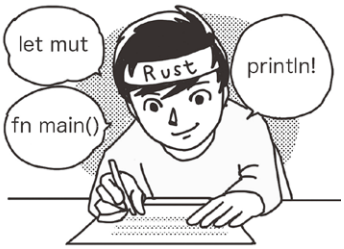
ダウンロード・データあります

Rustプログラミング問題集

ご購入はこちら



第3回　ラズベリー・パイのI/Oにアクセスする

中村 仁昭，懸川 岳

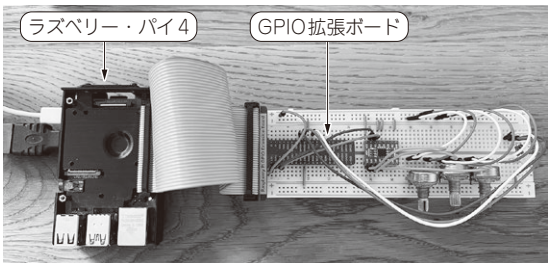


写真1　今回の実験の様子

本連載ではRustを使った問題と回答形式でRustを学んでいきます．

今回はラズベリー・パイのI/Oにアクセスしてみます．C言語で組み込み向けプログラムを作ると，範囲外のメモリ・アクセスや，システム・コールの引数間違いなどのプログラムが，エラーもなしにコンパイルでき，実行できてしまいます．Rustを使えば，そのような危険性をコンパイル時に検出することができます．（編集部）

● 実行環境

ラズパイ上でrustupでインストールした標準的な環境で動作確認しています．rustupでのインストール手順は`https://www.rust-lang.org/ja/tools/install`を参照してください．

今回の実験にはラズベリー・パイ4を使い，OSはRaspberry Pi OS 32-bit（raspios_armhf-2022-09-26）を使用しています．

問題1～問題6はラズベリー・パイのハードウェアにアクセスするため，rppal（Raspberry Pi Peripheral Access Library）クレートを使います．rppalクレートを使うためには，`Cargo.toml`に下記のような`dependencies`を記載する必要があります．

```
[dependencies]
rppal = "0.14.1"
```

問題7，問題8はrppalを使わず，GPIO character device ABI経由でアクセスするためにgpio-cdevクレートを使います．そのためには`Cargo.toml`に下記のような`dependencies`を記載する必要があります．

```
[dependencies]
gpio-cdev = "0.5.1"
```

今回の実験の様子を**写真1**に示します．ラズベリー・パイでブレッドボードを使って実験をするときは，**写真1**のようなGPIO拡張ボードを使うと便利です．

1　基本のLチカ

GPIO 23（16ピン）に接続したLEDを1Hzで点滅させてください．

● 回答

回答を**リスト1**に，このときの回路図を**図1**に示します．

Rustにはembedded-halというハードウェア抽象化クレートがあります．embedded-halは各種の機能のトレイトのみを提供しているので，各ボード/マイコンごとのHALを使用する必要があります．rppalはラズパイ向けのハードウェア・アクセス・ライブラリと，embedded-halのトレイトを実装したHALになります．ここでは単純に500msごとにピンの0/1を反転させています．

リスト1　GPIO 23に接続したLEDを点滅させる

```
use std::error::Error;
use std::thread;
use std::time::Duration;

use rppal::gpio::Gpio;

const GPIO_LED: u8 = 23;

fn main() -> Result<(), Box<dyn Error>> {
    let mut pin = Gpio::new()?.get(GPIO_LED)?
                                        .into_output();

    loop {
        pin.toggle();
        thread::sleep(Duration::from_millis(500));
    }
}
```