特集2

第10章 組み込みマイコンでもGUIでデバッガが使える

VSCode＋OpenOCD＋GDBでデバッグ

中林 智之

組み込み開発する上で重要なのが，デバッガでデバッグできるのかという点です．新しくデバイスを買ったらデバッガがつながるかどうかから始まります．新しいプログラミング言語を使うときでも，当然気になるポイントです．

RustはC言語のコードと同じようにOpenOCD+GDBでデバッグできます．

Rustではデバッガを使う機会は比較的少ないです．言語仕様により未定義動作が発生しにくいためです．

しかし，組み込み開発でRustを使っていると，ペリフェラル・レジスタへの設定が意図通りできているかどうかを中心に，どうしてもデバッガを使いたいことがあります．

ESP32シリーズのstd環境でも便利にデバッグできる仕組みが整っています．ここではVSCodeを使ったデバッグ方法を紹介します．

M5Stamp C3U Mateのデバッグ環境

ESP32-C3にはUSB-シリアル変換とJTAGコントローラが内蔵されています．今回使っているM5Stamp C3U MateはUSB Type-CポートがESP32-C3のUSB-シリアル変換とJTAGコントローラに接続されているので，USBケーブル1本でデバッグできます．そのためハードウェアとしての準備は全く必要ありません．

M5Stamp C3U MateをUSB Type-Cケーブルで接続するだけで完了です．名前が似ている製品のM5Stamp C3 Mate（C3の後にUがない）は，USB Type-Cコネクタが UARTに接続されているため，今回紹介するデバッグ方法を使えません．お気を付けください．

Linuxではパーミッションを設定するために次のudevルールを/etc/udev/rules.d/99-esp32c3.rulesとして作成しておきます．

```
SUBSYSTEMS=="usb", ATTRS{idVendor}=="303a", GROUP="plugdev", MODE="0666"
```

udevルールを追加した後は，ルールをリロードします．

```
$ sudo udevadm control --reload
```

OpenOCDとGDBの動作確認

デバッグに必要なOpenOCDとGDBはESP-IDFのインストール時にダウンロードされています．Linuxであれば，通常，ホーム・ディレクトリの.espressif/tool下です．

● OpenOCDの起動

M5Stamp C3U MateをUSBケーブルでホストPCに接続している状態で次のコマンドを実行すると，OpenOCDが起動します．OpenOCDのバージョンや細かいパスはインストールするESP-IDFによって多少異なる可能性があります．

```
$ ~/.espressif/tools/openocd-esp32/v0.11.0-esp32-20211220/openocd-esp32/bin/openocd -f ~/.espressif/tools/openocd-esp32/v0.11.0-esp32-20211220/openocd-esp32/share/openocd/scripts/board/esp32c3-builtin.cfg
```

図1のようなログが出力され，TCPポート3333でGDBの接続待ちになればOKです．

● GDBの接続

GDBからOpenOCDに接続します．ESP-IDFをアクティベートしている状態であれば，RISC-V系のツールチェインもパスが通っているため，今回使っているビルド済みのサンプル・プロジェクトに移動して，次のコマンドを実行します．

```
$ riscv32-esp-elf-gdb target/riscv32imc-esp-espidf/debug/c3-rust-template
```

GDBのコンソールが起動するため，**図2**のコマンドを順に実行します．

うまく動けば**図3**のようにesp_idf_sys::start::app_mainのブレークポイントでプログラムが停