



μT-Kernel 3.0 BSP2 入門ガイド

トロンフォーラム T3 WG
INIAD(東洋大学情報連携学部)
豊山 祐一



- μ T-Kernel 3.0 BSP2とは
- BSP2を使ったプログラミングの概要
- BSP2のサンプル・プロジェクト
- マイコンのペリフェラルの使用方法
- μ T-Kernel 3.0 各種情報

- 質疑応答



μT-Kernel 3.0 BSP2とは

μT-Kernel 3.0とは



- IEEE 2050-2018国際標準仕様に準拠した軽量RTOS
 - 小規模な組み込みシステムやIoTエッジノード向けの高機能RTOS
- TRONプロジェクトの最新のリアルタイムOS
 - ▶ T-Kernelとの高い互換性。μITRONからの移行も容易。
- GitHubからのオープンソースのライセンス(T-License)でソースコード公開
- BSPによる市販ボードへの対応



μT-Kernel 3.0

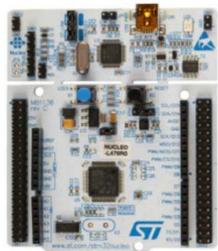
μT-Kernel 3.0 BSP (Board Support Package)



- μT-Kernel 3.0の入門や学習、評価用に市販マイコン・ボードですぐに使用可能なパッケージ
 - OS、基本デバイスドライバを実装済みのプロジェクトを提供
 - ソフトウェアはメーカーのファームウェアに依存しない
 - 2021年 GitHubから一般公開

対応マイコンボード

STM32L476 Nucleo-64	STM32H723 Nucleo-144	RX65N Renesas Target Board	RX65N Renesas Starter Kit+	Raspberry Pi Pico
ARM Cortex-M4	ARM Cortex-M7	RXv2	RXv2	ARM Cortex-M0+



μT-Kernel 3.0 BSP2



■ マイコンメーカー提供の開発環境、ファームウェアとの親和性を強化

- メーカー提供開発環境で作成したプロジェクトにそのまま組み込める。
- メーカー提供のHALなどファームウェアを簡単に使用可能
- 2023年12月から一般公開
- 対応マイコンボード

▶ STマイクロエレクトロニクス

- STM32L476 Nucleo-64
- STM32F401 Nucleo-64
- STM32F411 Nucleo-64
- STM32F446 Nucleo-64
- STM32G431 Nucleo-64
- STM32G491 Nucleo-64
- STM32F767 Nucleo-144
- **STM32H723 Nucleo-144**

▶ ルネサス エレクトロニクス

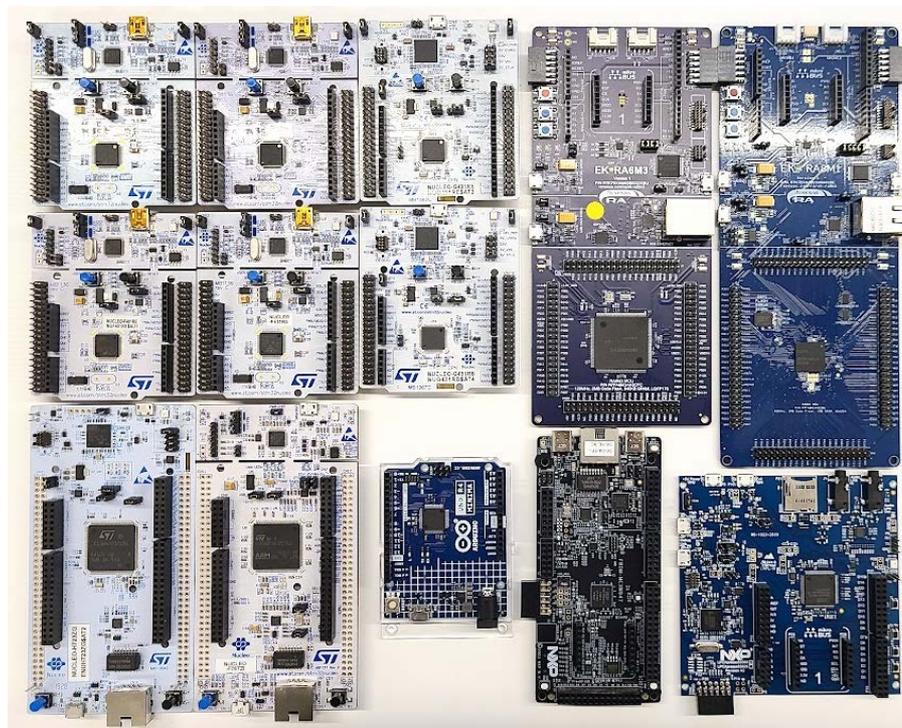
- EK_RA6M3
- **EK_RA8M1**
- Arduino UNO R4

▶ NXP

- **FRDM-MCXN947**
- LPC55S69-EVK

▶ インフィニオン

- **EVK-XMC7200**





BSP2を使った プログラミングの概要

マイコンメーカーの提供する開発環境・ツールを使用



- マイコンメーカーが提供するIDE(統合開発環境)、コンフィギュレータ、ファームウェアなどを使用

STマイクロエレクトロニクス

STM32CubeIDE

STM32CudeMX

STM32CubeFW

ルネサス エレクトロニクス

e²studio

Smart Configurator

Flexible Software Package (FSP)

インフィニオン テクノロジーズ

ModusToolbox

NXP

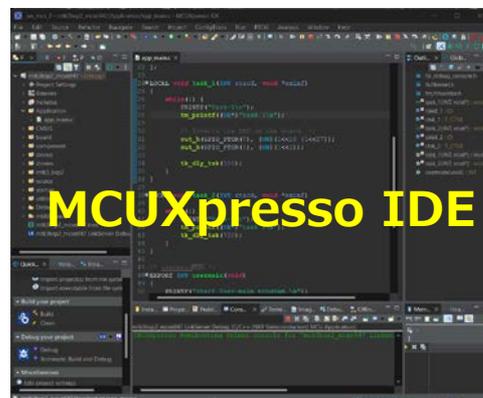
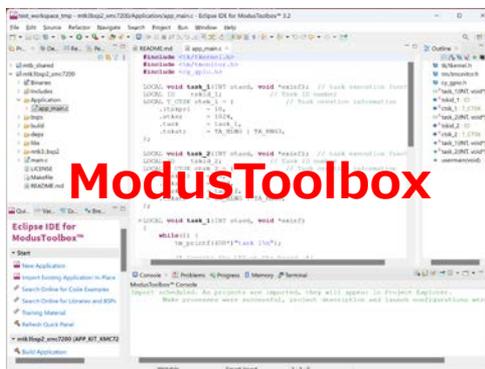
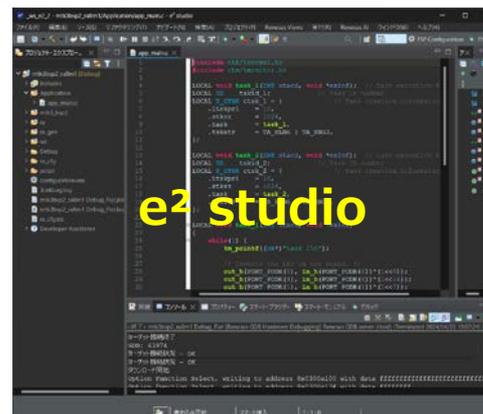
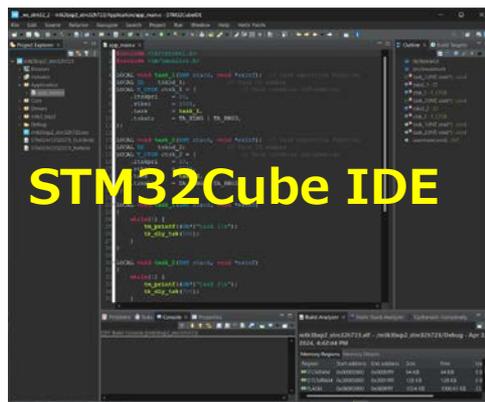
MCUXpresso IDE

IDE（統合開発環境）



■ Eclipse Embedded CDT (C/C++ Development Tools)をベースに各社のマイコンサポートツールが統合

- 基本的なデバッグ操作などは各社共通
- 各社のコンフィギュレータやファームウェアがIDEに統合



各社IDEの紹介ドキュメント



- IDEの使い方は各社が提供のドキュメントなどをご覧ください
- 5月にトロンフォーラムのウェビナーにて各社からの説明会を実施
 - 資料は以下からご覧いただけます
 - ▶ <https://www.tron.org/mt-kernel3/2024/05/30/post-82/>

各社のコンテスト向け説明会資料を公開しています

Edit

コンテスト 技術情報 最新情報



TRONプログラミングコンテスト

賞金総額 500万円

STMicroelectronics Infineon ST NXP RENESAS

TRONプログラミングコンテストに向けて、6月に実施しました各社のコンテスト説明会ウェビナーの資料を公開しています。コンテストのソフトウェア開発にお役立てください。

STマイクロエレクトロニクス

ルネサス エレクトロニクス

パーソナルメディア (micro:bit)

NXP (FRDM-MCXN947スタートアップガイド)

インフィニオンテクノロジーズ



- IDEではソフトウェアをプロジェクト単位で開発します

①ハードウェアの選択



IDE(コンフィギュレータ)を使用してプロジェクトを作成

②クロックの設定



③端子の設定



④ファームウェアの選択、設定



⑤μT-Kernel 3.0 BSP2の組み込み

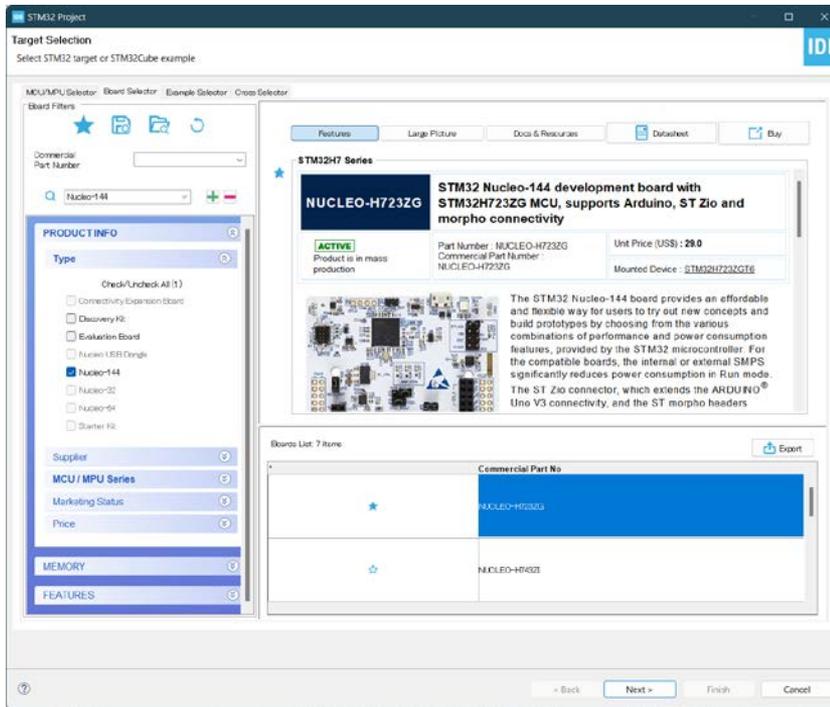


⑥アプリケーションの開発

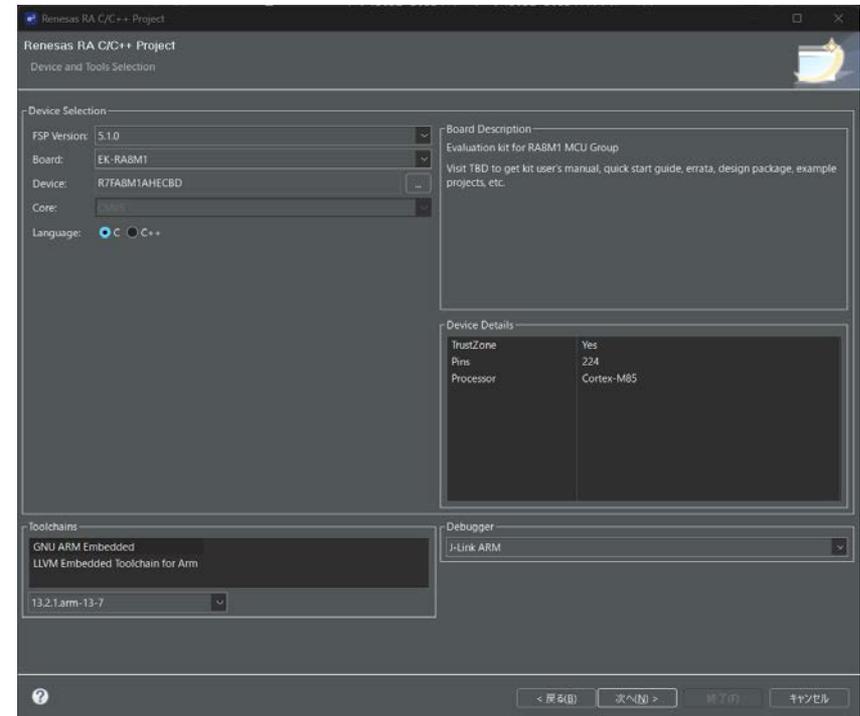
手順(1) ハードウェアの選択



- 使用するマイコン、ボードなど対象ハードウェアを選択しプロジェクトを作成します
 - メーカー純正のボードならばIDE上で選択可能です
- 選択したハードウェアに応じた基本設定が行われます



STM32Cube IDE

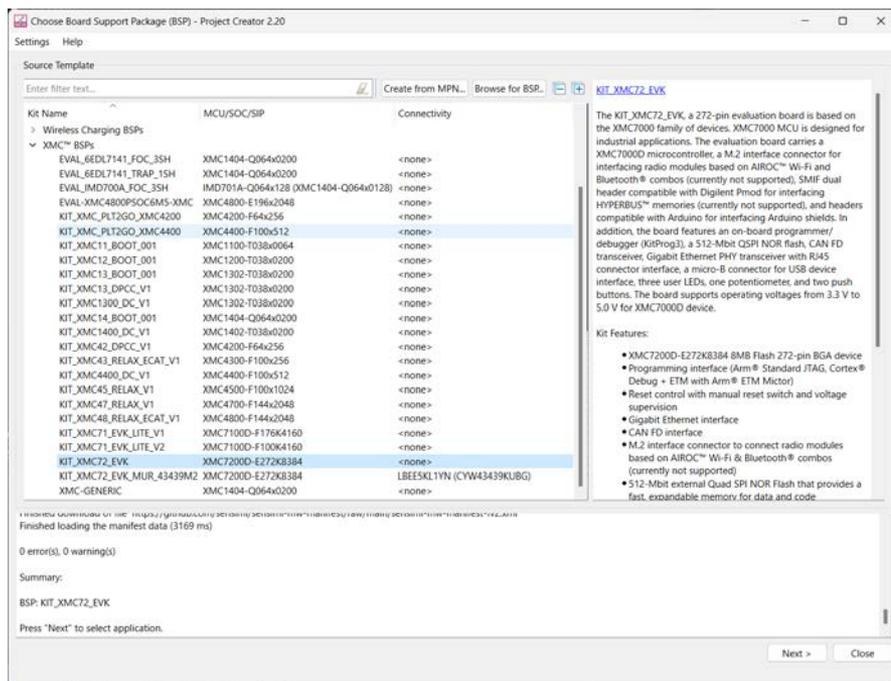


e² studio

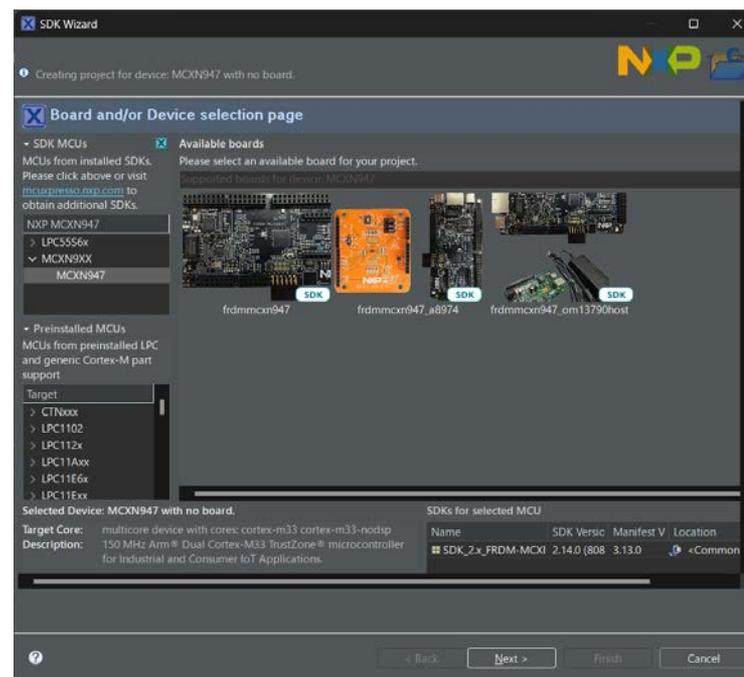
手順(1) ハードウェアの選択



- 使用するマイコン、ボードなど対象ハードウェアを選択します
 - メーカー純正のボードならばIDE上で選択可能です
- 選択したハードウェアの基本設定が自動生成されます



ModusToolbox

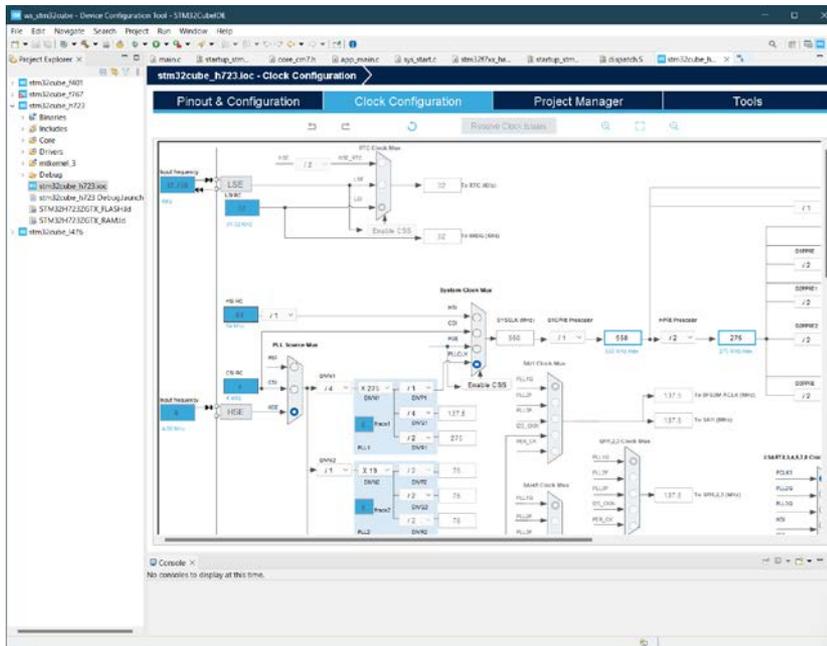


MCUXpresso IDE

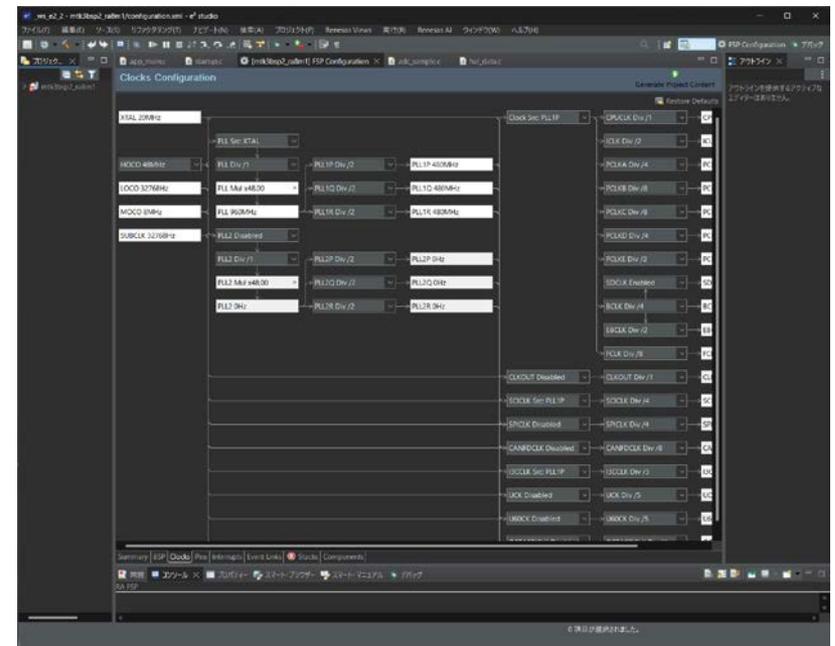
手順(2) クロックの設定



- マイコンの内部クロックの設定を行います
 - ボードを選択していれば基本的な設定はできています
- ペリフェラルに関しては基本的には設定する必要があります



STM32Cube IDE

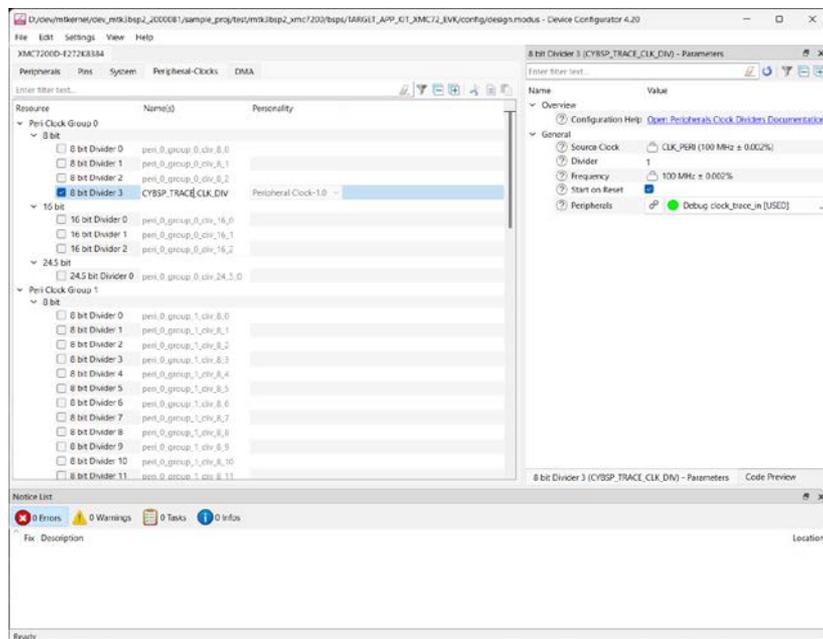


e² studio

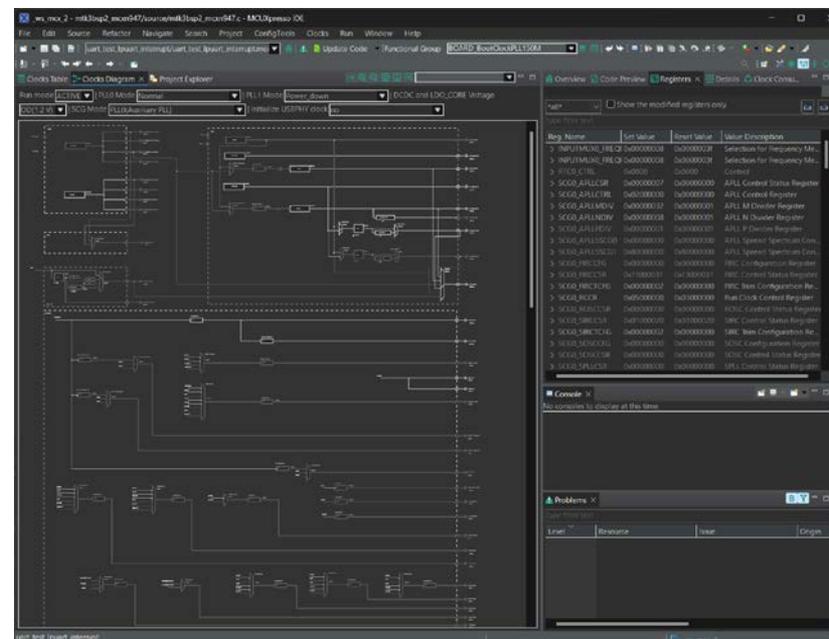
手順(2) クロックの設定



- マイコンの内部クロックの設定を行います
 - ボードを選択していれば基本的な設定はできています
- ペリフェラルに関しては基本的には設定する必要があります



ModusToolbox

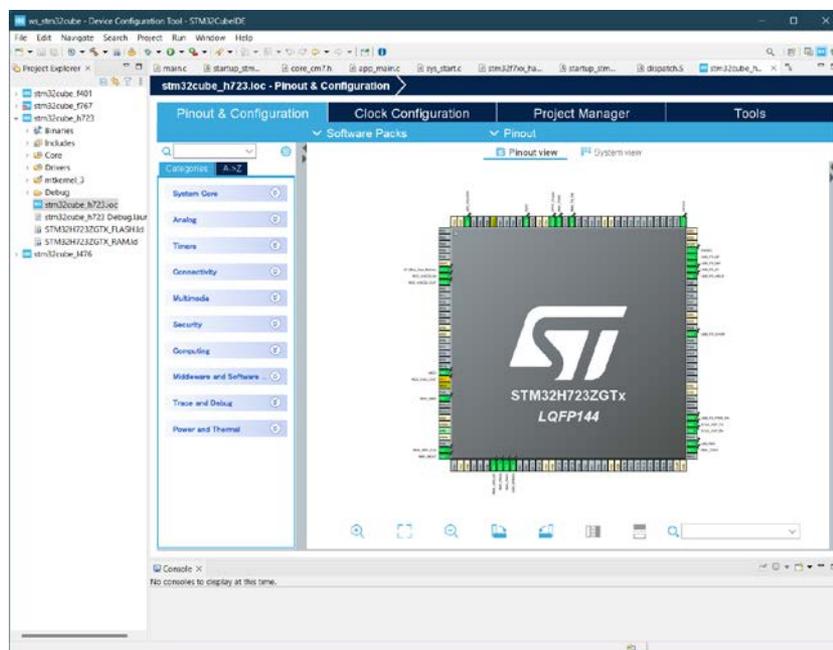


MCUXpresso IDE

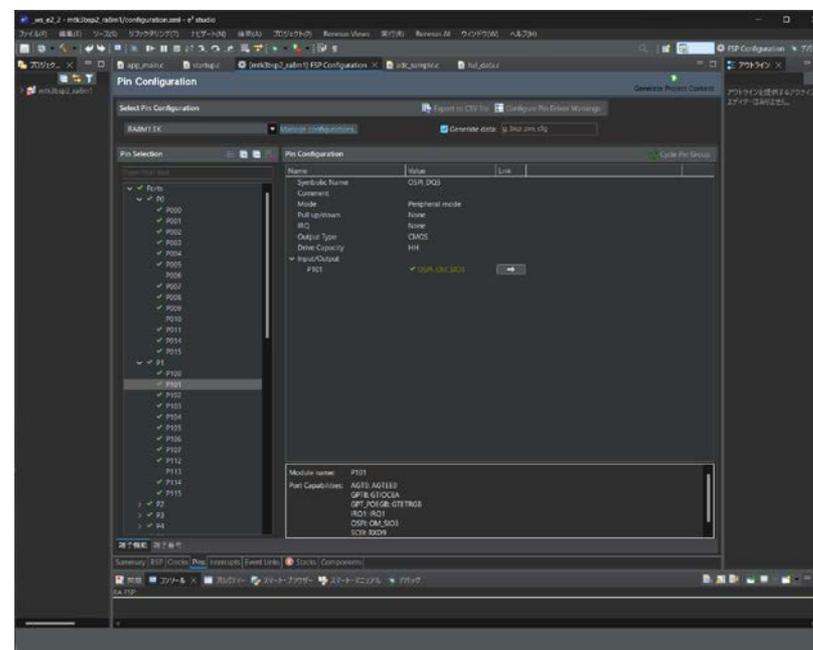
手順(3) 端子の設定



- マイコンの各端子の設定を行います
 - ボードを選択していれば基本的な設定はできています
- ペリフェラルに関しては基本的には設定する必要があります



STM32Cube IDE

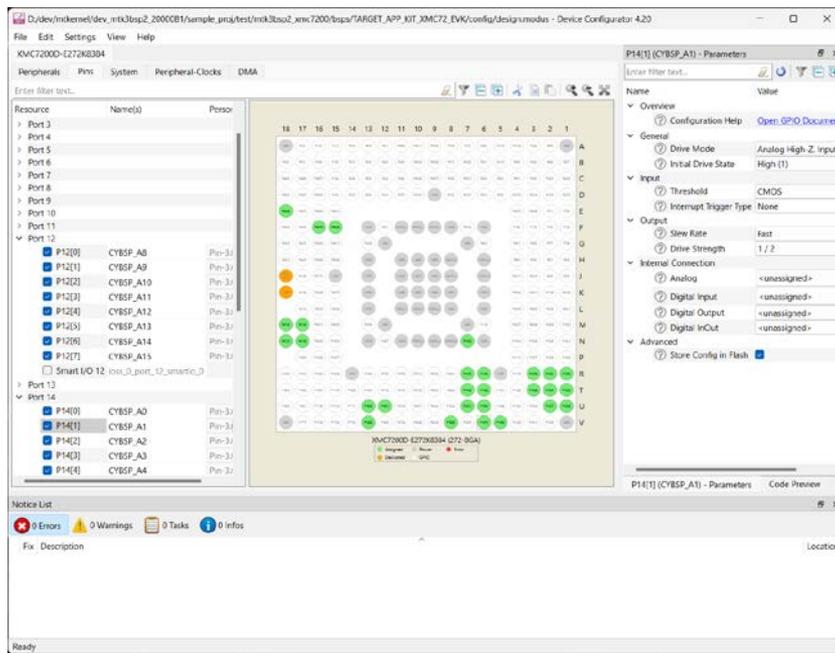


e² studio

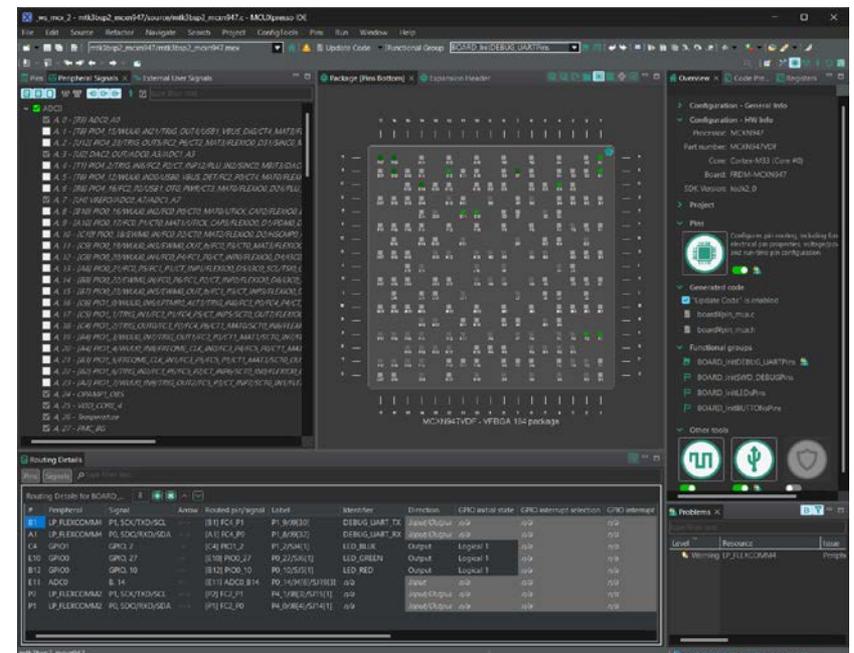
手順(3) 端子の設定



- マイコンの各端子の設定を行います
 - ボードを選択していれば基本的な設定はできています
- ペリフェラルに関しては基本的には設定する必要があります



ModusToolbox

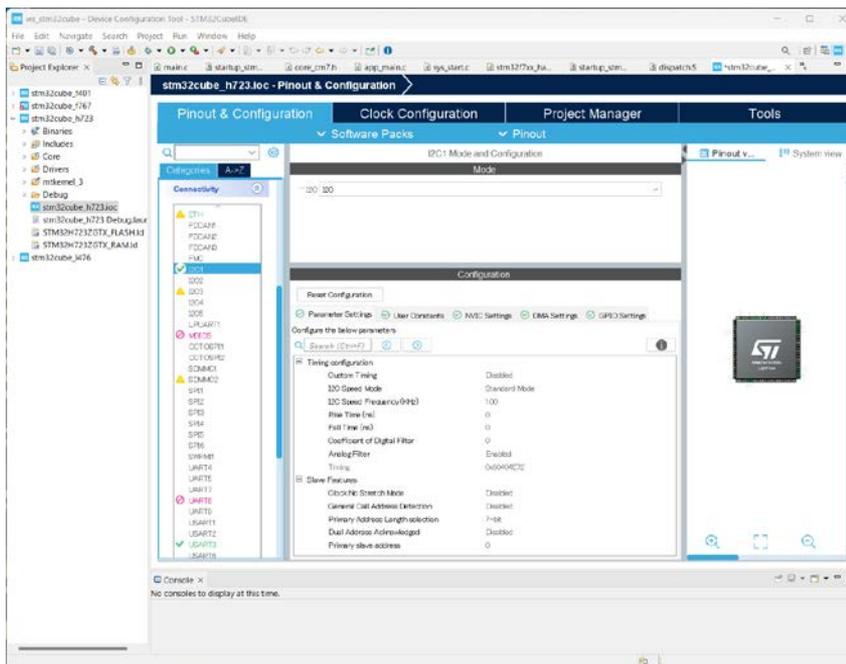


MCUXpresso IDE

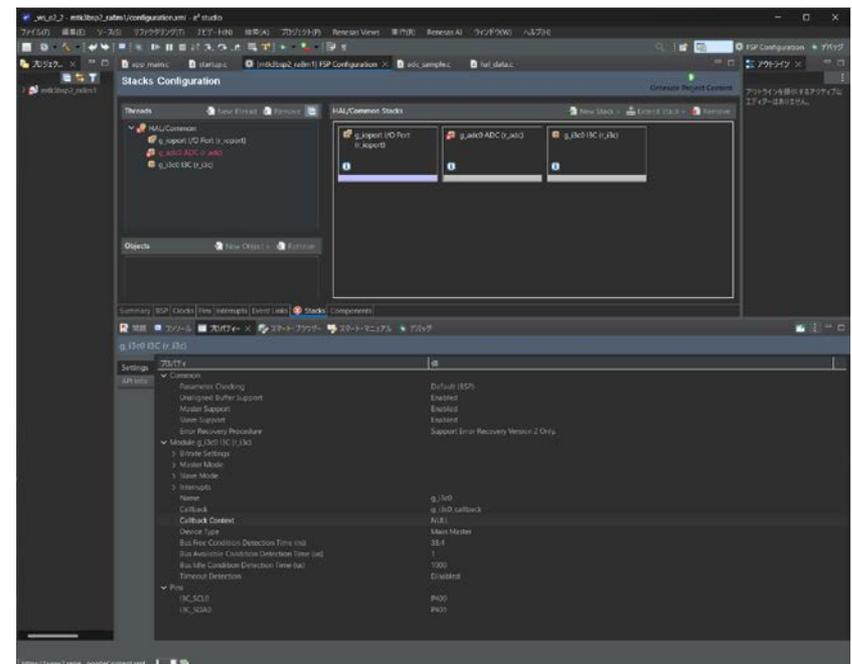
手順(4) ファームウェアの選択、設定



- 使用するHALやドライバなどを選択します
- デバイスの設定などを行います



STM32Cube IDE

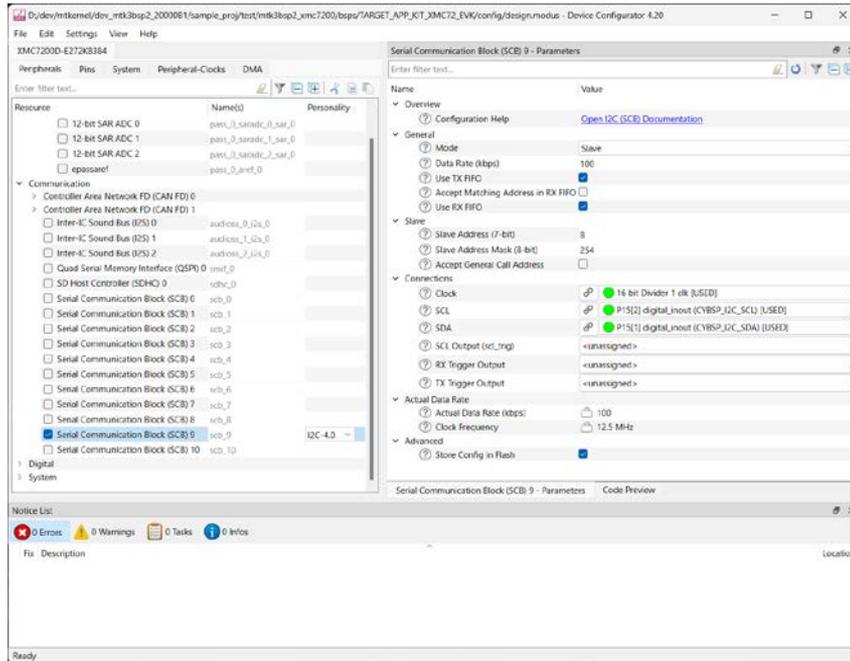


e² studio

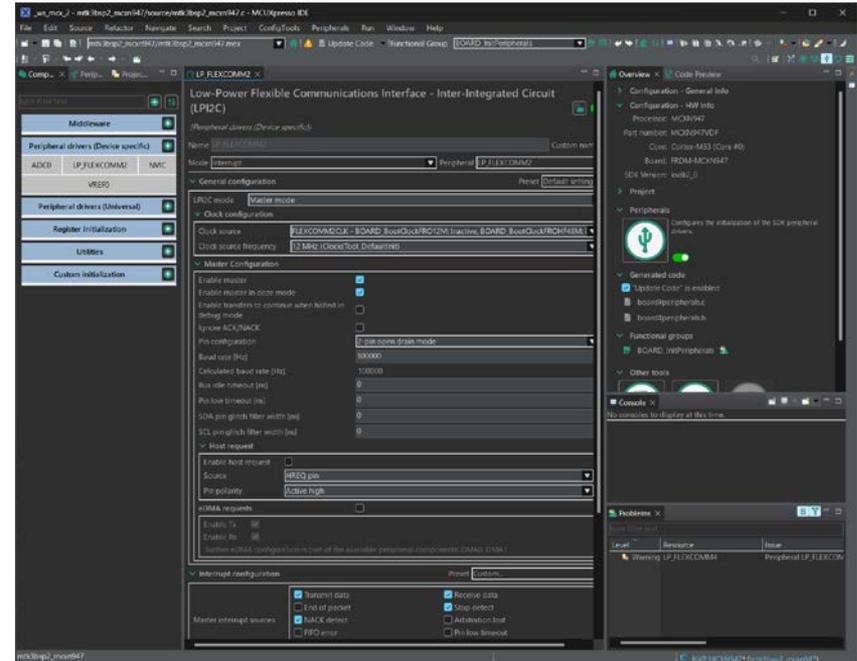
手順(4) ファームウェアの選択、設定



- 使用するHALやドライバなどを選択します
- デバイスの設定などを行います



ModusToolbox

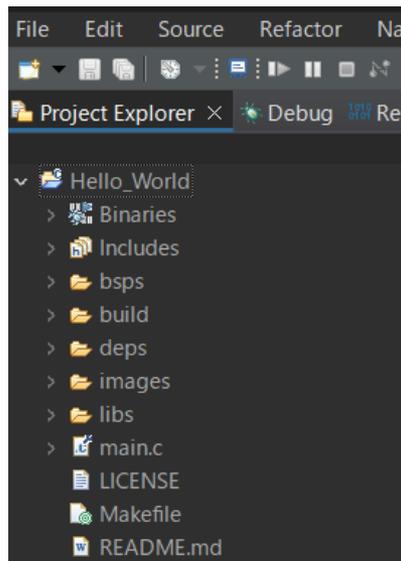


MCUXpresso IDE

いったんプロジェクトの動作確認



- ここまででプロジェクトの生成ができます
 - コンフィギュレータの設定に応じたソースコードが自動生成されます
- プロジェクトをビルドし、ボードで実行してみましよう
 - 何もしないアプリケーションが動作します
- 注意：自動生成されたファイルを直接編集しないように
 - 設定を変更すると上書きされてしまいます
 - アプリケーションのファイルのみ編集可能です



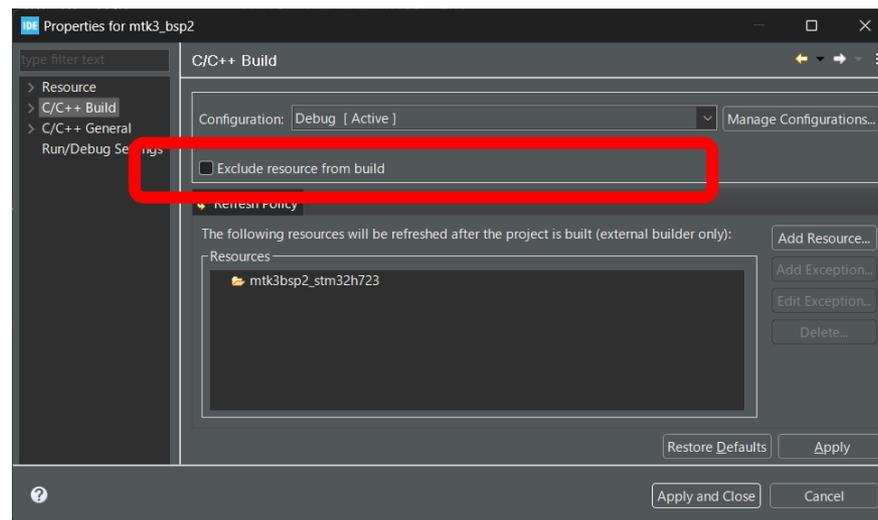
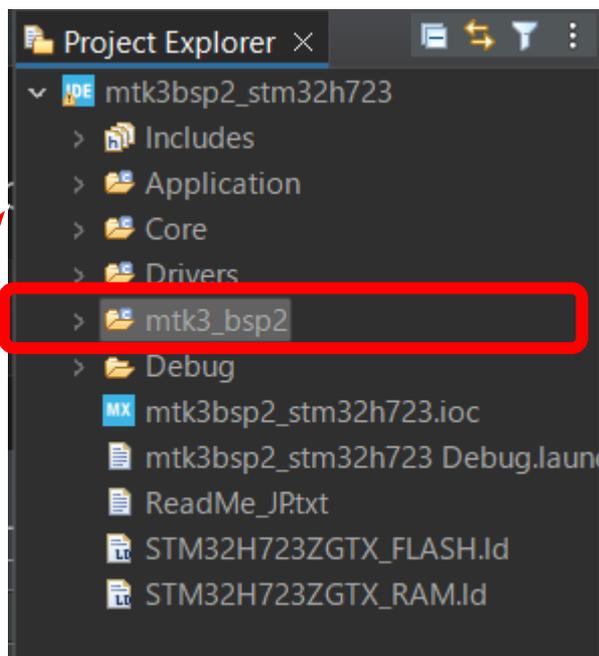
μT-Kernel 3.0 BSP2の組み込み(1)

ソースコードの取り込み



- μT-Kernel 3.0 BSP2のソースコードをダウンロードします
 - https://github.com/tron-forum/mtk3_bsp2/releases
- プロジェクトにBSP2のソースコードを追加します
 - ドラッグ&ドロップでソースコードをコピー
 - コピーした直後はビルドが無効になって場合がありますので有効にします
 - ▶ ModusToolboxはこの操作は不要です

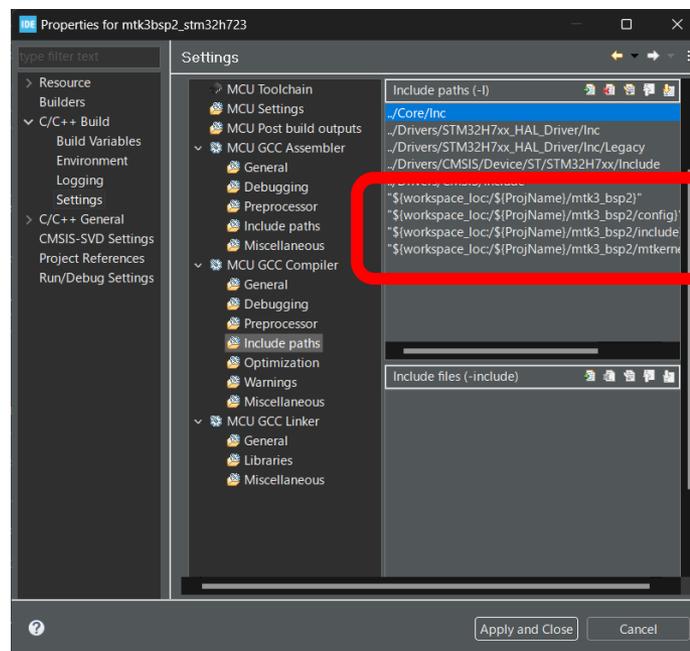
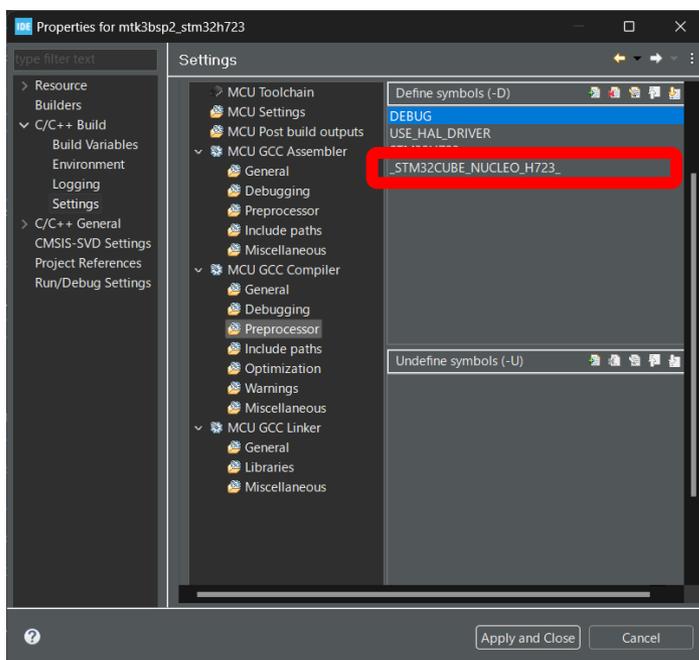
Drag&Drop



μT-Kernel 3.0 BSP2の組み込み(2) ビルドの設定



- μT-Kernel 3.0 BSP2のビルド設定を追加します
 - ターゲットボードを指定するシンボル定義
 - インクルードパスの設定
- 設定方法はμT-Kernel BSP2付属のマニュアルをご覧ください
 - ModusToolboxはmakefileを編集、それ以外のIDEはGUIで変更できます



μT-Kernel 3.0 BSP2の組み込み(3)

OSの起動関数の追加



- プロジェクトのアプリケーションの先頭からμT-Kernel 3.0の実行関数を呼び出します
 - `void knl_start_mtkernel(void);`
- アプリケーションの場所はIDEにより異なります。BSP2付属のマニュアルをご覧ください。
- ここまででいったんビルドし、ボードでの実行を確認してください
 - 何もしないμT-Kernelのアプリケーションが実行されます

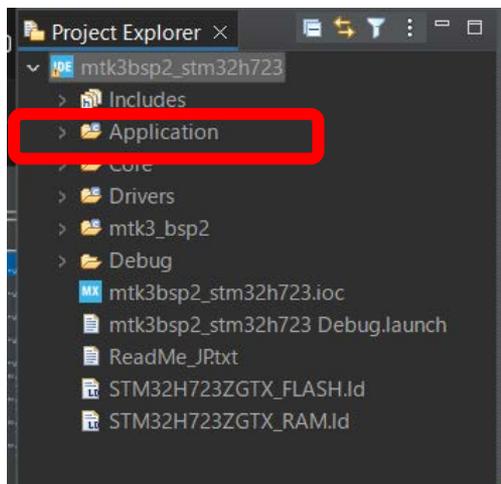


■ アプリケーションをプロジェクトに追加します

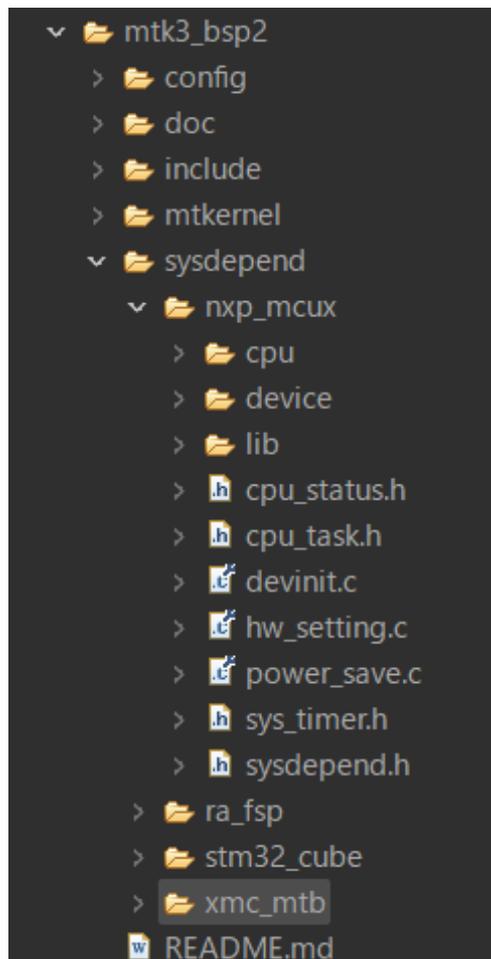
- 任意の場所に追加できますが、推奨はプロジェクトのルートディレクトリにアプリケーションのディレクトリを追加します
 - ▶ μT-Kernel 3.0 BSP2やIDEがバージョンアップした際に対応が楽です

■ アプリケーションの中でusermain関数を定義します

- μT-Kernel 3.0は起動後に初期タスクからusermain関数を実行します
- もしユーザ定義のusermain関数が無い場合はデフォルトのusermain関数を実行します。この関数は何もしません。



μT-Kernel 3.0 BSP2 ファイル構成



- config コンフィギュレーション定義
- doc ドキュメント
- include 定義ファイル
- mtkernel μT-Kernel 3.0本体
- sysdepend 対象ボード向けH/W依存部
 - nxp_mcux NXP
 - ra_fsp ルネサステクノロジー
 - stm32_cube STエレクトロニクス
 - xmc_mtb インフィニオン

- TRONプログラミングコンテストでは、
sysdependディレクトリ内のファイルは変更可能です。



BSP2の サンプル・プロジェクト

作成済みのサンプル・プロジェクトを配布



- TRONプログラミングコンテスト用のマイコンボードは作成済みのIDEプロジェクトをGitHubから配布しています

- https://github.com/tron-forum/mtk3bsp2_samples

mtk3bsp2_samples Public Watch 1

main 1 Branch 0 Tags Go to file Code

tron-forum Add Start Guide (Infineon KIT_XMC72_EVK) 101102b · 2 days ago 8 Commits

- IDE_Projects Add project (Infineon KIT_XMC72_EVK) 2 days ago
- Start_Guide Add Start Guide (Infineon KIT_XMC72_EVK) 2 days ago
- README.md Add Start Guide (Infineon KIT_XMC72_EVK) 2 days ago

mtk3bsp2_samples / IDE_Projects

tron-forum Add project (Infineon KIT_XMC72_EVK)

Name	Last commit message
..	
mtk3bsp2_mcxn947.zip	first commit
mtk3bsp2_ra8m1.zip	first commit
mtk3bsp2_stm32h723.zip	first commit
mtk3bsp2_xmc7200.zip	Add project (Infineon KIT_XMC72_EVK)



- サンプル・プロジェクトは以下が組込み済みです

- μ T-Kernel 3.0 BSP2
- サンプルのアプリケーション
 - ボード上のLEDをタスクから点滅
- A/DコンバータおよびI2Cのサンプル・デバイスドライバ
 - Arduino互換I/Fが使用可能に設定済み
 - 使用可能な端子はStart Guideをご覧ください
 - 設定等を変更すれば他の信号でも使用可能です

サンプル・プロジェクトのStart Guide



■ サンプル・プロジェクトについてはStart Guideをご覧ください

1 **μT-Kernel 3.0 BSP2 スタートガイド**
STM32Cube & NUCLEO-H723ZG編

2 **スタートガイドについて**
■本スタートガイドは、μT-Kernel 3.0 BSP2とマイコンボードの連携する仕組み(開発環境)を使用して、マイコンボードで実行するプログラムの作成、デバッグの基本的な方法を説明します。
■μT-Kernel 3.0 BSP2やIDEなどの詳細な情報は、他のドキュメントも参照してください。

3 **μT-Kernel 3.0 BSP2のダウンロード**
■μT-Kernel 3.0 BSP2のプロジェクト mik3bsp2_stm32h723.zip をダウンロードします。
■<https://www.st.com/en/development-tools/mik3bsp2.html>
■zipファイルを任意のディレクトリに展開します。
■このファイルを開発するディレクトリの名前には必ず「h723」と記載してください。

4 **IDEのインストール**
■STM32Cube IDEのインストールを以下よりダウンロードしインストールします。
■<http://www.st.com/en/development-tools/stm32cube-ide.html>
■STM32Cube IDEについて詳細は上記のWebサイトをご覧ください。

5 **STM32Cube IDEの実行**
■インストールしたSTM32Cube IDEを実行します。
■最初にワークスペースを開きます。任意のディレクトリを開いてください。この設定が保存されます。

6 **プロジェクトのインポート**
①メニュー[File]→[Import]を選択します。
②新しいタイプから[General]→[Existing Projects into Workspace]を選択しNextを押します。
③[Select root directory]の[Browse]ボタンを押し、BSP2のプロジェクトのディレクトリを指定します。
④BSP2のプロジェクトが表示されていることを確認のうえ[Finish]を押下します。
■[Select root directory]の[Browse]ボタンを押下し、開発するディレクトリを選択してください。
■[Import Existing Projects into Workspace]の[Next]ボタンを押下してください。

7 **プロジェクトの表示**
■インポートが正常に終了すると、プロジェクトマネージャ-μT-Kernel 3.0 BSP2のプロジェクトが表示されます。
■表示されているファイルをダブルクリックすると、その内容が表示され、編集ができます。
■BSP2のプロジェクトのディレクトリ構造を確認してください。

8 **プロジェクトのビルド**
■プロジェクトマネージャのプロジェクト名をクリックし、[Build Project]を選択します。
■プロジェクトのビルドが開始され、正常に終了すると「Build Finished」が表示されます。
■プロジェクトのビルドが完了すると、コンソールウィンドウにビルドログが表示されます。

9 **プログラムの実行とデバッグ(1)**
■ボード(NUCLEO-H723ZG)とPCをUSBで接続します。
■USBドライバのインストールを確認してください。
■プロジェクトマネージャのプロジェクト名をクリックし、メニュー[Run]から[Debug Configurations]を選びます。
■[Run]ボタンを押下し、実行を開始してください。

10 **プログラムの実行とデバッグ(2)**
■表示されたダイアログから[STM32 C/C++ Application]の[mik3bsp2_stm32h723_Debug]を選択します。
■[mik3bsp2_stm32h723_Debug]が表示されていない場合は、[STM32 C/C++ Application]ボタンをクリックしてください。
■前にビルドしたプロジェクトが対象となります。どこの名前を選択してください。

11 **プログラムの実行とデバッグ(3)**
■ダイアログの[Debug]ボタンを押すと、実行プログラムがボードに転送されて実行が開始されます。
■[Debug perspective]の切り替えが表示されますので[Switch]ボタンを押下します。デバッグ画面に切り替わります。
■[Debug]ボタンを押下してください。

12 **プログラムの実行とデバッグ(4)**
■デバッグが開始すると、app_main.cUsemain関数でブレークします。
■メニューのボタンから以下の基本的なデバッグ操作が可能です。
■STM32Cube IDEの操作方法は、メーカーのWebサイトなどをご覧ください。

13 **プログラムの実行とデバッグ(5)**
■ボードのプログラムからのdm_printf関数によるデバッグ用シリアル出力は、PCのUSBの仮想シリアルポートに入力されます。
■PCでターミナルソフトを開き、デバッグ用シリアル出力を表示することができます。
■PCのターミナルソフトにはTera Termなどが推奨されます。
■シリアル通信の速度は以下に設定してください。

14 **ユーザプログラムの作成**
■μT-Kernel 3.0 BSP2のApplicationディレクトリにユーザプログラムを記述します。
■ユーザプログラムのディレクトリは任意の名前に変更可能です。
■開発したプログラムは、BSP2のディレクトリ構造の参照してください。
■開発環境で、タスクを実行し、それぞれのスケッチボードにLEDを点灯させるシリアル出力を行うプログラムapp_main.cファイルに記述されています。

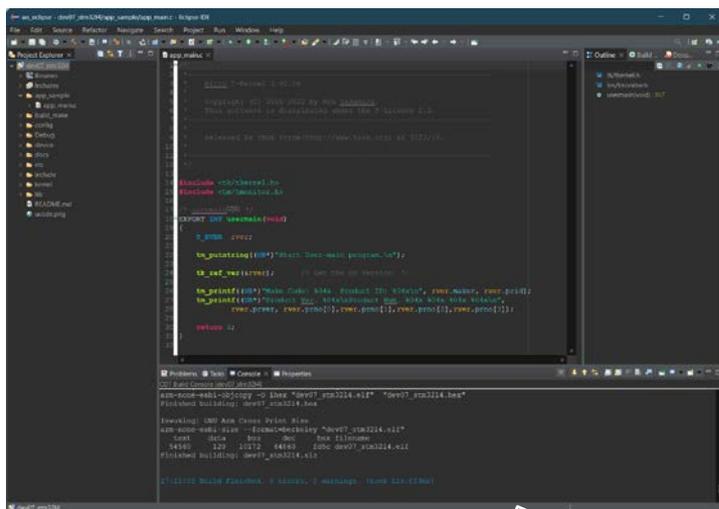
15 **ヘリファールの制御**
■μT-Kernel 3.0 BSP2は、A/Dコンバータと接続したサンプルデバイスドライバの制御ができています。
■サンプルデバイスドライバは、NUCLEO-H723ZG-TOArduinoB板のスケッチフォルダに格納されています。
■他の開発プロジェクトのコンフィギュレーション等の変更により利用できます。

Eclipse系IDEのポイント(1)

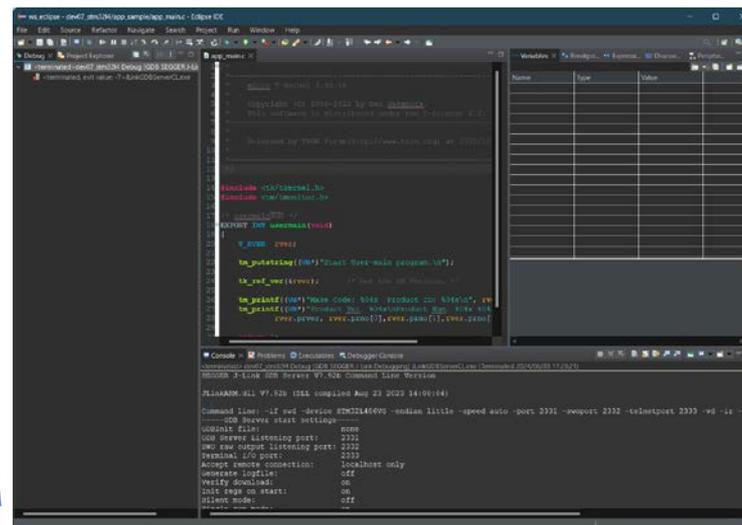


■ 複数の画面（パースペクティブ）を行き来します。

- Toolbarのボタンで移動できます



C/C++



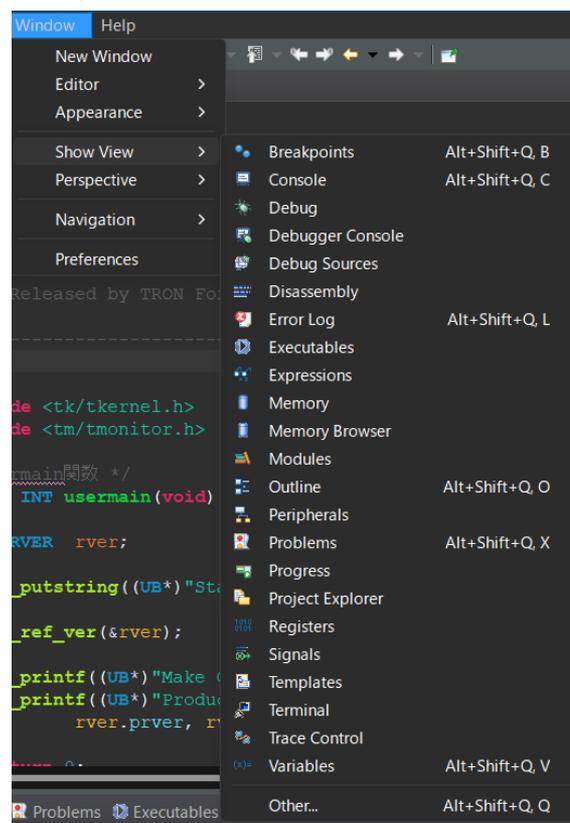
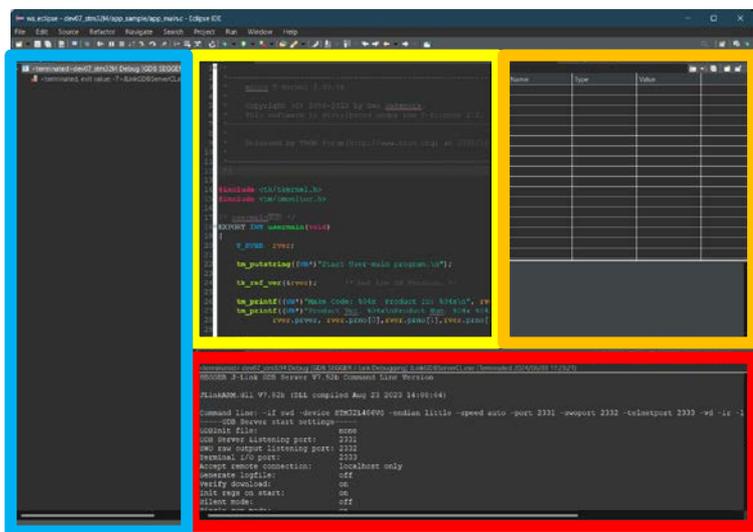
Debug

メーカー独自機能
(主にコンフィギュレーション関係)

Eclipse系IDEのポイント(2)



- パースペクティブ上のビューはカスタマイズできます
 - 画面分割、タブ化、非表示が自由にできます
- ビューはメニュー [Windows]>[Show View]で選択できます





- Eclipse は外部のソースコード・エディタが使用できます
 - 特に設定は不要
 - 外部のエディタでソースコードを変更するとEclipse に反映されま
す
 - ▶ 外部のエディタでソースコードを編集し、Eclipseでプロジェクトを選択して、ビルドするだけ
 - ▶ 外部エディタでソースコードのセーブを忘れずに
 - ▶ プロジェクトを選択せずにビルドすると反映されないときがあるので注意



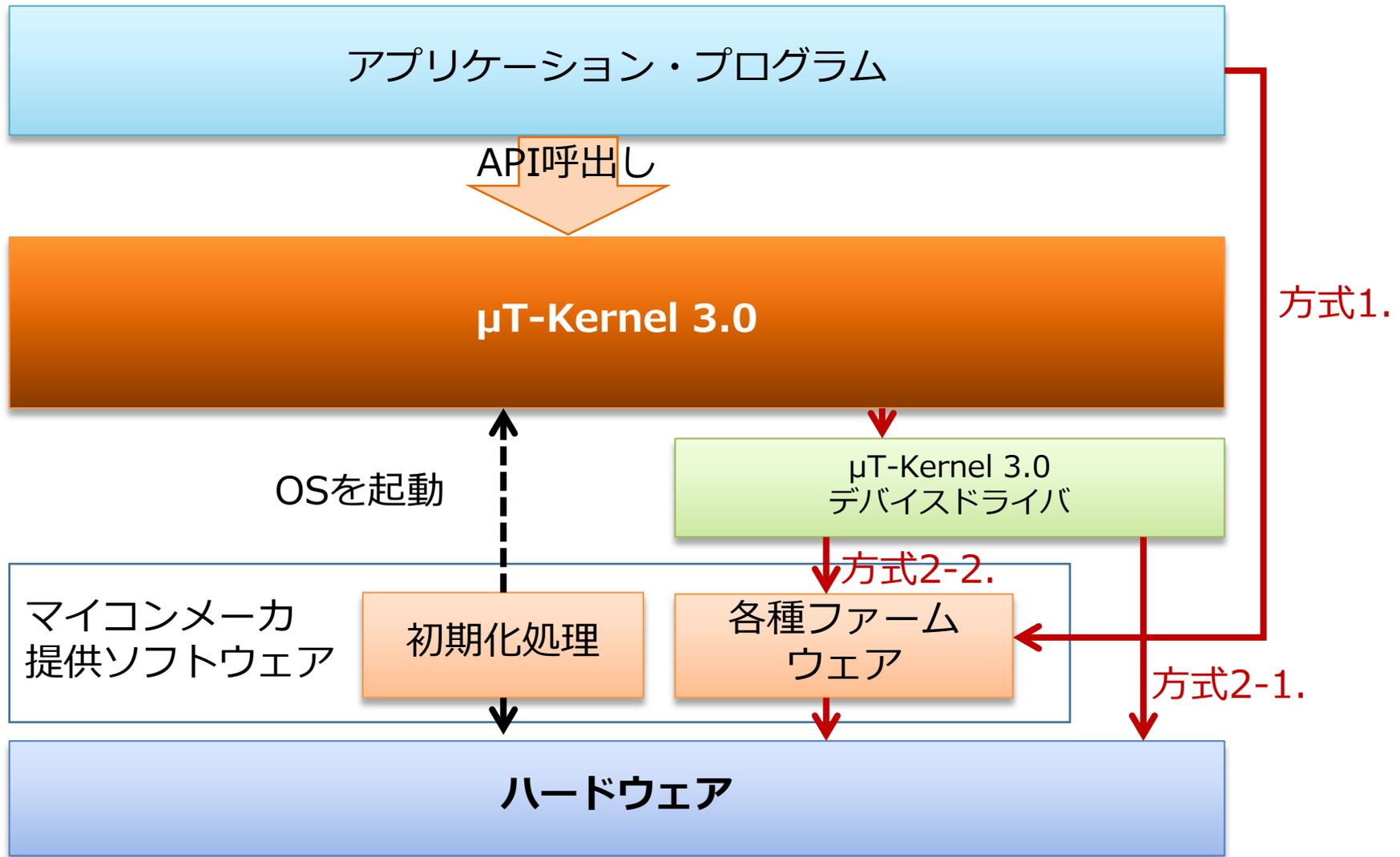
ペリフェラルの使用方法



- 以下の3種類の方法があります

- 1. マイコンメーカー提供のファームウェア(HALなど)を直接制御する
- 2. μ T-Kernel 3.0のデバイスドライバを作成する
 - 2-1. デバイスドライバからハードウェアを制御する
 - 2-2. デバイスドライバからメーカーのファームウェアを使用してハードウェアを制御する **推奨**

μT-Kernel 3.0 BSPのシステム構成



ペリフェラルを操作する各方式の違い



方式	メリット	デメリット
1. メーカー提供のファームウェアを直接制御	<ul style="list-style-type: none">ハードウェアのほぼ全ての機能を利用可能コンフィギュレータで設定の変更が可能	<ul style="list-style-type: none">操作方法やAPIの標準化はあまりなされていないRTOSのマルチタスクでの使用には注意が必要
2-1. μ T-Kernel 3.0デバイスドライバからハードウェアを制御	<ul style="list-style-type: none">操作方法やAPIの標準化されているRTOSから使用するにあたり制約のない効率の良い実装が可能	<ul style="list-style-type: none">基本的に個々に開発する必要がある
2-2. μ T-Kernel 3.0デバイスドライバからメーカーのファームウェアを使用してハードウェアを制御	<ul style="list-style-type: none">操作方法やAPIの標準化されているコンフィギュレータで設定の変更が可能RTOSから使用するにあたり制約のない実装が可能	<ul style="list-style-type: none">基本的に個々に開発する必要がある（ただし、2-1に比べれば容易）

メーカー提供のファームウェアを使用する場合



- アプリケーションから、メーカー提供のファームウェアを使用することができます
 - 使用方法については各メーカーからの情報を参照ください
- リアルタイムOS上で実行するために以下の注意が必要です
 - 複数のタスクから同時に使用する場合は、何らかの排他制御が必要です
 - ポーリング（ビジーループ）は他のタスクの動作を阻害しますので、割り込みの使用を推奨します
 - 割り込みからタスクへのイベントの通知はOSのAPIの使用を推奨します
 - ▶ グローバル変数のポーリングを行うと他のタスクの動作を阻害します

μT-Kernel 3.0 BSP2

サンプル・デバイスドライバ



- μT-Kernel 3.0 BSP2には2-2方式のサンプル・デバイスドライバが
付属しています (A/DC、I²C通信)
- サンプル・デバイスドライバはHALを使用しています
 - ペリフェラルの基本機能のみをサポートしています
 - デバイスドライバを開発するためのサンプルの位置づけです

A/Dコンバータの制御プログラム



- センサーからのアナログ入力を1秒毎に取り、UART(デバッグ)出力に表示するタスク・プログラム

```
LOCAL void task_1(INT stacd, void *exinf)
{
    UW      adc_val;
           ID      dd;          // デバイスディスクリプタ
           ER      err;

    dd = tk_opn_dev((UB*)"hadca", TD_UPDATE);    // デバイスのオープン
    if(dd <= E_OK) tm_printf((UB*)"ERR-0  %d¥n", dd);

    while(1) {
        err = tk_srea_dev(dd, 0, &adc_val, 1, NULL);    // A/DC チャンネル0からデータを取得
        if(err == E_OK) {
            tm_printf((UB*)"A/DC A0  =%06d¥n", adc_val);    // デバッグ出力
        } else {
            tm_printf((UB*)"ERR-ADC¥n");
        }
        tk_dly_tsk(1000);    // 1000ms待ち
    }
}
```



- センサーの値をI²C通信を使って1秒毎に読み取り、UART（デバッグ）出力に表示するタスク・プログラム
 - 温度・湿度センサー(SHT35)の例

```
LOCAL void task_2(INT stacd, void *exinf)
{
    UW          tmp0, hum0;
    UB          cmd[] = {0x2C, 0x06};
    UB          rcv[6];
    ID          dd;          // デバイスディスクリプタ
    ER          err;

    dd = tk_opn_dev((UB*)"htiica", TD_UPDATE);    // デバイスのオープン
    if(dd <= E_OK) tm_printf((UB*)"ERR-0 %d¥n", dd);

    while(1) {
        err = tk_swri_dev(dd, 0x45, cmd, 2, NULL);    // コマンドの送信
        if(err != E_OK) tm_printf((UB*)"ERR-1¥n");
        tk_dly_tsk(1);
        err = tk_srea_dev(dd, 0x45, rcv, 6, NULL);    // データの受信
        if(err != E_OK) tm_printf((UB*)"ERR-2¥n");

        tmp0 = rcv[0]<<8 | rcv[1]; hum0 = rcv[3]<<8 | rcv[4];
        tmp0 = (tmp0*175)/65535 - 45; hum0 = (hum0*100)/65535;
        tm_printf((UB*)"温度 %d °C 湿度 %d %¥n¥n", tmp0, hum0);    // デバッグ出力

        tk_dly_tsk(1000);    // 1000ms待ち
    }
}
```



μT-Kernel 3.0 情報



トロンフォーラムではμT-Kernel 3.0の各種セミナーを実施しています

■ 【実習】 μT-Kernel 3.0 組込みプログラミング入門

- 市販のマイコンボードを使用した1日のハンズオン実習
- 使用したマイコンボードなど教材を持ち帰れます。
- 予定
 - ▶ 6月28日 RXマイコン編 (Target Board for RX65N)
NIAD(東洋大学情報連携学部) 東洋大学赤羽台キャンパス
 - ▶ 以降、各種マイコンについて実施していきます



■ その他、中級プログラミング演習、ネットワーク演習などを実施

■ 今後の予定はトロンフォーラムのWebサイトに掲載します

- <https://www.tron.org/ja/seminar/schedule2024/>



INIAD (東洋大学情報連携学部) Open IoT 教育プログラム



- 高度なIoT技術を身に付けたい社会人の方を対象
- トロンフォーラムと連携し、リアルタイムOS μ T-Kernel 3.0を用いたシステム開発を中心に、Webアプリケーション開発やAI等のIoTと関連が深い分野の最新技術について学習します。
- **2024年度 '24/9月~'25/1月開催予定**
 - **受講生募集は2024年6月中旬開始を予定**
 - <https://enpit.iniad.org/>

μT-Kernel 3.0 解説書



リアルタイムOSの初歩から実践テクニックまで

基礎から学ぶ 組み込みμT-Kernel プログラミング

IEEE 世界標準
μT-Kernel 3.0

坂村 健 監修
豊山 祐一 著

IoT エッジノード向け世界標準 OS
マイクロカーネル
「μT-Kernel 3.0」を徹底解説

リアルタイムOSの基礎からデバイスドライバの開発や
OSの移植まで、理解のポイントとプログラム例を
分かりやすく説明した入門書の決定版

パーソナルメディア

- リアルタイムOSの基礎からデバイスドライバの開発、OSの移植まで説明
- μT-Kernel 3.0 BSPを使用し、市販のマイコンボードでプログラムを作成し実行
- 実習セミナーのテキストにも使用



- トロンフォーラムのWebからも技術情報を提供しています

- リアルタイムOS入門「 μ T-Kernel 3.0で学ぶリアルタイムOSプログラミング」
 - <https://www.tron.org/ja/page-722/rtos01/>
 - リアルタイムOSの基本と、リアルタイムOSを使用したプログラミングについて説明します

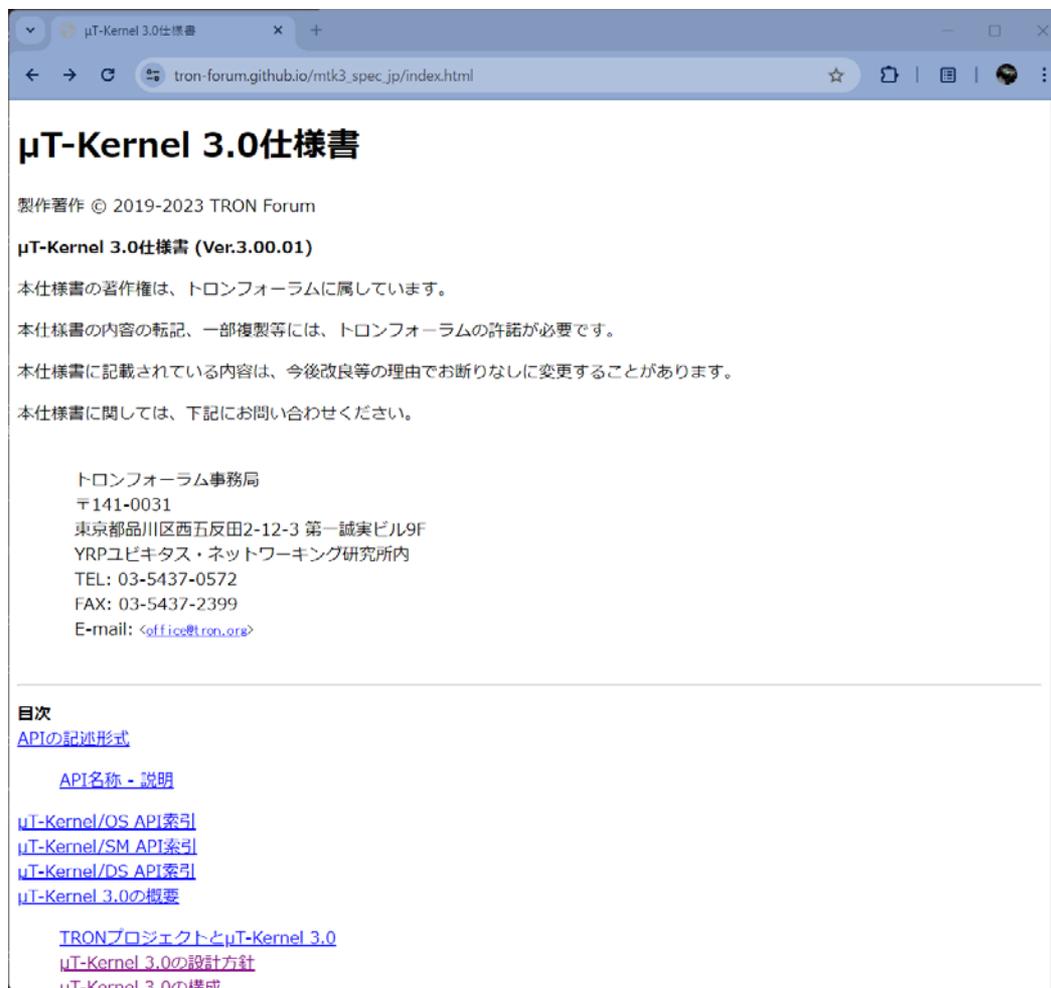
- 「新しいBSP（ボードサポートパッケージ）を使いこなす」
 - https://www.tron.org/ja/programming_contest/bsp/
 - μ T-Kernel 3.0 BSP2の紹介記事（TRONWARE誌掲載）

μT-Kernel 3.0 仕様書



■ μT-Kernel 3.0仕様書はWebから閲覧可能です。

- https://tron-forum.github.io/mtk3_spec_jp/index.html

A screenshot of a web browser displaying the μT-Kernel 3.0 specification page. The browser's address bar shows the URL "tron-forum.github.io/mtk3_spec_jp/index.html". The page content includes the title "μT-Kernel 3.0仕様書", copyright information "製作著作 © 2019-2023 TRON Forum", and the version "μT-Kernel 3.0仕様書 (Ver.3.00.01)". It also contains several paragraphs of text regarding copyright and permissions, followed by contact information for the TRON Forum office. At the bottom, there is a table of contents section with links to various parts of the specification, such as "APIの記述形式", "API名称 - 説明", "μT-Kernel/OS API索引", "μT-Kernel/SM API索引", "μT-Kernel/DS API索引", "μT-Kernel 3.0の概要", "TRONプロジェクトとμT-Kernel 3.0", "μT-Kernel 3.0の設計方針", and "μT-Kernel 3.0の構成".

μT-Kernel 3.0仕様書

製作著作 © 2019-2023 TRON Forum

μT-Kernel 3.0仕様書 (Ver.3.00.01)

本仕様書の著作権は、トロンフォーラムに属しています。

本仕様書の内容の転記、一部複製等には、トロンフォーラムの許諾が必要です。

本仕様書に記載されている内容は、今後改良等の理由でお断りなしに変更することがあります。

本仕様書に関しては、下記にお問い合わせください。

トロンフォーラム事務局
〒141-0031
東京都品川区西五反田2-12-3 第一誠実ビル9F
YRPユビキタス・ネットワークング研究所内
TEL: 03-5437-0572
FAX: 03-5437-2399
E-mail: <office@tron.or.jp>

目次

- [APIの記述形式](#)
- [API名称 - 説明](#)
- [μT-Kernel/OS API索引](#)
- [μT-Kernel/SM API索引](#)
- [μT-Kernel/DS API索引](#)
- [μT-Kernel 3.0の概要](#)
- [TRONプロジェクトとμT-Kernel 3.0](#)
- [μT-Kernel 3.0の設計方針](#)
- [μT-Kernel 3.0の構成](#)



- μT-Kernel 技術情報サイトではμT-Kernel 3.0に関する各種情報、関連リンクなどを掲載しています

- <https://www.tron.org/mt-kernel3/>





- TRONプログラミングコンテスト開催中です
 - ボードをご自身で用意いただければ参加できます
 - 応募締め切り 9月30日 まだ間に合います
 - 詳しくはWebを
 - ▶ https://www.tron.org/ja/programming_contest/

- ご質問があればお答えします