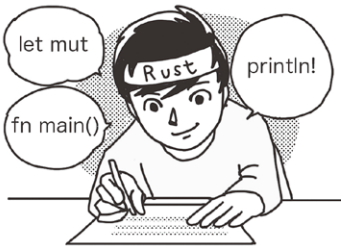
ダウンロード・データあります

Rustプログラミング問題集

ご購入はこちら



第4回　ESP32を使ったハードウェア制御

懸川 岳，中村 仁昭

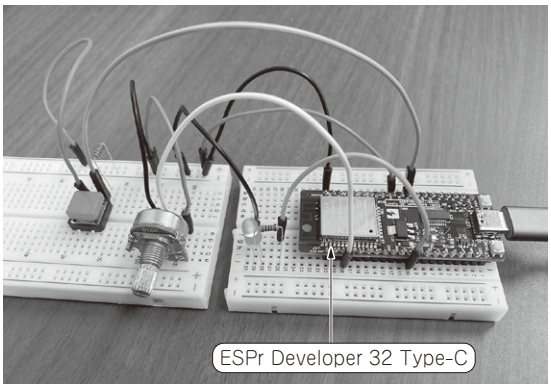


写真1　今回の実験の様子

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．

Rustは安全性が高いことから，組み込み向け用途でハードウェアをアクセスするプログラムの作成に応用することが期待されています．そこで今回はRustでESP32のハードウェアにアクセスしてみます．（編集部）

● 実行環境

今回の実験の様子を**写真1**に示します．ESPr Developer 32 Type-Cを使用し，ビルド環境としてはUbuntu 20.04のPCを使用しています．ビルド環境構築手順については下記URLを参照してください．

```
https://esp-rs.github.io/book/introduction.html
```

ESP32のハードウェアにアクセスするためにesp_idf_halクレートを使用します．そのためには`Cargo.toml`に以下のような記載が必要です．

```
[dependencies]
esp-idf-hal = "0.41"
```

以下のコマンドでUSB接続したESP32にファームウェアを書き込むことができます．

```
$ cargo espflash flash
```

また，次のコマンドではファームウェアを書き込み，シリアル・モニタを開始することができます．

```
$ cargo espflash flash --monitor
```

1　ESP32で基本のLチカ

GPIO 27に接続したLEDを1Hzで点滅させてください．

● 回答

回答例を**リスト1**に示します．また，このときの回路図を**図1**に示します．

GPIOへのアクセスは`PinDriver`構造体を使います．`toggle()`メソッドを使用して，500msごとに5VとGNDを切り替えることによって1HzでLEDを点滅させています．

リスト1　GPIO 27に接続したLEDを点滅させる

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

use esp_idf_hal::peripherals::Peripherals;
use esp_idf_hal::gpio::PinDriver;

fn main() -> Result<(), Box<dyn Error>> {
    let peripherals = Peripherals::take().unwrap();
    let mut led = PinDriver::output(
                        peripherals.pins.gpio27)?;
                        5VとGNDを切り替える
    loop {
        let _ = led.toggle();
        thread::sleep(Duration::from_millis(500));
    }
}
```

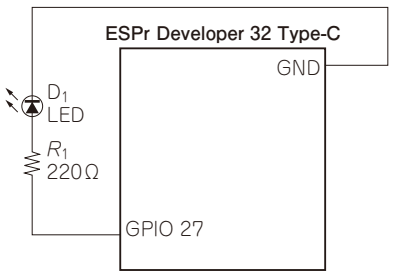


図1　GPIO 27にLEDを接続する

ダウンロード・データはこちら