第1章 Pythonだけでもここまで ハードウェアを制御できる

ダウンロード・データあります

お勧め1…pigpio

漆谷 正義

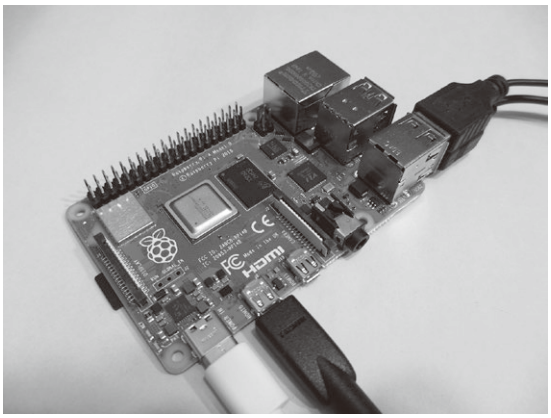

写真1 ラズベリー・パイのI/O端子をPythonから直接操作できるのがpigpio

● GPIOの実力研究&使いこなし

ラズベリー・パイには，PCなどのコンピュータにはないGPIO（汎用入出力）と呼ばれる端子が付いています．マイコンのようにGPIOが付いたコンピュータ・ボードとも言えます（**写真1**）．

LinuxベースのRaspberry Pi OSを通して，このGPIOを操作できるということが，IoT（ネットを通じたハードウェア制御）分野で大きな力を発揮します．複雑なネットワーク通信やファイル操作との連携は，OSなしでは至難の業です．

GPIOはハードウェアに接続する窓口です．センサ信号を受けたり，モータを制御したりします．ハードウェアに接続するため，この端子には使える電圧，電流，周波数，波形に加え，ノイズやタイミングなどの制約や影響を考慮する必要があります．このため，GPIOを使いこなすには，ある程度の電子回路の知識が必要となります．

ここでは，ラズベリー・パイ3 Model B＋および4 Model Bを例に，GPIOポートの特徴と使い方を解説します．さらに，ライブラリが豊富で，使いやすい言語ツールであるPythonを使って，GPIOを通じていろいろなハードウェアを制御するノウハウを紹介します．

標準で入っているGPIOライブラリ

● 選んだ理由

本稿ではGPIOライブラリとしてpigpioを選びました．PWMの性能（最高周波数，分解能，安定性）が良いこと，入力割り込みに対応していることなど，ハードウェア寄りに特化していることで選びました．

pigpioは，最新のRaspberry Pi OSには標準で含まれているので，インストール作業は不要です．

● ちょっとした使い方のコツ

唯一面倒な点ですが，pigpioは使用する前にデーモンをルート権限で走らせておく必要があります．

```
$ sudo pigpiod↵
```

しかし，このおかげでカーネル・ドライバを介さずにCPUレジスタ（BCM）を直接操作でき，GPIOの動作が格段に高速になります．そして次のように，コマンドラインからポートの操作もできます．

```
$ pigs w 18 1    ← GPIO18を"H"にする
$ pigs s 18 500  ← GPIO18に500μs幅のパルスを出力
$ pigs p 18 128  ← GPIO18にデューティ比50%のPWM出力
```

pigpioは，コールバック関数とポーリング用のスレッド採用などにより，サンプル・レートは最小1μs，標準5μsと高速になっています．

忘れがちなのがポート操作の後処理です．例えば，

```
pi.write(18,1)
```

としてプログラムを終了すると，GPIO18はいつまでも"H"のままになります．

```
pi.set_noise_filter(18, 1000, 5000)
```

とするとプログラムが終了した後，ノイズ・フィルタが設定されたままになりますし，

```
pi.set_pull_up_down(18, pigpio.PUD_UP)
```

とすると，プルアップ抵抗が入ったままになります．

```
pi.write(18,0)
pi.set_noise_filter(18,0,0)
```