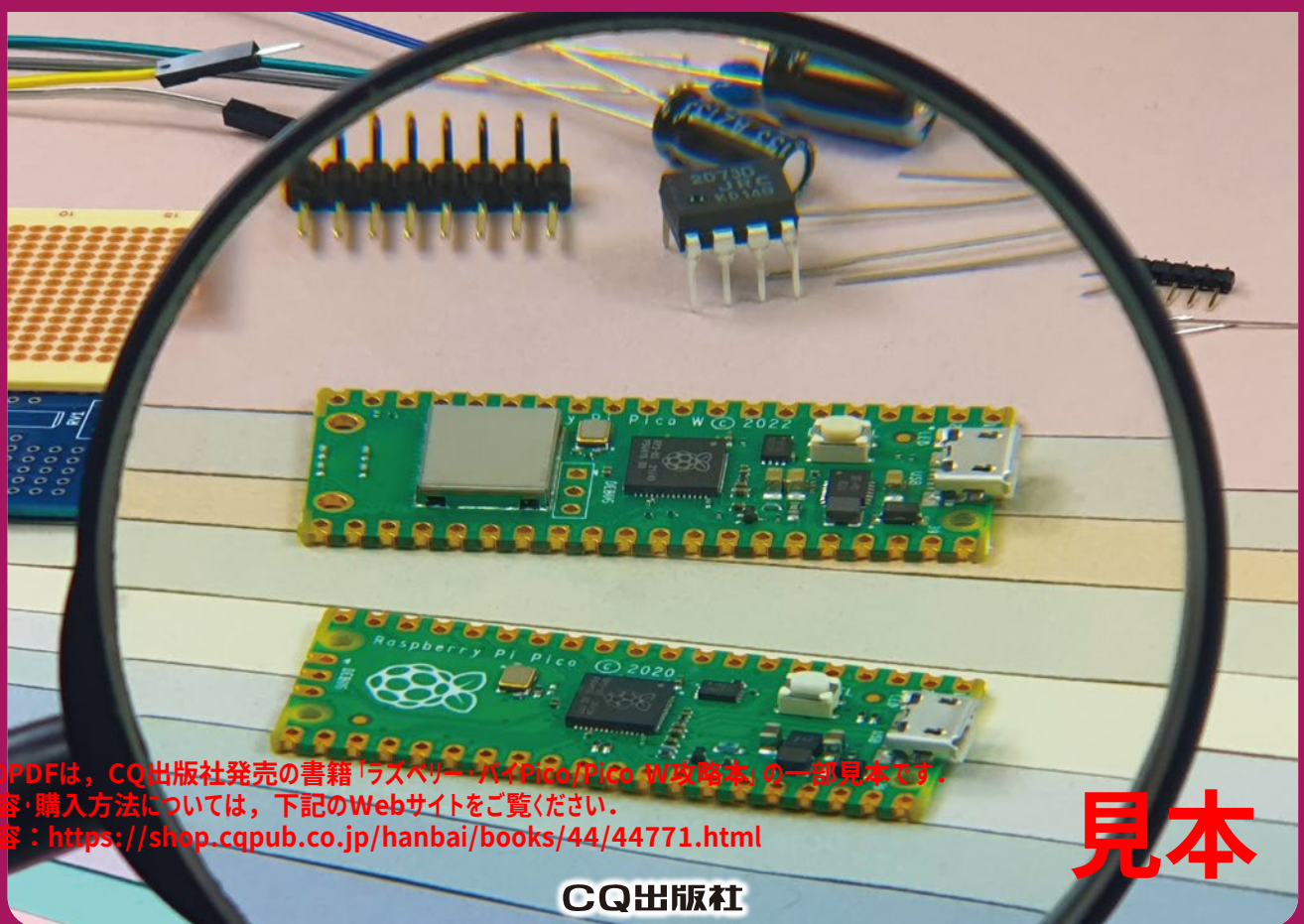
ボード・コンピュータ・シリーズ

MICRO CONTROLLER
マイコン

開発環境/PIO/USB/OS/人工知能/Wi-Fi
オーディオ/MicroPython/C/C++

ラズベリー・パイ Pico/Pico W 攻略本

Interface編集部 編



CQ出版社

第1章

51×21mmの小さなボディにデュアル・コアCPUと高機能I/Oコントローラを凝縮

これがラズベリー・パイ Picoだ

中森 章

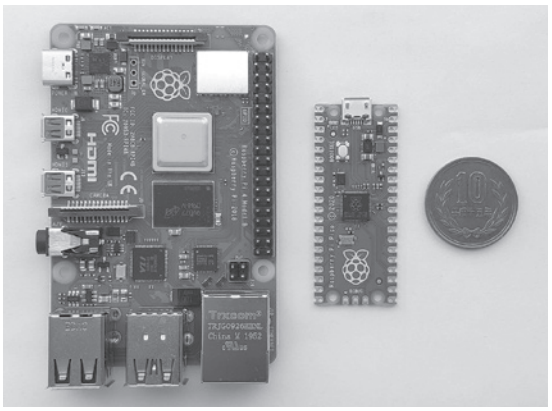


写真1　ラズパイ4とラズパイPicoの比較．クレジットカード・サイズのラズパイ4と比べて，ラズパイPicoは圧倒的に小さい

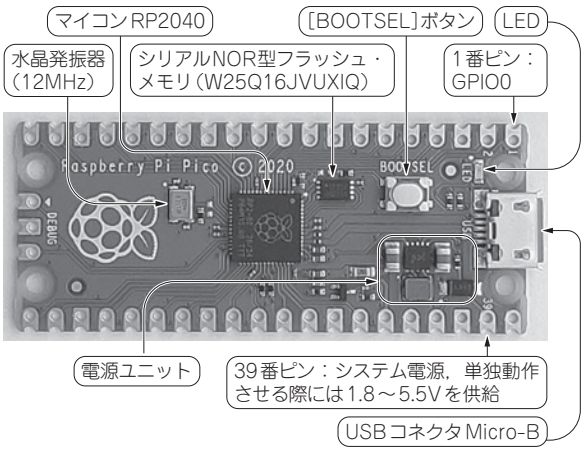


写真2　小さなボディにすごい機能が凝縮されている

● ファースト・インプレッション

ラズベリー・パイPico（以降，Pico）が発売されたという情報はTwitterで知りました．ラズベリーパイ財団がLinuxで動くSBC（シングル・ボード・コンピュータ）ではなく，ベアメタル（OSなし）を推奨しているようなマイコン・ボードを開発したことに多少の違和感を覚えました．恐らく，ラズベリー・パイのコプロセッサとして，I/O処理の負荷軽減（オフロード）をする位置付けだと思いました．搭載しているマイコン（RP2040）までもラズベリーパイ財団が開発したということに驚きです．

なぜ，わざわざ新規にマイコンを作る必要があったのでしょうか．しかも，Arm Cortex-M0+のデュアル・コアで，8個のステート・マシン（これらは，単純なCPUコアと言っても過言ではない）で動作する2系統のプログラマブルI/O（PIO）の搭載は豪華すぎます．

どのような経緯で，開発されたものか分かりませんが，試しに動かしてみる気持ちになるには十分すぎるリッチな仕様です．

そして，実際に実物を見たときの感想は「小さい…」でした（**写真1**）．あんなにすごい機能がこの小さなボディに凝縮されているのかと思うと，使いこなしてみたいと感じざるを得ません（**写真2**）．

● マイコンRP2040の良さをフルに引き出した

Picoは51×21mmの基板に，心臓部であるマイコンRP2040，2Mバイトのフラッシュ・メモリ，電源供給とデータ通信用のUSB Micro-Bポート，30本のGPIO端子（3基のA-Dコンバータを含む，4本は基板内で使用）[注1]，3ピンのSWD（デバッグ）ポートを搭載したものです．GPIOのヘッダ（というかスルー・ホールが空いているだけ）の端子ピッチは2.54mmで，標準的な間隔です．基本的に，Picoの機能はRP2040の機能そのものと言うことができるかもしれません．

ということは，PicoはRP2040を使いやすいように，DIP（Dual In-line Package）形式の配置にGPIOを並べ直した「部品」とみなすことができます．小型ですし，USBケーブルを挿入することで即起動できるので，非常に使いやすい部品となっています．

その使用目的は，明らかに，I/O処理の負荷軽減です．いわば，PICマイコンの高級版です．個人的には，2個のCPUコアやプログラマブルI/OはI/O制御の負荷軽減を行うために，どうしても必要な最小限の機能だったのではないかと推測します．

注1：RP2040には4基のA-Dコンバータを搭載していますが，Picoでは3基のA-Dコンバータのみ使用できます．

◆参考・引用*文献◆
(1) Raspberry Pi Pico Datasheet. https://datasheets.raspberrypi.org/pico/pico-datasheet.pdf
(2) RP2040 Datasheet. https://datasheets.raspberrypi.org/

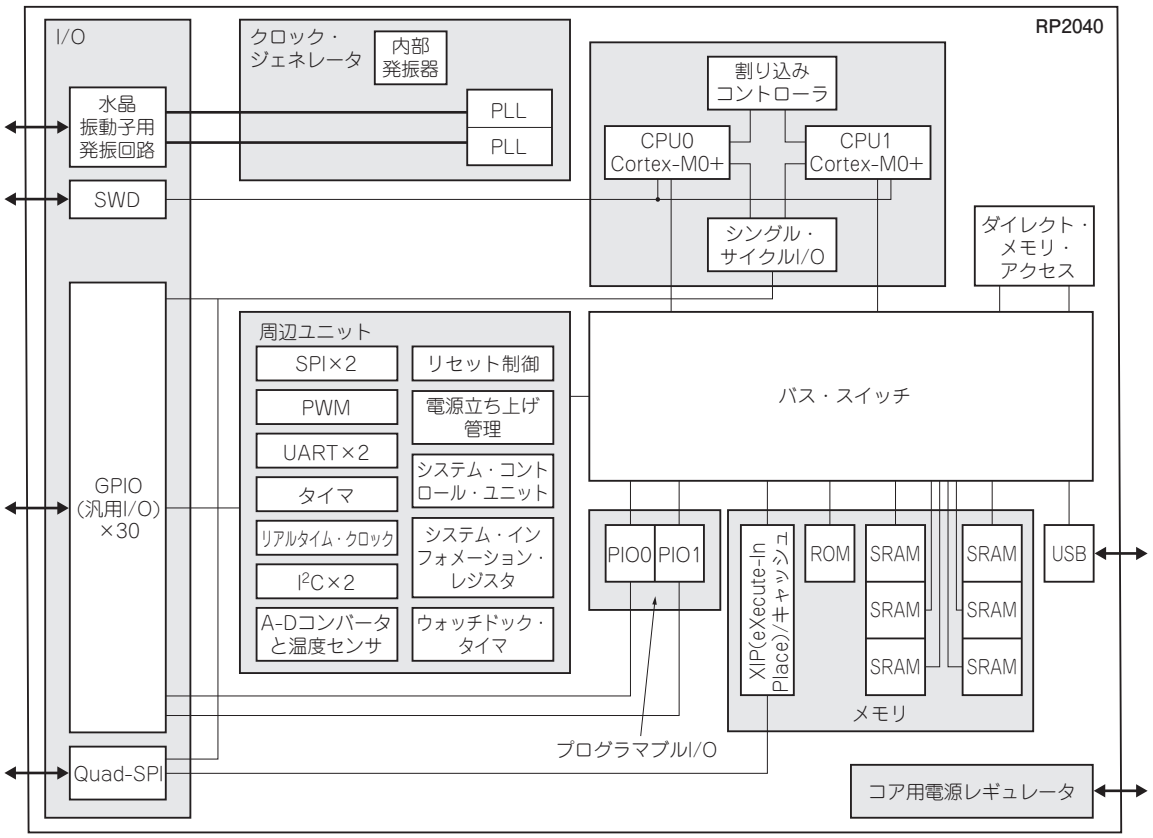


図1[2]　**マイコン RP2040 のブロック図**（2個の CPU コア（Arm Cortex-M0+）と2基の PIO が際立っている）

● RP2040の中身

図1にRP2040のブロック図を示します[1]．ブロック図だけを見ると，普通のデュアル・コア・マイコンのブロック図です．

UART，I²C，SPIといった周辺ユニットも通常のマイコンでよく見かけるものです．これらについて簡単に説明します．

- **SPI**：クロック，入力（受信）信号，出力（送信）信号の3線を基本とし，クロック同期で，データをやり取りします．
- **PWM**：実体は1本の出力信号です．カウンタを内蔵していて，カウンタの1周期の間で，出力する信号の“H”と“L”の駆動時間の割合を自由に指定できます．これにより，ディジタル出力でありながら，あたかもアナログ出力のような挙動を実現します．
- **UART**：上述のSPIによく似ていますが，UARTにはクロック線は存在せず，送信信号と受信信号の2本の信号線で，マイコン外部のデバイスと通信を行います．
- **Timer**：文字通り時間（タイム）を計測するユニットです．指定した時間に割り込みを発生させる機能があります．RP2040では64ビットのカウンタが実体です．
- **RTC**：リアルタイム・クロック（実体はカウンタ）です．1秒という時間を計測し，「年」，「月」，「日」，「時」，「分」，「秒」の値を計算して内蔵レジスタに記憶します．
- **I²C**：クロックに同期させてデータの通信を行う同期式シリアル通信の1つです．
- **ADC**：端子から入力されたアナログ信号を数ビット（RP2040の場合は12ビット）のディジタル信号に変換します．
- **TS**（温度センサ）：端子が感知した温度を数ビット（RP2040の場合は12ビット）のディジタル信号に変換します．
- **Watchdog Timer**：「番犬タイマ」の意味を持つタイマです．システムのデッドロックを防止する役割を持ちます．
- **QSPI**：上述のSPIの動作を4並列で行います．RP2040では，外部のシリアル・フラッシュ・メモリからデータを読み出すために使われます．
- **SRAM**：内蔵メモリです．合計264Kバイトの記憶領域が6分割（6バンク）されています．4バンク

rp2040/rp2040-datasheet.pdf
(3) ARMv6-M Architecture Reference Manual.
https://developer.arm.com/documentation/ddi0419/c/

第2章

カメラやLCDを搭載するAI向け高機能タイプも

RP2040搭載ボード＆Pico用拡張ボード図鑑

宮田 賢一

ラズベリー・パイPico（以降，Pico）を始め，ラズベリーパイ財団が開発した独自プロセッサRP2040を搭載するボードが各社から販売されています．

RP2040は，最大133MHzで動作するデュアル・コアCortex-M0+をベースに，リアルタイム信号処理を可能とするプログラマブルI/O，1クロック・サイクルで処理可能なSIO（GPIO），音声処理やグラフィックス・レンダリングへも応用可能な補間器など信号処理に向いた機能を備えています．さらに，ハードウェア整数除算器，ROMに内蔵された高速浮動小数演算ライブラリを持っており，高速な計算処理も実現できる可能性を持ったプロセッサです．

本稿では，RP2040を搭載するボード（**表1**）や，ボードと一緒に使える拡張ボード（**表2**）を紹介します．

みやた・けんいち

表1　個人で入手可能なRP2040搭載マイコン・ボード（2021年5月時点）

ボード名	メーカ	フラッシュ・メモリ	GPIO※1	LED	Qwiic※2	LiPo※3	SDカード・スロット	参考価格
Raspberry Pi Pico	ラズベリーパイ財団	2Mバイト	26（23+3）	1（緑）	×	×	×	550円
Tiny 2040	Pimoroni	8Mバイト	12（8+4）	1（RGB）	×	×	×	9.8ドル
Feather RP2040	Adafruit		21（17+4）	2（赤，RGB※4）	○	○	×	11.95ドル
QT Py RP2040			11（7+4）	1（RGB※4）	○	×	×	9.95ドル
ItsyBitsy RP2040			23（19+4）	2（赤，RGB※4）	×	×	×	9.95ドル
Thing Plus RP2040	SparkFun Electronics	16Mバイト	18（14+4）	2（青，RGB※4）	○	○	○	17.95ドル
Pro Micro RP2040			20（16+4）	1（RGB※4）	○	×	×	9.95ドル
MicroMod RP2040 Processor			29（25+4）	2（青，RGB※4）	×	×	×	11.95ドル
Arducam Pico4ML	Arducam	2Mバイト	26（23+3）	1（緑）	×	×	×	49.99ドル

※1：ディジタル＋アナログの内訳
※2：SparkFunの4線式通信規格Qwiicシステムの接続端子
※3：リチウム・ポリマ・バッテリの接続端子
※4：NeoPixelまたはその互換品

表2　個人で入手可能なラズベリー・パイPico用拡張ボード（2021年5月時点）

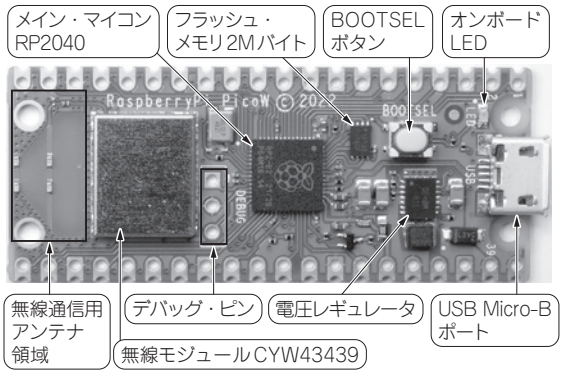
ボード名	メーカ	主な機能	参考価格
Grove Shield for Pi Pico	Seeed	Groveポート×10	4.3ドル
Pico Explorer Base	Pimoroni	LCD（IPS），圧電スピーカ，モータ・ドライブ	26.08ドル
Pico Decker		専用アドオン・ボード用ポート×4	14.1ドル
Pico Display Pack		LCD（IPS），RGB LED	15.86ドル
Pico Unicorn Pack		RGB LED 16×7マトリクス	23.26ドル
Pico Scroll Pack		白色LED 17×7マトリクス	15.86ドル
Pico Audio Pack		オーディオ出力	15.86ドル
Pico VGA Demo Base		ビデオ出力	21.15ドル
Wireless Pack		無線通信［IEEE 802.11b/g/n（2.4GHz）］	14.1ドル

第3章

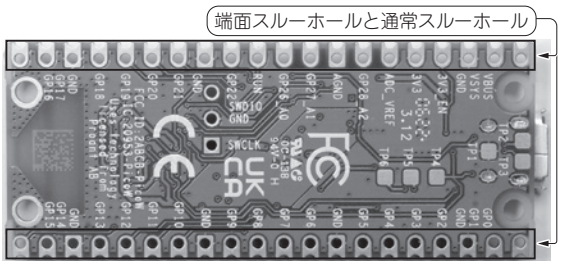
1200円で買える無線マイコン・ボード！
CとMicroPythonでLチカ＆HTTPクライアントを試す

PicoのWi-Fi版「Pico W」

宮田 賢一

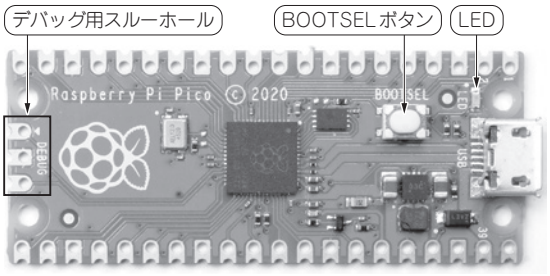


（a）表

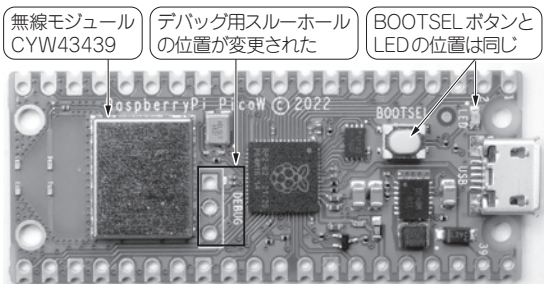


（b）裏

写真1　Pico Wの構成部品



（a）Pico



（b）Pico W

写真2　PicoとPico W搭載部品の位置関係

Wi-Fiに対応したラズベリー・パイPico W（以下，Pico W）が発売されました．技適の認証がなされていないため国内販売はまだですが，既に一部の通販サイトでは1200円程度で販売することが予告されています．本章では，先行して入手したPico Wで，何が新しくなったのかを紹介します．

なお，本章の一部にはPico Wの無線機能を実際に動作させて得られた実験結果が含まれています．この実験に先立ち，総務省の「技適未取得機器を用いた実験等の特例制度」[注1]に基づいて，筆者が所有しているPico Wを短期間の実験を目的とした無線設備として届け出を行った上で，関連する法令を順守して運用していることを記しておきます．読者の方には以下の枠内の注意を喚起するとともに，Pico Wを入手して日本国内で使用する場合には，現時点では読者自身による特例制度への届け出が必要となりますので十分に気を付けてください（関連記事：314ページ）．

この無線設備は電波法に定める技術基準への適合が確認されておらず，法に定める特別な条件の下でのみ使用が認められています．この条件に違反して無線設備を使用することは，法に定める罰則その他の措置の対象となります．

注1：https://www.tele.soumu.go.jp/j/sys/others/exp-sp/

ボード構成

● 部品配置

Pico Wの外観を**写真1**に，ラズベリー・パイPico（以下，Pico）とPico Wとの比較を**写真2**に示します．

表面にはこれまでのPicoと同じく，メイン・マイコンであるRP2040と2Mバイトのフラ[redacted]

第1章

開発環境の構築方法からPico C/C++ SDK付属サンプル・プログラムの試し方まで

開発環境1…ラズベリー・パイ4を使った公式推薦のプログラミング

小野寺 康幸

表1　Picoの情報入手先

文書名	内　容	URL
Raspberry Pi Pico Pinout	Picoの端子割り当て情報	https://datasheets.raspberrypi.com/pico/Pico-R3-A4-Pinout.pdf
RP2040 Datasheet	Picoに搭載するマイコン・チップ RP2040のデータシート	https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf
Raspberry Pi Pico Datasheet	Picoのデータシート．基板外形や電気的特性などが記載されている．端子割り当て情報もある	https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf
Getting started with Raspberry Pi Pico	Picoの開始ガイド．言語C/C++を使う開発環境の準備からデバッグ方法まで記載されている	https://datasheets.raspberrypi.com/pico/getting-started-with-pico.pdf

ラズベリー・パイPico（以降Pico）は単体で販売されるため，手がかりがないことには何も始められません．マニュアルや開発ソフトウェアは付属しませんので，まずはどこから手を付ければよいのか解説します．

言い換えればPicoを手に入れたら，手始めにやることを次の手順で解説を進めます．

(1) 情報を手に入れる
(2) 開発ソフトウェアをインストールする
(3) サンプル・プログラムを動かす

まずは基本に従ってPicoを動かしてみます．本稿の内容をPico開発の手始めにしていただければと思います．

公式ウェブ・サイトから情報入手

Picoの公式情報は以下のウェブ・ページにあります．

```
https://www.raspberrypi.com/documentation/microcontrollers/
```

ここから必要な情報を入手します．特に重要なファイル（PDF）をダウンロードして目を通しておきましょう．もちろん読破する必要はありませんが，どこにどんな情報が記述されているか把握しておき，必要になったときに参照できるようにしておきます．**表1**に主な情報を示します．いずれも上記URLから辿れます．

Pico開発のあらまし

● 本章では開発マシンにラズベリー・パイ4Bを使うことを前提とする

Picoは，ラズベリー・パイ4Bで開発することを基本にしています．PCでも開発できますが，まずは基本に従ってみましょう．さらに，C/C++を開発言語の基本にしています．Python（正確にはMicroPython）でも開発できますが，実装されていない機能は動作しません．C/C++なら細かいところまで手が届きます．機能を割り切って簡単に開発したいならPythonを使ってもよいでしょうが，ここでは基本に従ってC/C++で開発します．

Pico以外に使用する物は以下の通りですので，あらかじめ用意しておきます．

- ラズベリー・パイ4B：Picoの開発用
- ACアダプタ（USB Type-C）：ラズベリー・パイ4Bへの電源供給
- microSDカード（以降SDカード）：ラズベリー・パイ4BにOSを書き込む
- マウス：マウス操作をする
- キーボード：キーボード操作をする
- ディスプレイ：画面表示する
- Micro HDMIケーブル：モニタ出力する
- PC：ラズベリー・パイ4のOSをSDカードに書き込む
- Wi-Fi環境：インターネットに接続する

第2章

ラズパイ4向けの公式C/C++ SDKをWindowsやLinuxで使う

開発環境2…PCだけでPicoをプログラミング

井田 健太

PCだけでPicoの開発ができる

ラズベリー・パイPico（以降，Pico）に搭載されているマイコンはRP2040です．これを含むRP2シリーズのマイコン向けのC/C++開発環境がRaspberry Pi Pico C/C++ SDK（以降，公式SDKと呼ぶ）として提供されています．

基本的にはラズベリー・パイ4で動作しているRaspberry Pi OSで使用することを前提としていますが，Windows，macOS，LinuxなどのOSでも使用できます．

ここでは，Windows 10およびUbuntu上に開発環境を構築する手順を示します．

Picoに接続してブレーク・ポイントの設定やステップ実行ができるデバッガも使えるようにします（**写真1**）．

● Windows…WSL2のUbuntuで

公式SDKはLinux用となっているため，そのままではWindows上では使用できません．

公式ドキュメントでは，Visual Studio（マイクロソフト）付属のビルド・ツールや，Windows版の幾つかのソフトウェアをインストールする方法が紹介されています．しかし，この手順は環境構築の手間がかかる上，問題が発生しやすいです．

ここではWindows 10の1903以降で使用できる，Windows Subsystem for Linux 2（WSL2）を使って，Windows上にUbuntu環境を構築し，Ubuntu上で公式SDKを使用します．

WSL2上にUbuntu環境を構築する手順は，Interface誌2021年5月号，または次のページを参照してください．

```
https://interface.cqpub.co.jp/202108wsl/
```

Windows環境の場合，WSL2上にUbuntu 20.04 LTSをセットアップした後の手順の多くは，Ubuntu上で使う場合の手順と同じになります．

以降の手順で，WindowsとUbuntuでコマンドが異なる場合などは，それぞれを分けて解説します．

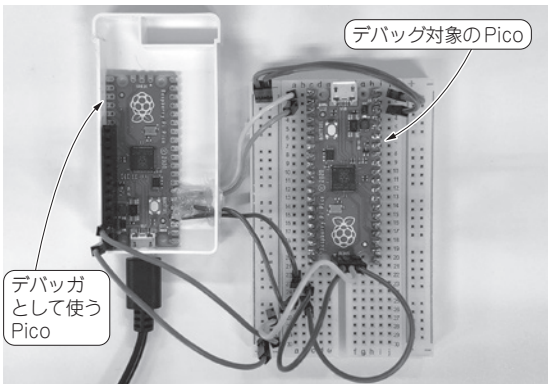


写真1　PC1台でPicoのクロス開発ができる．デバッガも使用可能

● Linux…Ubuntuで

執筆時点で最新のLTS版（長期サポート版）は，Ubuntu 20.04 LTSです．以降では，これを使った構築手順を示します．古いバージョンの場合，追加のパッケージのインストールなどが必要になる可能性があります．

環境構築

■ ツール類のインストール

以降の作業はUbuntu上のターミナルなどで行います．

● C/C++ビルドツールのインストール

以下のコマンドで，C/C++を使った開発に必要なパッケージをインストールします．

```
$ sudo apt update↵
$ sudo apt install -y build-essential cmake gcc-arm-none-eabi git curl unzip↵
```

● 書き込み用picotoolのインストール

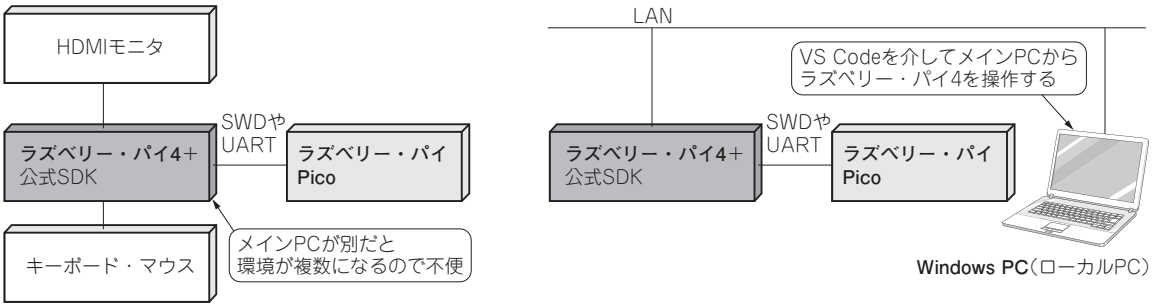
RP2040の組み込みブートローダを操作するツールとして，picotoolが用意されています．

第3章

VS Codeからリモート接続!
メインPC環境に手を入れずに公式SDKが使える

開発環境3…PCからラズパイ経由でPicoをプログラミング

丸石 康



（a）ラズベリー・パイ4の公式SDKを直接使う場合　　（b）VS CodeのRemote Development拡張機能を介した場合

図1　本章でやること…ラズベリー・パイ4に構築した公式SDKをWindows PCから便利に使えるようにする
VS Codeの拡張機能を介せば，あたかも直接ソースコードを編集したりデバッグしたりするような操作感が得られる

本章では，**図1**のようにラズベリー・パイ4に構築した公式SDKをWindows PCから快適に使う方法を紹介します．

第2章では，公式SDKをWindows PCにインストールする方法を紹介していますが，ここで紹介する方法は，あまりメインPCの環境に手を入れたくないという人にお勧めです．　**<編集部>**

きっかけ…ラズパイ上の公式SDKをPCから快適に操作したい

● ラズパイならコマンド3行だけで環境構築できる

ラズベリー・パイPico（以降，Pico）のC/C++用開発環境であるSDK（Software Development Kit）のインストール方法は，公式資料Getting started with Raspberry Pi Pico(1)で説明されています．資料の中には，ラズベリー・パイ4（Raspberry Pi OS），とmacOS，Windows上にインストールする方法がそれぞれ紹介されています．

その中でも，ラズベリー・パイ4へのインストール方法は，3つのコマンドを実行するだけなので，PCの手順よりもかなりシンプルです．

● メインPCと連携しにくいのがデメリットだった

ところが，ラズベリー・パイ4にSDKをインストールしてPico用ソフトウェアを開発するには，PCなどからSSHやVNCなどで接続するか，直接モニタやキーボード，マウスを接続するかのどちらかになります．

SSH接続では，Visual Studio Code（VS Code）のようなGUI環境を使った開発ができません．またVNCでは，操作方法の違うWindowsとRaspberry Pi OSを行ったり来たりするので，作業しづらいと感じるでしょう．

直接モニタやキーボード，マウスを接続する方法では，メインPCが別にある場合に複数の環境を操作する不便さを感じると思います．

● ラズパイ上の公式SDKをWindowsの作法で快適に操作する

本章では，Getting started with Raspberry Pi PicoのChapter 7で解説されているVS Codeを使った公式SDKの開発環境をWindows PCからリモートで操作できるようにします．

図1（b）に本章で紹介する環境の全体像を示します．この環境では，Windows PCであたかも直接ソースコードを編集したりデバッグしたりするような操

第4章

Arduino IDEで開発したスケッチをステップ実行!
ブレーク・ポイント設定や変数などの情報を見える化

汎用デバッガ× VS Code で構築するデバッグ環境

丸石 康

● おなじみArduino IDEでもPicoの開発ができる

Arduinoは，ラズベリー・パイ Pico（以降，Pico）に搭載されているRP2040を公式にサポートすることを発表しました．Arduinoが提供している統合開発環境Arduino IDEは，ラズベリーパイ財団から提供されている公式SDK（Software Development Kit）に比べると，環境構築の手順がシンプルなことや，豊富なArduino用ライブラリの資産が利用できることなどのメリットがあります．

● 汎用デバッガとVS Codeがあればデバッグも OK!

Arduino IDEは，スケッチ（C++ベースのArduino専用プログラミング言語）のコンパイルと，シリアル・モニタを使ったデバッグができますが，ステップ実行や任意の変数の値確認など，いわゆるデバッガ機能はありません．2021年3月にベータ版がリリースされたArduino IDE 2.0には将来的にソースコード・デバッグ機能が追加されるようですが，本原稿執筆時点（2021年5月）では対応が十分ではありません．

そこで本章では，現時点での正式リリース版であるArduino IDE 1.8.15でコンパイルした実行ファイルをデバッグする方法を紹介します．本章で紹介するデバッグ方法では，**写真1**に示す汎用デバッガJ-Link（SEGGER）とVS Codeを使います．

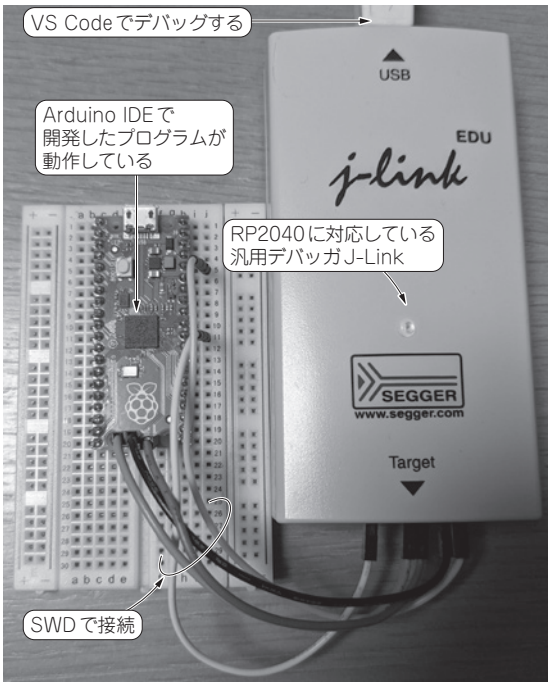


写真1　Arduino IDEで開発したPico用プログラムを汎用デバッガとVS Codeでデバッグしてみる
Picoと汎用デバッガJ-Link EDUを接続した様子

ステップ1：Arduino IDEにPicoの開発環境を追加する

● Picoの開発にArduino IDEを使うメリット

▶（1）豊富なArduinoライブラリが利用できる

Pico用のC++開発環境にArduino IDEを使えば，豊富なArduino用ライブラリの資産をPico用プログラムの開発に利用できます．

ライブラリ資産の中には，Picoの性能を生かせないものやPicoで使用できないものもありますが，コミュニティ・ベースの開発が進むにつれて問題は解消されていくと考えられます．

▶（2）少ない学習コストで利用できる

Arduino IDEと公式SDKは，どちらもコンパイラにGCC（GNU Compiler Collection）を使っていますが，サポートしているAPI（Application Programming Interface）が異なります．Arduino IDEは，Arduino APIをサポートしています．公式SDKには独自のAPIが用意されています．

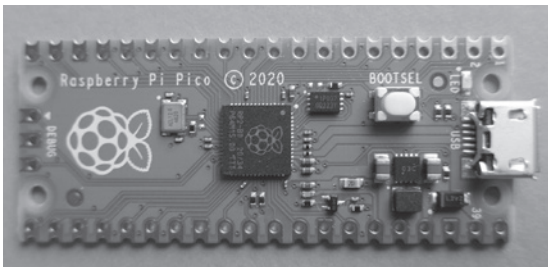
Arduino APIは，通常だとマイコンによって異なるドライバなどのインターフェースの差分を吸収してくれるので，新規のマイコンやボードであっても少ない学習コストで開発できるようになります．

第6章

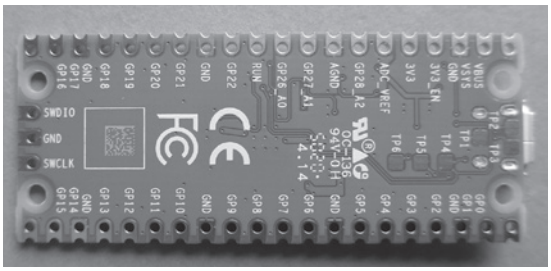
流入/流出電流値や内部プルアップ/プルダウン抵抗値など

実験でチェック！ I/O端子の実力

漆谷 正義



（a）表面



（b）裏面

写真1　ラズベリー・パイ Picoはマイコン基板…今回はI/O端子の実力をチェック

Picoは従来のラズベリー・パイの拡張基板であるとともに，単独でも動作するマイコン・ボードです（**写真1**）．

写真1を見ると，センサやモータに接続するためのGPIOの他に，A-D変換入力，PWM出力，シリアル通信端子などがあります．おのおのの端子には，動作電圧，電流，入力/出力インピーダンスが定められています．その多くは仕様書に記載されていますが，目を通す時間がない方，内容が難解と感じられる方もいると思います．また，仕様書に明記されていない項目もあります．ここでは，GPIOなどの外部端子の電気的仕様，要点，使い方を解説します．

搭載マイコンは，Arm Cortex-M0+コアを2個搭載したRP2040です．開発言語はC/C++以外に，MicroPythonおよびCircuitPythonに対応しています．おのおの，Pico専用のファームウェア（UF2ファイル）をインストールして使います．

以下の実験ではMicroPythonを使っています．

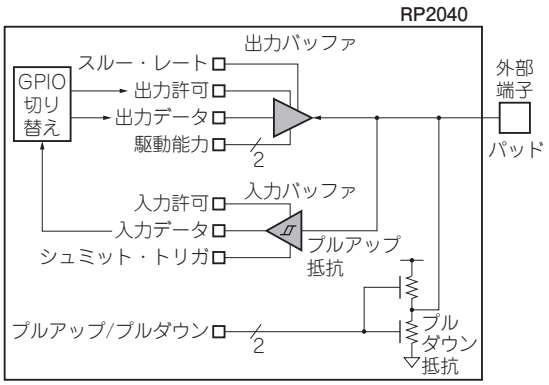


図1　ICチップを外部端子と接続するインターフェースをパッド回路と言う

GPIOを動かす前に知っておきたい基礎知識

● 外部端子接続回路

GPIOの外部端子はパッドに接続されています．パッドは，ICチップを外部端子（足）と接続する場所のことです．パッドとマイコン内部回路とのインターフェースをパッド回路と言い，**図1**のようになっています．

出力や入力の許可，駆動能力の切り替え，シュミット・トリガのON/OFF，プルアップ/プルダウン抵抗のON/OFFなどが，次項で示すGPIOレジスタを使って設定できます．

● GPIOの制御レジスタ

RP2040のGPIOは全部で30本あります．このうちGPIO23 ～ GPIO25とGPIO29は，Picoのボード回路の制御に使われているので，外部に出ているのは**表1**の26本です．

表1にはGPIOレジスタのオフセット・アドレス

第7章

多数ある拡張ボードを利用してサクッと試作

Arduino互換ボード利用のススメ

関本 健太郎

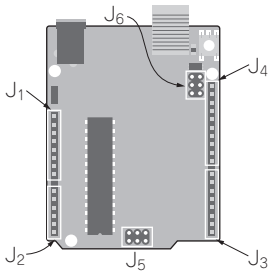


図1　Arduino Unoボードのピン割り当て

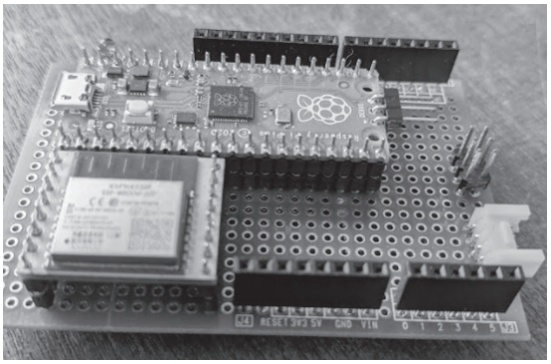


写真1　手持ちマイコン・ボードのピン割り当てをArduino互換に仕立てるといろいろなシールドを利用できる

2005年にArduinoボードが登場し，Arduinoボード向けにたくさんの拡張ボードが販売され，Arduino IDEというソフトウェア開発プラットフォームが整備されました．それ以来，搭載されるマイコンが追加され，さまざまなボードがArduinoファミリとして商用化されてきました．現在では数十種類あります．特にArduino Uno, Arduino Nanoなどが広く認知されています．このArduinoのエコ・システムを利用すべく，多くのベンダからのマイコンの評価基板は，Arduinoのボードのピン配置に合わせたものが出荷されるようになっています．

ご多分に漏れず筆者も，評価ボードのないマイコン向けに評価基板を作成する際には，コネクタをArduino互換ボードのピン配置にするようにしています．

本章では，最も一般的なArduino Unoを取り上げ，互換のピン配置について整理し，Arduino互換ボードのデザインのカギについて解説します．その情報をもとに，ラズベリー・パイPicoボードをArduino互換ボード化する拡張ボードを作成し（**写真1**），Arduino IDEによるプログラム作成例を説明します．

Arduino Uno系の基板のピン配置

● 6つのコネクタの主な機能

Arduinoというとほとんどの場合，Arduino Unoを指します．ピン配置を**図1**に示します．コネクタがJ1～J6の6つありますが，初期のArduino DuemilanovaやArduino Diecimilaでは，コネクタJ6はなく，J1～J5の5つとなっていたり，コネクタJ4のピン数が8ピンまたは10ピンとなっていたりして，Arduinoのモデルによって，コネクタ配置は若干異なっています．なお，コネクタJ3とコネクタJ4はピン間ピッチが半ピッチずれており，拡張ボードが左右逆に挿入されることを防ぐ工夫がされています．

Arduino Unoの場合には，マイコンはATmega328Pであり，MCUのポートCがJ2，ポートDがJ3，ポートBがJ4に割り当てられています．

マイコンのポートC，D，Bを J2，J3，J4に配置した結果，マイコンの周辺機能としては主に以下が割り当てられています．

- J1…電源関連
- J2…アナログ入力およびI^2C機能
- J3…ディジタル入出力，シリアル通信，PWM機能
- J4…ディジタル入出力，SPI，I^2C機能
- J5…SPIおよびICSP（ファームウェア書き換え）
- J6…USB-シリアル機能を提供するチップ（Atmega8U2）のためのICSP機能が割り当てられ，USB-シリアル機能のファームウェアの書き換えのときのみ利用される

第2章

正確なタイミングの高速I/Oを作れる

プログラマブルI/Oの機能と簡易ライブラリ

森岡 澄夫

ラズベリー・パイPico（以降，Pico）には，普通のマイコン・ボードにはない，とてもユニークな機能があります．プログラマブルI/O，略してPIOと呼ばれるものです．ここでは，PIOはパラレルI/Oの略ではありません．

PIOは，GPIO端子のアクセスを極めて正確なタイミングで高速に（125MHzクロック精度）行えるカスタムI/Oペリフェラルを最大8個作れます．

従来のラズベリー・パイやPCの弱点は，センサやアクチュエータなどI/Oデバイスの制御を行いにくいことです．Picoは，それを代行する高速I/Oインターフェース・ボードになります．同じ目的でFPGAを使うよりも，簡単で安価です．本章ではPIOの基本的な使い方を説明します．

プログラマブルI/O機能のメリット

● 正確なタイミングで信号入出力をするインターフェースを作れる

ラズベリー・パイやPCを使うときにしばしば悩みの種となるのが，センサやアクチュエータなどのI/Oデバイスを接続しにくいことです．ラズベリー・パイにはGPIO端子があり，SPIやI²Cなどのペリフェラル回路も幾つかあるのですが，デバイス個数や自由度は大きく制限されます．特にソフトウェアからGPIO端子を直接読み書きしてデバイスを操作する方法[注1]は，時間精度を気にしないLEDのようなデバイスを除き，とても実用に耐えません．

表1は，ラズベリー・パイ4のLinux（Raspberry Pi OS）からソフトウェアでGPIO出力をトグルさせた例です．ウェイト設定値は信号をトグルさせるための待ち時間を意味します．ウェイト設定値がxマイクロ秒（μs）であるとき，1000000/（2x）Hzの周波数が得られれば，正確な時間でのコントロールができていると結論できます．例えば，x＝10000μsの場合，50Hzの周波数が出力されるならば，正確な時間でのコントロールができているということです．しかし，**表1**によると，xが小さいところでは，そのような周波数は得られていません．例えば，x＝100μsならば5000Hz，x＝10μsならば50000Hzが出ていなければなりませんが，そのような周波数は出ていません．したがってラズベリー・パイ4では，正確な時間でのコントロールができていないことが分かります．**表1**では，出力される周波数を同一設定にしても大きな幅で揺らぐことを示すために，最小周波数，最大周波数，および平均周波数の3種類を示しています．

表1　ラズベリー・パイ4においてC言語上で関数usleep()でウェイトし出力トグルさせた結果
従来のラズベリー・パイではソフトウェアから高速なGPIO操作をすることは難しかった

ウェイト設定値x［μs］	最小周波数［Hz］	最大周波数［Hz］	平均周波数［Hz］
10	4474.0	5335.0	4600.0
100	2000.0	2455.0	2390.0
1000	413.4	447.2	441.4
10000	48.0	49.4	49.2

多くのデバイスを利用したり正確なタイミング・コントロールをしたりするためには，マイコンやFPGAをインターフェースとして使うのがこれまでの定石です（**表2**）．ただし，どれも一長一短あり，万能な方法はありません．マイコンは安価で手軽に使えますが，GPIOによるデバイス・アクセスはそれほど高速ではなく通信に広い帯域は取れません（例えばカメラを接続するなどは難しい）．FPGAはピン数が多くて帯域を広く取れ，タイミング管理も数10nsの精度でできますが，設計は誰でもできるとは言えず，価格も高いです．

● 8つの並列動作するミニ・プロセッサとして使える

PicoのPIOは，FPGAほどではありませんが，正確にタイミング制御できる高速インターフェースを安価に作れる面白い機能です．**図1**がPIOのブロック構成

注1：ビット・バンギングと呼ばれる．自由なプロトコルやビット幅で通信できる．

第3章

GPIOにはできない速度! リアルタイムがうれしい

簡易ライブラリで複数のRCサーボや測距センサを動かす

森岡 澄夫

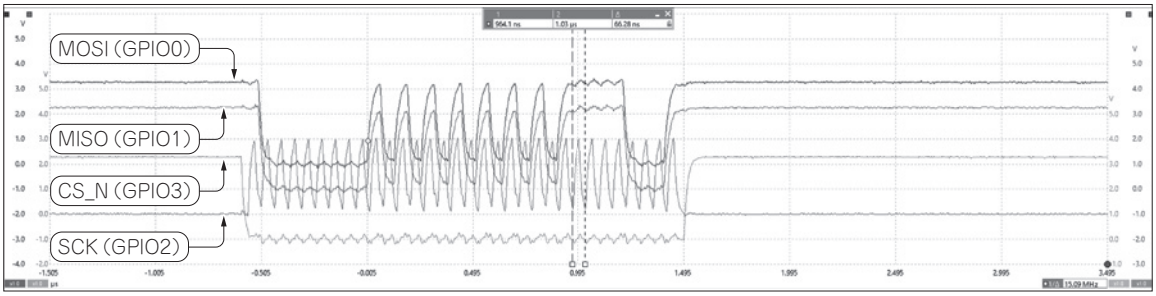


図1　PIOによるSPI入出力の実力
CPOL=0，CPHA=0，MSBファーストに相当する動作になっているが，容易に変更できる．通信周波数もステート・マシンの動作周波数を変えることで簡単に調整できる

前章に続き，プログラマブルI/O（以降，PIO）を実際に使ってみた4つの例と，それらの製作に当たってつまずいた点などを紹介します．ラズベリー・パイPico（以降，Pico）単独で完結するのではなく，PCやラズベリー・パイと接続し，拡張I/Oインターフェースとして使うことを想定しています．作例は前章で用いた頒布アーカイブに収録されています．

作例1：SPI入出力の処理

● 製作物の機能と特徴

信号入出力の実際的な例題として，センサとの通信に用いるSPIのマスタ・インターフェースを作ってみます．

4線式のSPIでは，マスタ出力MOSI，マスタ入力MISO，クロックSCK（またはSCLK），チップ・セレクトCS_N（またはSS）があります．**図1**のようにクロックに同期して出力と入力が同時並列で行われます．送受信データ長は32ビットにしていますが，作り替えは難しくありません．Armプロセッサからステート・マシンへ送信データを送ると送受信が始まり，完了すると受信データがArmプロセッサへ送られます．

SPIにはクロックの極性やデータ入出力のタイミングによる幾つかのモードがありますが，作例では立ち上がりでサンプリング動作するようにしています（いわゆるモード0）．しかし，どのモードへも簡単に作り替えできます．

SPIクロックの周波数はステート・マシンの動作周波数の1/4であり，簡単に変えられます．筆者が実機で試したところ，SPIクロックは15.625MHzで安定した入出力ができました．一般的なマイコンに搭載されているSPIペリフェラルと同等かそれ以上の性能で，十分な実用性があります．通常，マイコンのGPIO制御ではこのような速度は出せませんので，PIOがとても強力であることがよく分かります．

● ソフトウェア構成とコード

ステート・マシンのコードが**リスト1**，Armプロセッサのコードが**リスト2**です．前章で例を示した通り，out命令やin命令を使ってGPIOの読み書きを行い，push命令やpull命令を使ってArmプロセッサとのデータ交換をします．OSRレジスタとISRレジスタを，それぞれ1ビットずつGPIO入出力するためのバッファとして利用しています．

全32ビットを送受信したかは，OSRレジスタが空になったかをjmp命令で見ることによって判定しています．

● サイドセットを使ったおかげで高速インターフェースができた

この例では，SPIのSCK出力やCS_N出力

第4章

PIOアセンブリ・コードが手軽に作れる

実機がなくてもデバッグできる簡易PIOエミュレータ

森岡 澄夫

ラズベリー・パイPico（以降，Pico）のPIO（プログラマブルI/O）は，カスタマイズした高速インターフェースを作れるという，他のマイコン・ボードにない画期的な機能を提供しています．しかし，若干クセのあるアセンブリ言語でのプログラミングが必要な上，デバッグ環境が整っておらず，開発がやりにくい問題がありました．

開発のきっかけ

● PIOはインターフェース作成に高いポテンシャルを持つ

ラズベリー・パイを含む従来のマイコン・ボードは，マイコンの動作クロックに近い高速インターフェースを作ることが困難です．例えば，マイコン・チップ内のUARTの数が足りなくなれば，処理を自作するしかありません．ペリフェラルにない独自インターフェースが必要になるときも同様です．

しかし，ソフトウェアでGPIOを制御するやり方では，高帯域を出せません．また，ラズベリー・パイのように，リアルタイムOSではないLinuxを使っている場合には，GPIOのスイッチング速度だけでなく，タイミング精度もかなり悪く，実用には耐えません．

PicoのPIO(3)は，これを解決できる画期的な機能です．高速（入出力信号の周波数が62.5MHzくらいまで）かつ，高いタイミング精度（クロック単位）のI/O処理を，FPGA（Field Programmable Gate Array）のような回路設計をすることなく，プログラミングだけで実現できます．実際に，Pico SDKのPIO設計サンプルには，UARTだけでなくSPIやI²Cも含まれているほど，実用度の高い機能です．

● PIO開発環境が抱えていた問題

利用価値の高いPIOですが，筆者が第7部第4章を参考に実際に試してみたところ，次の2点がネックでした．

1. アセンブリ言語での開発になる．命令セットはあまり一般的な設計ではない独特なもので，命令動作の細部まで頭に入れていないと分かりにくいバグを発生させてしまう．
2. ステート・マシンを1ステップずつ実行させてモニタリングできるようなデバッガが提供されていない．GPIOピンにオシロスコープをつないで波形観測しながら開発することになる．しかしステート・マシン内部のレジスタ値などを観測しにくく，ちょっとした間違いであってもなかなか特定できない．

特に後者は問題で，オシロスコープとにらめっこしながらのデバッグの面倒さは，PIOを使ってみる気を削ぐものでした．

● ステート・マシン1つのデバッグに特化したエミュレータ

そこで「PIOを使おうと思い立ったら，すぐに開発できるようにする」ことをコンセプトに，ステート・マシン実行を可視化するエミュレータを作ることにしました．

図1に作業の全体的な流れを，**表1**に本エミュレータでの制限事項を示します．詳細は後で説明しますが，エミュレータからさまざまなC言語API関数を提供します．それらの関数を使ってアセンブリ・コードを書き［第8部で紹介されているMicroPythonによるコード記述の仕方と類似］，任意のCコンパイラでコンパイルします．また，GPIO信号や割り込み信号などステート・マシンへの入力を，別途CSVファイルにあらかじめ書いておきます．

コンパイルで得られたバイナリを実行するとエミュレーションが行われます．その結果，各クロック・サイクルで実行した命令，レジスタ値，GPIO出力値を記録したCSVファイルと，Pico C/C++SDKでコンパイル可能なアセンブラ・ソースファイルが生成されます．これらを使ってデバッグと実機実行用バイナリの作成を行います．

提供エミュレータの特徴は，Linux/Windowsなど好きな環境ですぐ使え，操作が簡単で，実機を持ち出さなくてもおおよそのデバッグができる点です．大きな制限事項は，1つのステート・マシンのエミュレーションしかできないことです（簡便さの[illegible]あえ

アマゾンのバックアップで機能が充実！
複数の処理もマルチタスクでシンプルに書ける

第1章 FreeRTOSを載せる方法

石岡 之也

ワンチップ・マイコンでリアルタイムOSを用いる大きな利点としては，マルチタスクによって複数の処理を，あたかも同時に動かすようなプログラミングが可能になることだと思います．

Arduinoのように1つのループで複雑な処理を行うことも可能ですが，何かを処理中に別の処理を行うには，プログラミングや設計方法などの技術が必要になってきます．こういったときにRTOSを使うと，全ての問題を解決してくれるわけではありませんが，技術不足を補ってくれると思います．

リアルタイムOSの導入は楽ではありませんが，FreeRTOSのように利用者が多いものは，ウェブ上で情報が見つけやすかったり，掲示板へ質問したりすることで，解決も可能です．

RAMサイズが小さいマイコンでは，リアルタイムOSが占有するサイズがネックになることがありますが，ラズベリー・パイPicoは，ワンチップ・マイコンとしては大きな264Kバイトものを搭載していることから，リアルタイムOSを有効に利用できると思います．

FreeRTOSは2000年代前半にリリースされたソースが公開されているリアルタイムOS（RTOS）です．マイコン・チップ・ベンダ各社が提供するサンプル・プログラムに同梱されていることがあります．また，安価なマイコン・モジュールに使われるなど，有名なRTOSの1つです．

2017年にアマゾンが買収し，付加機能が充実したことと，Wi-Fi搭載マイコン・モジュールESP32で使われていることから，日本でも名前をよく目にするようになりました．

● Pico向けFreeRTOS

GitHub上に，Pico向けのFreeRTOSが公開されています．

```
https://github.com/PicoCPP/RPI-pico-FreeRTOS
```

このソースコードを使って，

- ビルドに必要な環境構築
- 必要なソースコードのダウンロード方法
- ビルド方法
- 製作事例

を紹介します．なお，執筆時点ではシングル・コアでの動作のようです．

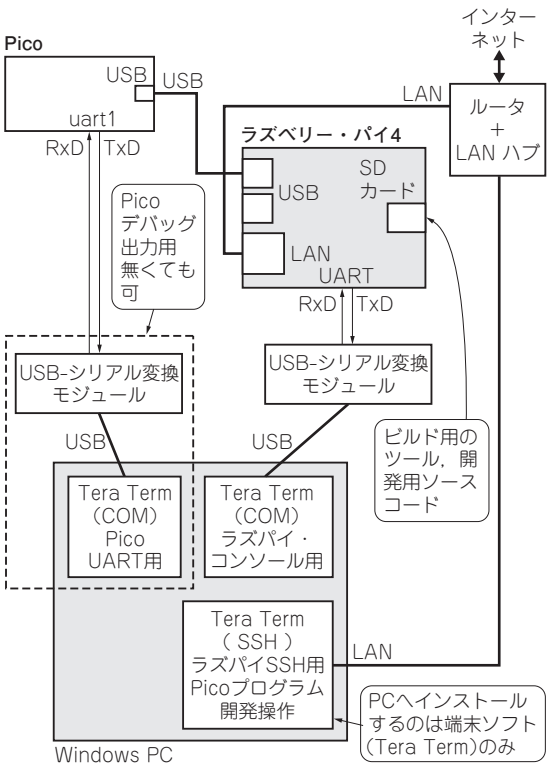


図1　Picoの開発環境はラズベリー・パイに構築した

開発環境

● 長期運用を考慮してラズベリー・パイにUbuntuを使って構築した

Pico向けのFreeRTOSの開発環境を，今回はラズベリー・パイ4とUbuntuを使って構築しました（図1）．ラズベリー・パイのLinuxは，最初にRas[illegible] IT

第1章

開発環境のセットアップからマイコン向けサンプルの試し方まで

フレームワークTensorFlowの準備

大沢 健太郎，谷本 和俊

オープンソースのマイコン用機械学習フレームワークTensorFlow Lite for Microcontrollers（以下，TFLM）を，Picoで動かす手順について解説します．TFLMのビルド環境にはセットアップが簡単なラズベリー・パイ4を利用します．

Pico用のTensorFlowリポジトリがある

Pico用のTFLMは，既にGitHub上に公開されています．ただし，グーグルが運営しているTensorFlowのリポジトリではなく，ラズベリーパイ財団が運営している公式リポジトリ（https://github.com/raspberrypi/pico-tflmicro）にあります．

READMEを見ると，このリポジトリは自動生成されたもので，リード・オンリとあり，issueやpull requestは，TensorFlow側のリポジトリにファイルされるとあります．最新版を反映するには，TensorFlowのリポジトリにあるスクリプトgenerate.py（tensorflow/lite/micro/tools/project/generate.py）を使って生成するとの記載がありますが，現在，TFLMのリポジトリはTensorFlow本体から独立（https://github.com/tensorflow/tflite-micro）しました．独立後のプロジェクト生成ツール（tflite-micro/tensorflow/lite/micro/tools/project_generation/create_tflm_tree.py）もそのままでは動作しませんので，今回はラズベリー・パイ側のリポジトリをそのまま利用します．

Pico用のTFLMには，**表1**に記載したhello_world，micro_speech，magic_wand，person_detectionの4つのサンプル・アプリケーションがあります．これらのうち，hello_worldだけはPicoのLEDで動作が確認できるように実装されています．hello_world以外のサンプルは，Picoにセンサが搭載されていないため，マイク，加速度，カメラなどのセンサからデータを取得するコードは自分で記述する必要があります．まずはビルド環境構築の確認の意味で，Pico単体で動作確認が可能なhello_worldを動かしてみます．

表1　TFLMのサンプル・アプリケーション一覧

サンプル名	内　容
hello_world	入力値に対してsin波（サイン関数）の予測値を返す．PicoのLEDに対して予測値をPWM出力することでPicoのみで動作確認が可能
micro_speech	センサにマイク入力を使用したシンプルなスピーチ（キーワード）認識
magic_wand	加速度センサを使用したジェスチャ推定
person_detection	カメラ（画像入力）を使用した特定の人検出．make時に指定するサンプル名はperson_detection_int8

ビルド環境として利用するラズパイ4のセットアップ

ビルド環境にラズベリー・パイ4（または3）を利用する場合，Pico用の公式SDK環境をセットアップし，TFLMのリポジトリをクローンするだけで環境を立ち上げることが可能です．以下の手順は，ラズベリー・パイ4の/home/piで実行していきます．

● Pico用の公式SDK環境セットアップ

Picoの公式ドキュメント(1)の通りにスクリプトをダウンロードし，実行するだけでセットアップが完了します．

```
$ wget https://raw.githubusercontent.com/raspberrypi/pico-setup/master/pico_setup.sh
$ chmod +x pico_setup.sh
$ ./pico_setup.sh
```

● TFLMのクローン

ラズベリー・パイ公式のリポジトリから，Pico用のリポジトリをクローンします．

```
$ git clone https://github.com/raspberrypi/pico-tflmicro.git
```

Pico用の環境セットアップ・スクリプトが用意されているため，簡単に環境構築できます．ラズベリー・パイ4上でスクリプトを実行して20分程度ラス

第1章

せっかくだからPico Wを簡単なウェブ・サーバにしてみた

公式サンプルの活用法

小野寺 康幸

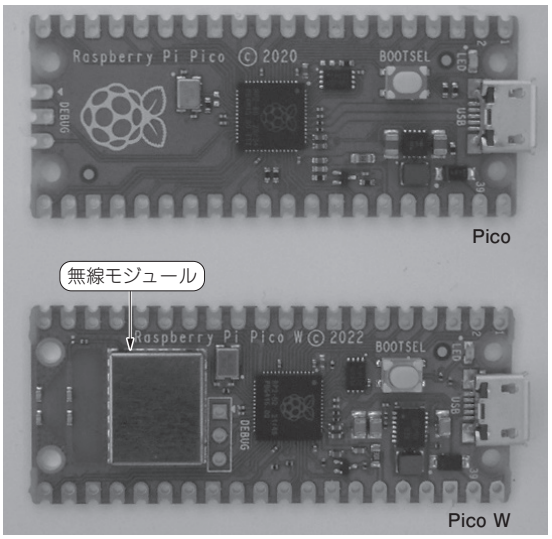


写真1　ラズベリー・パイ PicoとPico W

ラズベリーパイ財団は，2022年6月30日Pico Wを発表しました．Pico Wは，2022年10月7日に技術基準適合証明（008-220422）を取得しました．既に最新の開発環境Pico C/C++SDKは，Pico Wに対応しています．

● Pico Wの特徴

Pico W（**写真1**）最大の特徴は2.4GHz帯を利用したWi－Fi（802.11n）機能です．無線モジュールCYW43439（インフィニオン・テクノロジーズ）注1とオンボード・アンテナで実現しました．Pico W搭載マイコンRP2040とCYW43439とは，SPIで通信します．PicoとPico Wとでは，ピン配置やサイズに違いはありません．

● Pico Wのアンテナ

アンテナ付近に金属を配置したり，金属で遮へいしないようにしましょう．電波伝搬に悪影響を及ぼします．4つのチップ・コンデンサを含む部分がアンテナです（**写真2**）．

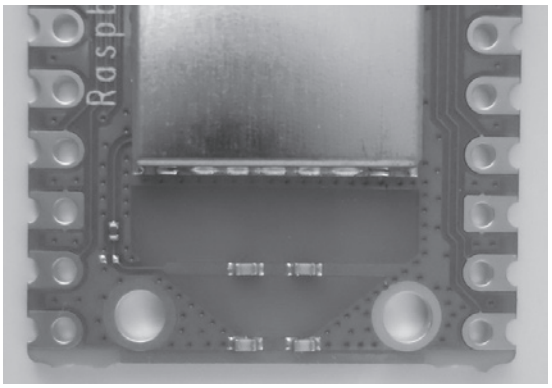


写真2　オンボード・アンテナ

Picoからのハードウェアの変更点

Picoから見てPico Wは，内部的に幾つかの変更点があります．PicoはRP2040のピンを全て使いきっており，無線モジュールCYW43439を制御する余分なピンはありません．そこで，Picoからどのように変更しているのか見ていきましょう．

PicoとPico Wの違いを**図1**と**図2**に示します．

● 違い1…GPIO29

PicoのGPIO29はアナログ入力ADC3として機能し，電源電圧V_{SYS}を取得するために使用します．

Pico WのGPIO29は，CYW43439のクロック信号WL_CLKとして使用します．さらにアナログ・スイッチを経由して，従来と同じADC3としても兼用します．なおV_{SYS}とは抵抗を介して接続しているため，信号の衝突はありません．

● 違い2…GPIO25

PicoのGPIO25はLED出力に使用しますが，Pico

注1：CYWの型名はサイプレスがインフィニオン・テクノロジーズに買収された名残です．

第2章

Pico Wファームウェア書き込み/
Wi-Fiルータ接続プログラムの入手/HTTP通信

Wi-Fiのアクセス・ポイントに接続して通信する

関本 健太郎

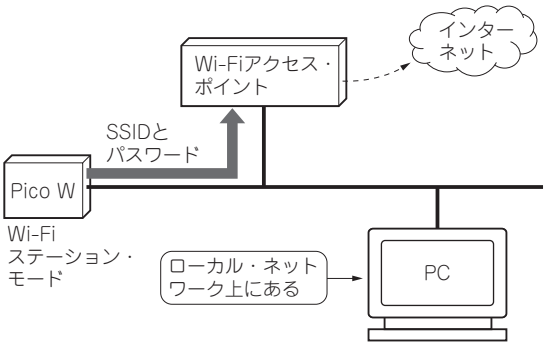


図1　PicoWをPCやWi-Fiアクセス・ポイント経由でインターネットに接続する

MicroPythonを利用して，Pico WをWi-Fiのアクセス・ポイントに接続するまでの手順について説明します．Wi-Fiのアクセス・ポイントを経由して，ローカル・ネットワークにあるPCや，インターネット上のウェブ・サイトとの通信を想定しています(**図1**)．

● ステップ1：MicroPythonファームウェアをPico Wに書き込む

Pico WでMicroPythonを使う場合，まずはPico W向けのMicroPythonファームウェアを書き込む必要があります．これは，ウェブ・サイトから最新のuf2形式のファイルをダウンロードして，Pico Wに書き込みます．

```
https://micropython.org/download/rp2-pico-w/
```

● ステップ2：統合開発環境「Thonny」のインストール

MicroPythonのプログラム開発，実行環境として「Thonny」という統合開発環境を利用します．Thonnyは初心者向けに設計されたPythonの統合開発環境です．

インストールは，ウェブ・サイト(https://thonny.org/)より，Windows向けのインストーラをダウンロードし，パソコン環境にインストールします．

以前のバージョンと比較し，特に，MicroPython本体のフラッシュ・メモリ中のファイル・システムへのファイルの転送機能など，随分使いやすくなった印象があります．また，これまではVisual Studio CodeのPyMakrプラグインをよく利用していましたが，最近はThonnyに代わってきています．

本稿ではThonnyの使い方については割愛しますが，**図2**にはプログラムの編集画面を示します．

● ステップ3：Wi-Fiルータへの接続プログラムを入手する

Wi-Fiルータへの接続プログラム(**リスト1**)は，Raspberry Pi Foundationが提供しているプログラムを切り出したものです．このプログラムは，文献(1)のURLから入手してください．

これは，connect関数として実装されていますので，後から別のプログラムで再利用しやすくなっています．なお，Wi-Fi機能を利用するには，networkモジュールをインポートする必要があります．

▶プログラムの処理内容

前半部分では，MicroPythonのnetworkモジュールのWLAN関数を利用してWi-Fiステーション・モードで起動し，Wi-FiルータのSSIDとパスワードを設定します．

また，接続するconnect関数を定義して，後半部分でconnect関数を呼び出しています．Wi-Fiルータに接続するプログラムを作成する際には，このコード・スニペットを利用すると良いでしょう．

▶利用するプロトコル

IoTの活用でよく利用されるネットワーク・プロトコルは，HTTP(HTTPS)(**図3**)，あるいはMQTT(**図4**)ですが，本稿の通信はHTTP通信とします．なお，MQTT通信例は，次章で紹介します．

第3章

Eclipse Mosquitto/MQTT Explorer/Grafanaの導入

Pico Wで得たセンサ・データをWi-Fi経由で取得する

関本 健太郎

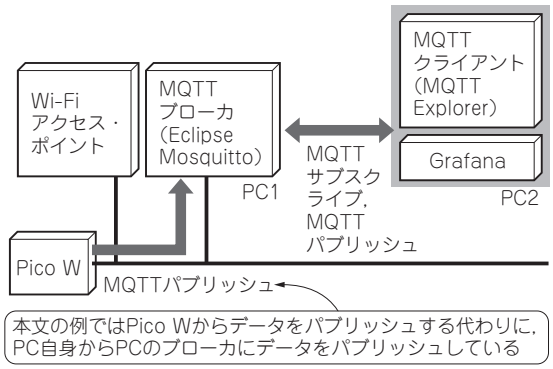


図1　Eclipse Mosquitto/MQTT Exploer/Grafanaを利用してセンサ・データの取得と確認ツールの作成を行う

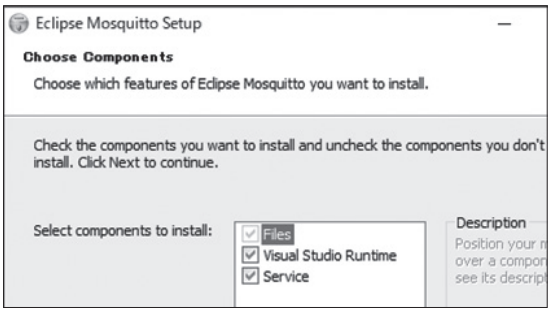


図2　Eclipse Mosquittoはインストーラ画面に従ってインストールする

● データの収集方法はその「量」で決まる

現在，エンタープライズの企業でオンプレミスの多量のセンサ・データを取得して解析し，知見を得る仕組みとして，最も手軽かつセキュアで，スケーラブルなアーキテクチャはパブリック・クラウドのマネージド・サービスを活用することです．

一方，ホーム・オートメーションなど，個人でセキュリティを度外視し，少量のセンサ・データを扱う場合には，パブリック・クラウドを利用する必要はありません．そのような場合はオープンソースのツールを用いることで，手軽にセンサ・データを取得，解析する仕組みをパソコン環境で実現することができます．

ここでは，MQTTブローカとしてEclipse Mosquitto，MQTTクライアントとしてMQTT ExploerおよびGrafanaの2つのツールを利用します．そして，センサのデータを取得し，確認できるツールを導入します（図1）．

ブローカ/クライアント・ツールのインストール

● MQTTブローカ（Eclipse Mosquitto）の導入

Eclipse Mosquitto（https://mosquitto.org/download/）は，Windows，Linux，およびMacで利用できる，MQTTプロトコル・バージョン5.0，3.1.1，および3.1を実装するオープンソース（EPL/EDLライセンス）のメッセージ・ブローカです．

▶インストール

Windows環境では，インストーラ（この例ではmosquitto-2.0.15-install-windows-x64.exe）を実行することで，簡単にMQTTブローカをWindowsのサービスとしてインストールできます．インストーラを起動すると，図2のようなダイアログが表示されますので，ウィザードに従ってインストールします．

▶Mosquittoの設定

まずは，mosquitto.confファイル（C:¥Program Files¥mosquitto¥mosquitto.conf）中の“# listener port-number…”行をコメント・アウトして，“listener 1883”に変更します．また，“# allow_anonymous false”行をコメント・アウトして，“allow_anonymous true”に変更します．

▶Windowsファイアウォールの設定

さらに，ホストPCのWindowsファイアウォールの設定にTCPの1883ポートの入力の許可する規則を作成します．手順を以下に示します．

- 1，「Windows Defender ファイアウォール」を起動します．
- 2，「受信の規則」-「新しい規則」と選択して

ISBN978-4-7898-4477-2
C3055 ￥2800E
CQ出版社
定価 3,080円（本体2,800円）⑩
9784789844772
1923055028005

MICRO CONTROLLER
マイコン

ラズベリー・パイ Pico/Pico W 攻略本