電気電子工学実験3：白井研究室

ボードコンピュータを用いたロボット実験

ボードコンピュータを用いたロボットを題材に、 マイコンのプログラミングやロボットの制御などの技術を学ぶ

Raspberry Pi を用いたマイコン制御のロボットについて学ぶ。1 週目では Raspberry Pi の機能、LinuxOS と Python プログラム、GPIO の制御について学ぶ。2 週目では、PWM の使い方、DC モーターによるロボットの移動制御と、Wifi による PC と Raspberry Pi の通信について学ぶ、3 週目ではロボットのリモコン操作、ウェブカメラからのビデオ配信と、ロボットの操作との連携について学ぶ。4週目では今までの実験を踏まえてより高性能なロボット作成に挑戦する。

1 週目： Raspberry Pi に慣れよう

(1) Raspberry Pi のハード的構成

キーボード、マウス、wifi、HDMI（モニタ）、電源

(2)ソフト構成： Linux OS (Raspbian)

エディタ(leafpad)、Web ブラウザ(firefox),

sudo コマンド

(3) プログラミング言語 Python

その基本、GPIO の制御

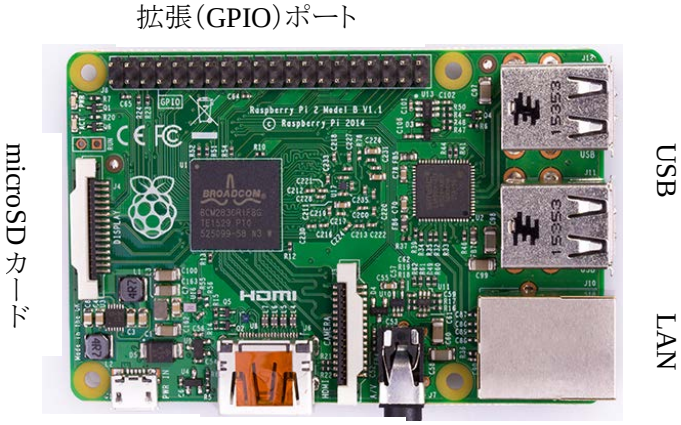

図 1.1. Raspberry Pi 2 B+

実験： Raspberry Pi の環境設定（1時間程度）

大事なポイント： 電源は最後に接続、最初に外す

電源を繋いでいるときは、他の機器の抜き差し禁止 （壊したら弁償すること）

GPIO とは汎用の入出力のこと。Raspberry Pi の GPIO の出力は 3.3V が最大値

注意点: (1) LED は必ず抵抗を直列に接続する、(2) 電源を入れたまま配線作業しない、

(3) IC には必ず電源と GND をつなぐ、(4) GPIO の出力ピン同士を接続しない

- LED チカ （注意: ほぼ同じ内容が RaspberryPiStart.pdf にある。どちらを実行してもよい）

  ハード: ブレッドボード、LED, 抵抗で図 1.2 に示す回路を組む

  ソフト: leafpad を用い、以下のコードを書き、LED.py というファイルにする

```
# LED.py
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
pin = 17
GPIO.setup(pin, GPIO.OUT)
for i in range(5):
   GPIO.output(pin,GPIO.HIGH)
   time.sleep(0.5)
   GPIO.output(pin, GPIO.LOW)
   time.sleep(0.5)

GPIO.cleanup()
```

プログラムリスト 1

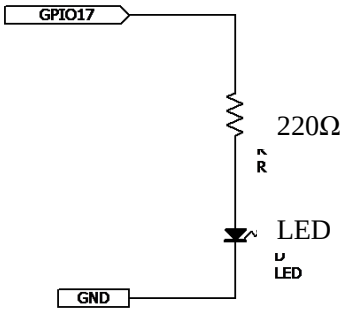

図 1.2. LED 回路

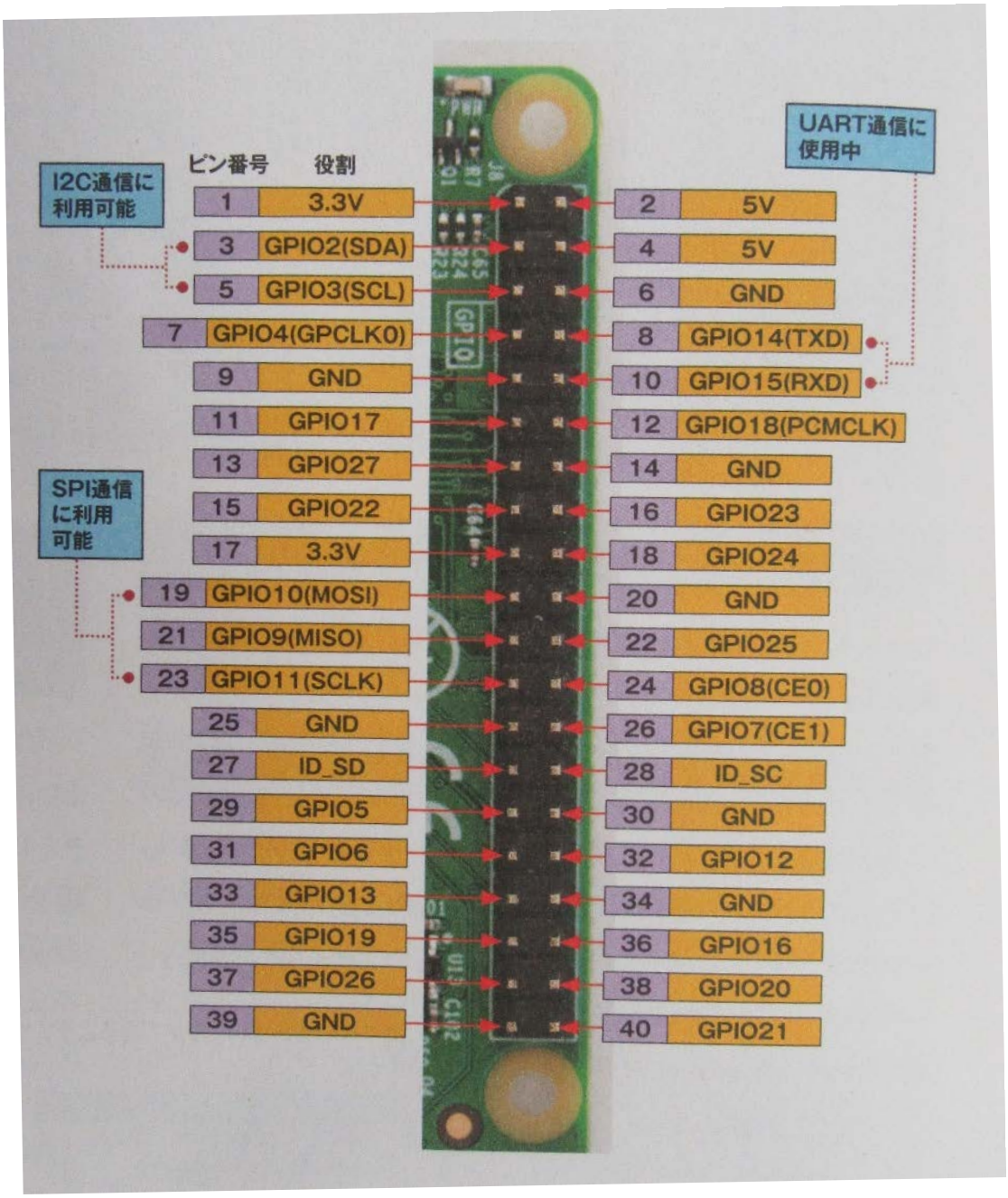

図 1.3.　Raspberry Pi の拡張ポート(GPIO とピン番号)　日経ソフトウェア　2015 年 7 月号付録　p.8　より借用

プログラムリスト1で(ピン番号) 17 とあるのは、図 1.3　の『ピン番号 11』のことである(混乱しやすいので注意しよう)。GPIO のピンには、ボード上の通し番号(BOARD で指定)と GPIO 番号（BCM で指定）の二通りの番号がついている。見て分かるように、ピン番号 2 と 4 は常時 5V が出力され、1 と 17 は 3.3V が出力されている。また、6,9,14,20,25,30,34,39 番は GND（グラウンド、アース）である。図 1.2 の回路を Raspberry Pi に組み込むには、それぞれピン番号 11 を抵抗に、GND は上記の GND のいずれかと接続する。

これだけでは LED は光らない。LED を光らせるにはプログラムによる GPIO ピンの制御(ON/OFF の切り替え)が必要である。それがプログラムリスト 1 である。プログラムリスト1について簡単に説明する。

1 行目は「コメント」であり、行の先頭にある # がそのマークである。C プログラミング言語における // と同じ機能

をするが、C プログラミング言語とは異なり // は Python では「割り算」を意味するので注意する。
2 行目と 3 行目はいずれも import から始まっているが、これは「ライブラリ」を使えるようにするためのコマンドである。C プログラミング言語では #include に相当する(と思って欲しい)。3 行目の `import time` は時間(time)に関する関数のパッケージを使えるようにするもので、下から 3 行目の `time.sleep(0.5)` は `time` パッケージで定義された関数 `sleep` の実行である。これにより 0.5 秒間時計が止まる。また 2 行目の

```
import RPi.GPIO as GPIO
```

は RPi モジュールにある GPIO 制御のためのパッケージを取り込め、という命令である。そしてそのパッケージで定義されている変数や関数はみな GPIO から始まる単語を使え、ということを表している。使用例が 4 行目にある
`GPIO.setmode(GPIO.BCM)`
である。setmode は GPIO パッケージで定義された関数であるため、 GPIO.setmode (GPI と setmode の間に「.」があるのに注意)として使用される。また、GPIO.BCM は GPIO パッケージで定義された変数(定数)である。ちなみにこの命令によって、GPIO のピンは「ピン番号」ではなく「役割番号」で参照されることが決まる。

5 行目の `pin = 17` は、変数 `pin` に `17` という数を代入するものである。`Python` では変数の型宣言というものは存在しない。必要な変数は型宣言なしにいつでもどこでも導入して良い(関数も同じで、「関数」であることの宣言は必要だが、どのような値を返すかという宣言は存在しない)。
6 行目の `GPIO.setup(pin, GPIO.OUT)` は、`17` 番(ピン番号ではなく「役割番号」であることに注意)のピンを「出力」に使用するための設定である。`GPIO` は `GND` や電源のような特殊なピンを除き、入力にも出力にも使える。ただし、そのための宣言が必要である。

その次からは `Python` の繰り返しの制御構造の例である(最後の「:」に注意)。

```
for i in range(5):
```

は繰り返しの回数をカウントする変数として `i` を用いること、また、その値の範囲`(range)`が5であることを表している。実際には、`range(5)`は `[0,1,2,3,4]` というリスト(配列)を返し、`i` の初期値に `0` がセットされ、繰り返しが終わるたびに `1, 2, 3, 4` という値を取る。それが終わったら次の命令(ここでは `GPIO.cleanup()`)が処理される。

繰り返される内容が、`GPIO.output(pin,GPIO.HIGH)` から始まる `4` 行の命令である。`C` とは異なり、繰り返される内容は波括弧`{ }` ではくくられない。`Python` ではインデントを揃える(必ずスペースを使うこと、`TAB` は使えない)ことでブロック(ここでは繰り返しの命令の集まり)を表現する。ここでインデントがそろっていないと(なんと!)エラーになる。(このように `Python` はスタイル(形)にこだわる-美しく書かせられる-プログラミング言語である)。ここで繰り返される内容は

```
GPIO.output(pin,GPIO.HIGH)      ―― pin の出力を HIGH(3.3V)にする
time.sleep(0.5)                 ―― 0.5 秒間そのまま保持
GPIO.output(pin,GPIO.LOW)       ―― pin の出力を LOW(0V)にする
time.sleep(0.5)                 ―― 0.5 秒間そのまま保持
```

である。つまり `LED` の回路が正しく接続されていれば、`0.5` 秒間点いて `0.5` 秒間消える、ということが `5` 回繰り返される。
最後の `GPIO.cleanup()` は `GPIO` の設定の解除命令であり、`GPIO` についてのプログラムを実行した場合は、これを必ず実行しておくこと。なお、この前の「空行」は(スペースをいれてはいけない!)、ブロックの終わり(繰り返しの範囲の指定)を意味する。

課題. 実験(1)の手順に従い、回路を組んでから Raspberry pi を起動する。

Terminal を起動し leafpad を呼出し、LED.py ファイルをつくる。

そして(LED.py があるディレクトリを作業用ディレクトリにして、)

sudo python LED.py

を実行し、LED が点滅するのを確認せよ。ここで python が Python プログラムの実行コマンドである。また GPIO を制御するには root (管理者)権限が必要なので、一時的に管理者権限を与えるコマンドである。sudo と python の順番を間違えないようにしよう。

次に、2 個の LED を用い、同じプログラムを用いて 2 つの LED が交互に点滅を繰り返すようにせよ。

プログラムで点滅の間隔を制御してみよ。

PWM (Pulse Width Modulation)とは何か、調べよ。またこれを用いて LED の明るさを制御せよ(参考:プログラムリスト 2)

```
import time
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
pin = 17      # using GPIO 17
freq = 50     # frequency = 50Hz
dc = 0.0      # duty cycle
GPIO.setup(pin, GPIO.OUT)
p = GPIO.PWM(pin, freq)
p.start(dc)

try:
   while True:
      for dc in range(0, 101, 20):
         p.ChangeDutyCycle(dc)
         time.sleep(0.1)
      time.sleep(1)
      for dc in range(100, -1, -20):
         p.ChangeDutyCycle(dc)
         time.sleep(0.1)
except KeyboardInterrupt:
   pass

p.stop()
GPIO.cleanup()
```

プログラムリスト 2