µT-Kernel3.0 BSP ユーザズ・マニュアル

- STM32H723 Nucleo-144編 -

Version.01.00.01

2023.05.26

目次

1. 概要

本書は、マイコンボードSTM32H723 Nucleo-144 向けのμT-Kernel 3.0 BSP (Board Support Package)の使用方法を説明します。
μT-Kernel 3.0 BSPは、特定のマイコンボード等のハードウェアに対して移植したμT-Kernel 3.0の開発および実行環境一式を提供するものです。

1.1. 対象ハードウェアおよびソフトウェア

本BSPの対象ハードウェアおよびソフトウェアは以下となります。

分類	名称	開発元
マイコンボード	Nucleo-144 for STM32H723ZG	STマイクロエレクトロニクス
搭載マイコン	STM32H723ZG (ARM Cortex-M7 コア)	STマイクロエレクトロニクス
OS	μT-Kernel3.0 v3.00.06A	TRONフォーラム ※1
BSP	STM32H723 Nucleo-144向けμT-Kernel3.0 BSP v1.00.00.B5	TRONフォーラム ※2

※1 μT-Kernel 3.0は以下のGitHubのリポジトリから公開 https://github.com/tron-forum/mtkernel_3

※2 μT-Kernel3.0 BSPは以下のGitHubのリポジトリから公開 https://github.com/tron-forum/mtk3_bsp

1.2. 使用する開発環境

開発を行うホストPCのOSはWindowsとします。確認はWindows 11で行いました。
本BSPで使用する開発環境、ツールは以下となります。

(1) 統合開発環境 STM32CudeIDE STマイクロエレクトロニクスのSTM32マイコン開発用の統合開発環境です。Eclipseをベースとし、STM32マイコンの開発のためのプラグインやCコンパイラのツールチェーンがパッケージされています。
動作検証を行ったバージョンは 1.12.1 です。

1.3. 関連ドキュメント

以下に関連ドキュメントを示します。

分類	名称	発行
OS仕様	μT-Kernel 3.0仕様書 Ver.3.00.01	TRONフォーラム TEF020-S004-3.00.01 ※1
OS実装仕様	μT-Kernel 3.0 STM32H7マイコン向け実装仕様書 Ver.1.00.02	TRONフォーラム TEF033-W012-221014 ※2

分類	名称	発行
デバイスドライバ	μT-Kernel 3.0 デバイスドライバ説明書 Ver.1.00.5	TRONフォーラム TEF033-W007-221007 ※2

※1 TRONフォーラムWebページから公開
https://tron-forum.github.io/

※2 μT-Kernel 3.0正式版に含まれる（以下のGitHubのリポジトリから公開）
https://github.com/tron-forum/mtkernel_3

2. 開発環境の準備

µT-Kernel 3.0 BSPを使用するにあたり、開発環境を準備する手順を説明します。

2.1. STM32CubeIDEのインストール

STM32CubeIDEを以下から入手し、Webページの指示に従いインストールします。
Windowsではインストーラを使用しSTM32CubeIDEをインストールすれば、必要なツール一式が入手できます。

STM32CubeIDEのダウンロードページ https://www.st.com/ja/development-tools/stm32cubeide.html

3. プロジェクトの作成

STM32CudeIDEではプログラムはプロジェクトの単位で開発します。µT-Kernel 3.0 BSPのプロジェクトを以下の手順で作成します。

3.1. GitHubからのプロジェクトの入手

TRONフォーラムのGitHubの以下のレポジトリにµT-Kernel 3.0 BSPが公開されてています。

https://github.com/tron-forum/mtk3_bsp

µT-Kernel 3.0 BSPは対象ハードウェア毎にブランチが作成されています。STM32H723 Nucleo-144のBSPのブランチ名は「stm32h723_nucleo」です。
また、最新のリリース版は、GitHubのReleaseから取得することができます。「µT-Kernel 3.0 BSP for STM32H723 Nucleo」を選んでダウンロードしてください。
ダウンロードしたZipファイルは任意の場所に展開すると、STM32CudeIDEのプロジェクトになります。

3.2. プロジェクトのファイル構成

µT-Kernel 3.0 BSPのプロジェクトは、以下のディレクトリおよびファイルの構成となっています。基本的にはµT-Kernel 3.0の正式リリース版に準じています。
以下にディレクトリおよびファイルの構成を示します。

ディレクトリまたはファイル名	内容
app_program	アプリケーション・プログラム用のディレクトリ
build_make	Make構築ディレクトリ（BSPでは使用しない）
config	コンフィギュレーション
device	デバイスドライバ
docs	ドキュメント
etc	リンカファイル等
include	各種定義ファイル

ディレクトリまたはファイル名	内容
kernel	µT-Kernel本体
lib	ライブラリ
.settings	STM32CudeIDEの設定ファイル
.cproject	STM32CudeIDEのプロジェクトファイル
その他ファイル	STM32CudeIDEの各種ファイルなど

ユーザが作成するアプリケーション・プログラムは、ディレクトリ「app_program」に格納します。初期状態では、サンプルコードを記述したapp_main.cファイルが格納されています。通常、ユーザが操作するのは、このディレクトリ「app_program」となります。

3.3. プロジェクトのインポート

前項で入手したプロジェクトをSTM32CudeIDEに以下の手順で取り込みます。

(1) メニュー[File]から[Import...]を選択するとImportダイアログが開きます。

(2) [General]→[Existing Projects into Workspace]を選択し[Next]ボタンを押します。

(3) [Select root directory]の[Browse]ボタンを押し、前項のµT-Kernel 3.0 BSPプロジェクトのディレクトリを指定します。

(4) [Projects]に[mtk3bsp_nucleo_stm32h723]が表示されるので、それをチェックします。

(5) [Finish]ボタンを押下すると、プロジェクトがワークスペースに取り込まれます。インポートが完了すると、STM32CudeIDEのProject Explorerに[mtk3bsp_nucleo_stm32h723]]が表示されます。

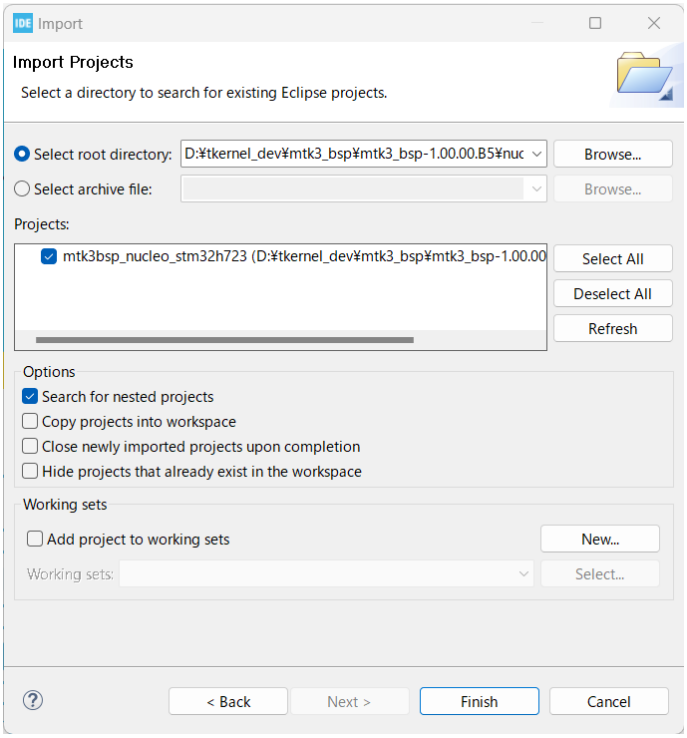

図1 STM32CudeIDEに取り込んだプロジェクト

3.4. プロジェクトのビルド

µT-Kernel 3.0 BSPプロジェクトをビルドします。 Project Explorer上で[mtk3bsp_nucleo_stm32h723]が選択されている状態で、メニュー[Project]→[Build project]を選択します。もしくはProject Explorer上の[mtk3bsp_nucleo_stm32h723]を右クリックし、メニュー[Build project]を選択します。
プログラムのビルドが実行され、[Console]に「Build Finished. 0 errors」が表示されれば、正常にビルドが完了しました。

4. アプリケーションの作成

4.1. プログラムのソースファイル

本プロジェクトでは、アプリケーション・プログラムのソースファイルは、プロジェクトのディレクトリ下の[app_program]ディレクトリに置くことを前提としています。
初期状態では、サンプルコードを記述したapp_main.cファイルが置かれています。ユーザは本ファイルの内容を変更することにより、プログラムを記述することができます。また、アプリケーションが複数のファイルから構成される場合は、このディレクトリの中にすべてを置いてください。

4.2. ワーキングセットの選択

STM32CudeIDEでは、プログラムの各種設定のセットをワーキングセットと呼びます。ワーキングセットは、一つのプロジェクトに複数作ることができます。
本プロジェクトは初期状態では、「Release」と「Debug」の二つのワーキングセットが作られています。通常、デバッグ時には「Debug」を使用します。
ワーキングセットは、メニュー[Project]→[Build configurations]→[Set Active]から選択することができます。
「Release」と「Debug」の設定内容の相違を以下に示します。なお、ワーキングセットの各種設定は、アプリケーション・プログラムに応じて変更してください。

項目	Debug	Release
最適化レベル	None(-O0)	Optimize for size (-Os)
デバッグ・レベル	Maximum(-g3)	Default(-g)

4.3. プログラムのビルド

Project Explorer上で[mtk3bsp_nucleo_stm32h723]が選択されている状態で、メニュー[Project]→[Build Project]を選択すると、プログラムがビルド（コンパイル、リンク）され、実行コード・ファイル「mtk3bsp_nucleo_stm32h723.elf」が生成されます。
実行コード・ファイルは、プロジェクトのディレクトリの下の、ワーキングセットと同名のディレクトリの中に生成されます。

5. アプリケーションの実行

生成したアプリケーション・プログラムの実行コードをマイコンボード上で実行する方法を説明します。

5.1. デバッグ環境の準備

STM32CudeIDEを実行しているパソコンとマイコンボード（Nucleo-144）をUSBケーブルで接続します。ボードにはデバッグプローブの機能を備わっていますので、USB経由でパソコンからマイコンボードへのプログラムの転送およびデバッグ実行を行うことができます。

また、パソコンから仮想COMポート(シリアル通信ポート)として、マイコンボードとの接続が認識されます。パソコン上でターミナルエミュレータ（例：TeraTerm ※1）を実行することにより、マイコンボードと通信を行うことができるようになります。
µT-Kernel 3.0 BSPでは、デバッグ用シリアル出力をこの仮想COMポートに送信します。また、µT-Kernel 3.0のシリアル通信デバイスを使用してパソコンと通信を行うことも可能です。デバッグ用シリアル出力のシリアルポート設定は以下とします。

項目	設定内容
通信速度	115200bps
データ	8bit
パリティ	none
ストップビット	1bit

※1 Tera Term Home Page https://ttssh2.osdn.jp/

5.2. デバッグ実行

アプリケーション・プログラムのデバッグ実行は、以下の手順で行います。なお、(1)から(3)までの操作は最初に1回のみ行うだけです。
デバッグ実行を行うことにより、プログラムの実行コードはマイコンのFlash ROMに書き込まれます。よって以降は、STM32CudeIDEを使用せずに単独でマイコンボードに電源を入れれば、プログラムは実行されます。

(1) アプリケーション・プログラムのビルドを行った後にメニュー[Run]→[Debug configurations]を選択します。

(2) 開いたダイアログから項目[STM32 C/C++ Application]を選択し、[New configuration]ボタンを押します。「mtk3bsp_nucleo_stm32h723」が新規構成として追加されます。

(3) 追加された構成「mtk3bsp_nucleo_stm32h723」を選択して、以下の設定を行っていきます。すでに正しい値が設定されている場合は変更の必要はありません。その他の設定についても作成するプログラムの必要に応じて変更してください。

- [Main]タブ [Project:]プロジェクトを指定します。 [C/C++ Application:]にビルドしたELFファイルを指定します。

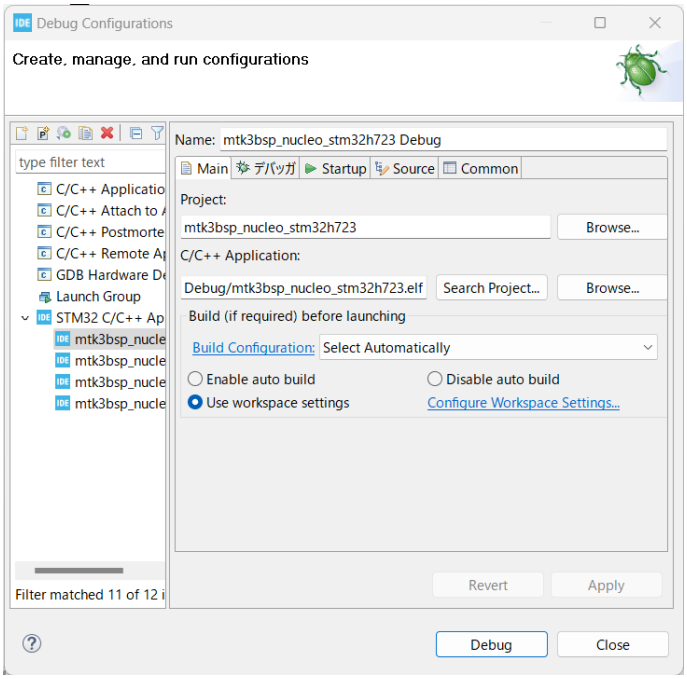

図2 [Debug configurations]の[Main]タブ

- 「Startup」タブ [Set breakpoint at:]の設定にデバッグ開始時にブレーク(一時停止)させる関数名を設定します。初期値の「main」だとμT-Kernel 3.0 の内部でブレークしてしまうので、アプリケーション・プログラムに合わせて変更します。通常はアプリケーション・プログラムの開始関数である「usermain」とします。

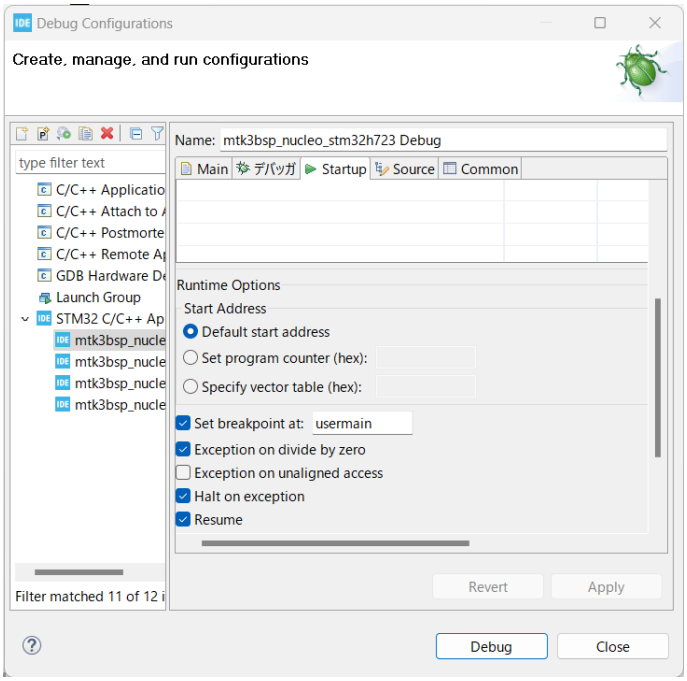

図3 [Debug configurations]の[Startup]タブ

(4) [Debug]ボタンを押すとプログラムがマイコンボードに転送され、STM32CudeIDEはデバッグ画面に変わります。

(5) [Resume]ボタンを押しプログラムを実行すると設定したブレークポイントで停止します。再び[Resumeボタン]を押すとサンプル・プログラムが実行されます。サンプル・プログラムは、マイコンボード上のLEDを点滅させます。また、デバッグ出力に「User program started」の文字列を出力しますので、パソコンでターミナルエミュレータを実行していれば、その画面上に文字列が表示されます。

6. ハードウェアに関わる実装仕様

STM32H723 Nucleo-144向けのμT-Kernel 3.0の ハードウェアに関わる実装仕様を記します。なお、STM32H7マイコンのOSの実装仕様は、「μT-Kernel 3.0 STM32H7マイコン向け実装仕様書」も参考にしてください。

6.1. 基本実装仕様

6.1.1. マイコン関連

実装対象のマイコンであるSTM32H723の基本的な仕様を以下に示します。

項目	内容
CPUコア	ARM Cortex-M7
最大CPU動作周波数	80MHz
ROM	1MB(フラッシュメモリ)
RAM	SRAM(D1) 320KB, SRAM(D2) 32KB, SRAM(D3) 16KB ITCM RAM 64KB, DTCM RAM128KB
内蔵周辺モジュール	タイマー、I2C、SPI、UARTなど

本BSPではメモリはROM(フラッシュメモリ)およびSRAM(D1ドメイン)を使用します。

6.1.2. 割込み定義

割込み関連の定義を以下に記します。

名称	値	意味
N_INTVEC	190	外部割込み数
N_SYSVEC	16	例外数
INTPRI_BITWIDTH	4	割込み優先度の幅(ビット数)
INTPRI_MAX_EXTINT_PRI	1	外部割込みの最高優先度
INTPRI_SVC	0	SVC例外の優先度
INTPRI_SYSTICK	1	システムタイマ例外の優先度
INTPRI_PENDSV	15	PendSV例外の優先度

割込み優先度の範囲は0～15です。ただし、ユーザプログラムから使用可能な範囲は1～15です。

6.1.3. OS内部で使用する割込み

OSの内部で使用する割込みには、以下のようにマイコンの割込みまたは例外が割り当てられます。

種類	割込み番号	割り当てられる割込み・例外	優先度

種類	割込み番号	割り当てられる割込み・例外	優先度
システムタイマ割込み	15	SysTick	1
ディスパッチ要求	14	PendSV	15
強制ディスパッチ要求	14	PendSV	15

6.1.4. クロックの設定

マイコンのクロックは以下のように設定されます。

項目	値(MHz)	備考
SYSCLK	550	システムクロック
HCLK	SYSCLK/2	周辺クロック(AHB3)
D1PCLK1	SYSCLK/4	周辺クロック(APB3)
HCLK12	SYSCLK/2	周辺クロック(AHB1&2)
PCLK1	SYSCLK/4	周辺クロック(APB1)
PCLK2	SYSCLK/4	周辺クロック(APB2)
HCLK4	SYSCLK/2	周辺クロック(AHB4)
D3PCLK1	SYSCLK/4	周辺クロック(APB4)
TMCLK	SYSCLK	システムタイマクロック

6.1.5. 周辺モジュールの初期設定

(1) リセットの解除 OS起動時の初期化で以下の周辺モジュールのクロック供給が有効になります。

- コンフィギュレーションUSE_SDEV_DRVが無効(0)の場合

周辺モジュール名	用途
SYSCFG	システム設定コントローラ
GPIO B, C, D, E	I/Oポート
USART3	デバッグ用シリアル入出力(T-Monitor)
TIM2～TIM5	物理タイマ用タイマ

- コンフィギュレーションUSE_SDEV_DRVが有効(1)の場合

周辺モジュール名	用途
SYSCFG	システム設定コントローラ
GPIO A, B, C, D, E	I/Oポート
USART3	デバッグ用シリアル入出力(T-Monitor)

周辺モジュール名	用途
TIM2～TIM5	物理タイマ用タイマ

デバイスドライバが使用する周辺モジュールのクロック有効化は、各デバイスドライバの初期化処理で行います。これはクロック・ソースの選択などの設定がデバイスドライバで可能とするためです。よって、USE_SDEV_DRVが有効でもOSの初期化処理では周辺モジュールのクロック供給を有効にはしません。

(2) I/O端子の設定 OS起動時の初期化で以下のようにI/O端子が設定されます。

● コンフィギュレーションUSE_SDEV_DRVが無効(0)の場合

I/Oポート名	機能名	用途
PB0	出力ポート	オンボードLED (Green)
PB14	出力ポート	オンボードLED (Red)
PC13	入力ポート	オンボードボタン
PD8	USART3_TX	デバッグ用シリアル入出力(T-Monitor)
PD9	USART3_RX	同上
PE1	出力ポート	オンボードLED (Yellow)

● コンフィギュレーションUSE_SDEV_DRVが有効(1)の場合

I/Oポート名	機能名	用途
PA3	A/DC in	アナログ信号(A/DCドライバ)
PB0	出力ポート	オンボードLED (Green)
PB14	出力ポート	オンボードLED (Red)
PC13	入力ポート	オンボードボタン
PD8	USART3_TX	デバッグ用シリアル入出力(T-Monitor)
PD9	USART3_RX	同上
PE1	出力ポート	オンボードLED (Yellow)

6.2. 物理タイマ機能

6.2.1. 使用するハードウェアタイマ

物理タイマ機能は、マイコン内蔵の以下の汎用タイマを使用します。物理タイマとして使用していない汎用タイマは他の用途に使用可能です。

物理タイマ番号	対応するタイマ	ビット幅
1	TIM2	32
2	TIM3	16

物理タイマ番号	対応するタイマ	ビット幅
3	TIM4	16
4	TIM5	32

6.2.2. 物理タイマの設定

物理タイマの各設定は以下のファイルで定義されています。

```
include\sys\sysdepend\cpu\stm32h7\sysdef.h
```

物理タイマのカウンタ周波数は、初期設定では汎用タイマへのクロック周波数（550MHz）を未分周で使用します。
以下の設定値を変更することにより周波数の分周値を変更できます。この値は汎用タイマのTIMx_PSCレジスタに設定されます。

```
#define TIM2PSC_PSC_INIT    0
#define TIM3PSC_PSC_INIT    0
#define TIM4PSC_PSC_INIT    0
#define TIM5PSC_PSC_INIT    0
```

物理タイマの割込み優先度は、以下の設定値を変更することにより変更できます。

```
include\sys\sysdepend\cpu\stm32h7\sysdef.h
```

```
#define INTPRI_TIM2     5
#define INTPRI_TIM3     5
#define INTPRI_TIM4     5
#define INTPRI_TIM5     5
```

6.2.3. 物理タイマ割込み

物理タイマはその内部処理で汎用タイマの割込みを使用します。汎用タイマの割込みは以下のように定義されます。

物理タイマ番号	汎用タイマ	割込み番号	割込み優先度(※1)
1	TIM2	28	5
2	TIM3	29	5
3	TIM4	30	5

物理タイマ番号	汎用タイマ	割込み番号	割込み優先度(※1)
4	TIM5	50	5

※1 割込み優先度は前述のINTPRI_TIMxを変更することにより変更可能です。

6.2.4. デバイスドライバ

本BSPは、TRONフォーラムが提供するμT-Kernel 3.0のサンプル・デバイスドライバを、STM32H723 Nucleo-144向けに移植し実装しています。

種別	対象I/Oデバイス	デバイス名
シリアル通信デバイスドライバ	USART1	sera
	USART2	serb
	USART3	serc
	UART4	serd
	UART5	sere
	USART6	serf
	UART7	serg
	UART8	serh
	UART9	seri
	USART10	serj
A/D変換デバイスドライバ	ADC1	adca
	ADC2	adcb
	ADC3	adcc
I2C通信デバイスドライバ	I2C1	iica
	I2C2	iicb
	I2C3	iicc
	I2C4	iicd
	I2C5	iice

6.3. T-Monitor互換ライブラリ

6.3.1. コンソール入出力

T-Monitor互換ライブラリのAPIによるコンソール入出力の仕様を以下に示します。

項目	内容

項目	内容
使用デバイス	内蔵UART (USART0)
ボーレート	115200bps
データ形式	data 8bit, stop 1bit, no parity

7. 更新履歴

版数	日付	内容
1.00.02	2023.05.26	・タイトルを「マニュアル」から「ユーザズ・マニュアル」に変更 ・内容および構成の全面見直し
1.00.00	2022.02.14	・初版