開発環境＆処理系百科

宮田 賢一

ここではESP32で使えるプログラミング言語とその開発環境を紹介します．

マイコン向けのプログラミング言語や開発環境の特徴は，限られたCPUリソースでも動かせる実行ファイルを生成できることや，マイコン用の周辺デバイスを使うためのライブラリが用意されていることです．**表1**と**表2**に，ESP32に対応しているプログラミング言語や開発環境を挙げました．スタンダードなArduinoやC言語だけではなく，PythonやJavaScript，Ruby，Luaなど，メジャーなプログラミング言語は網羅されています．対応言語の多さからもESP32がマイコン開発者に注目されていることが分かります．

プログラム実行形態

個々のプログラミング言語を説明する前に，マイコン上でプログラムがどのように実行されるか，その実行形態について説明しておきます．

表1に示した通り，マイコン向けプログラムの実行形態として以下の3通りに分類しました．

①単独実行型
②他言語埋め込み型
③対話型

● その1：単独実行型

そのプログラミング言語だけで全ての処理を記述できるタイプです．ビルドしてできた実行ファイルはESP32に転送することで，ESP32のリセットのたびに自動的に実行されます［**図1**(**a**)］．C言語が代表例です．

● その2：他言語埋め込み型

埋め込み型プログラミング言語で記述したプログラム・コードを，他のプログラミング言語のソースコード中へ埋め込んでコンパイルします［**図1**(**b**)］．

メインの処理はC言語などで記述し，埋め込み言語のプログラム・コードはC言語の文字列として扱います．それを専用のAPIに渡すことで実行するタイプです．少し分かりにくいので例を示します．プログラム（**リスト1**）はJavaScriptの一種であるduktapeの利用例です．`main`関数はCで記述し，“1+2”というJavaScriptのプログラムを文字列としてduktape専用のAPI（`duk_eval_string`）に渡すと，`main`関数実行時にこのJavaScriptプログラムが実行されます．

この方式のメリットは，C言語のビルド環境さえあればJavaScriptなどの他の言語のプログラムを実行できることにあります．

● その3：対話型

REPL（Read-Eval-Print Loop）やインタープリタとも言います．あらかじめ対象言語の実行環境だけESP32に転送しておき，プログラムの実行はシリアル・ターミナルなどを使って対話的に入力するタイプです［**図1**(**c**)］．命令に対するデバイスの挙動を少しずつ試しながらプロトタイプを作り上げるような場合に便利です．MicroPythonが代表例です．

対話的に実行する以外にも，PCで作ったMicroPythonのソースコードをESP32のMicroPython実行環境上へ転送し，ESP32のリセット後，自動的に実行することもできます．

3種類の方法それぞれに特徴がありますので，実際に触って試してみてください．

言語処理系

ESP32で使えるプログラミング言語を少し深堀して，それぞれの言語の以下の点について解説します．

● ボードによって対応状況が変わってくる

ESP32という同じモジュールを搭載していても，ボードによって，ESP32モジュールとボード上の周辺回路との接続方法は千差万別です．例えばLEDを搭載したボードはたくさんありますが，LEDが接続されるGPIOはボードによって異なります．

ここで紹介する言語はどのESP32ボードでも動作しますが，一部の開発環境ではボードによる違いをな