特集2

第6章 ①ログ・システム，②エラー・ハンドリング，③メモリ・アロケータ，④スレッド，⑤データ共有

ステップ4…便利なライブラリを試してみる

中林 智之

組み込み向けマイコンでPC並の高度な機能が使えるstdとその他の便利クレート（ライブラリ）をESP32-C3上で試します．どの機能もPCでは当然のように動かせるものですが，リソースの厳しい組み込みマイコンの環境では動ない高級品でした．それが今，ESP32-C3で動くようになりました．

ここではこの後のサンプル・プログラムで使うものを中心に少しESP32-C3上で動かしてみます．

ビルド用のソースコードはGitHubのplay-stdディレクトリ下にあります．また，本誌に載せきれなかった**リストA**～**リストP**，**図A**～**図J**は本誌ウェブ・ページに掲載しています．

```
https://interface.cqpub.co.jp/2305rust2/
```

〈筆者のGitHub〉

```
https://github.com/tomoyuki-nakabayashi/interface202305-c3-std-rust
```

①ログ・システムEspLogger

● イベントをログに残すために使える

ログ・システムはシステムが稼働中に発生したイベントを記録するためのもので，いつ，何が発生したのかを調査するときに使います．ESP-IDFには標準のログ・システムがあり，次の機能を備えています．

- ログ・レベルによるフィルタ処理や文字装飾（色）の変更
- システム起動からのタイム・スタンプの表示
- ログを出力しているモジュール（タグ）の付与

info!レベルのログ例（緑色で装飾される）

```
I (297) cpu_start: Starting scheduler.
```

error!レベルのログ例（赤色で装飾される）

```
E (10302) task_wdt: Task watchdog got triggered. The following tasks did not reset the watchdog in time:
```

表1 ログ・レベル

レベル	内 容
error!	エラー．非常に深刻なエラー
warn!	警告．危険な状態
info!	情報．有用な情報
debug!	デバッグ．優先度の低い情報
trace!	トレース．優先順位が非常に低く，多くの場合，非常に冗長な情報

一方で，単純に標準出力に文字列を出力しただけではコンソールの標準色になります．

```
Hello, world!
```

esp-idf-svcにあるEspLoggerを使えば，このESP-IDF標準のログ・システムと同じログ出力が可能です．しかもRustで標準的なログ・システムのlog[1]と互換性のある方法です．

● ログ・システムAPIを定義するlogクレート

logクレートはログ・システムのAPIを定義するクレートです．**表1**のようにログ・レベルが高いものから順に，error!，warn!，info!，debug!，trace!というマクロが定義されています．これらのマクロを，ログ・システムの実装を切り替えながら使うことができます．

ログ・システムの実装としてよく使われているのはenv_logger[2]で，ホストPCの環境変数によってログ・レベルを切り替えることができる実装です．RUST_LOG環境変数を，例えば，INFOに設定するとinfo!レベル以上のログが出力される，という具合です．

```
$ RUST_LOG=INFO cargo run↵
```

他にもさまざまなログ・システムの実装があるのですが，興味があれば，クレート登録サイトcrates.io[3]を探してみてください．