

micro:bitで μT-Kernel 3.0を使う 2026

2026年2月5日
パーソナルメディア株式会社

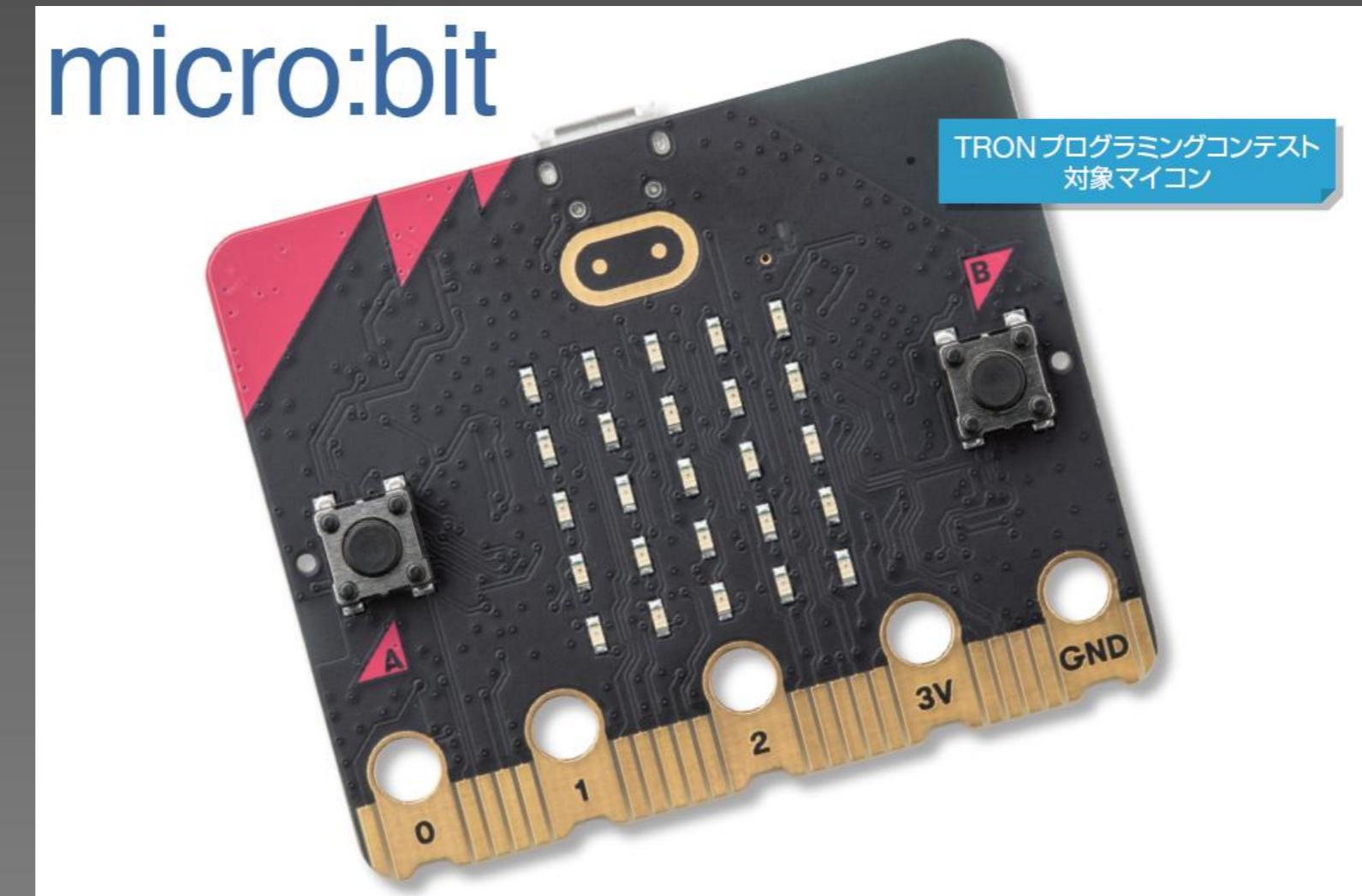
Agenda

- micro:bitとは
- micro:bitでμT-Kernel 3.0を動かす
- micro:bit用μT-Kernel 3.0の開発環境
- Eclipseを使ったプログラム開発の実際
- 周辺デバイスを使う
- コンテスト応募とAI活用のヒント
- Q and A

micro:bitとは

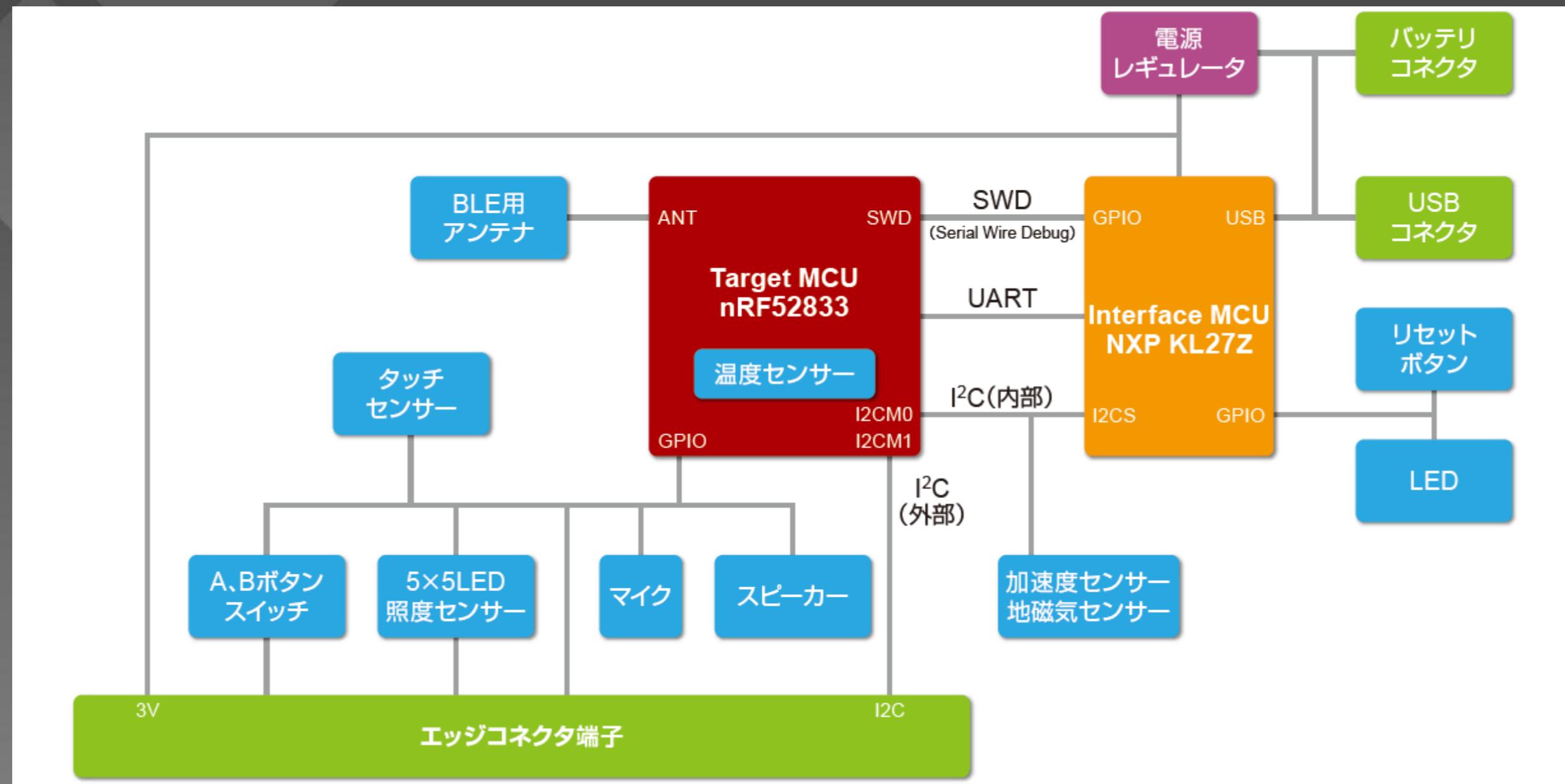
micro:bitとは

- 英国放送協会BBCが開発した超小型の教育用ボードコンピュータ
- CPUはNordicのnRF52833 (Arm Cortex-M4コア)



micro:bitとは

■ 豊富な周辺デバイスを搭載



micro:bitとは

■ 標準的にはビジュアル言語MakeCodeを使ってソフト開発



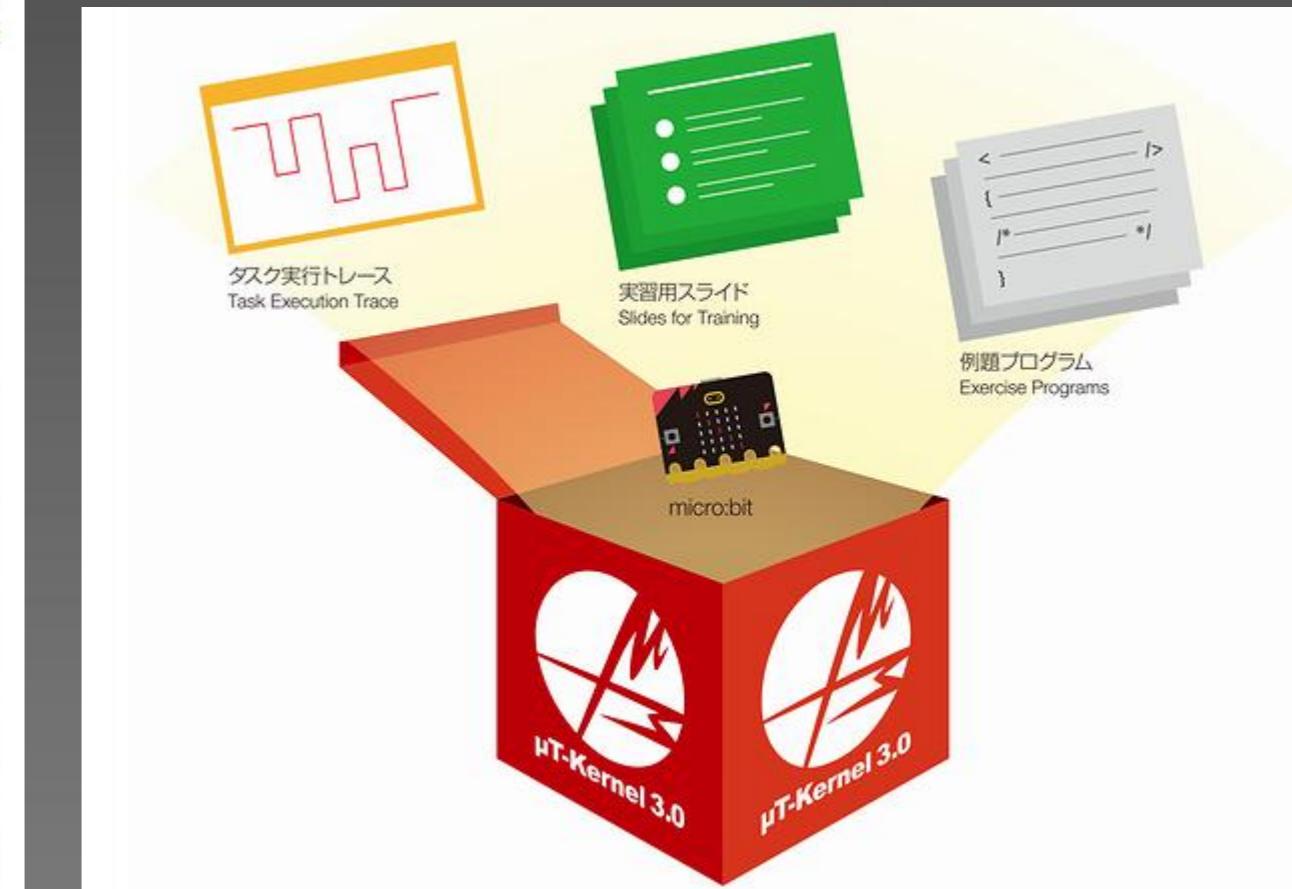
micro:bitで μT-Kernel 3.0を動かす

micro:bitでμT-Kernel 3.0を動かす

- 方法1: 「IoTエッジノード実践キット/micro:bit」を利用
- 方法2: TRONWARE記事 「micro:bitでμT-Kernel 3.0を動かそう」
- 方法3: 上記の「方法2: TRONWARE記事」をウェブサイトで閲覧
 - パーソナルメディアのサイトから公開中
 - <https://www.t-engine4u.com/info/mbit/index.html>
- 使用するμT-Kernel 3.0や開発環境、開発方法はいずれも同じ
 - トロンフォーラムのμT-Kernel 3.0をmicro:bitに移植したものを利用
 - 開発環境はEclipseを利用

micro:bitでμT-Kernel 3.0を動かす

- 方法1: 「IoTエッジノード実践キット/micro:bit」を利用
http://www.t-engine4u.com/products/ioten_prackit.html



micro:bitでμT-Kernel 3.0を動かす

■ 方法2: TRONWARE記事 「micro:bitでμT-Kernel 3.0を動かそう」

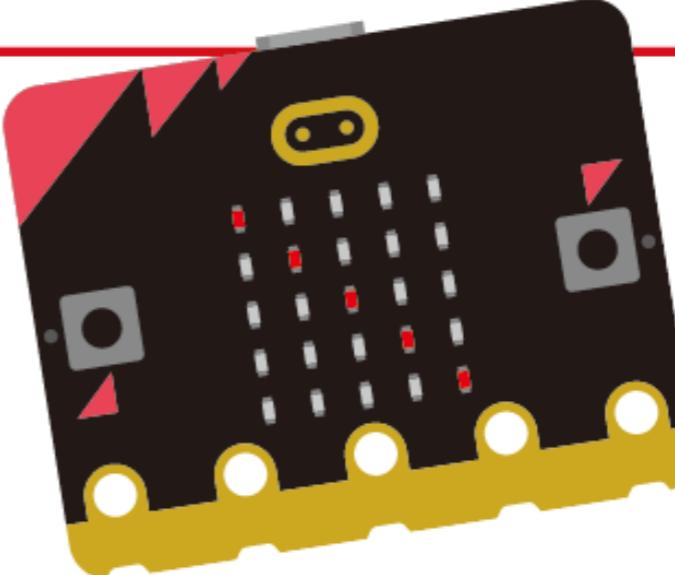
連載  micro:bit で μT-Kernel 3.0 を動かそう

micro:bit で μT-Kernel 3.0 を動かそう

[第8回] ドレミファ音階の再生とティック時間

トロンフォーラム T3 WG

本連載では、小学生向きのプログラミング教育などに使われているBBC micro:bit（以下「micro:bit」）の上で動くようになつたμT-Kernel 3.0をご紹介している。連載第8回の本号では、micro:bitから音を出してみよう。μT-Kernel 3.0の持つ時間関連の機能をいろいろ試しながら、ドレミファの音階を再生するところまで進めていく。



micro:bitでμT-Kernel 3.0を動かす

方法3: Web版 「micro:bitでμT-Kernel 3.0を動かそう」

micro:bitで μT-Kernel 3.0を動かそう

本ページは、TRON & オープン技術情報マガジン「TRONWARE」の連載記事「micro:bitでμT-Kernel 3.0を動かそう」をWebで公開したもので、小学生向きのプログラミング教育などに使われているBBC micro:bitの上で動くようになったμT-Kernel 3.0の使い方や、周辺デバイスの制御方法などをご紹介しています。

目次

- [第1回] micro:bitの概要
- [第2回] 開発ツールの準備とμT-Kernel 3.0のコンパイル
- [第3回] μT-Kernel 3.0の実行
- [第4回] マルチタスクの動作確認
- [第5回] ボタンスイッチ入力
- [第6回] 割込みを使ってみよう
- [第7回] LEDのダイナミック点灯

[第7回] LEDのダイナミック点灯

本連載では、小学生向きのプログラミング教育などに使われているBBC micro:bit（以下「micro:bit」）の上で動くようになったμT-Kernel 3.0をご紹介している。連載第7回の本号では、micro:bitのボード上についているLEDを点滅したり、25個のLEDを制御したりしてドット文字の表示を試みる。「ダイナミック点灯」とよばれる方法で多数のLEDを制御するが、そのためにμT-Kernel 3.0の物理タイマの機能を使う。LED関連のハードウェア技術情報と合わせて理解しよう。

LEDのハードウェア

micro:bitの回路図^{(*)1}のLEDの部分を図1に示す。25個のLEDが、ROW1..ROW5 の5本の線とCOL1..COL5の5本の線との間のすべての組み合わせになる場所に、マトリックス（行列）状に配置されており、LEDマトリックスとよばれている。

まずは、左上のD2と書かれたLEDを一つ点滅させてみよう。このLEDを点灯させるためには、LEDの両端に電圧をかけて電流を流せばよい。LEDもダイオードの一種なので、電流の流れる向きは一方向に決まっており、この回路図の左上から右下の方向にのみ流れ。つまり、D2の左上の電圧を高く、右下の電圧を低くすれば、このLEDが点灯する。D2の左上側はROW1という線につながり、D2の右下側はCOL1という線につながっているので、ROW1を電圧の高いレベルに、COL1を電圧の低いレベルにすればよい。

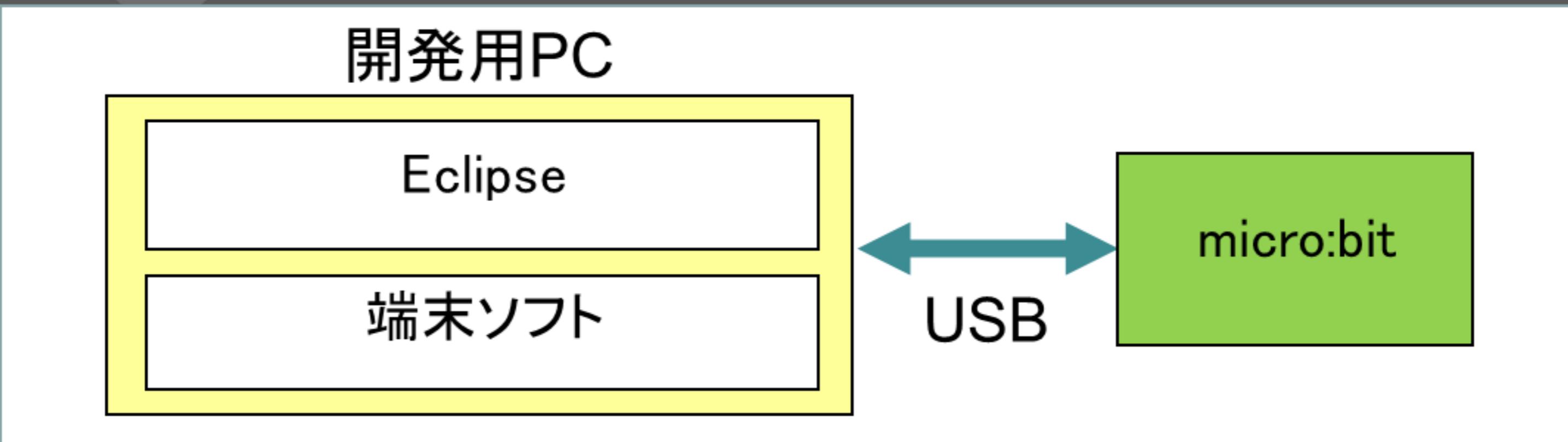
ROW1..ROW5とCOL1..COL5の10本の線は、すべてmicro:bitのGPIOに接続されており、その対応関係は表1のとおりである。この情報は、micro:bitの回路図^{(*)1}のTarget MCUの部分や、V2 pinmapの表^{(*)2}から得られる。表1より、ROW1に接続されているpinは、ROW1..ROW5に接続されるpinと並んで、D2..D10に接続されるpinである。

micro:bit用 μT-Kernel 3.0の開発環境

micro:bit用μT-Kernel 3.0の開発環境

■ 統合開発環境Eclipseを使ったクロス開発

- プログラムの作成、コンパイル、ビルド、実行、デバッグが可能
- USBシリアル経由で開発用ホストの端末コンソールとの文字入出力が可能



micro:bit用μT-Kernel 3.0の開発環境

- アプリ開発にはμT-Kernel 3.0の開発言語であるC言語を利用
 - MakeCodeは使えない
- 周辺デバイスのハードウェア制御は自分自身で行う
 - 自分でドライバを開発するか、アプリから周辺デバイスのハードウェアを直接制御する
 - 以下のデバイスは、TRONWARE連載記事を参考にプログラミング可能
 - ✓ LEDマトリックス、ボタンスイッチ、
 - ✓ スピーカー、PWM、A/D変換、
 - ✓ I2C、加速度センサー
 - ✓ 2台のmicro:bit同士でシリアル通信

Eclipseを使った プログラム開発の実際

Eclipseを使ったプログラム開発の実際

■ [Step-1] μT-Kernel 3.0の開発用ツールの準備

- (1) Eclipseのインストール
- (2) GNU Arm Embedded Toolchainのインストール
- (3) xPack Windows Build Toolsのインストール
- (4) PythonとpyOCDのインストール

■ [Step-2] μT-Kernel 3.0のインポートとコンパイル

- (5) micro:bit用μT-Kernel 3.0のソースコード入手と展開
- (6) EclipseにμT-Kernel 3.0のプロジェクトを作成
- (7) Eclipseの操作によるコンパイルとビルド

Eclipseを使ったプログラム開発の実際

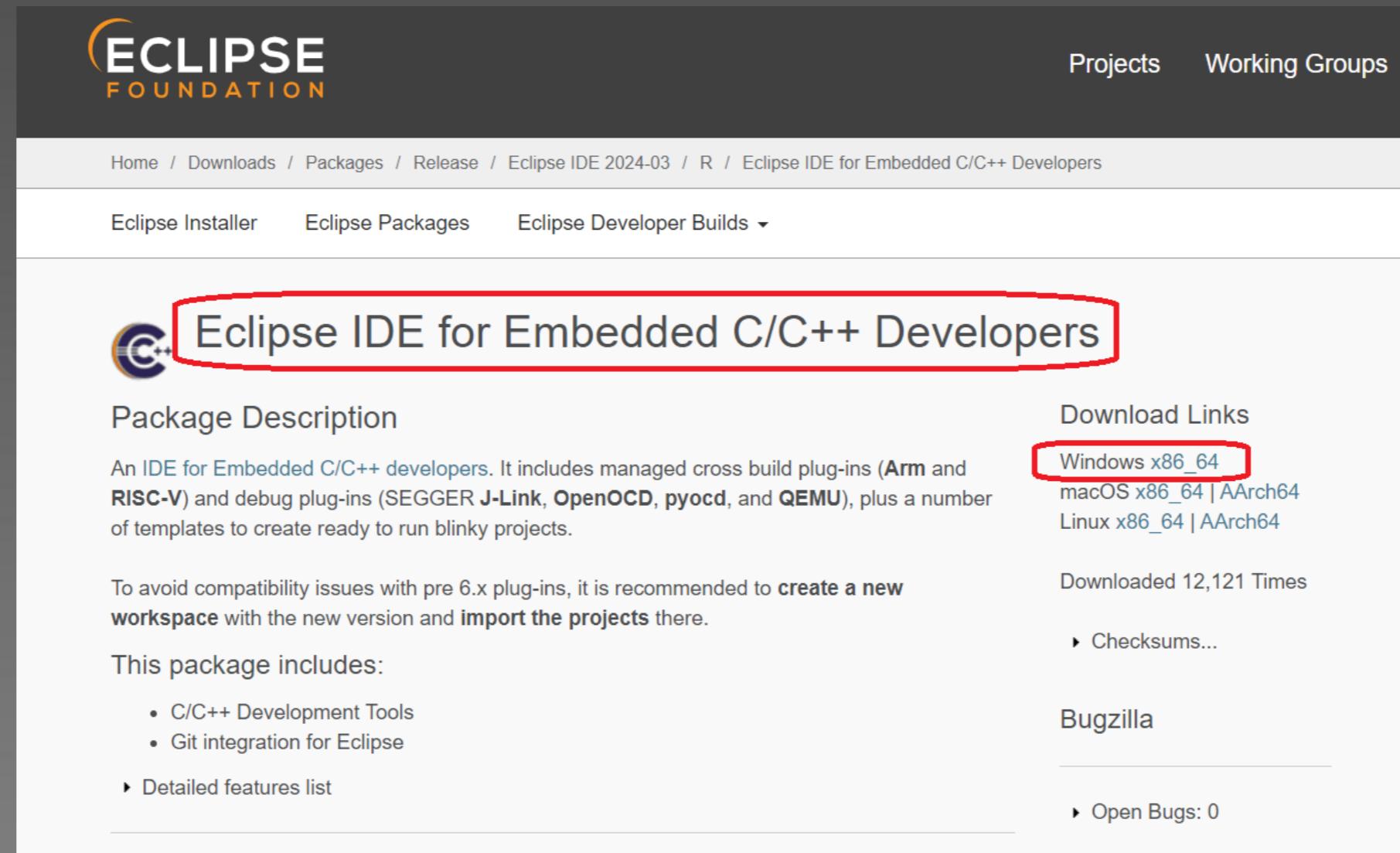
■ 使用する開発用ツールとソースコード

名称	説明	URL
Eclipse IDE for Embedded C/C++ Developers	GUI画面からソースコードの作成や編集、コンパイル、デバッグなどを行う統合開発環境	https://www.eclipse.org/downloads/packages/release/2024-03/r/eclipse-ide-embedded-cc-developers
GNU Arm Embedded Toolchain	Armマイコンを使った組込み開発用のGNU Cコンパイラおよび関連ツール一式	https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-rm/downloads
xPack Windows Build Tools	プログラムの構築（ビルド）のためのWindows用のツール一式	https://github.com/xpack-dev-tools/windows-build-tools-xpack/releases
Python	プログラミング言語Pythonの言語処理系	https://www.python.org/downloads/
pyOCD	Python上で動作するArmマイコン用のデバッグツール	(Pythonからインストール)
micro:bit用μT-Kernel 3.0	ソースコード一式（パスワード付きZIPファイル）	https://www.personal-media.co.jp/book/tw/tw_index/362.html

Eclipseを使ったプログラム開発の実際

■ (1) Eclipseのインストール

- GUI画面からソースコードの作成や編集、コンパイル、デバッグなどを行う統合開発環境



Eclipseを使ったプログラム開発の実際

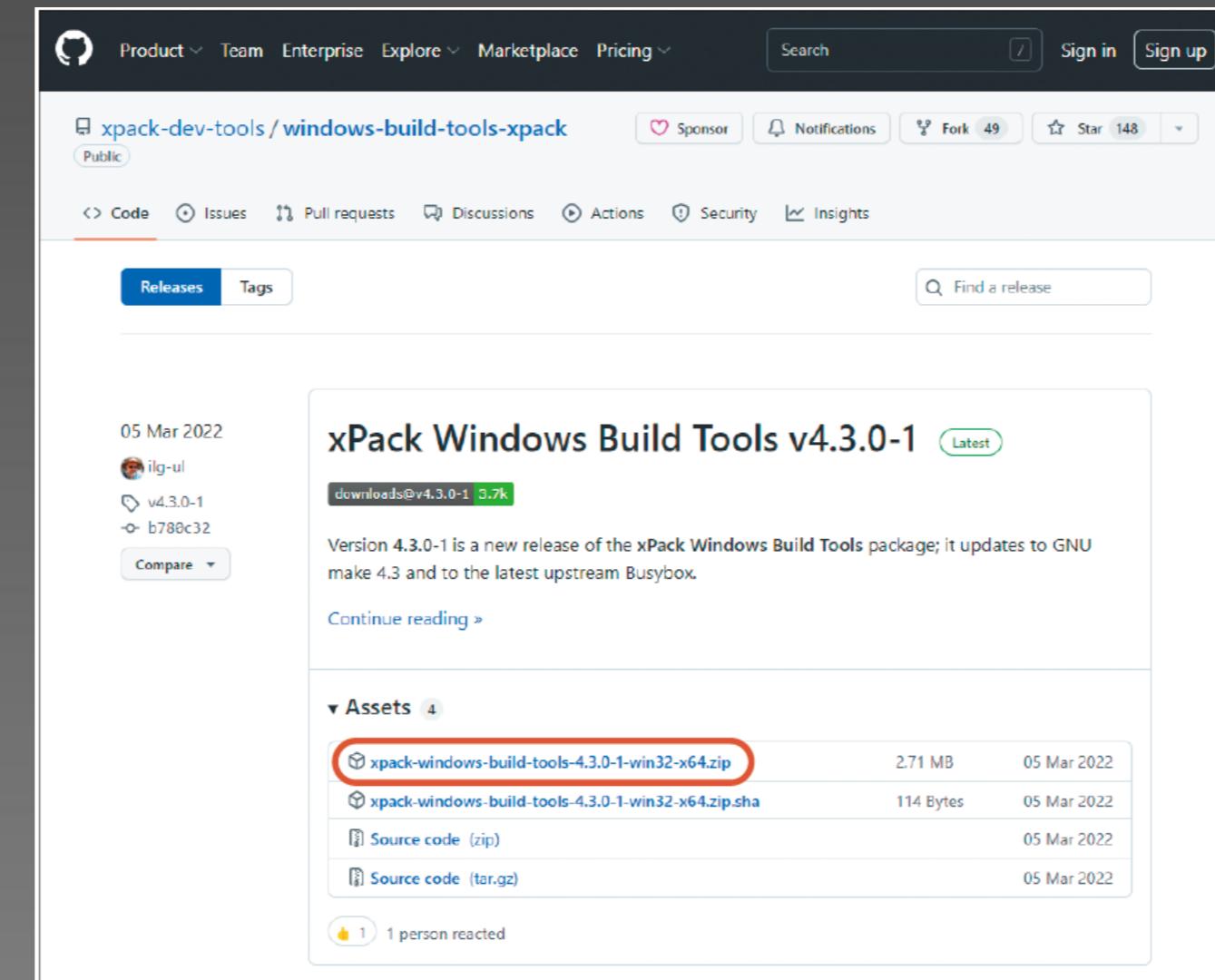
- (2) GNU Arm Embedded Toolchainのインストール
 - Armマイコンを使った組込み開発用のGNU Cコンパイラおよび関連ツール一式

The screenshot shows the "What's new in 10.3-2021.10" section of the Arm Developer website. It highlights a fix for the VLLDM instruction security vulnerability. The page lists three download options:

- 1 [gcc-arm-none-eabi-10.3-2021.10-win32.exe](#)
Windows 32-bit Installer (Signed for Windows 10 and later) (Formerly SHA2 signed binary)
MD5: 8d0f75f33f9e3d5f9600197626297212
- 2 [gcc-arm-none-eabi-10.3-2021.10-win32.zip](#)
Windows 32-bit ZIP package
MD5: 2bc8f0c4c4659f8259c8176223eeafc1
- 3 [gcc-arm-none-eabi-10.3-2021.10-x86_64-linux.tar.bz2](#)
Linux x86_64 Tarball
MD5: 2383e4eb4ea23f248d33adc70dc3227e

Eclipseを使ったプログラム開発の実際

- (3) xPack Windows Build Toolsのインストール
 - プログラムの構築（ビルド）のためのWindows用のツール一式



Eclipseを使ったプログラム開発の実際

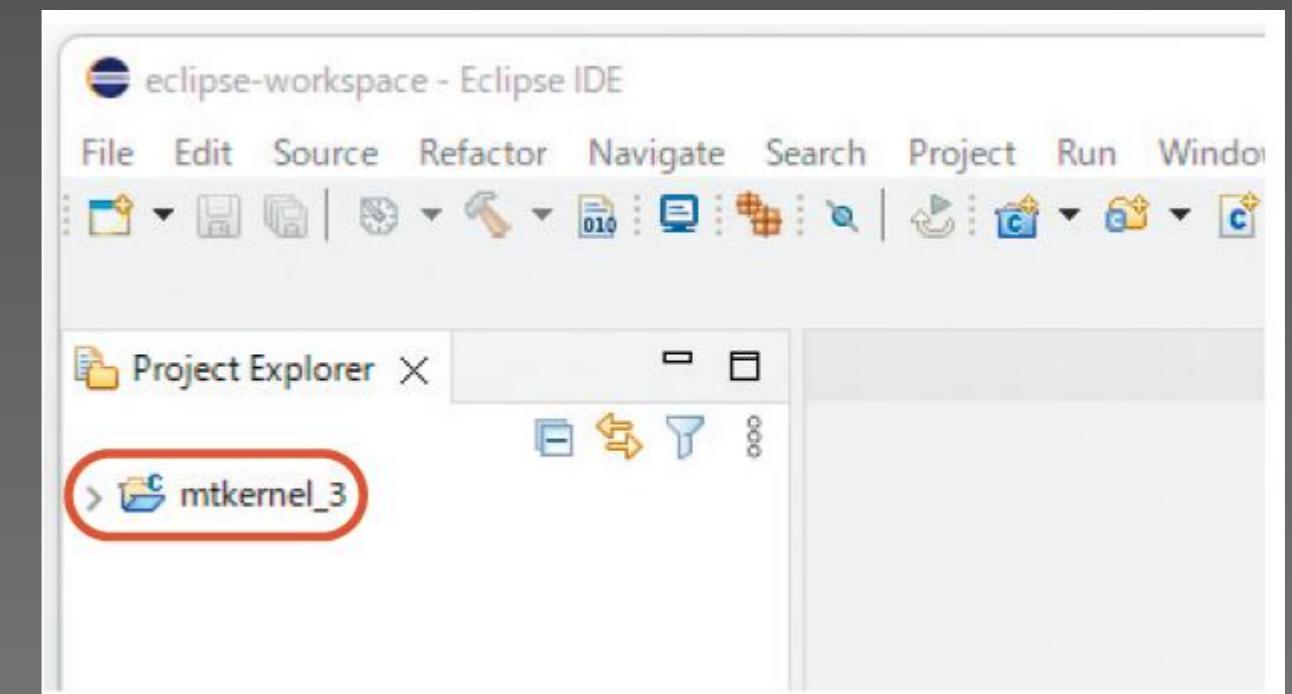
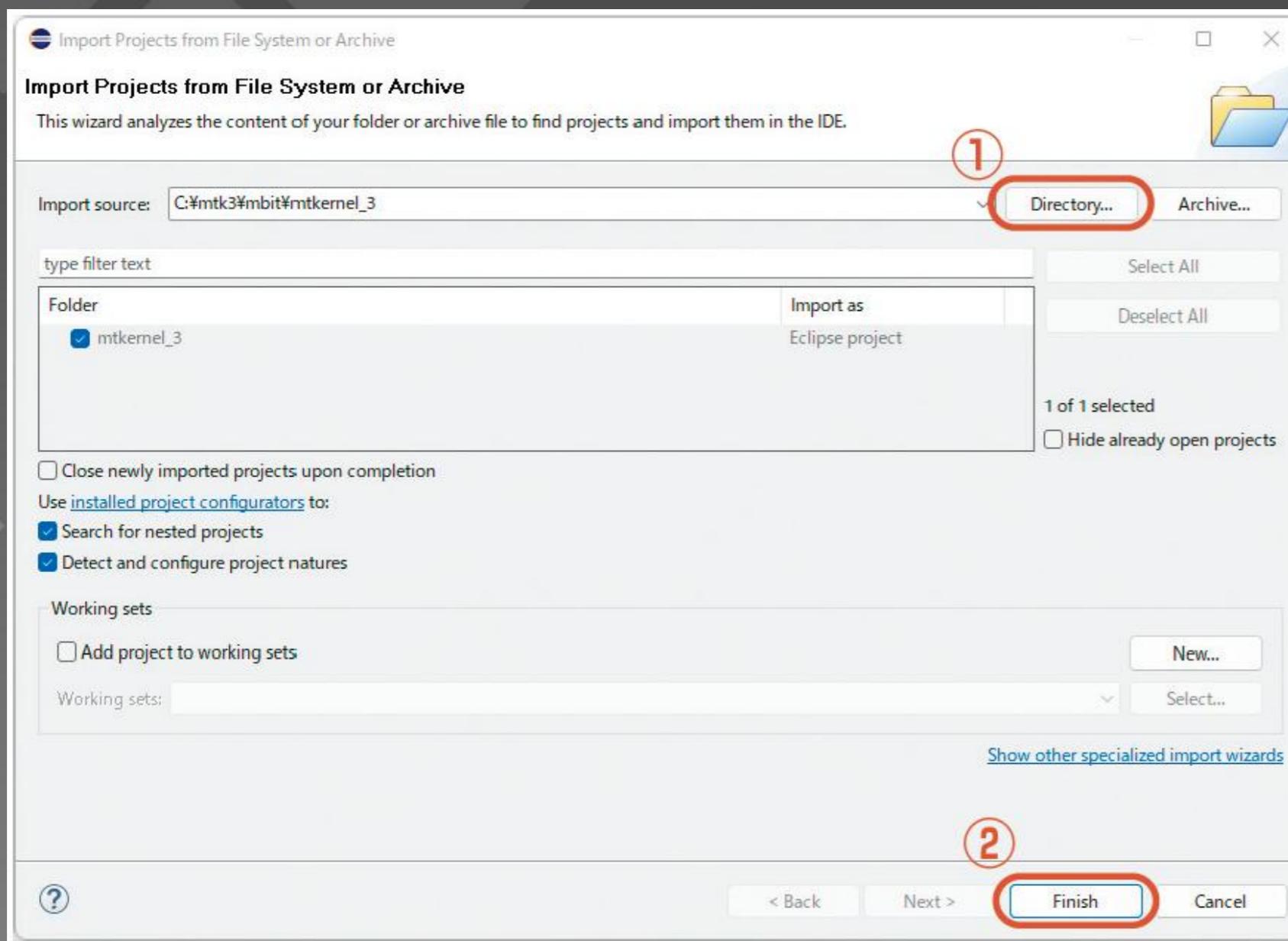
■ (4) PythonとpyOCDのインストール

- Python上で動作するArmマイコン用のデバッグツール
- Pythonを入れた後に “python -m pip install -U pyocd” を実行



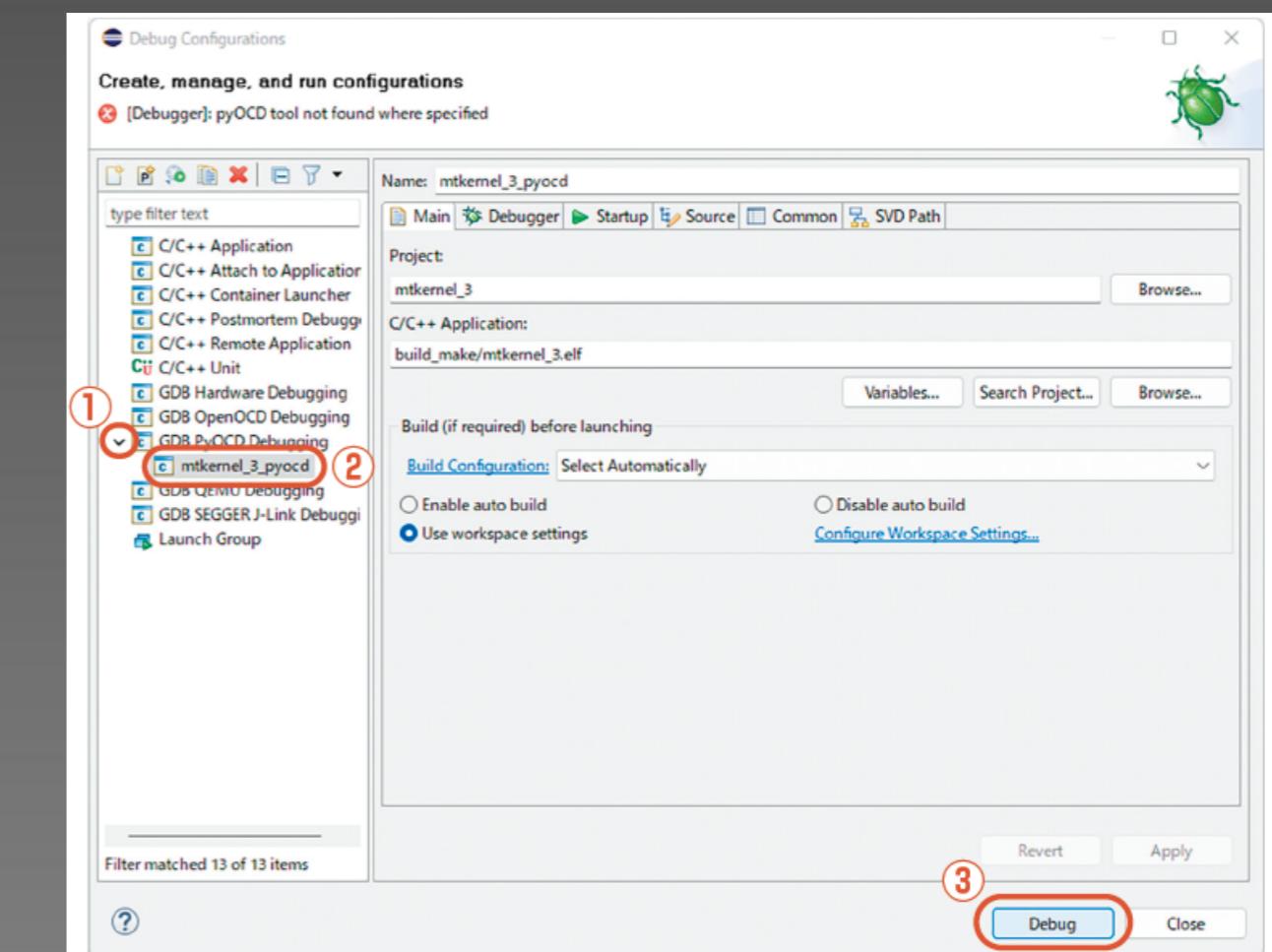
Eclipseを使ったプログラム開発の実際

■ [Step-2] μT-Kernel 3.0のインポートとコンパイル



Eclipseを使ったプログラム開発の実際

- [Step-3] micro:bit実機へのμT-Kernel 3.0の転送と実行
 - (8) micro:bitのFlash ROMの消去
 - (9) 端末ソフトの起動とmicro:bit実機への接続
 - (10) デバッグ実行用のEclipseの設定
 - (11) micro:bit実機での実行



Eclipseを使ったプログラム開発の実際

■ usermain() の先頭のブレークポイントでプログラムの実行が停止

eclipse-workspace - mtkernel_3/app_sample/app_main.c - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Debug X Project Expl... □

mtkernel_3_pyocd [GDB PyOCD Debug] □
mtkernel_3.elf
Thread #1 1 (Suspended : Breakpoint)
usermain() at app_main.c:86 0x
init_task_main() at inittask.c:12
0x0
pyocd.exe
arm-none-eabi-gdb
Semihosting

app_main.c X

```
80
81 EXPORT INT usermain( void )
82 {
83     T_RVER rver;
84     ID id1, id2;
85
86     TM_PUTSTRING( (UB*) "Start User-main program.\n" )  

87
88     tk_ref_ver(&rver); /* Get the OS Version.
89
90 #if USE_TMONITOR
91     tm_printf((UB*)"Make Code: %04x Product ID: %0
92     tm_printf((UB*)"Product Ver. %04x\nProduct Num.
93                         rver.prver, rver.prno[0],rver.prno[1],r
94 #endif
95
96     id1 = tk_cre_tsk(&ctsk1);
97     tk_sta_tsk(id1, 0);
```

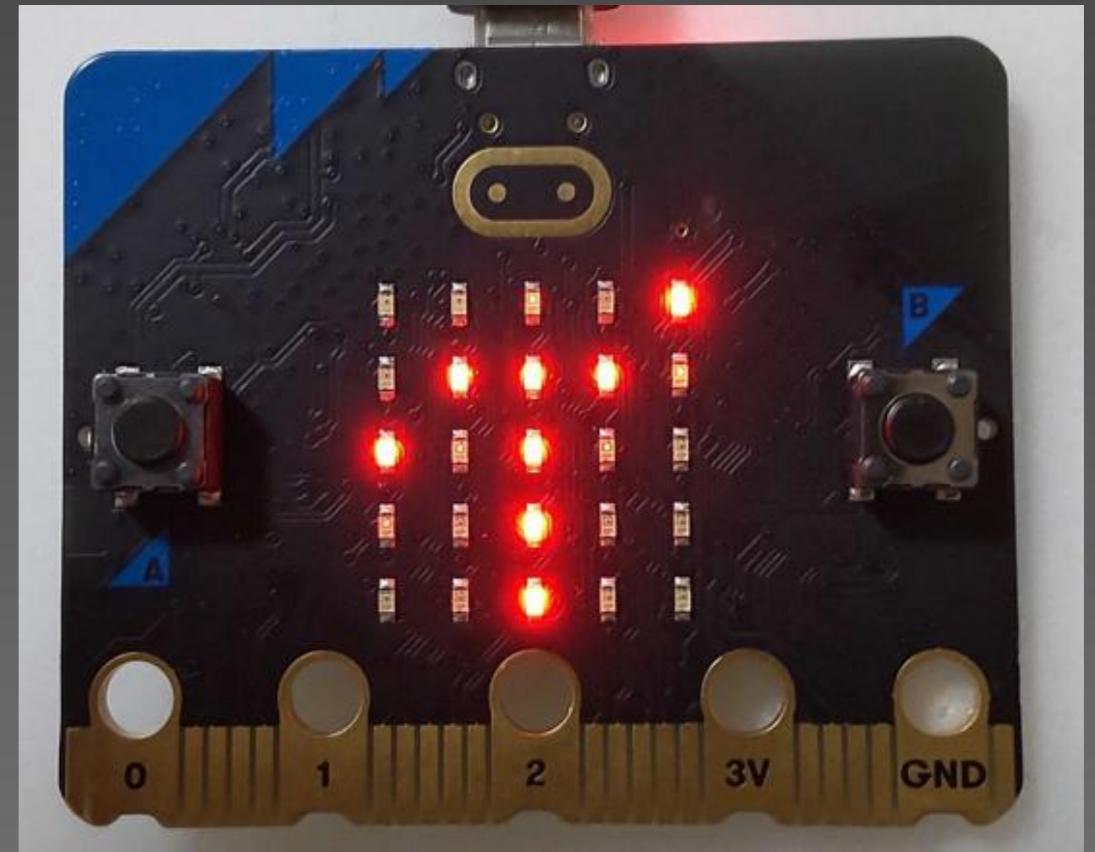
(x)= V X B E P □

Name	Type	Value
(x)= rver	T_RVER	{...}
(x)= id1	ID	41837
(x)= id2	ID	536881808

周辺デバイスを使う

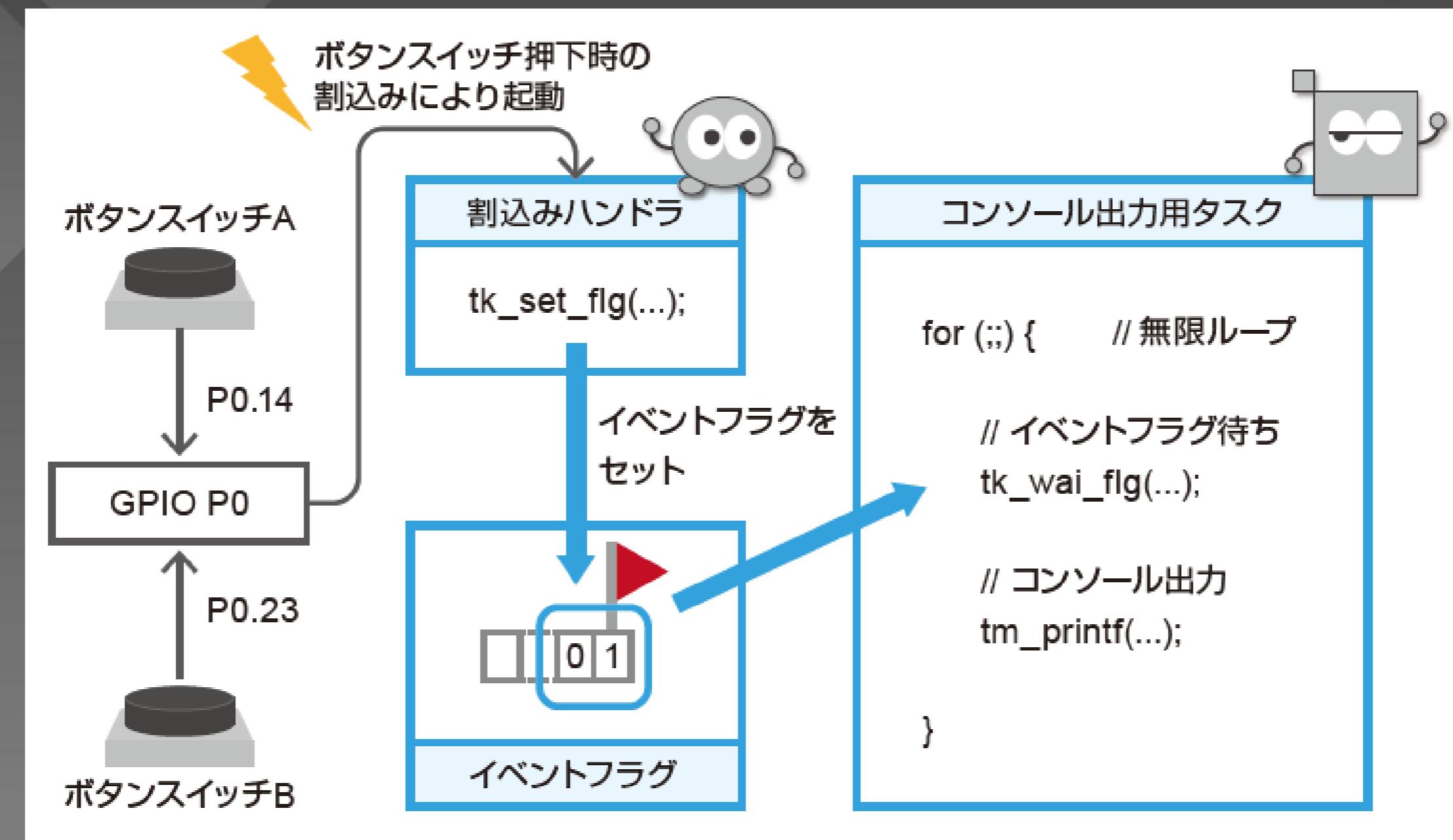
周辺デバイスを使う

- (1)GPIOとボタンスイッチ割込み
- (2)LEDマトリックスに「イ」を表示
- (3)PWMによる音階再生とLEDの調光制御
- (4)A/D変換とジョイスティック
- (5)I2Cと加速度センサー
- (6)別のmicro:bitとシリアル接続



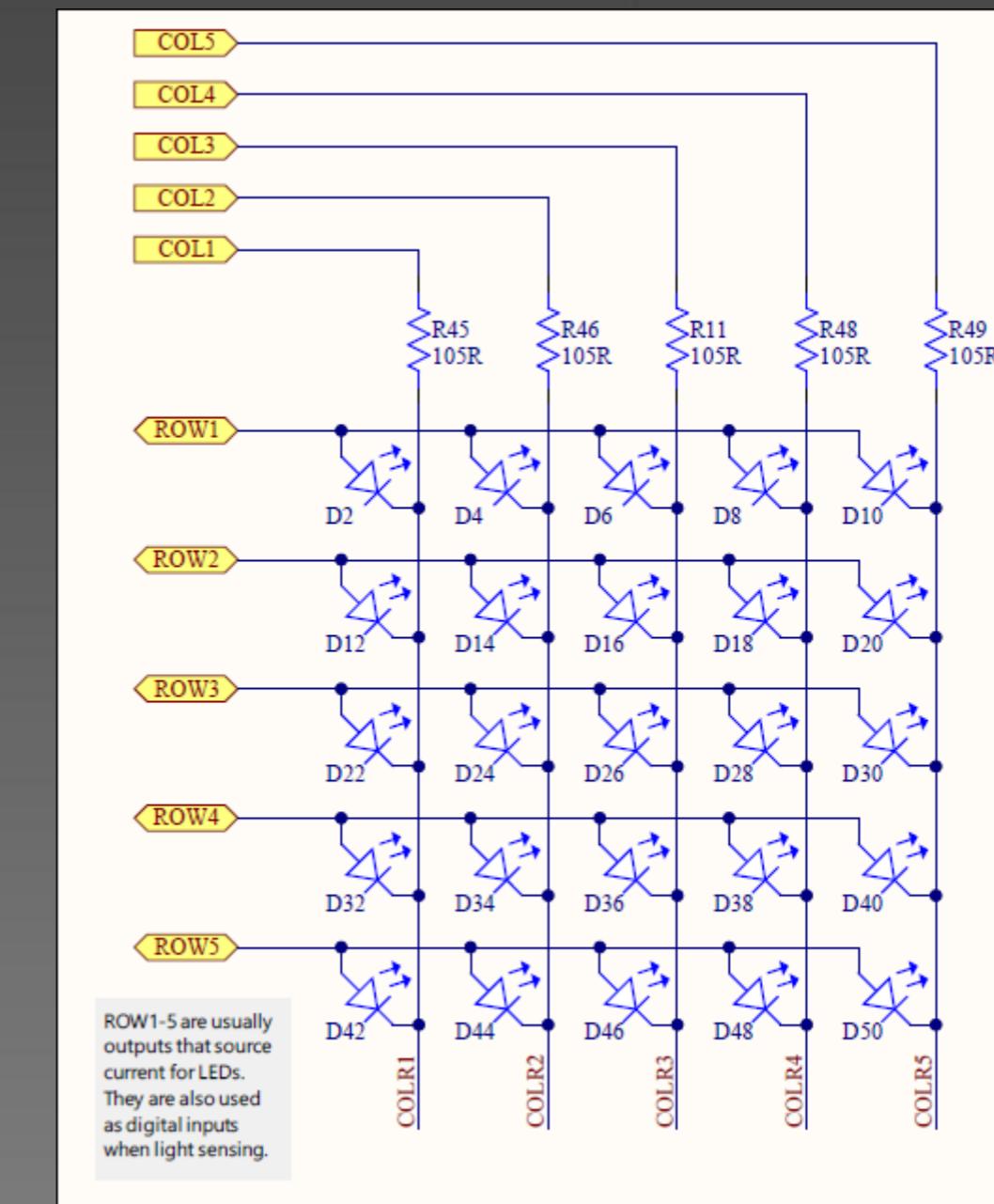
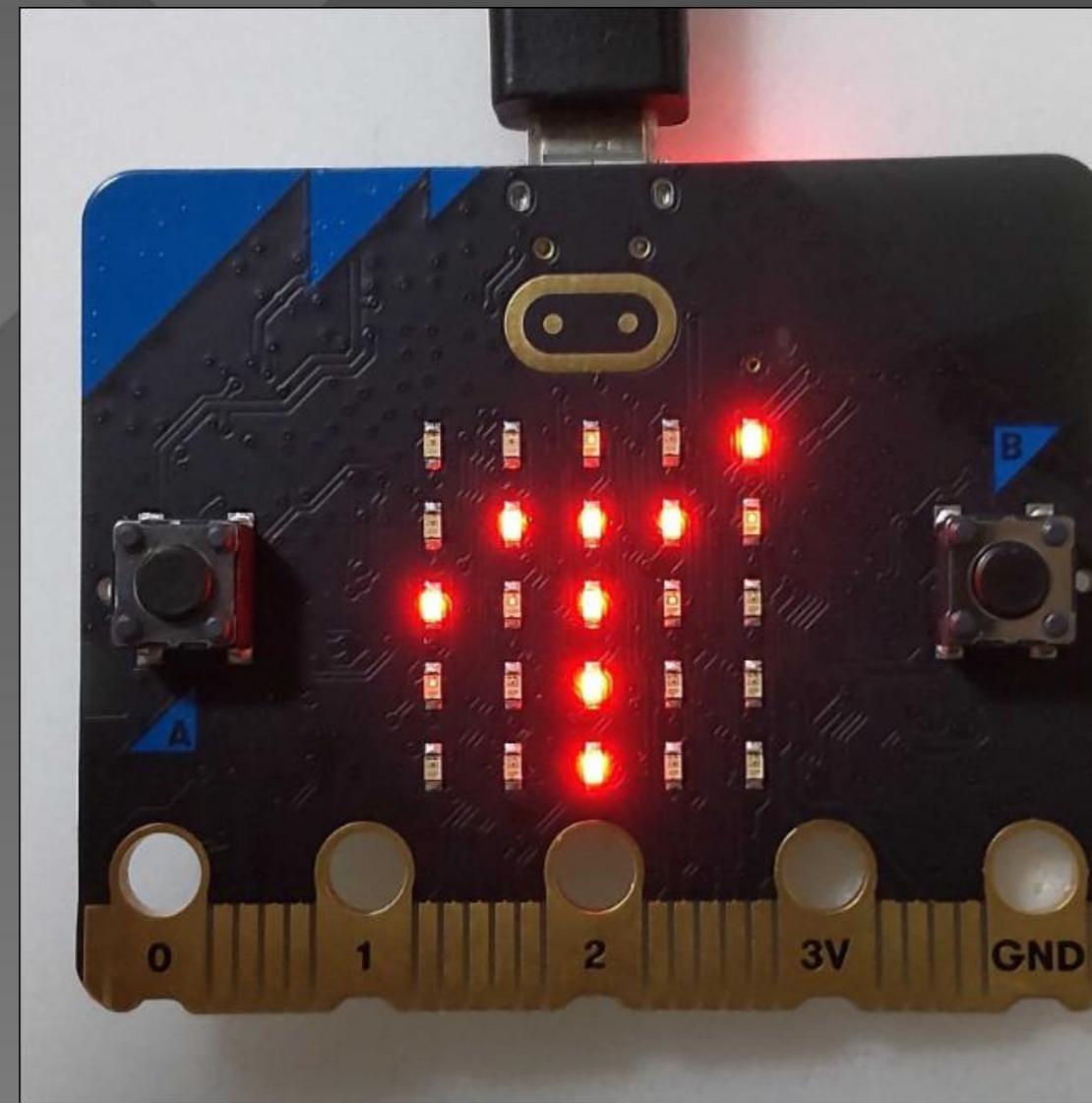
周辺デバイスを使う

■ (1)GPIOとボタンスイッチ割込み



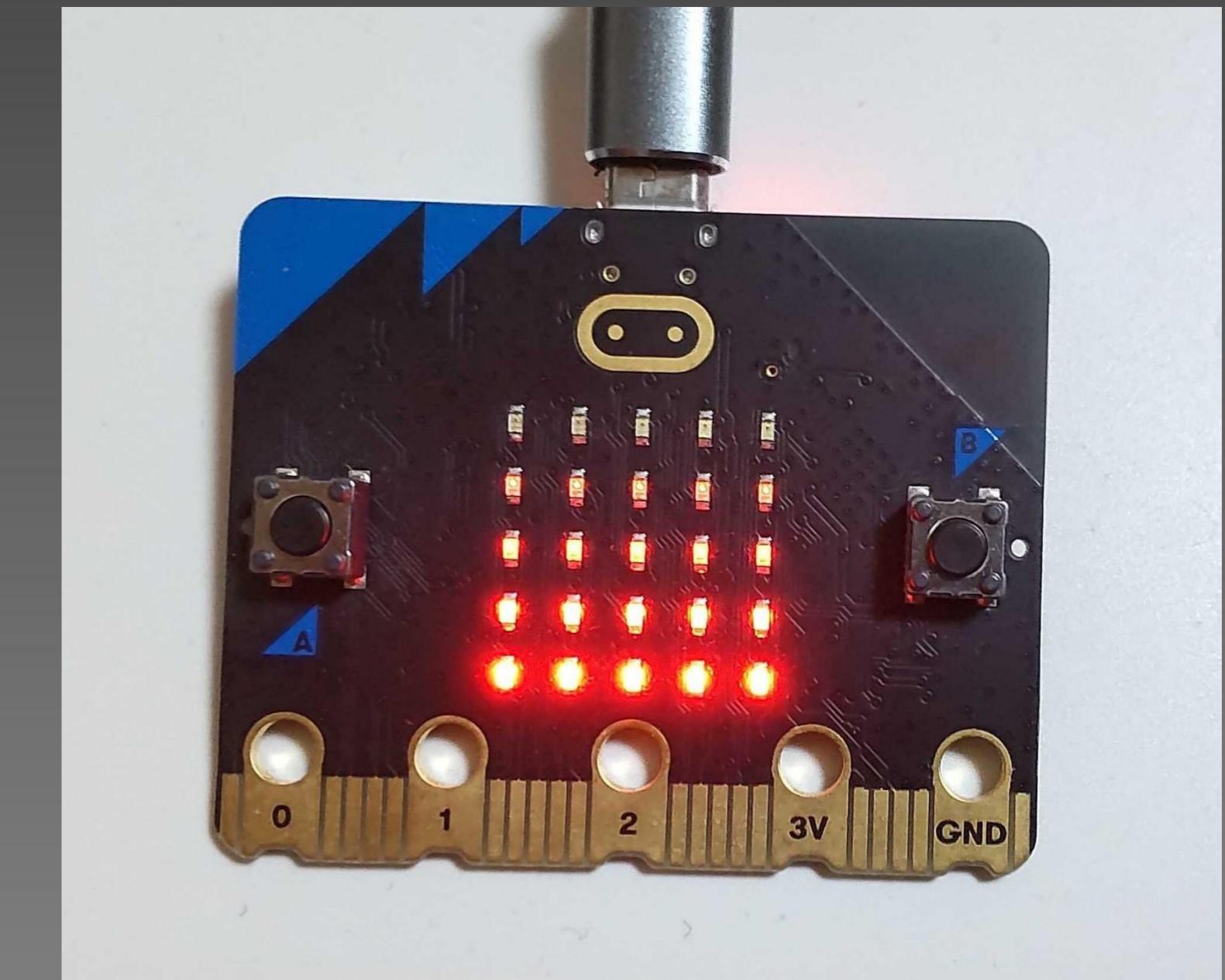
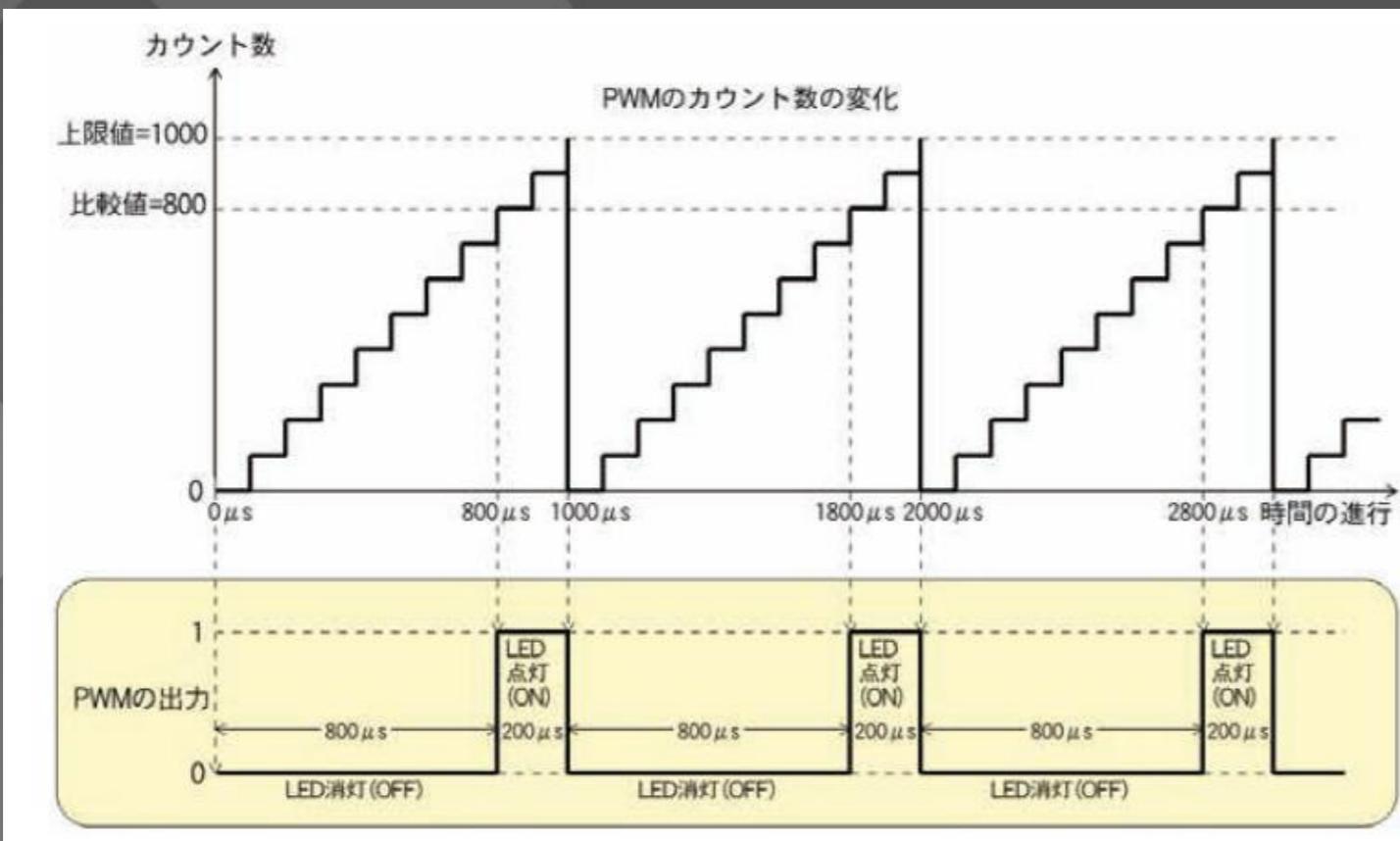
周辺デバイスを使う

■ (2) LEDマトリックスに「イ」を表示



周辺デバイスを使う

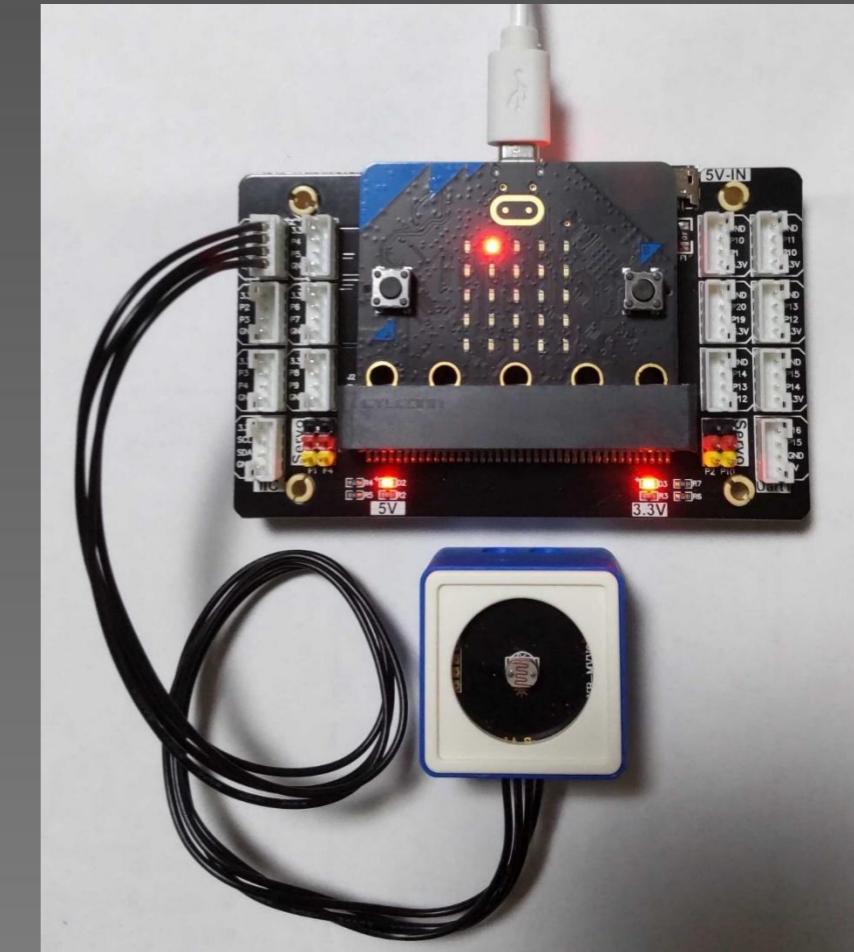
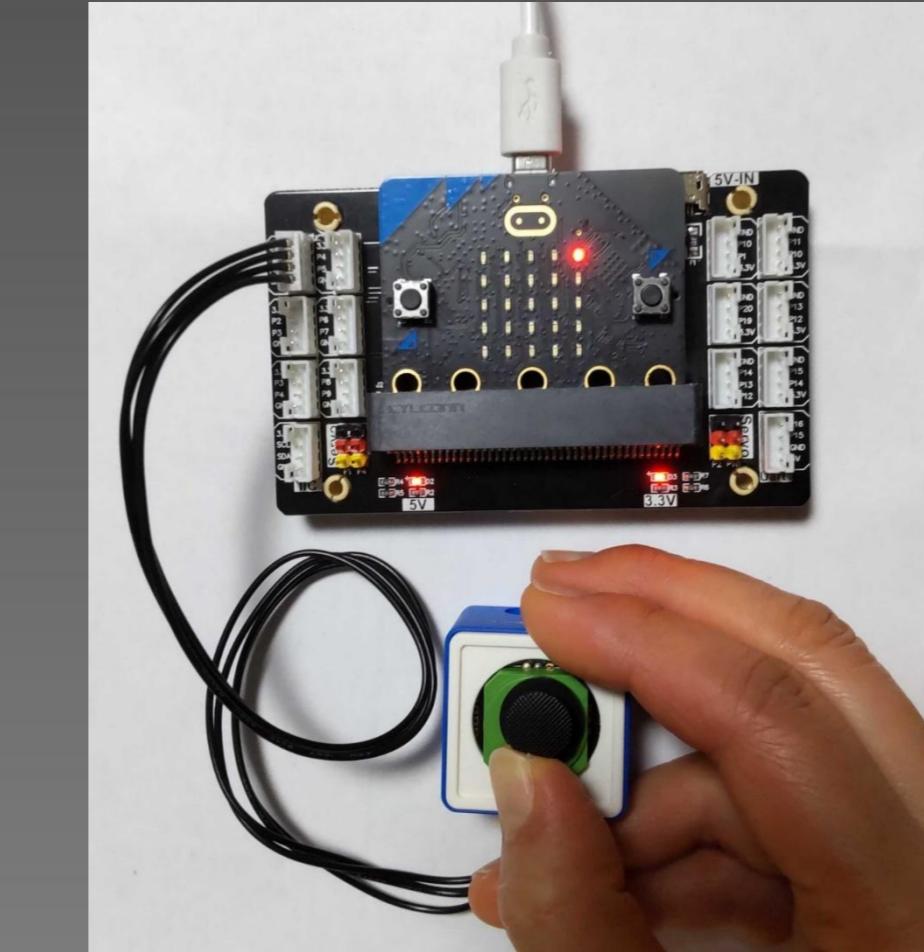
■ (3) PWMによる音階再生とLEDの調光制御



周辺デバイスを使う

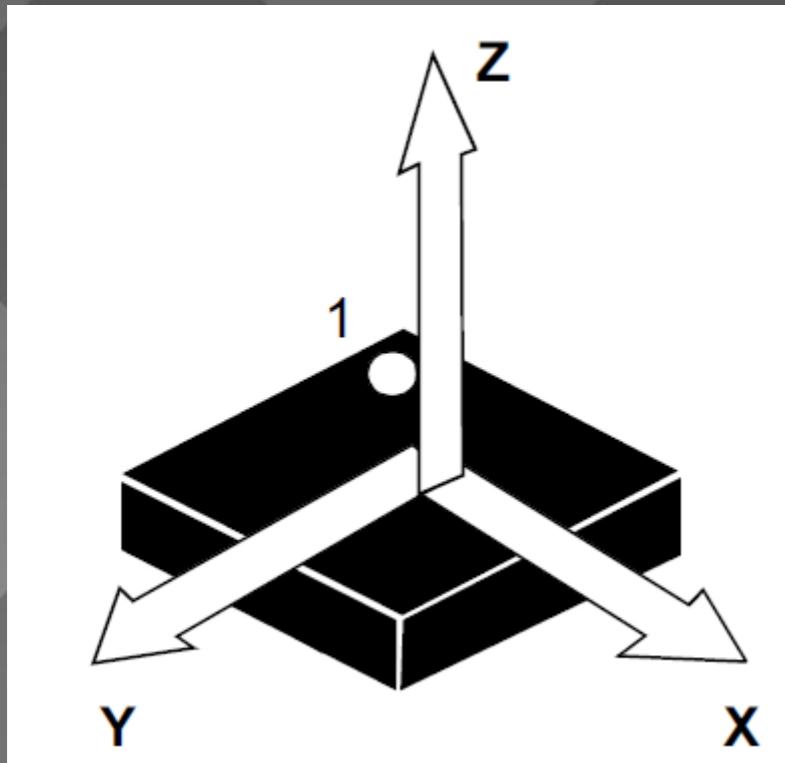
■ (4)A/D変換とジョイスティック

- イフティニーストアの「ワールド オブ モジュール センサー キット」を利用
 - ✓ <https://store.iftiny.com/products/yahboom-world-of-module-programmable-sensor-kit-for-microbit-v2>



周辺デバイスを使う

■ (5)I2Cと加速度センサー



microT-Kernel Version 3.00

WHO_AM_I_A = 0x33

CTRL_REG1_A = 0x57

Acc: x,y,z= -10, -6, 237

Acc: x,y,z= -8, -4, 247

Acc: x,y,z= -14, -11, 240

Acc: x,y,z= 36, 2, 241

Acc: x,y,z= 97, 3, 224

Acc: x,y,z= 127, 17, 208

Acc: x,y,z= 144, 6, 201

Acc: x,y,z= 11, 12, 259

Acc: x,y,z=-112, -5, 214

Acc: x,y,z=-165, 2, 192

← ボードが水平(micro:bitの文字が上)

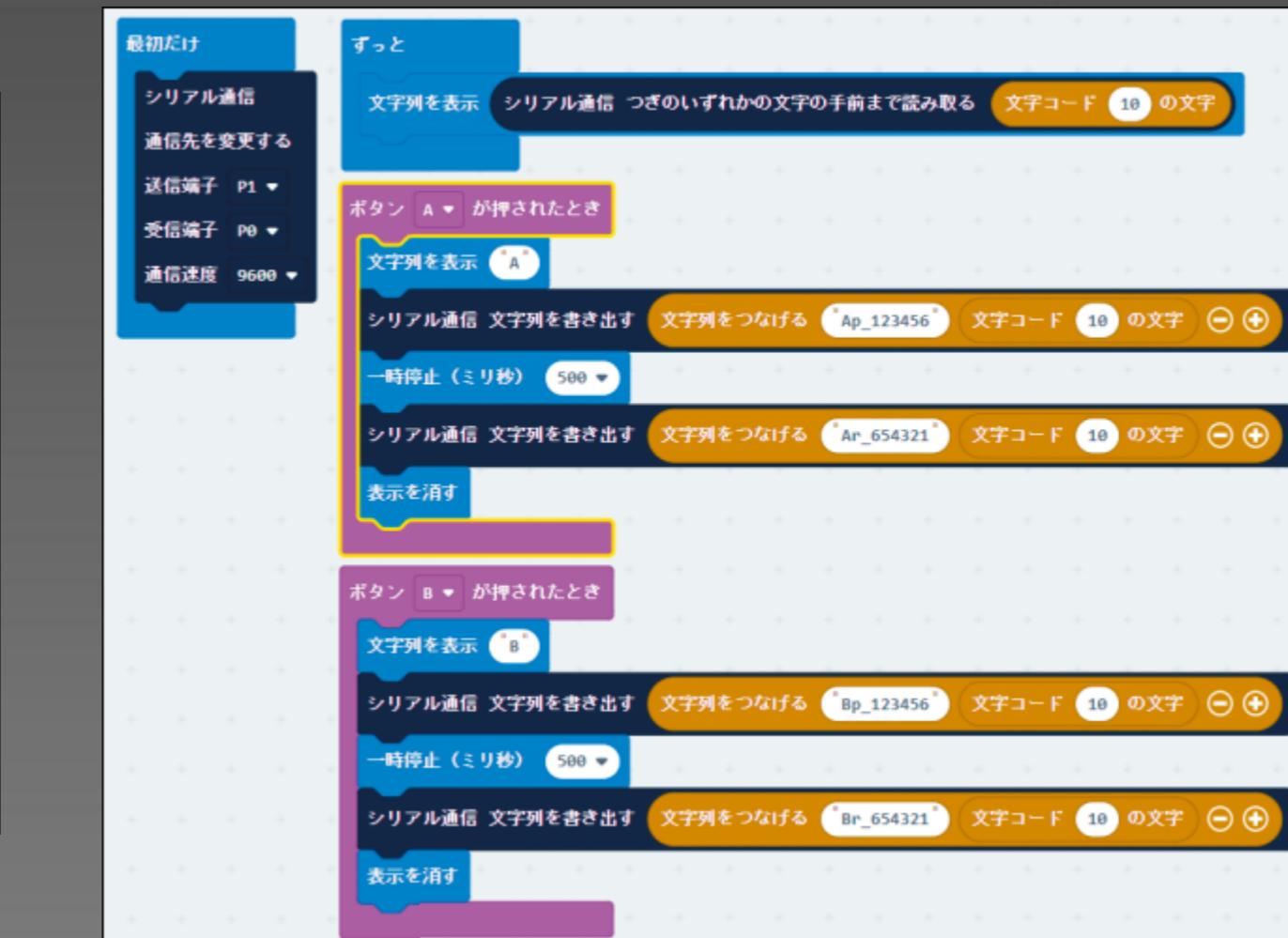
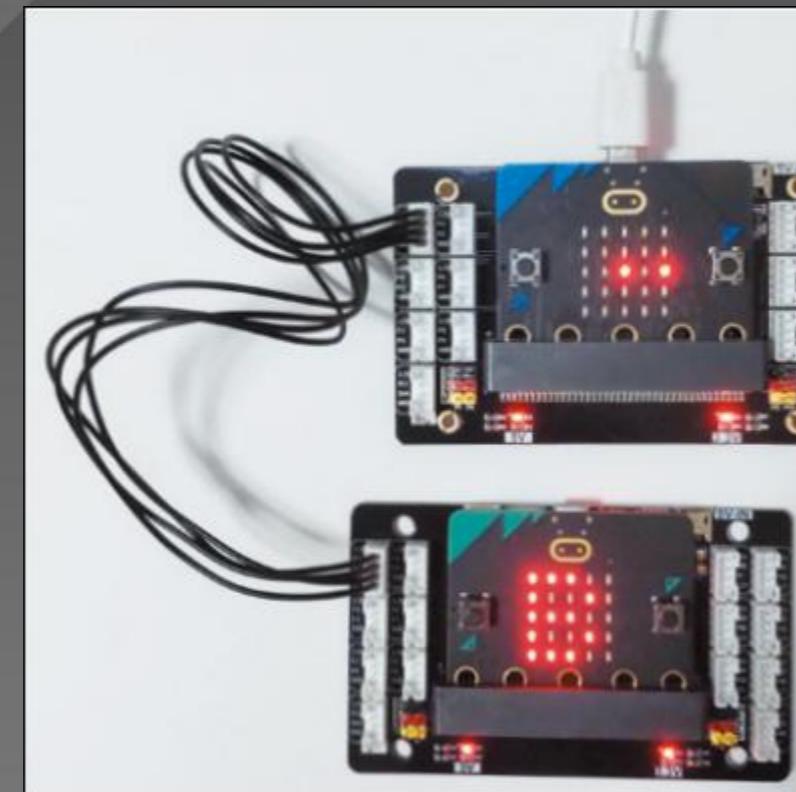
← ボードの左側を下に傾ける

← ボードの右側を下に傾ける

周辺デバイスを使う

■ (6)別のmicro:bitとシリアル接続

- もう1台のmicro:bitではMakeCodeを使うことも可能



周辺デバイスを使う

- 紹介したサンプルプログラムはTRONWAREの連載記事で掲載、Web版もあり
 - (1)GPIOとボタンスイッチ割込み (VOL.201)
 - <https://www.t-engine4u.com/info/mbit/5.html>, <https://www.t-engine4u.com/info/mbit/6.html>
 - (2)LEDマトリックスに「イ」を表示 (VOL.202)
 - <https://www.t-engine4u.com/info/mbit/7.html>
 - (3)PWMによる音階再生とLEDの調光制御 (VOL.205)
 - <https://www.t-engine4u.com/info/mbit/10.html>
 - (4)A/D変換とジョイスティック (VOL.206)
 - <https://www.t-engine4u.com/info/mbit/11.html>
 - (5)I2Cと加速度センサー (VOL.207)
 - <https://www.t-engine4u.com/info/mbit/12.html>
 - (6)別のmicro:bitとシリアル接続 (VOL.208)
 - <https://www.t-engine4u.com/info/mbit/13.html>

micro:bit + μT-Kernelのプログラム開発方法：番外編

- 生成AIに適切なプロンプトを与えると、ある程度のプログラムはできる
 - プロンプト例
 - micro:bitで動くμT-Kernel 3.0を使って、LEDマトリックスに「イ」を表示するC言語のプログラムを書いてください。
 - TRONWARE VOL.217 (最新号) で紹介



2025 TRON Symposium —TRONSHOW—

- 最初の部分はまったく正しい。GPIOのアドレス定義も正しい。
- nRF51822はmicro:bit V1.5のCPUであり、現在入手できるV2.2ではnRF52833。nRF51822はmicro:bit V1.5のCPUであり、現在入手でき

```

/*
 * micro:BIT LED表示プログラム - カタカナ"の表示
 * μL-Kernel 3.0対応
 */

#include <tk/tkernel.h>
#include <tmb/tmonitor.h>
...
...
...
// GPIOレジストラのベースアドレス (mNFS1822) */
#define GPIO_BASE 0x50000000
#define GPIO_OUT (~(volatile unsigned int) (GPIO_BASE + 0x504))
#define GPIO_OUTSET (~(volatile unsigned int) (GPIO_BASE + 0x508))
#define GPIO_OUTCLR (~(volatile unsigned int) (GPIO_BASE + 0x50C))
#define GPIO_DIRSET (~(volatile unsigned int) (GPIO_BASE + 0x510))

```

- LED マトリックスのpin配置で、行(Row)は正しいが、列(Column)は間違い。
- また、COL4のみ GPIO P0ではなく P1をを使っため、GPIO アドレスの定義も別に必要。
[\(a\) microcontroller GPIOリストと GPIOアドレス](#)

```

/* 行(Row)ビン: 21, 22, 15, 24, 10 */
static const int row_pins[LED_ROWS] = {21, 22, 15, 24, 10};
/* 列(Column)ビン: 4, 5, 6, 7, 8, 9, 10, 11, 12 (5x5なので
   最初の2つを省略)
static const int col_pins[LED_COLS] = {4, 5, 6, 7, 8};
...

```

```

文字のドットパターンが不自然。特に難し
ろではないと思うが...
```

```

/*ターンデータ(5x5マトリックス) */
// パターン数: ピックマ4左側。ピックが右側 */
// パターン名: 1...5 (中央に配置)
00, // 1回目: ●●●●●
00, // 2回目: ●●●●●
00, // 3回目: ●●●●●
00, // 4回目: ●●●●●
10, // 5回目: ●●●●● (下部を広げる)
```

図3 AIの生成したC言語プログラム

- 「幻覚」のAPIと危険なコード

GPIOの設定などは正しく、一見動きそうに見えます。しかし、詳しく見ていくと問題がありました。まず、GPIOのピン番号の定義です。micro:bitのハードウェア仕様では、LEDに接続されているGPIOのピン番号が不規則に割り当てられているのですが、AIは列(Column)方向のピン番号を「4、5、6、7、8」といった連番で定義してしまいました。これではLEDは点灯しません。また、「イ」のドットパターンも正しく実現できませんでした。そこで課題を変えて、「1秒ごとに『○』と『×』を交互に表示して」と指示しま

こうすると、LEDのドットパターンを正しく定義されました。また、1行ごとに表示を切り替えるために、`kernel`の「周期ハンドラ」を使うと新しいアプローチを取ってきました。

待ち)のAPIを使っていました
周期ハンドラの中で待ち状態に
システムコールを使うことはでき
ますが、こういったリアルタイ
の基本的なルールを、AIは理解
ていませんでした

- 「○」と「×」のビットパターンは正しく定義されている

```

    /*「□」のパターンデータ (5x5マトリックス) */
    static const unsigned char ledpin_circle[LED_ROWS] = {
        0b1110, // 1行目: ●●●●
        0b0110, // 2行目: ●●●●
        0b0011, // 3行目: ●●●●
        0b0001, // 4行目: ●●●●
        0b0010, // 5行目: ●●●●
    };

    /*「×」のパターンデータ (5x5マトリックス) */
    static const unsigned char ledpin_cross[LED_ROWS] = {
        0b10001, // 1行目: ●●●●●
        0b01001, // 2行目: ●●●●●
        0b00101, // 3行目: ●●●●●
        0b00010, // 4行目: ●●●●●
        0b00101, // 5行目: ●●●●●
    };

```

Eclipseに入れると、残念ながら元のマスター

A screenshot of a Java code editor showing the file 'Main.java'. The code contains a single method definition: 'public static void main(String[] args)'. The word 'main' is highlighted in red, indicating it is the entry point of the program.

「イマ」にはDefinePhysicalTimerHandlerが必要がある
周期ハンドラーのtk_cre_cycからの類推で、
pymrやtk_sta_pmrといった架空のス
ルを呼んでいる
「は」DefinePhysicalTimerHandlerと
PhysicalTimerを使う必要がある
「ータ」の与え方も違う
「ひ」、プログラムのコンパイルやリンクが
「る」

図4 AIの生成したC言語プログラム(2)

コンテスト応募とAI活用のヒント

■ micro:bitの特性を活かす

- ボードのコンパクト性
- 電池駆動の携帯性
- 豊富なセンサー類

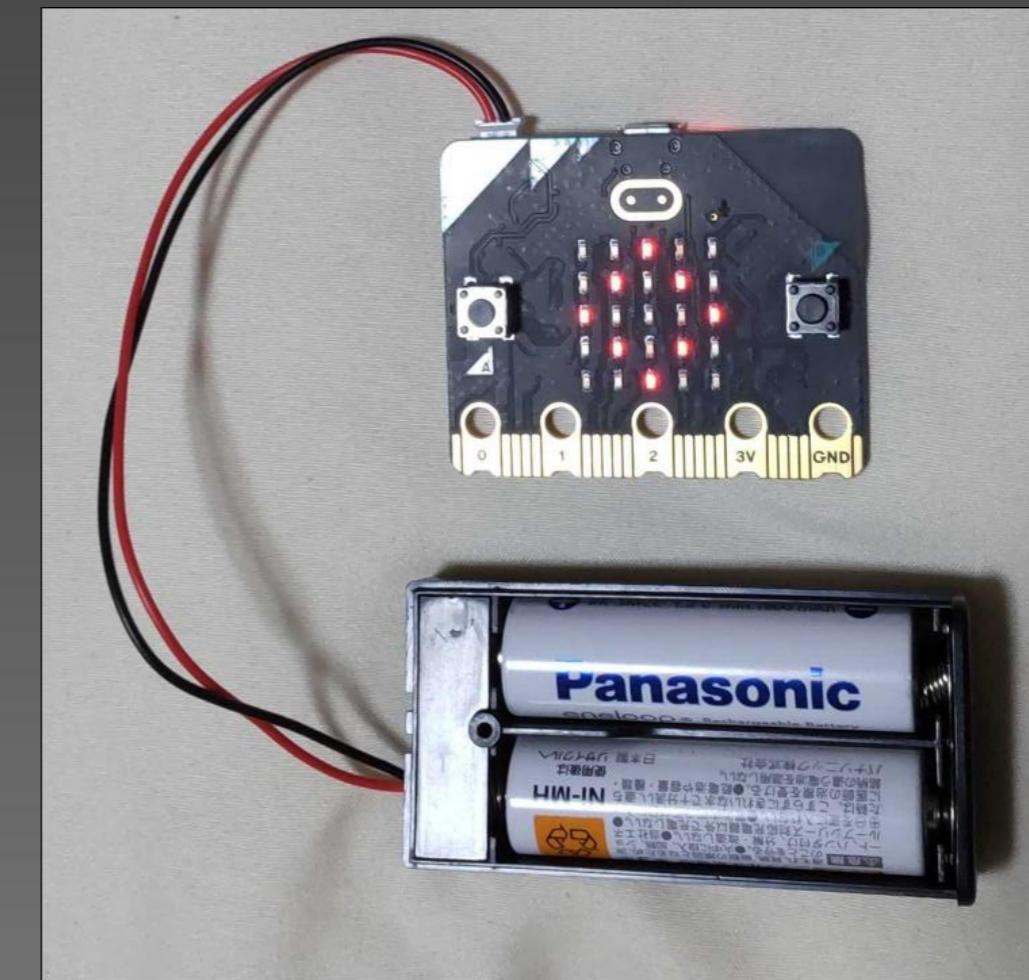
■ たとえば、以下を組み合わせたミニ楽器

- (3) PWMによる音階再生
- (5) 加速度センサー

■ AIの活用例

- micro:bitでも実行可能な超コンパクトなAI機能、フレームワーク、機械学習ソフトウェア等の利用
- クラウド上のAIと連携
- 開発環境や開発ツールとしてAIを活用

■ とはいえ、上記にとらわれない広範囲なアイデアを募集



Q and A

■ 参加者の皆様からのご質問

www.tron.org
www.t-engine4u.com