで CdS 素子が GND に接続、またはその逆で ADC の測定点が異なる) であった可能性が高いと考えられる。本レポートでは、実験データと物理的整合性を優先し、前述の計算式を採用した。

6.3 PWM デューティ比の解釈

Python スクリプトでは duty_func(i) が i * 1000 を返しており、pigpio ライブラリの hardware_PWM 関数のデューティサイクル指定範囲 (0-1,000,000) を考慮すると、スクリプト中の i (0-99) は、実際のハードウェア PWM デューティ比の $i/10$ パーセント (例: i=50 であれば 5% デューティ) に相当する。グラフの横軸はこのスクリプト変数 i をそのまま「デューティ比 (%)」として表示しているが、このスケーリング関係に留意する必要がある。

6.4 PWM の具体的な用途 (調査)

PWM (Pulse Width Modulation: パルス幅変調) は、パルス信号のデューティ比を変化させることで、アナログ的な制御をデジタル回路で実現する技術であり、多様な分野で利用されている。

- **LED の調光**: 本実験のように、LED に流れる平均電流をデューティ比で制御し、明るさを連続的に変化させる。人間の目には高速な点滅が平均的な明るさとして認識される。
- **モータ制御**: DC モータの回転速度制御や、サーボモータの位置制御に用いられる。デューティ比を変えることでモータへの供給電力を調整し、速度を制御する。サーボモータでは、特定のパルス幅が特定の位置に対応する。
- **スイッチング電源 (DC-DC コンバータ)**: 電圧を効率的に昇圧または降圧するために PWM が利用される。スイッチング素子 (MOSFET など) のオン・オフ時間を PWM で制御し、インダクタやキャパシタと組み合わせて所望の出力電圧を得る。
- **オーディオアンプ (クラス D アンプ)**: 音声信号を PWM 信号に変換し、スイッチング素子を駆動する。高効率な電力増幅が可能である。
- **ヒータ制御**: 電気ヒータへの電力供給を PWM で制御し、温度を精密に調整する。
- **通信**: 赤外線リモコンなどのデータ送信にも、パルス列のパターンとして PWM の考え方が応用されることがある。

これらの用途では、PWM の周波数やデューティ比の分解能が制御対象や要求精度に応じて選択される。

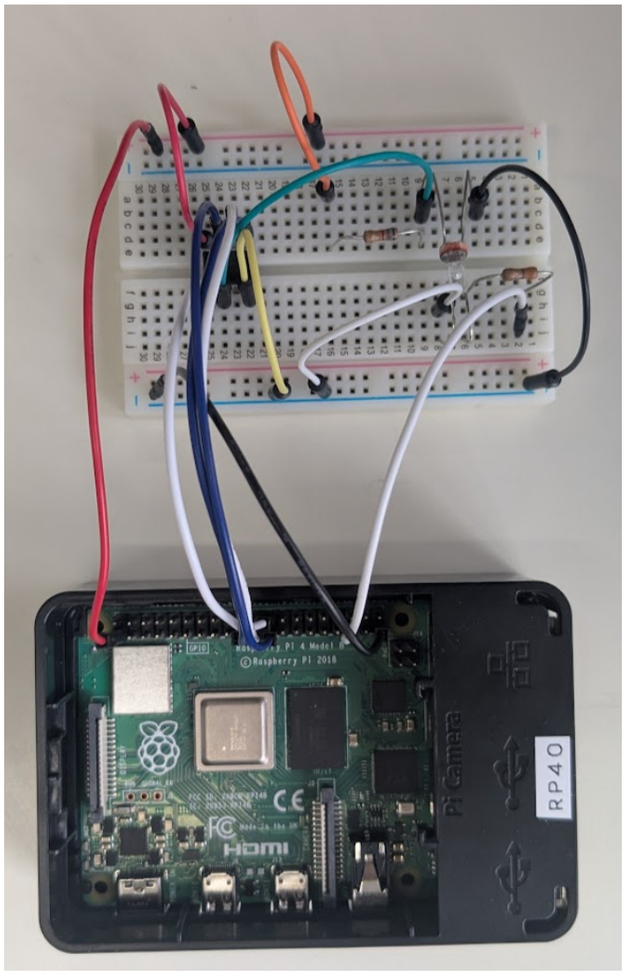

図 2　Raspberry Pi 4B の GPIO ピン配置と接続回路

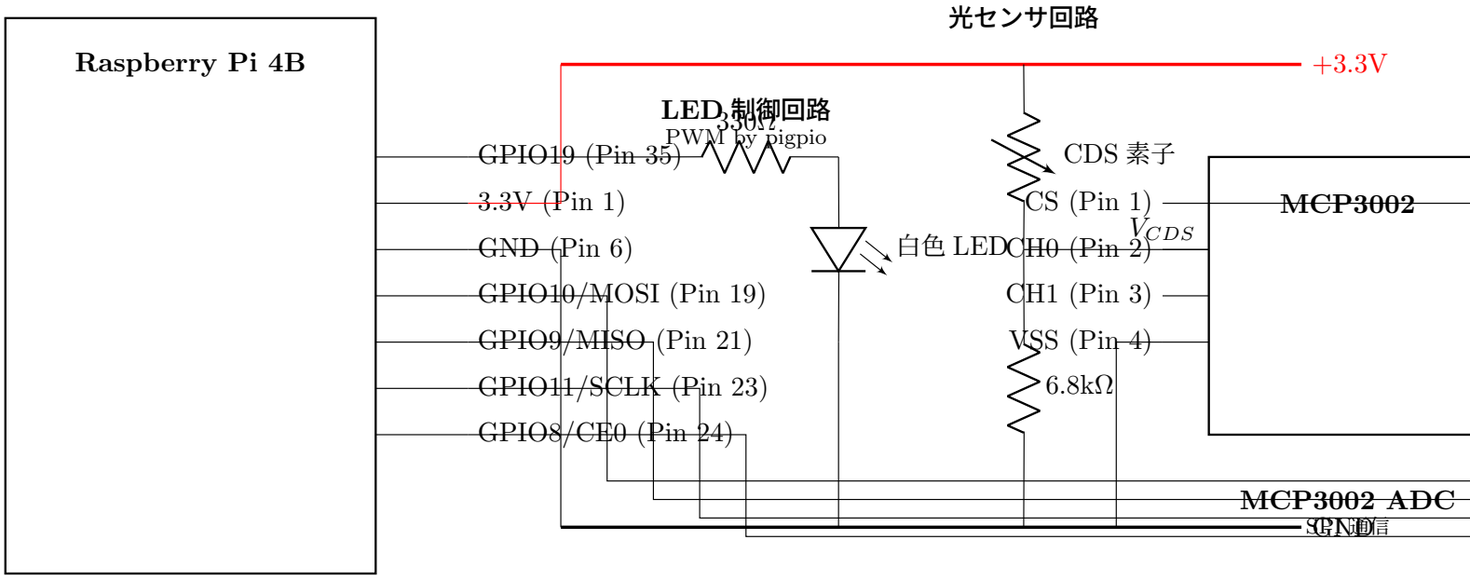

図 3　Raspberry Pi 4B を用いた LED 制御と光センサの AD 変換回路