



life.augmented



マイコン初心者にも簡単！ 高性能STM32H7シリーズ入門

STマイクロエレクトロニクス株式会社
マイクロコントローラ & デジタル製品グループ
マイクロコントローラ製品技術部 部長

原文雄



拡大し続けるSTM32ポートフォリオ



マイクロ
プロセッサ

★
ハイパフォーマンス
マイコン

»
メインストリーム
マイコン

🔋
超低消費電力
マイコン

📶
ワイヤレス
マイコン

STM32MP1
Up to 1 GHz Cortex-A7
209 MHz Cortex-M4

STM32MP2
Dual 1.5 GHz Cortex-A35
400 MHz Cortex-M33

STM32F7
1082 CoreMark
216 MHz Cortex-M7

STM32H7
Up to 3224 CoreMark
Up to 550 MHz Cortex -M7
240 MHz Cortex -M4

STM32N6
MCU with neural
processing unit

STM32F2
Up to 398 CoreMark
120 MHz Cortex-M3

STM32F4
Up to 608 CoreMark
180 MHz Cortex-M4

STM32H5
Up to 1023 CoreMark
250 MHz Cortex-M33

STM32F3 245 CoreMark 72 MHz Cortex-M4
STM32G4 569 CoreMark 170 MHz Cortex-M4
ミックスド・シグナル・マイコン

STM32C0
114 CoreMark
48 MHz Cortex M0+

STM32F0
106 CoreMark
48 MHz Cortex-M0

STM32G0
142 CoreMark
64 MHz Cortex-M0+

STM32F1
177 CoreMark
72 MHz Cortex-M3

STM32L0
75 CoreMark
32 MHz Cortex-M0+

STM32U0
118 CoreMark
48 MHz Cortex M0+

STM32L4
273 CoreMark
80 MHz Cortex-M4

STM32L4+
409 CoreMark
120 MHz Cortex-M4

STM32L5
443 CoreMark
110 MHz Cortex-M33

STM32U5
651 CoreMark
160 MHz Cortex-M33

STM32WL
162 CoreMark
48 MHz Cortex-M4
48 MHz Cortex-M0+

STM32WB
216 CoreMark
64 MHz Cortex-M4
32 MHz Cortex-M0+

STM32WBA
407 CoreMark
100 MHz Cortex-M33



■ 直近に量産移行した製品 □ 2024年発表の新シリーズ □ 開発中



STM32H7シリーズの特徴

STM32マイコンのポートフォリオを拡張する新製品

シリーズ最高性能の実現

- デュアルコア品 最大 2424 + 800 CoreMark
(Arm®Cortex®-M7 @480MHz + Cortex-M4 @240MHz)
- シングルコア品 最大 3174 CoreMark
(Cortex-M7 @600MHz)



柔軟なシングル / デュアルコア・アーキテクチャ

産業・セキュリティ・AIアプリケーションに最適

- グラフィック制御アシスト
- 高速データ転送
- 高度な周辺機能



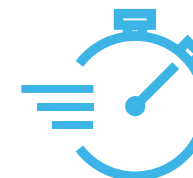
高度なセキュリティ機能

- 暗号ハッシュ
- Cortex-M7セキュリティ・サービス



開発を加速する豊富なエコシステム

- 各種SWツール
- 評価用ボード
- コミュニティ & 豊富なパートナー





170品種以上をラインアップ！ STM32H7シリーズ 製品ポートフォリオ

ブート Flash

STM32H7R3/7S3	STM32H7R7/7S7
600MHz / 1284DMIPS	600MHz / 1284 DMIPS
SRAM 620KB	SRAM 620KB
64K user flash	64K user flash
128K ST-iRoT	128K ST-iRoT
Chrom-ART	NeoChrom

デュアル コア

STM32H745/755	STM32H747/757
480 + 240MHz	480 + 240MHz
1027 + 300DMIPS	1027 + 300DMIPS
RAM 1 MB	RAM 1 MB
Flash up to 2 MB	Flash up to 2 MB

シングル コア

STM32H7A3/B3	STM32H742	STM32H743/753	STM32H725/735	STM32H723/733
280MHz / 599DMIPS	480MHz / 1027DMIPS	480MHz / 1027DMIPS	550MHz / 1177DMIPS	550MHz / 1777DMIPS
RAM 1.4MB	RAM 692 KB	RAM 1 MB	RAM 564KB	RAM 564 KB
Flash up to 2MB	Flash up to 2 MB	Flash up to 2 MB	Flash up to 1MB	Flash up to 1 MB

バリュー ライン

STM32H7B0	STM32H750	STM32H730
280MHz / 599DMIPS	480MHz / 1027DMIPS	550MHz / 1177DMIPS
RAM 1.4MB	RAM 1 MB	RAM 564 BB
Flash 128KB	Flash 128 KB	Flash 128 KB

Arm® Cortex®

-M7

-M7 & -M4





STM32H723 / H733 汎用ライン

豊富な周辺機能、高性能、大容量メモリ、3つのパワー・ドメイン、LDO、
STM32H753 / H743ラインとのパッケージ互換

- パフォーマンス
- RAM配置
- ECC（メモリ全領域）
- 複数の高速 A/DC
- グラフィック
- セキュリティ
- 数値演算アクセラレータ
- メモリ・インタフェース
- 通信インタフェース
- 電源供給スキーム
- STM32 DNA（互換性を考慮、パッケージ、エコシステム等）



STM32H723 / H733 汎用ライン



(*) : STM32H733 でのサポート(暗号機能あり)



STM32開発エコシステム

ソフトウェア

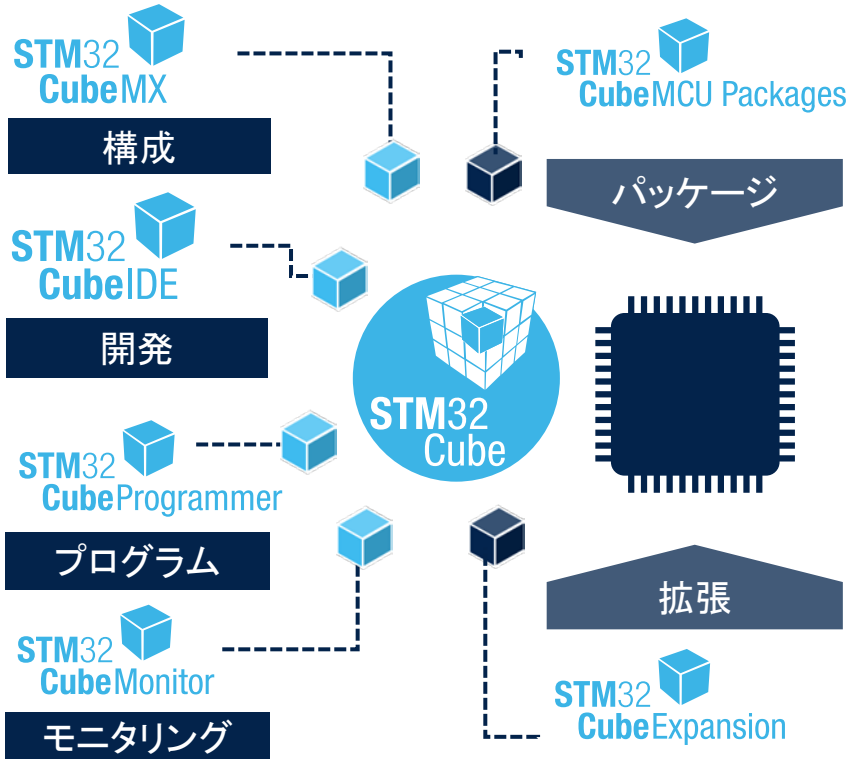
ハードウェア

カスタマーサポート

ソフトウェア ツール 組み込みソフトウェア

STM32 Nucleo-144ボード

FAE - Worldwide カスタマ・サポート



ディスカバリ・キット



FAE - Worldwide カスタマ・サポート



community.st.com

MOOC

Partner Program

life.augmented

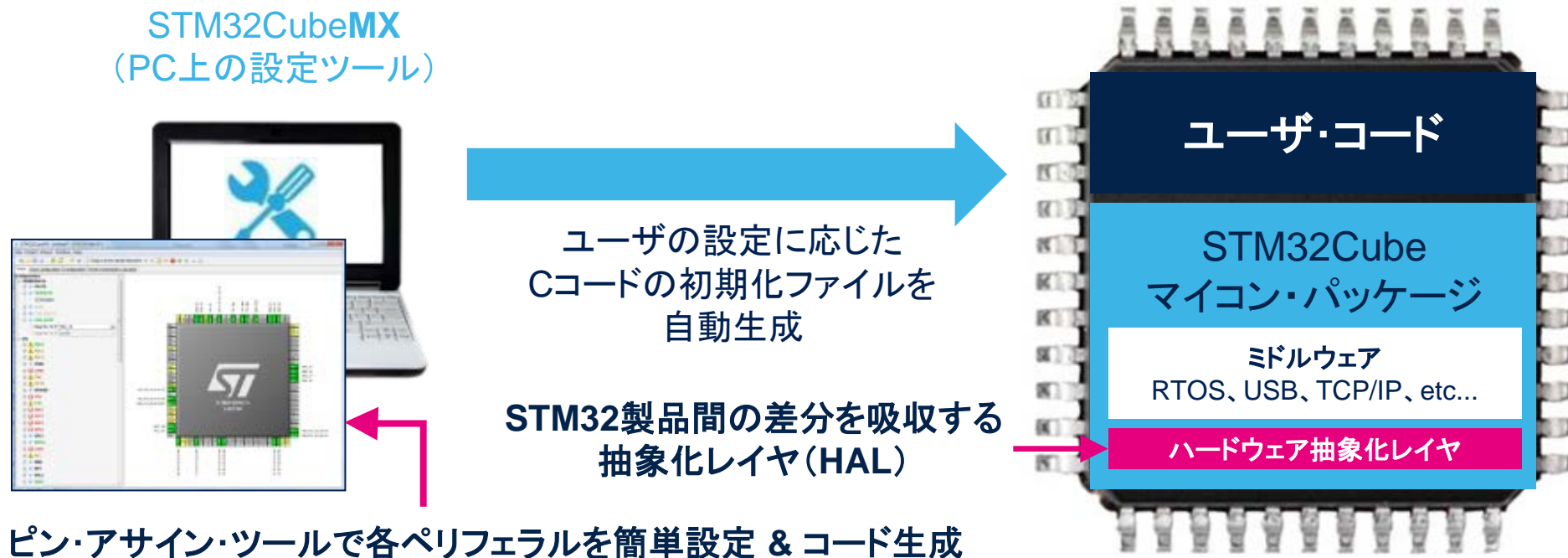
STM32CubeMX 組み込みソフトウェア開発支援ツール





組み込みソフトウェア開発支援ツール STM32CubeMX

STM32CubeMXを使用した開発形態

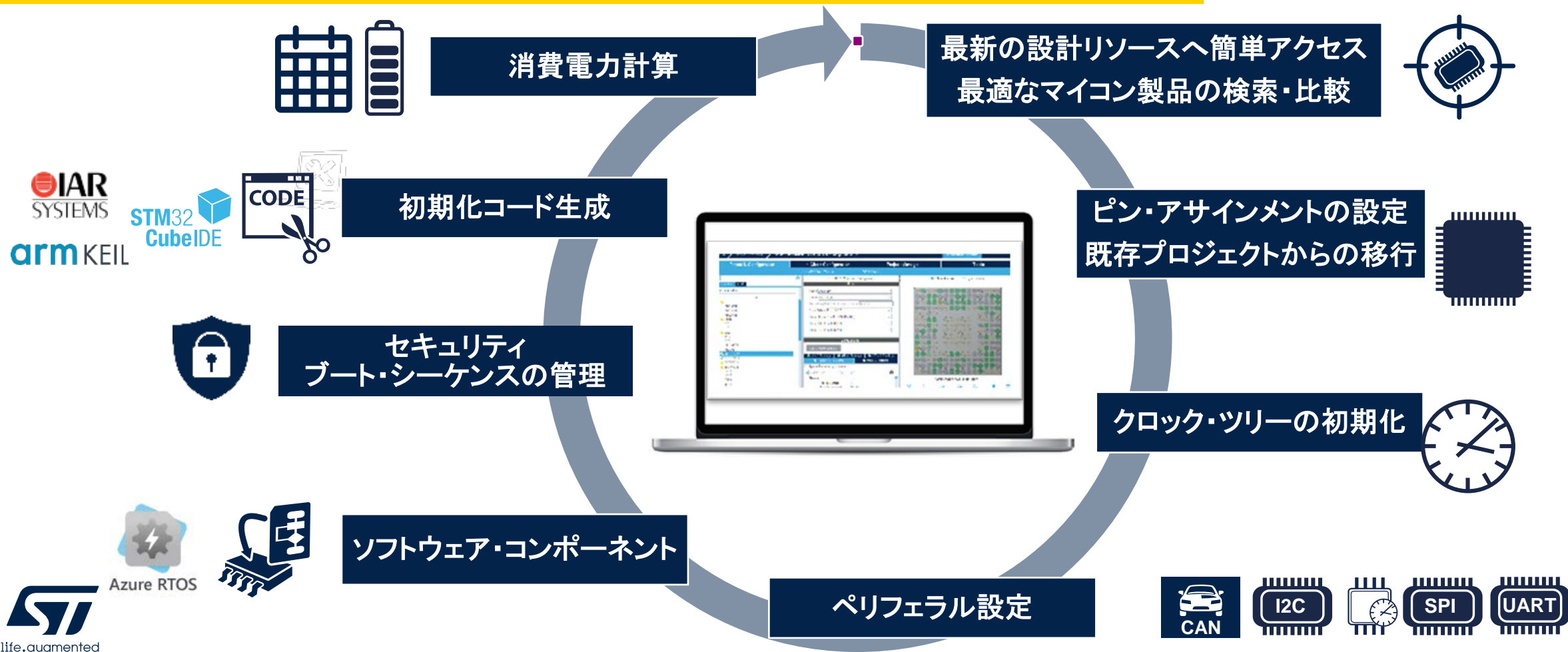


- ペリフェラル、クロック、割り込み、DMA等の設定コードを自動生成するツール
- ユーザの開発負荷を削減 & STM32間のソフトウェア移植性を実現



STM32CubeMX 開発支援ツール

さまざまなマイコン設定をGUIで簡単制御
初期プロジェクトを生成し、組込み製品開発の期間短縮に貢献



Our technology starts with You



Find out more at [STM32H7](https://www.st.com/STMicroelectronics/Products/Processors/32-bit/ST32H7)

© STMicroelectronics - All rights reserved.

ST logo is a trademark or a registered trademark of STMicroelectronics International NV or its affiliates in the EU and/or other countries.

For additional information about ST trademarks, please refer to www.st.com/trademarks.

All other product or service names are the property of their respective owners.



life.augmented



life.augmented

NUCLEO-H723ZG +CubeIDE プログラミング

2024年05月10日

STマイクロエレクトロニクス株式会社
マイクロコントローラ製品技術部



STM32 マイコン開発ボード (NUCLEO)

STM32マイコン搭載プロトタイプ作成用
オープン開発プラットフォーム



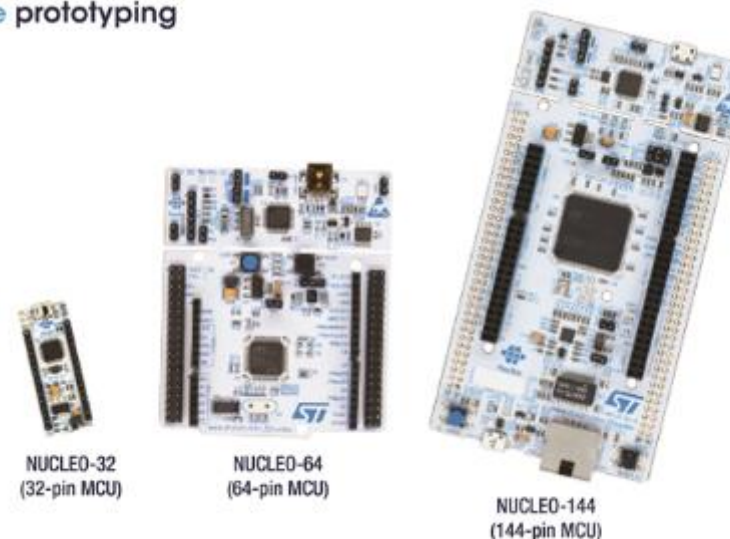


STM32 Nucleoボード 概要

特 徴

- 32 / 64 / 144ピン・パッケージのSTM32マイコン搭載
- ST-LINK/V3 デバッガ / プログラマ搭載
 - 仮想のcomポート (Virtual com port)
 - マス・ストレージ
- 専用シールドによる広範な拡張機能
 - Arduino Uno Rev3接続に対応 (Nucleo-64 / Nucleo-144)
 - ST Zioコネクタを介して幅広いペリフェラルにアクセス可能 (Nucleo-144)
 - ST Morphoコネクタを介してマイコンの全端子にアクセス可能 (Nucleo-64 / Nucleo-144)
 - Arduino-Nanoコネクタ (Nucleo-32)
- mbedオンライン・リソースへの直接アクセス
- IAR、Keil、arm mbedオンライン、GCCベースのIDEによるサポート

Flexible prototyping



arm
MBED



全72種類

STM32 Nucleoボードのラインアップ (2024年4月時点)

★ ハイパフォーマンスマイコン

STM32F2 144: F207ZG*	STM32F4 64: F401RE, F410RB, 64: F411RE, F446RE 144: F412ZG, F413ZH 144: F439ZI* 144: F446ZE	STM32H5 64: H503RB 64: H533RE ● 144: H563ZI	STM32F7 144: F722ZE 144: F756ZG*, F767ZI*	STM32H7 144: H753ZI* 144: H723ZG*, H7A3ZI-Q 144: H755ZI-Q* 255: H7S3L8* ●●
--------------------------------	---	---	--	---

➡➡ メインストリームマイコン

STM32C0 64(48): C031C6	STM32F0 32: F031K6, F042K6 64: F030R8, F070RB 64: F072RB, F091RC	STM32G0 32: G031K8 64: G070RB, G071RB 64: G0B1RE	STM32F1 64: F103RB	STM32F3 64: F302R8, F303K8 64: F303RE, F334R8 144: F303ZE	STM32G4 32: G431KB 64: G431RB, G474RE 64: G491RE
----------------------------------	--	--	------------------------------	---	--

🔋 超低消費電力マイコン

STM32L0 32: L011K4, L031K6 64: L010RB, L053R8 64: L073RZ	STM32U0 64: U031R8, ● U083RC ●	STM32L4 32: L412KB, L432KC 64: L452RE, L476RG 64: L412RB-P, L433RC-P 64: L452RE-P 144: L496ZG*, L4A6ZG 144: L496ZG-P*	STM32L4+ 144: L4P5ZG, L4R5ZI 144: L4R5ZI-P	STM32L5 144: L552ZE-Q*	STM32U5 144: U575ZI-Q ● 144: U5A5ZJ-Q ● 64: U545RE-Q
--	--	--	---	----------------------------------	--

📶 ワイヤレスマイコン

STM32WL 64: WL55JC	STM32WB WB55 48: WB15CC 64: WB55RG	STM32WBA 64: WBA52CG, WBA55CG ●
------------------------------	--	---

マセンダ : SMPS搭載 (-P: 外部SMPS、-Q: 内蔵SMPS)

*: イーサネット : 10/100Mbps

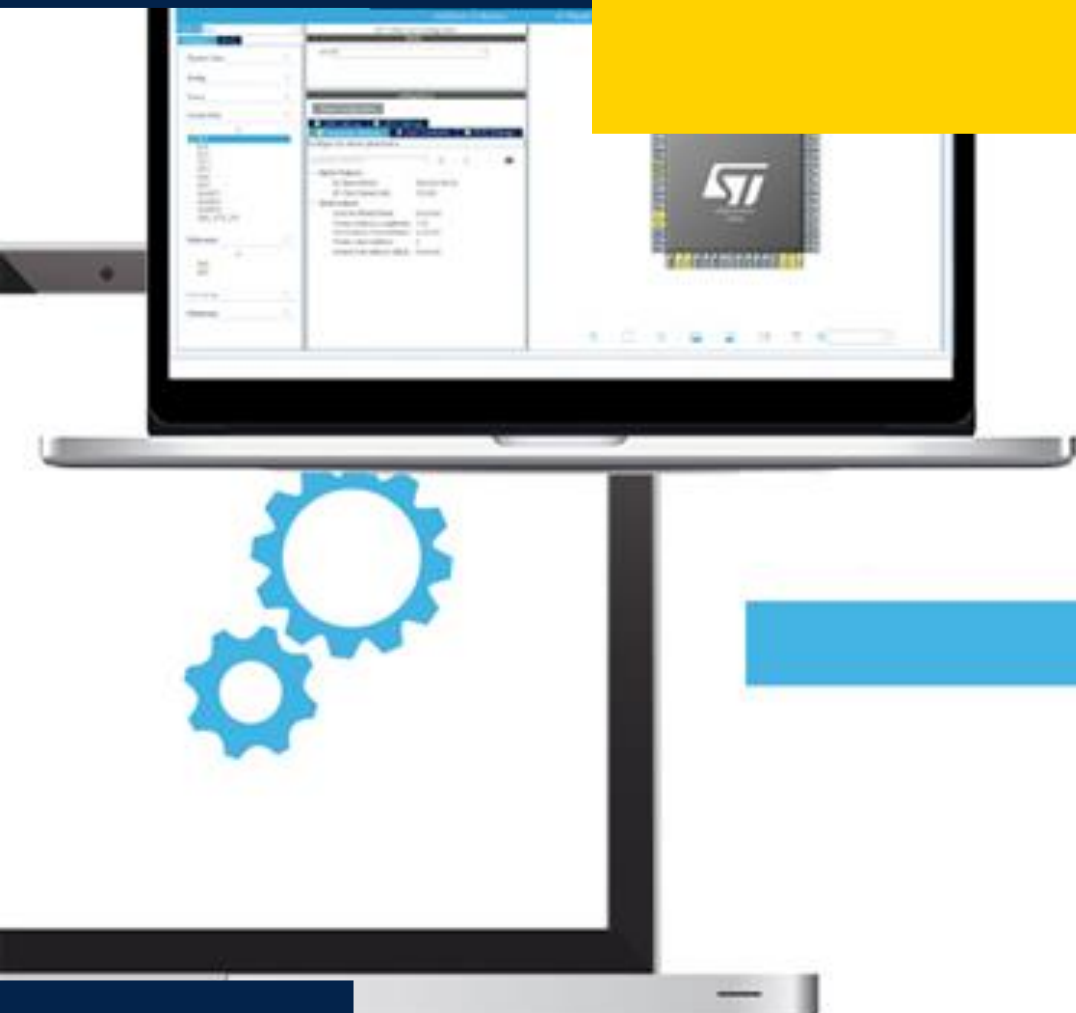
USB dongle付き68ピン

HW暗号化 / ハッシュ・プロセッサ搭載

- 新ボード
- USB Type-C® シンク
- USB-PD



ツールの説明





開発環境

評価ボード

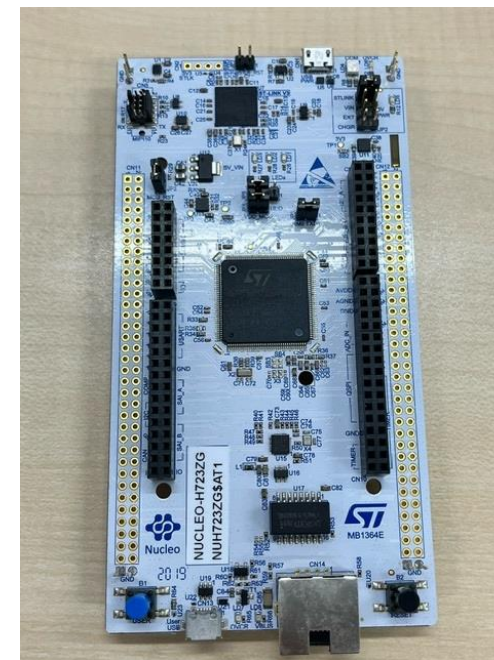
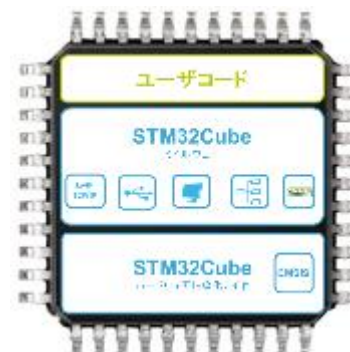
- NUCLEO-H723ZG

統合開発環境

- ST社 STM32CUBEIDE

ソフトウェア・ライブラリ

- STM32CubeMX
- STM32CubeH7





確認事項

1. NUCLEO-H723ZG

- URL: <https://www.stmcu.jp/design/hwdevelop/nucleo/77343/>
- 関連ドキュメント
 - UM2407 STM32H7 Nucleo-144 ボード (MB1364) ユーザー マニュアル
 - URL: https://www.stmcu.jp/design/document/users_manual/65701/

2. STM32CubeIDE

- URL: https://www.stmcu.jp/design/sw_dev/pc_soft/66469/
- 関連ドキュメント
 - UM2609 STM32CubeIDE ユーザ・ガイド (和訳版あり)
 - URL: https://www.stmcu.jp/design/document/users_manual/76085/
- Wiki Page
 - [URL:https://wiki.st.com/stm32mcu/wiki/STM32CubeIDE:Introduction_to_STM32CubeIDE](https://wiki.st.com/stm32mcu/wiki/STM32CubeIDE:Introduction_to_STM32CubeIDE)

注意！
CubeIDEは管理者権限でインストールしましょう。



確認事項

3. STM32CubeMX

- URL: https://www.stmcu.jp/design/sw_dev/pc_soft/52798/
- 関連ドキュメント
 - UM1718 STM32CubeMX : STM32の設定と初期化のCコード生成
 - URL: https://www.stmcu.jp/design/document/users_manual/53167/
- Wiki Page
 - https://wiki.st.com/stm32mcu/wiki/Introduction_to_STM32CubeMX

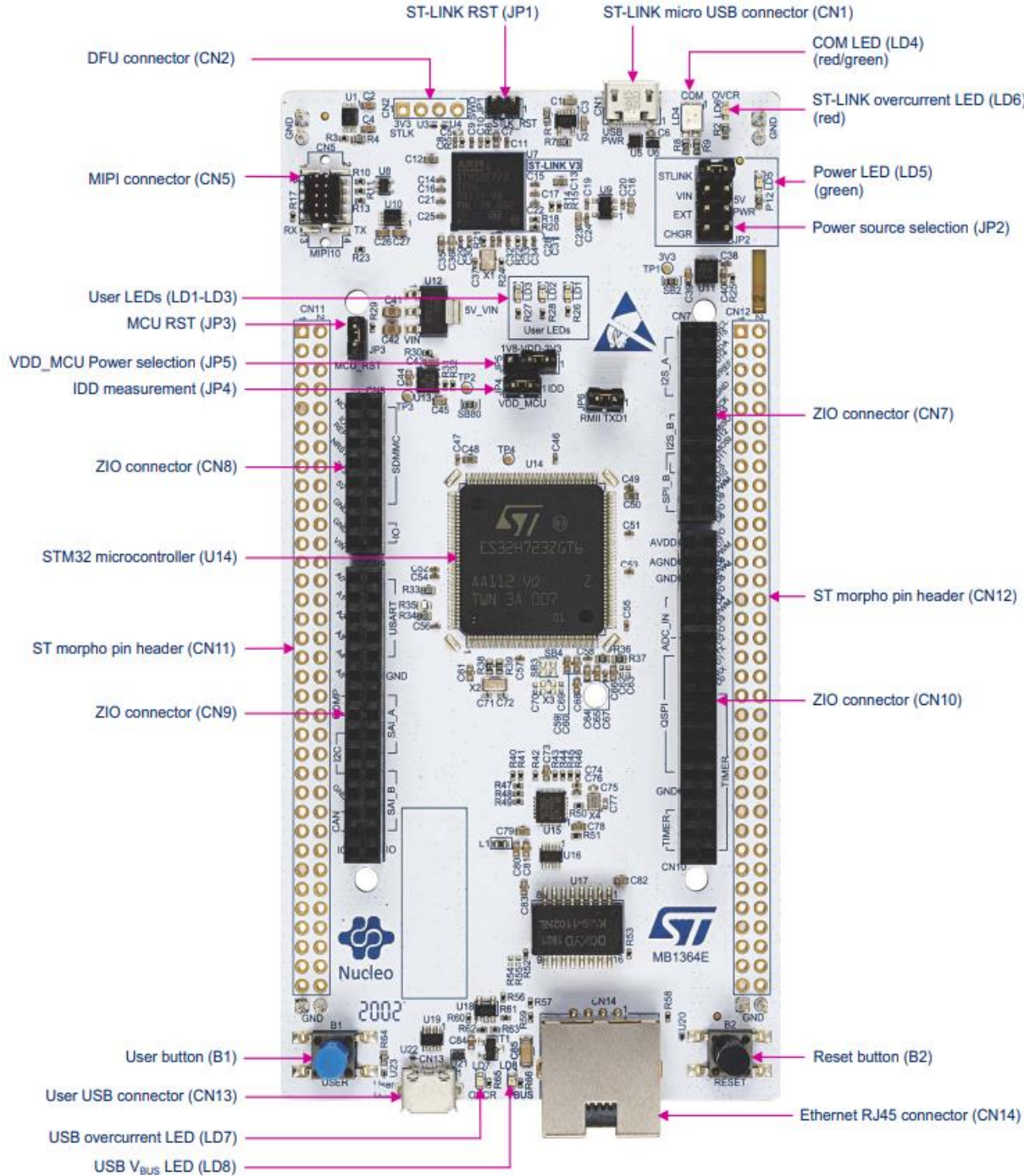
注意！
CubeMXは管理者権限でインストールしましょう。

4. STM32CubeH7

- URL: https://www.stmcu.jp/design/sw_dev/firmware/53422/
- 関連ドキュメント
 - UM2217 Description of STM32H7 HAL and low-layer drivers
 - URL: https://www.stmcu.jp/design/document/users_manual/53956/

注意！
CubeH7のインストール時には日本語表記のディレクトリ名はなるべく控えましょう。

Nucleo ボード説明



USBケーブルを使用してCN1コネクタとPCを接続するとボードを動作、デバックすることができます。



STM32Cube_FW_H7_V1.11.2

- _htmresc
- Documentation
- Drivers
- Middlewares
- Projects
 - NUCLEO-H7A3ZI-Q
 - NUCLEO-H723ZG
 - Applications
 - Demonstrations
 - Examples
 - BSP
 - COMP
 - CORDIC
 - CRC
 - DAC
 - DMA
 - DTS
 - FLASH
 - FMAC
 - GPIO
 - GPIO_EXTI
 - EWARM
 - Inc
 - MDK-ARM
 - Src
 - STM32CubeIDE

プログラム起動～ロード、デバックまで
順を追って説明します。

CubeIDE起動 サンプル・プログラムを展開

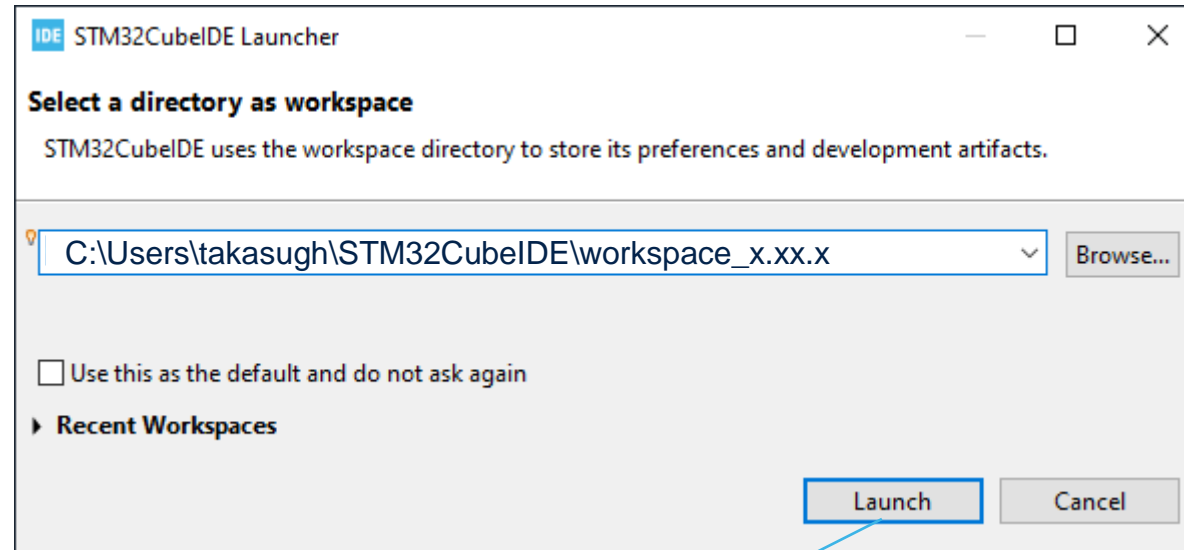
- 既存プロジェクトを開く
- ファイル>開く>ワークスペースにて以下のファイルを指定
 - STM32Cube_FW_H7_VX.X.X \Projects \NUCLEO-H723ZG \Examples \GPIO \GPIO_EXTI \STM32CubeIDE \.project

.settings	5/7/2024 5:24 PM	File folder
Doc	5/7/2024 5:21 PM	File folder
Drivers	5/7/2024 5:21 PM	File folder
Example	4/2/2024 4:33 PM	File folder
IDE .cproject	4/2/2024 5:07 PM	CPROJECT File
IDE .project	4/2/2024 5:07 PM	PROJECT File
STM32H723ZGTX_FLASH.Id	4/2/2024 5:07 PM	LD File

プログラム起動～ロード、デバックまで
順を追って説明します。

CubeIDE起動

- ワークスペースの確認画面



Launchをクリック



プログラム起動～ロード、デバックまで
順を追って説明します。

STM32CubeIDE 説明

STM32CubeIDE 起動画面

現在のワークスペースの名前

メニュー・バー

ツール・バー

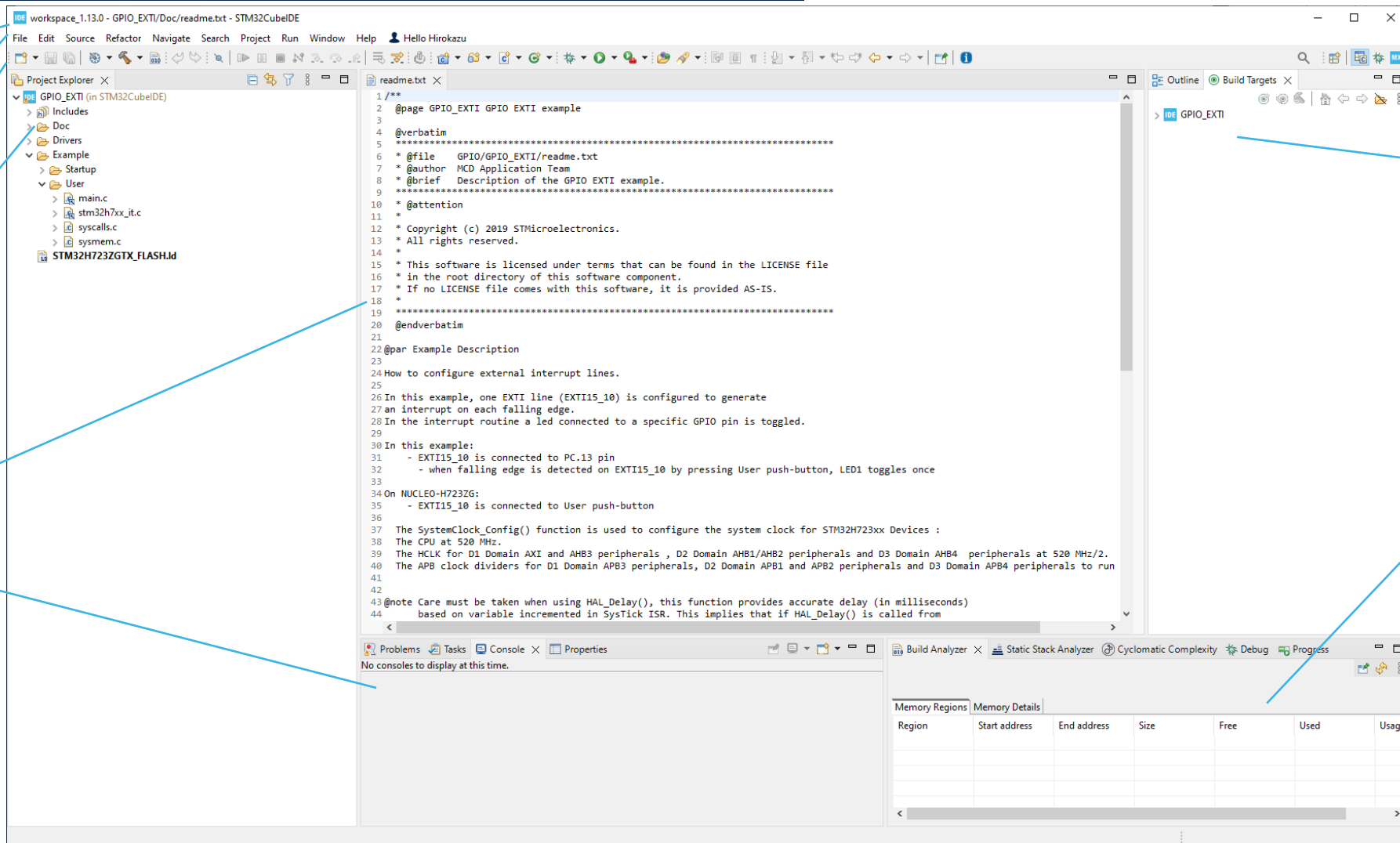
プロジェクト・
エクスプローラ

エディター・
エリア

コンソール・
ビュー

アウトライン・
ビュー

ビルド・
アナライザ





プログラム起動～ロード、デバックまで
順を追って説明します。

ビルド (コンパイル)

プロジェクトの再ビルドを行う

The screenshot shows the STM32CubeIDE interface. The 'Project' menu is open, and 'Build Project' is selected. A blue arrow points from the menu to the build console. The console output shows the build process for the GPIO_EXTI project, including the generation of the GPIO_EXTI.list file and the final build completion message: 19:42:56 Build Finished. 0 errors, 0 warnings. (took 4s.992ms). A blue box with the text '成功' (Success) is overlaid on the console output.

```
workspace_1.13.0 - GPIO_EXTI/Doc/readme.txt - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help Hello Hirokazu
Project Explorer
  GPIO_EXTI (in STM32CubeIDE)
    Includes
    Doc
    Drivers
    Example
    STM32H723ZGTx_FLASH.ld
system.c
GPIO_EXTI GPIO EXTI example
.....
GPIO/GPIO_EXTI/readme.txt
MCD Application Team
Description of the GPIO EXTI example.
.....
.....
Copyright (c) 2019 STMicroelectronics.
All rights reserved.
This software is licensed under terms that can be found in the LICENSE file
in the root directory of this software component.
If no LICENSE file is provided with this software, it is provided AS-IS.
.....
19 *****
20 @endverbatim
21
22 @par Example Description
23
24 How to configure external interrupts.
25
26 In this example, one EXTI line (EXTI15_10) is configured to generate
27 an interrupt on each falling edge.
28 In the interrupt routine a led connected to a specific GPIO pin is toggled.
29
30 In this example:
31   - EXTI15_10 is connected to PC.13 pin
32   - when falling edge is detected on EXTI15_10 by pressing User push-button,
33     LED1 toggles once
34 On NUCLEO-H723ZG:
35   - EXTI15_10 is connected to User push-button
36
37 The SystemClock_Config() function configures the system clock for STM32H723xx Devices :
38 The CPU at 520 MHz.
39 The HCLK for D1 Domain AXI
40 The APB clock dividers for
41
42
43 @note Care must be taken when using HAL_Delay(), this function uses a
44 counter based on variable incremented in SysTick ISR. This
CDT Build Console [GPIO_EXTI]
arm-none-eabi-size GPIO_EXTI.elf
arm-none-eabi-obfdump -h -S GPIO_EXTI.elf > "GPIO_EXTI.list"
text data bss dec hex filename
8148 28 1572 9748 2614 GPIO_EXTI.elf
Finished building: default.size.stdout
Finished building: GPIO_EXTI.list
19:42:56 Build Finished. 0 errors, 0 warnings. (took 4s.992ms)
```




プログラム起動～ロード、デバックまで
順を追って説明します。

ロード

ターゲットにダウンロードしてデバッガを起動

The screenshot shows the STM32CubeIDE interface. The 'Run' menu is open, and the 'Debug As' option is selected. A dialog box titled 'Confirm Perspective Switch' is displayed, asking to switch to the '1 STM32 C/C++ Application' perspective. The 'Switch' button is highlighted.

workspace_1.13.0 - GPIO_EXTI/Doc/readme.txt - STM32CubeIDE

File Edit Source Refactor Navigate Search Project **Run** Window Help Hello Hirokazu

Project Explorer

- GPIO_EXTI (in STM32CubeIDE)
- Binaries
- Includes
- Debug
- Doc
- Drivers
- Example
 - GPIO_EXTI.launch
 - STM32H723ZGTX_FLASH.Id

Resume
Suspend
Terminate
Disconnect
Step Into
Step Over
Step Return
Run to Line
Use Step Filters Shift+F5
Run
Debug F11
Run History
Run As
Run Configurations...
Debug History
Debug As
Debug Configurations...
Breakpoint Types
Toggle Breakpoint Ctrl+Shift+B
Toggle Line Breakpoint
Toggle Watchpoint

GPIO_EXTI example

GPIO_EXTI/readme.txt

ication Team

ion of the GPIO EXTI example.

9 STMicroelectronics.

ed.

licensed under terms that can be
tory of this software component.
e comes with this software, it is

ernal interrupt lines.

EXTI line (EXTI15_10) is configur
falling edge.

line a led connected to a specific GPIO pin is toggled.

IDE Confirm Perspective Switch

This kind of launch is configured to open the Debug perspective when it suspends.

This Debug perspective supports application debugging by providing views for displaying the debug stack, variables and breakpoints.

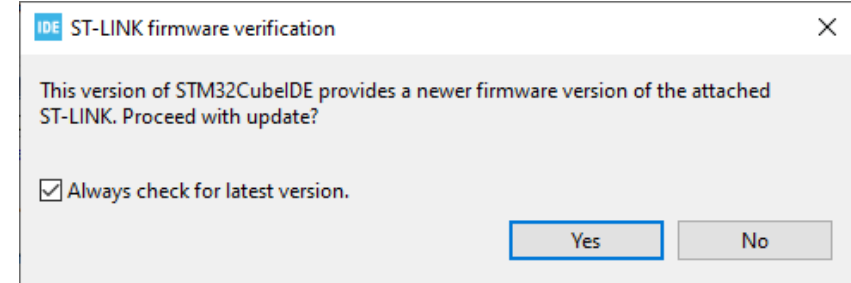
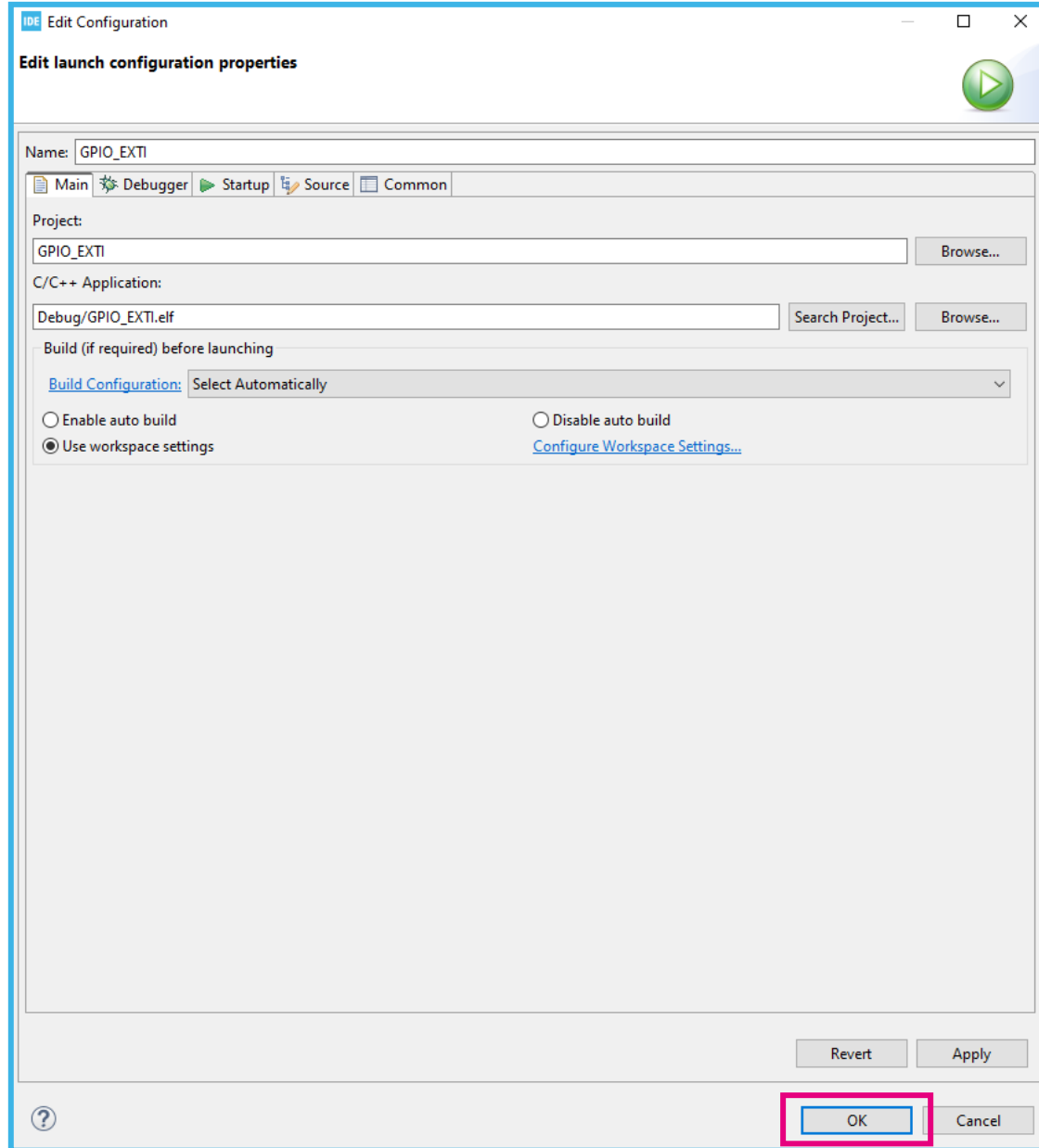
Switch to this perspective?

Remember my decision

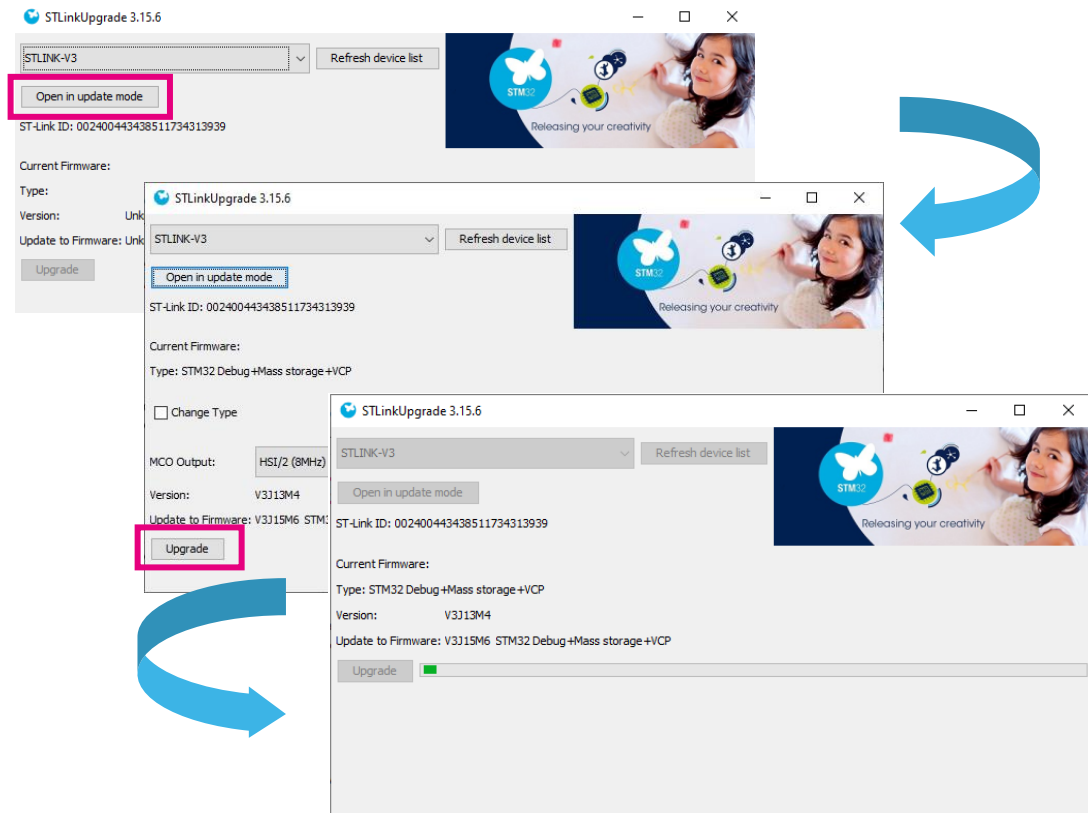
Switch No

プログラム起動～ロード、デバックまで
順を追って説明します。

ロード



↑ ST-Linkのファームウェアが古い場合こちらのメッセージが表示されます。YESをクリックして更新してください。



プログラム起動～ロード、デバックまで
順を追って説明します。

ロード & RUN

The screenshot shows the STM32CubeIDE interface. The main.c file is open, and the function `main()` is visible. A callout box highlights the line `MPU_Config();` at line 50, which is circled in pink. A blue arrow points from this line to a blue box containing the text "プログラム・カウンタ (PC) の位置". The console at the bottom shows the message "Download verified successfully".

```
38 static void EXTI15_10_IRQHandler(void);
39 static void CPU_CACHE_Enable(void);
40 /* Private functions -----*/
41
42 /**
43  * @brief Main program
44  * @param None
45  * @retval None
46  */
47 int main(void)
48 {
49     /* Configure the MPU attributes */
50     MPU_Config();
51
52     /* Enable the CPU Cache */
53     CPU_CACHE_Enable();
54
55     /* STM32H7xx HAL library initialization:
56     - SysTick timer is configured by default as source of time base, but user
57     can eventually implement his proper time base source (a general purpose
58     timer for example or other time source), keeping in mind that time base
59     duration should be kept 1ms since PPP_TIMEOUT_VALUEs are defined and
60     handled in milliseconds basis.
61     - Set NVIC Group Priority to 4
62     - Low Level Initialization
63     */
64     HAL_Init();
65
66     /* Configure the system clock to 520 MHz */
67     SystemClock_Config();
68
69     /* -1- Initialize LEDs mounted on NUCLEO-H723ZG board */
70     BSP_LED_Init(LED1);
71
72     /* -2- Configure EXTI15_10 (connected to PC.13 pin) in interrupt mode */
73     EXTI15_10_IRQHandler_Config();
74
75     /* Infinite loop */
76     while (1)
77     {
78     }
79 }
80
81 /**
```



プログラム起動～ロード、デバックまで
順を追って説明します。

ツール・バー 説明 ロード&RUN

デバッグ用のツールバー

リセット及び 再スタート	全ての ブレーク・ポイント のスキップ	リジューム	ターミネート	ステップ イン	ステップ リターン	
		ターミネート 及び リジューム	サスペンド	接続解除	ステップ オーバー	インストラクション ステップング モード

↑ クリック

リジュームをクリックすると、プログラムがRUNします。
ボード上の青色ボタン（ユーザーボタン）を押す度に緑色LED（LD1）が点灯、消灯を繰り返します。



プログラム起動～ロード、デバックまで
順を追って説明します。

ツール・バー 説明 サスペンド

デバッグ用のツールバー

リセット及び再スタート 全てのブレーク・ポイントのスキップ リジューム ターミネート ステップイン ステップリターン

ターミネート及びリジューム サスペンド 接続解除 ステップオーバー インストラクションステップモード

クリック

サスペンドをクリックすると、
プログラムが停止します。
停止すると、停止した位置の
プログラムが表示されます。

```
72 /* -2- Configure EXTI15_10 (connected to PC.13 pin) in interrupt mode */  
73 EXTI15_10_IRQHandler_Config();  
74  
75 /* Infinite loop */  
76 while (1)  
77 {  
78 }  
79 }  
80
```



プログラム起動～ロード、デバックまで
順を追って説明します。

ツール・バー 説明 ステップ実行～Cソース

デバッグ用のツールバー



ステップイン、ステップオーバーでプログラムの実行内容をより深く見ていくことができます。

```

63
64 HAL_Init();
65
66 /* Configure the system clock to 520 MHz */
67 SystemClock_Config();
68

```

```

103 static void SystemClock_Config(void)
104 {
105     RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
106     RCC_OscInitTypeDef RCC_OscInitStruct = {0};
107     HAL_StatusTypeDef ret = HAL_OK;
108
109     /*!< Supply configuration update enable */
110     HAL_PWREx_ConfigSupply(PWR_LDO_SUPPLY);

```

```

67 SystemClock_Config();
68
69 /* -1- Initialize LEDs mounted on NUCLEO-H72
70 BSP_LED_Init(LED1);
71

```

ステップ・オーバー

ステップ・イン



プログラム起動～ロード、デバックまで
順を追って説明します。

ツール・バー 説明 ステップ実行～アセンブラ

デバッグ用のツールバー

リセット及び再スタート 全てのブレーク・ポイントのスキップ リジューム ターミネート ステップイン ステップリターン

ターミネート及びリジューム サスペンド 接続解除 ステップオーバー インストラクションステップングモード

アセンブル言語レベルでのより細かい実行を確認することができます。

The screenshot shows an IDE window with several tabs: readme.txt, system.c, startup_stm32h723zgtx.s, and main.c. The main.c tab is active, showing C code for the main function. The assembly window on the right shows the corresponding assembly instructions for the main function, with the instruction at address 08001c0a highlighted: `b1 0x8001e84 <MPU_Config>`. The registers window is also visible, showing the current state of the processor registers.



プログラム起動～ロード、デバックまで
順を追って説明します。

ツール・バー 説明 再スタート

デバッグ用のツールバー

リセット及び 再スタート	全ての ブレーク・ポイント のスキップ	リジューム	ターミネート	ステップ イン	ステップ リターン
	ターミネート 及び リジューム	サスペンド	接続解除	ステップ オーバー	インストラクション ステップング モード

CPUをリセットし、プログラムを再起動します



プログラム起動～ロード、デバックまで
順を追って説明します。

ツール・バー 説明 プログラム終了時

デバッグ用のツールバー

リセット及び 再スタート	全ての ブレーク・ポイント のスキップ	リジューム	ターミネート	ステップ イン	ステップ リターン
		ターミネート 及び リジューム	サスペンド	接続解除	ステップ オーバー
					インストラクション ステップング モード

デバッカの接続を解除します。



プログラム起動～ロード、デバックまで
順を追って説明します。

ツール・バー 説明 再接続

デバッグ用のツールバー

リセット及び 再スタート	全ての ブレーク・ポイント のスキップ	リジューム	ターミネート	ステップ イン	ステップ リターン
	ターミネート 及び リジューム	サスペンド	接続解除	ステップ オーバー	インストラクション ステップング モード

デバッカの接続を解除し再接続します。

ブレークポイント

The screenshot shows the STM32CubeIDE interface. The 'Run' menu is open, and 'Toggle Breakpoint' is highlighted with a red box. The code editor shows a C file with a breakpoint (a blue dot) set on line 64, which is also circled in red. The code includes comments and function calls like HAL_Init().

メニューバーのRUN→Toggle Breakpointでブレークポイントが設定できます。CPUがこのポイントを通過（実行）すると強制的に停止します。

ブレークポイントが設定されると、命令のところに●が表示されます。

デバック中はアウトライン・ビューで確認可能

Name	Type	Value
MPU_InitStruct	MPU_Region_InitTypeDef	{...}
(x)= Enable	uint8_t	204 'i'
(x)= Number	uint8_t	31 '\037'
(x)= BaseAddress	uint32_t	0
(x)= Size	uint8_t	241 'ñ'
(x)= SubRegionDisable	uint8_t	2 '\002'
(x)= TypeExtField	uint8_t	0 '\0'
(x)= AccessPermission	uint8_t	8 '\b'
(x)= DisableExec	uint8_t	137 '\211'
(x)= IsShareable	uint8_t	31 '\037'
(x)= IsCacheable	uint8_t	0 '\0'
(x)= IsBufferable	uint8_t	8 '\b'

変数

Expression	Type	Value
(x)= analogValue	volatile uint32_t	10
+ Add new expression		

変数

ブレーク
ポイント

```
08001c0c: b1 0x8001e90 <MPU_Config>
53 CPU_CACHE_Enable();
08001c10: b1 0x8001db4 <CPU_CACHE_Enable>
64 HAL_Init();
08001c14: b1 0x80004fc <HAL_Init>
67 SystemClock_Config();
-> 08001c18: b1 0x8001c34 <SystemClock_Config>
70 BSP_LED_Init(LED1);
08001c1c: movs r0, #0
08001c1e: b1 0x800030c <BSP_LED_Init>
73 EXTI15_10_IRQHandler_Config();
08001c22: b1 0x8001d34 <EXTI15_10_IRQHandler_C
74 analogValue = 10;
```

アセンブラ

デバック中はアウトライン・ビューで確認可能

デバック

Name	Value
General Registers	
r0	536870912
r1	1476543488
r2	134217728
r3	134218481
r4	0
r5	0
r6	0
r7	537001976
r8	0
r9	0
r10	0
r11	0
r12	0
sp	0x2001fff8
lr	134224875
pc	0x8001c0a <main+4>
xpsr	1627389952

レジスタ <CPU>

Register	Address	Value
Cortex_M7		
Cache		
Control		
FPE		
ID		
ImpDef		
MPU		
NVIC		
SysTick		
STM32H723		
ADC1		
ADC2		
ADC12_Common		
ADC3		
ADC3_Common		

レジスタ <ペリフェラル>

CubeMXとの連携



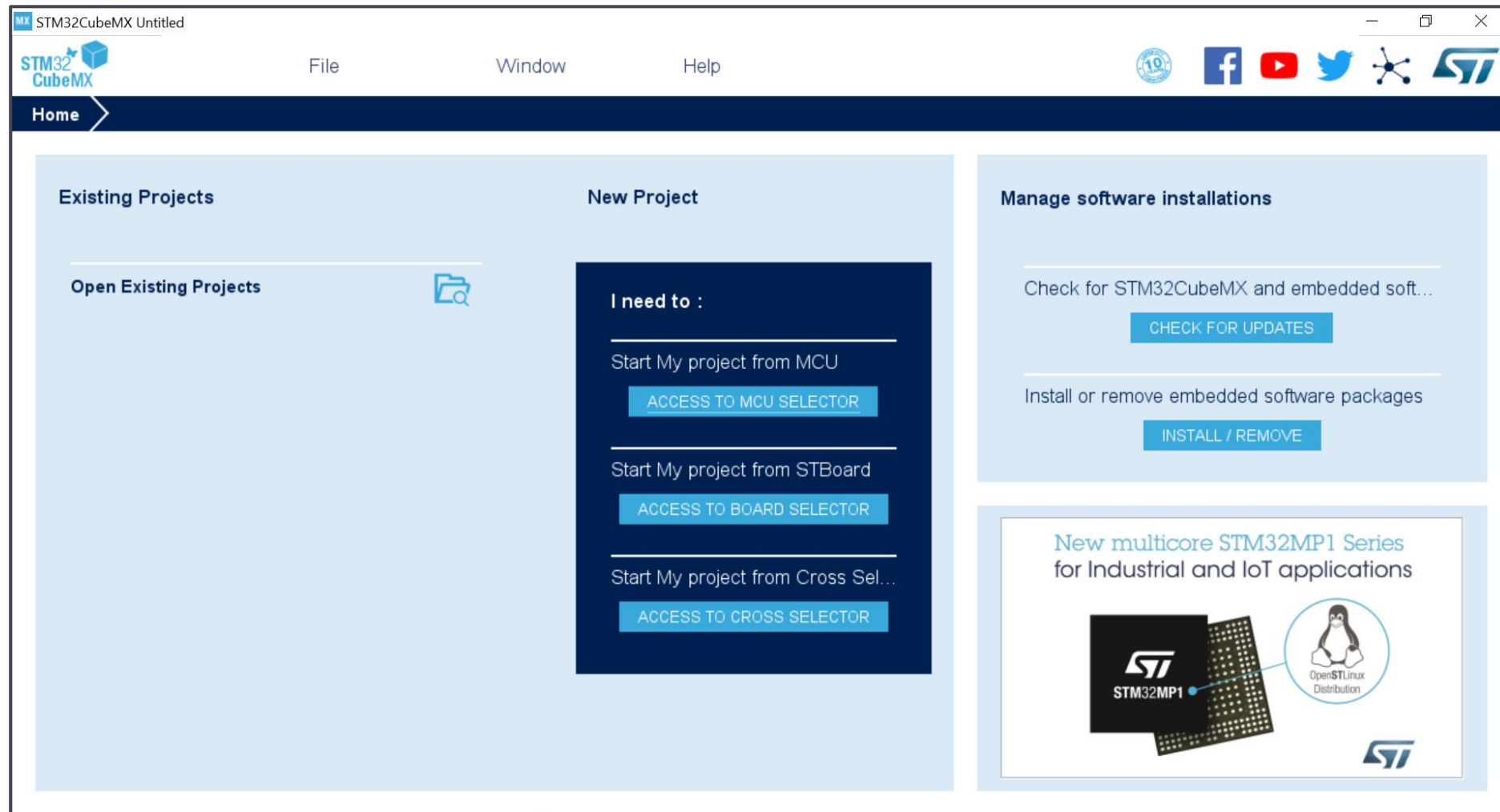
CubeMXを使用したプロジェクト作成例

セットアップ



- STM32CubeMXを使用したGPIOトグルソフトウェアの作成
- サンプル・プログラム：汎用IOをトグルしてLEDを点滅するプログラム
- 手順
 - ① NUCLEO-H723ZGを準備
 - ② ボードとPCの接続
 - ③ CubeMXでピンの設定
 - ④ クロックの設定
 - ⑤ プロジェクトの登録
 - ⑥ コード生成

CubeMXの起動



CubeMXでのデバイス選定

ACCESS TO MCU SELECTORから

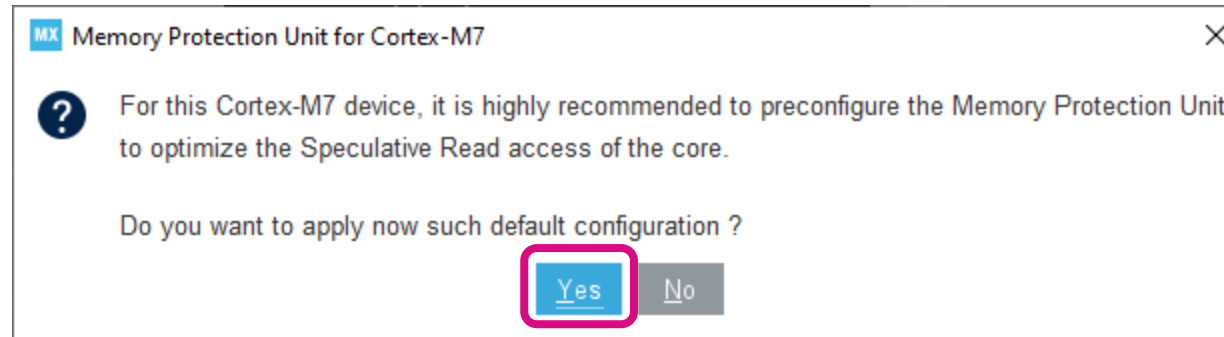
Series : **STM32H7** Lines : **STM32H723/733** Package : **LQFP144**
を選択し、リストから**STM32H723ZGT6**をダブルクリックします。

The screenshot shows the CubeMX interface for selecting a device. The 'New Project' dialog is open, and the 'ACCESS TO MCU SELECTOR' button is highlighted. The 'PRODUCT INFO' section shows 'Series' set to 'STM32H7' and 'Line' set to 'STM32H723/733'. The 'Package' section shows 'LQFP 144 20x20x1.4 mm' selected. A table at the bottom lists available MCUs/MPUs, with 'STM32H723ZGT6' highlighted.

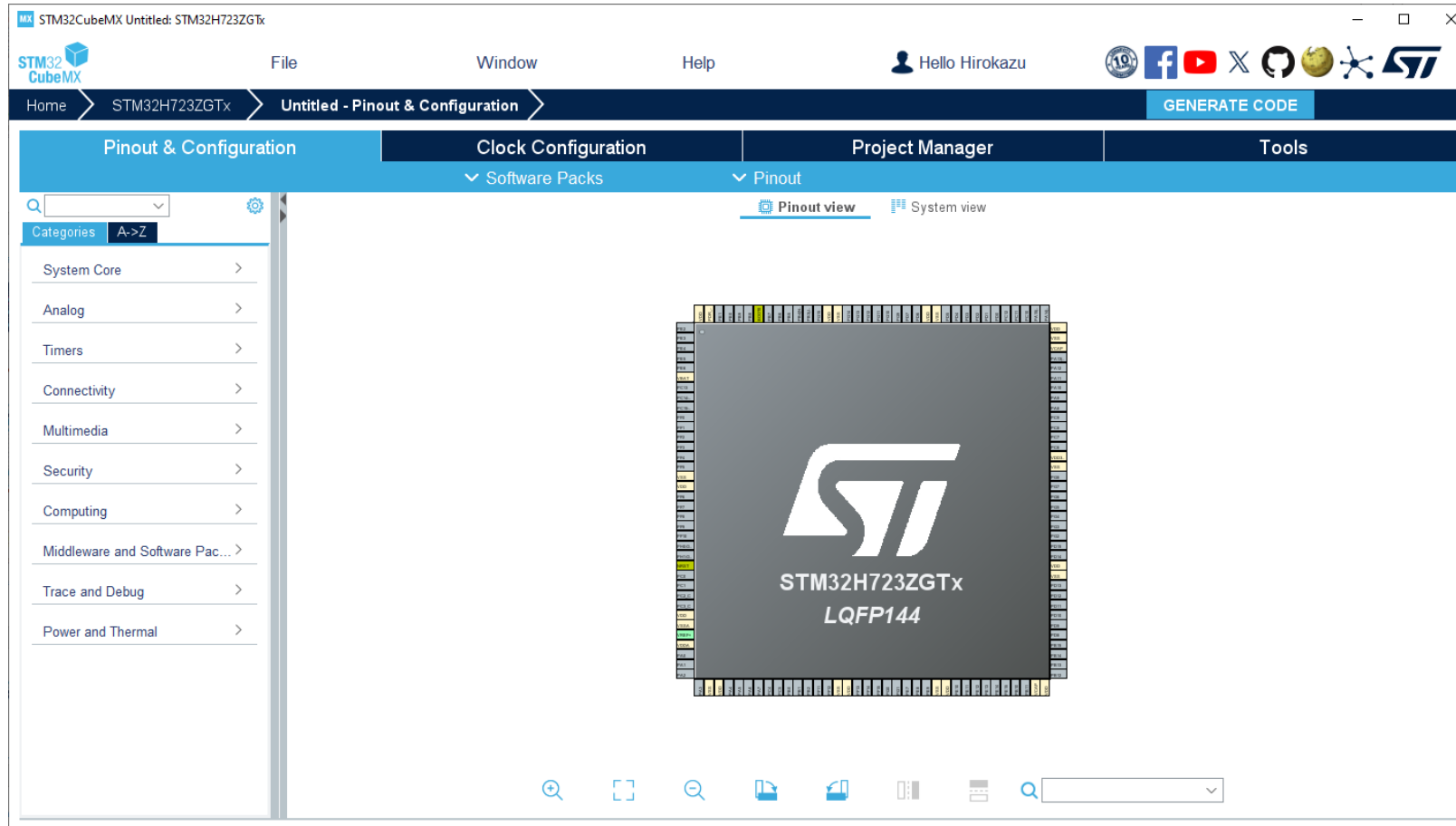
*	Commercial Part No	Part No	reference	Board
☆	STM32H723ZET6	STM32H723ZE	STM32H723ZETx	
☆	STM32H723ZGT6	STM32H723ZG	STM32H723ZGTx	NUCLEO-H723ZG
☆	STM32H733ZGT6	STM32H733ZG	STM32H733ZGTx	

CubeMXでのデバイス選定

- このメッセージが出てくるのでYesをクリック



プロジェクト画面が開きます。
ここから設定を進めていきます。



ピンの選択

以下のピン設定を行います。

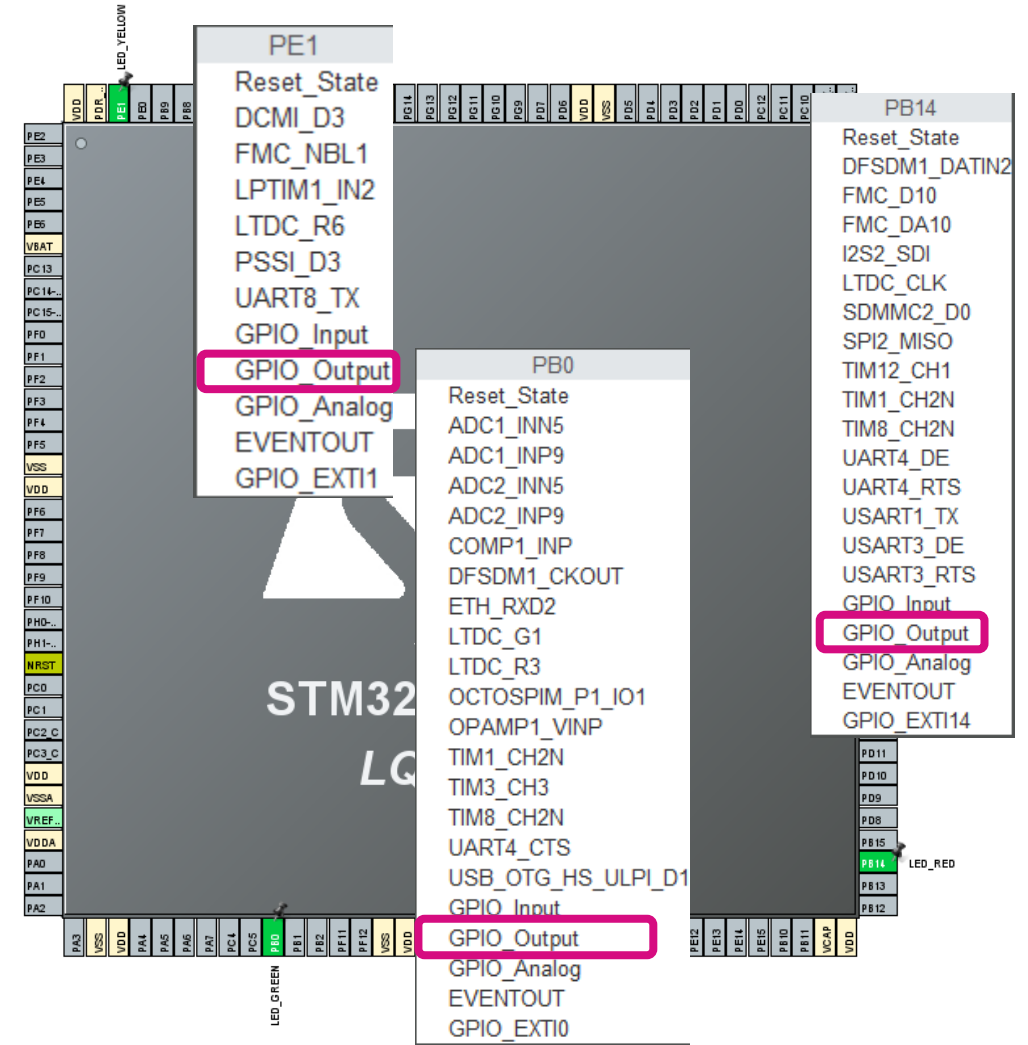
設定はマイコンイメージのピン部分をクリックすることで設定可能です。

PB0:LED_GREEN (LD1)
GPIO_Output

PE1:LED_YELLOW (LD2)
GPIO_Output

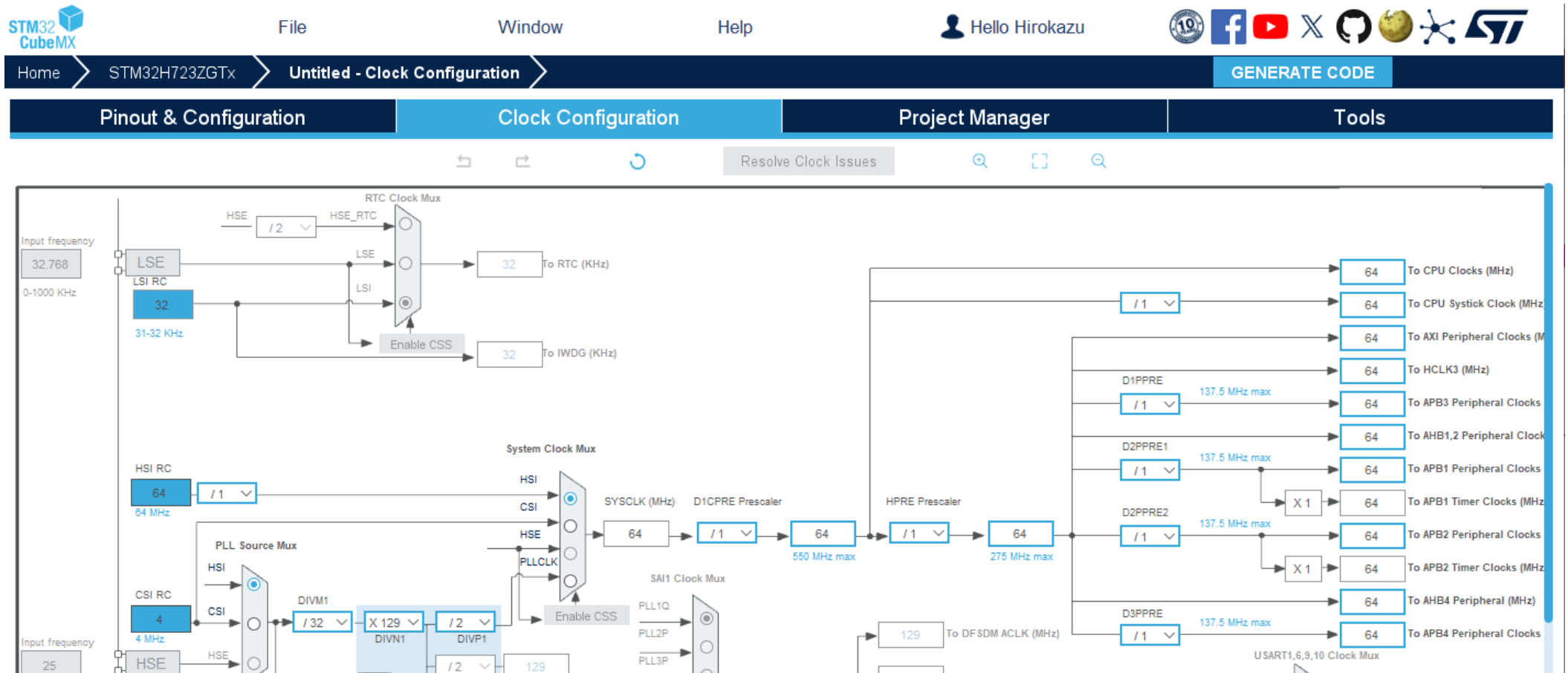
PB14:LED_RED (LD3)
GPIO_Output

設定されたピンは色が変わります。



クロックの設定

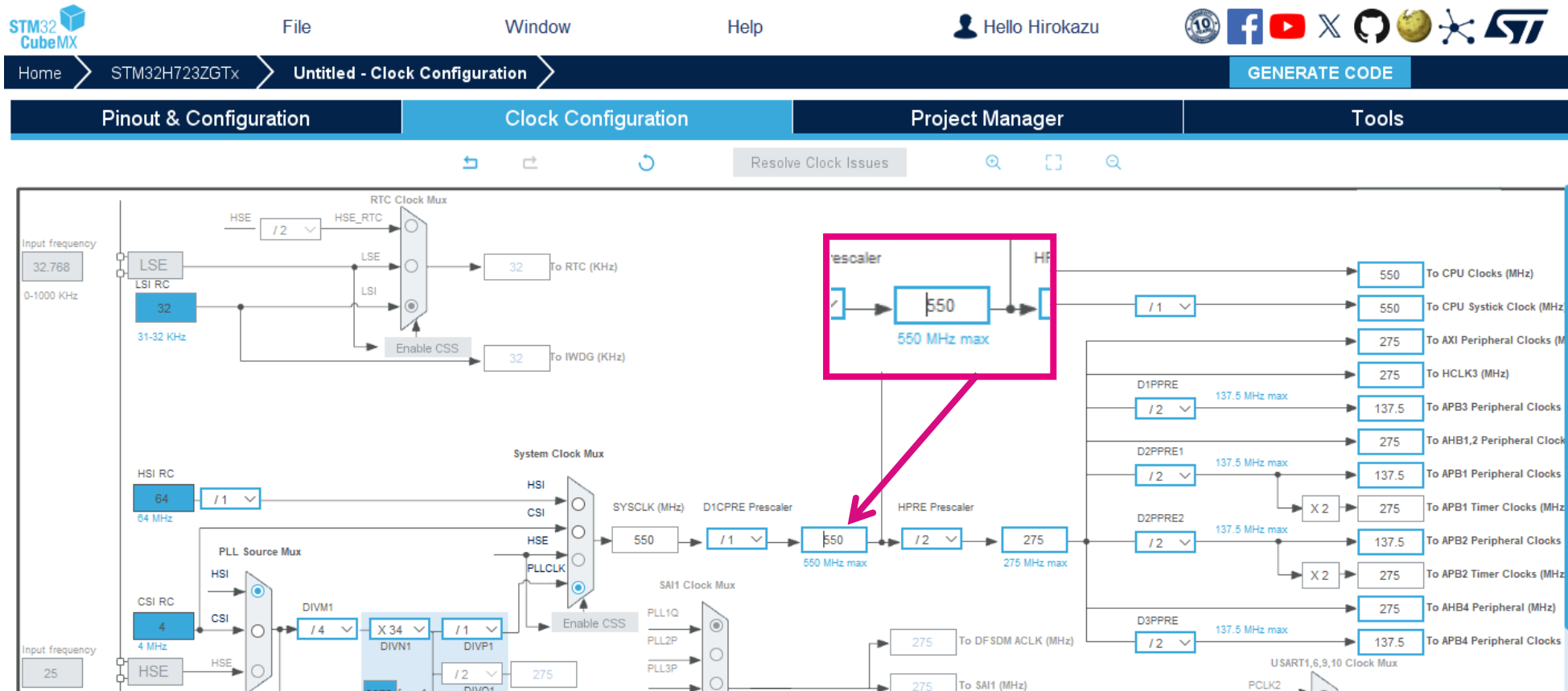
次にクロックの設定を行います。
Clock Configurationのタブを開きます。



クロックの設定

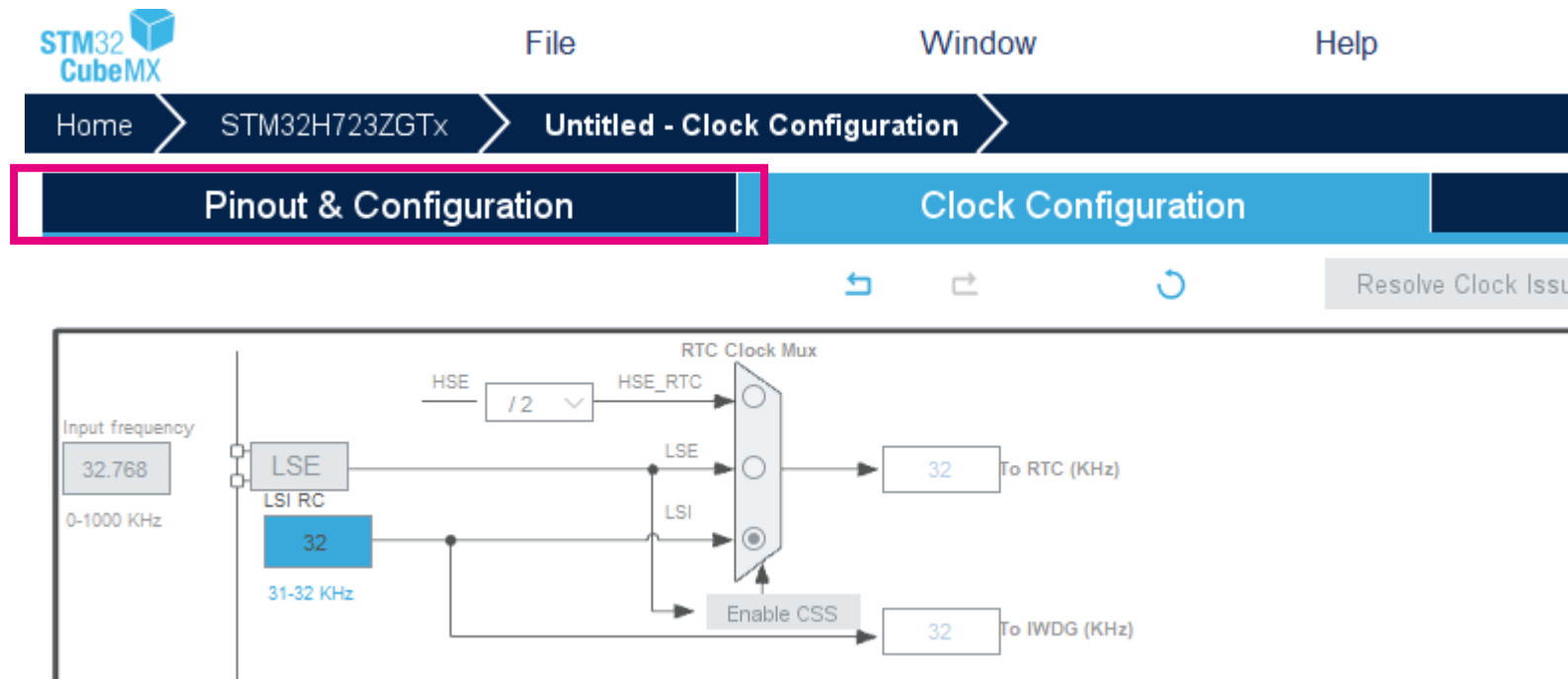
以下のようにクロックの設定を行います。

HCLK 64 → 550 本設定で、CPUクロック550MHz
ペリフェラルクロック275MHz/137.5MHz
に設定されます。



ピンの設定

ピン設定の詳細を設定しますので、再度**Pinout & Configuration**タブを開きます。



ピンの設定

System Core → GPIO → GPIO Mode and Configuration

各Pinの詳細設定を行います。

System Core

- BDMA
- CORTEX_M7
- DMA
- GPIO**
- IWDG1
- MDMA
- NVIC
- RAMECC
- RCC
- ✓ SYS
- WWDG1

Group By Peripherals

✓ GPIO

Search Signals

Search (Ctrl+F) Show only Modified Pins

Pin Name	Signal on Pin	GPIO outp...	GPIO mode	GPIO Pull-...	Maximum ...	Fast Mode	User Label	Modified
PB0	n/a	Low	Output Pus...	No pull-up ...	Low	n/a		<input type="checkbox"/>
PB14	n/a	Low	Output Pus...	No pull-up ...	Low	n/a		<input type="checkbox"/>
PE1	n/a	Low	Output Pus...	No pull-up ...	Low	n/a		<input type="checkbox"/>

ピンの設定

リストの各ピンを選択し詳細設定を行います。
PB0、PB14およびPE1の設定を以下のように設定します。

GPIO

Search Signals
Search (Ctrl+F)

PB0 Show only Modified Pins

Pin Na...	Signal on ...	GPIO outp...	GPIO mode	GPIO Pull...	Maximum ...	Fast Mode	User Label	Modified
PB0	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_GRE...	<input checked="" type="checkbox"/>
PB14	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_RED	<input checked="" type="checkbox"/>
PE1	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_YEL...	<input checked="" type="checkbox"/>

PB0 Configuration :

GPIO output level: Low

GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: Low

User Label: LED_GREEN

LED_GREENと入力

GPIO

Search Signals
Search (Ctrl+F)

PB14 Show only Modified Pins

Pin Na...	Signal on ...	GPIO outp...	GPIO mode	GPIO Pull...	Maximum ...	Fast Mode	User Label	Modified
PB0	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_GRE...	<input checked="" type="checkbox"/>
PB14	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_RED	<input checked="" type="checkbox"/>
PE1	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_YEL...	<input checked="" type="checkbox"/>

PB14 Configuration :

GPIO output level: Low

GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: Low

User Label: LED_RED

LED_REDと入力

GPIO

Search Signals
Search (Ctrl+F)

PE1 Show only Modified Pins

Pin Na...	Signal on ...	GPIO outp...	GPIO mode	GPIO Pull...	Maximum ...	Fast Mode	User Label	Modified
PB0	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_GRE...	<input checked="" type="checkbox"/>
PB14	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_RED	<input checked="" type="checkbox"/>
PE1	n/a	Low	Output Pu...	No pull-up ...	Low	n/a	LED_YEL...	<input checked="" type="checkbox"/>

PE1 Configuration :

GPIO output level: Low

GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: Low

User Label: LED_YELLOW

LED_YELLOWと入力

プロジェクト選択

ここまででマイコンの設定は完了ですので、次は

Project Manager

プロジェクトの名前や保存先等を設定します。

Home > STM32H723ZGTx > Untitled - Project Manager > GENERATE CODE

Pinout & Configuration | Clock Configuration | **Project Manager** | Tools

Project Settings

Project Name:

Project Location: Browse

Application Structure: Do not generate the main()

Toolchain Folder Location:

Toolchain / IDE: Min Version: Generate Under Root

Linker Settings

Minimum Heap Size:

Minimum Stack Size:

Thread-safe Settings

Project

プロジェクト選択

プロジェクトの設定を行います。

STM32 CubeMX

File Window Help Hello Hirokazu

Home > STM32H723ZGTx > 0426_NEW_32H7.ioc - Project Manager

Pinout & Configuration Clock Configuration Project Manager

Project

Project Settings

Project Name 0426_NEW_32H7

Project Location C:\ST\ Browse

Application Structure Advanced Do not generate the main()

Toolchain Folder Location C:\ST\0426_NEW_32H7

Toolchain / IDE EWARM Min Version V8.50 Generate Under Root

Linker Settings

Minimum Heap Size 0x200

Minimum Stack Size 0x400

Thread-safe Settings

Cortex-M7NS

プロジェクトの名称を入力します。

プロジェクトの保存先を入力します。

*可能な限り、フォルダ名に関して、
1. スペース/空白文字を含めない
2. 日本語名を含まない
を守るようお勧めします。

プロジェクト選択

Project

Toolchain / IDEを「STM32CubeIDE」に設定します。

The screenshot shows the STM32CubeIDE Project Manager interface. The breadcrumb navigation at the top indicates the path: Home > STM32H723ZGTx > Untitled - Project Manager. The interface is divided into three main sections: Pinout & Configuration, Clock Configuration, and Project Settings. The Project Settings section is currently active and contains the following fields:

- Project Name: [Empty text input field]
- Project Location: C:\ST
- Application Structure: Advanced (dropdown menu) Do not generate
- Toolchain Folder Location: C:\ST\
- Toolchain / IDE: STM32CubeIDE (dropdown menu) Generate Under Root

The 'Project' and 'Code Generator' sections on the left side of the Project Settings panel are highlighted with red boxes. The 'Toolchain / IDE' dropdown menu and the 'Generate Under Root' checkbox are also highlighted with a red box.

プロジェクト選択

Code Generator

コード生成に関する設定を行います。

The screenshot shows the STM32Cube IDE Project Manager interface. The breadcrumb navigation at the top indicates the path: Home > STM32H723ZGTx > 0426_NEW_32H7.ioc - Project Manager. The interface is divided into three main sections: Pinout & Configuration, Clock Configuration, and Project Manager. The Project Manager section is active and contains the following settings:

- STM32Cube MCU packages and embedded software packs
 - Copy all used libraries into the project folder
 - Copy only the necessary library files
 - Add necessary library files as reference in the toolchain project configuration file
- Generated files
 - Generate peripheral initialization as a pair of '.c/.h' files per peripheral
 - Backup previously generated files when re-generating
 - Keep User Code when re-generating
 - Delete previously generated files when not re-generated

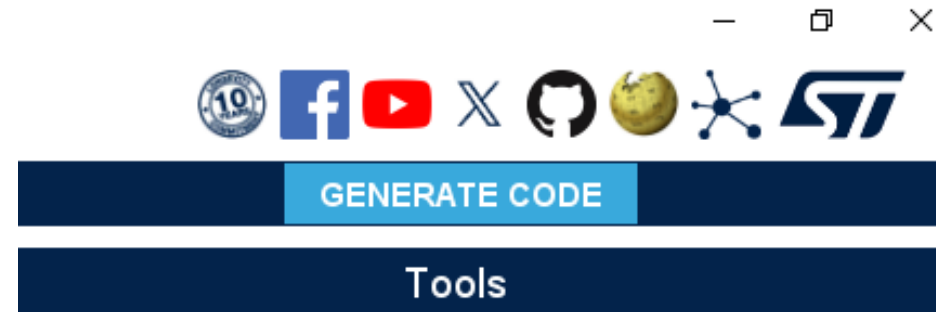
Red boxes highlight the 'Project' and 'Code Generator' sections on the left sidebar, and the 'Copy only the necessary library files' option in the settings. Red arrows point from the 'Code Generator' section to the selected option and from the explanatory text to the same option.

必要なファイルだけ
コピーするようにします

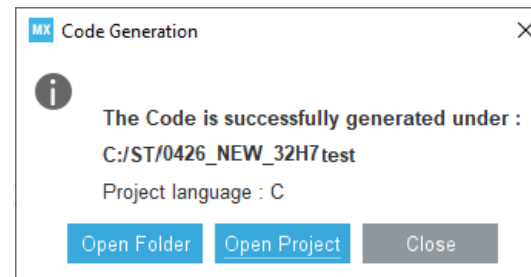
*この設定に切り替えないと、ビルドに不必要なファイルまで、プロジェクトフォルダにコピーされ、PCのストレージを圧迫します。

お疲れさまでした！

ここまでに設定した内容で、コードを生成します。
GUIの右上の**GENERATE CODE**ボタンを押します。



コード生成が完了すると以下のダイアログが開きます。



Open Projectボタンを押下することでSTM32CubeIDEが開きます。

CubeIDE コード追記

ここからユーザ処理を追加していきます。

編集するファイルはmain.cになります。

STM32CubeMXで生成したコードには、

```
/* USER CODE BEGIN xxx */ と /* USER CODE END xxx */
```

の記載があり、このコメントの範囲内に
ユーザコードを追加していきます。

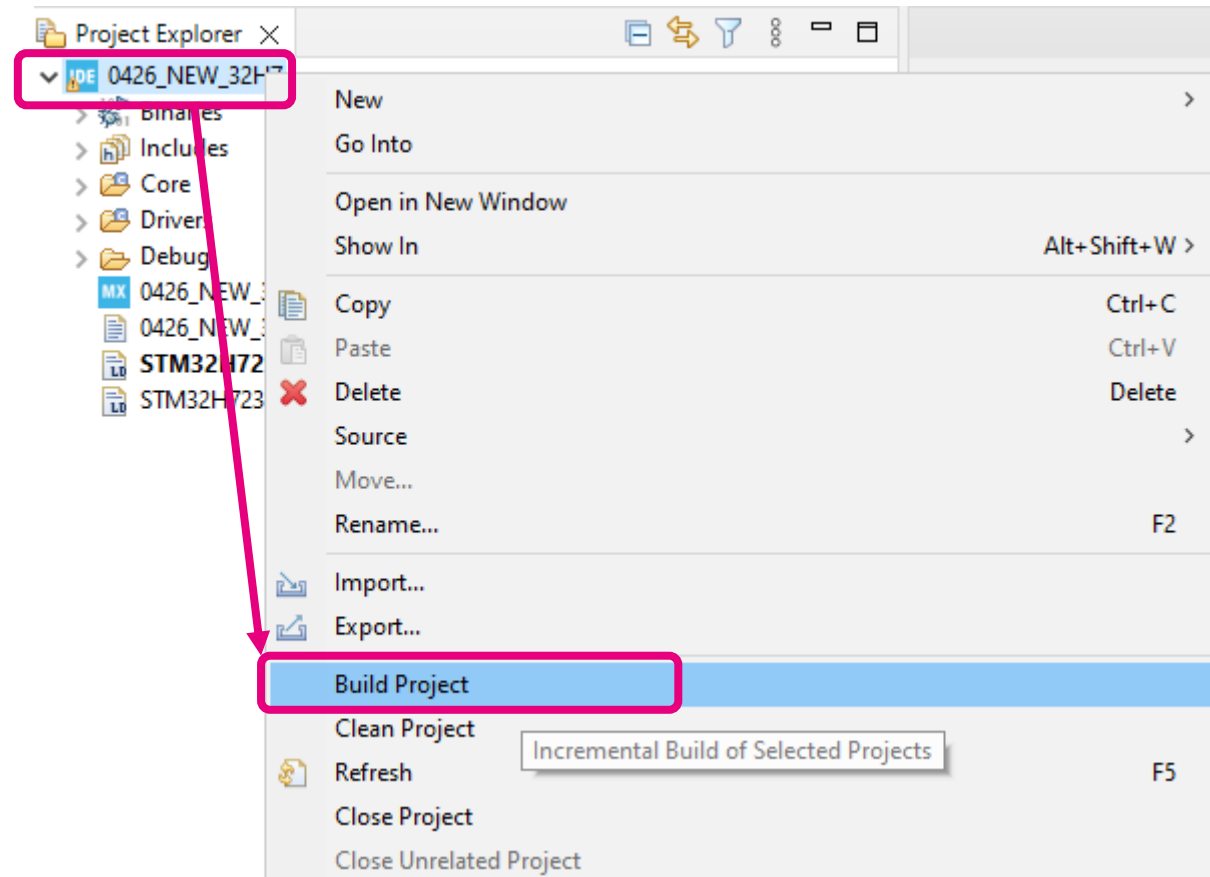
上記のルールに則って追記したユーザーコードは、
再度CubeMXにてコード生成した場合でも保護されます。

CubeIDE コード追記

STM32CubeIDEが起動したら、ビルドが通ることを確認します。

プロジェクトを右クリック → Build Project

右クリック



追記箇所

```
/* USER CODE BEGIN 3 */ ~ /* USER CODE END 3 */
```

ループ処理を追加します。

HAL_Delay関数を使用して定期的にGPIOをトグルします。

```
/* USER CODE BEGIN 3 */
```

```
    HAL_GPIO_TogglePin(LED_RED_GPIO_Port,LED_RED_Pin);
```

```
    HAL_Delay(750);
```

```
    HAL_GPIO_TogglePin(LED_YELLOW_GPIO_Port,LED_YELLOW_Pin);
```

```
    HAL_Delay(500);
```

```
    HAL_GPIO_TogglePin(LED_GREEN_GPIO_Port,LED_GREEN_Pin);
```

```
    HAL_Delay(250);
```

```
} /* end of while */ *このかっこの前の行に挿入です
```

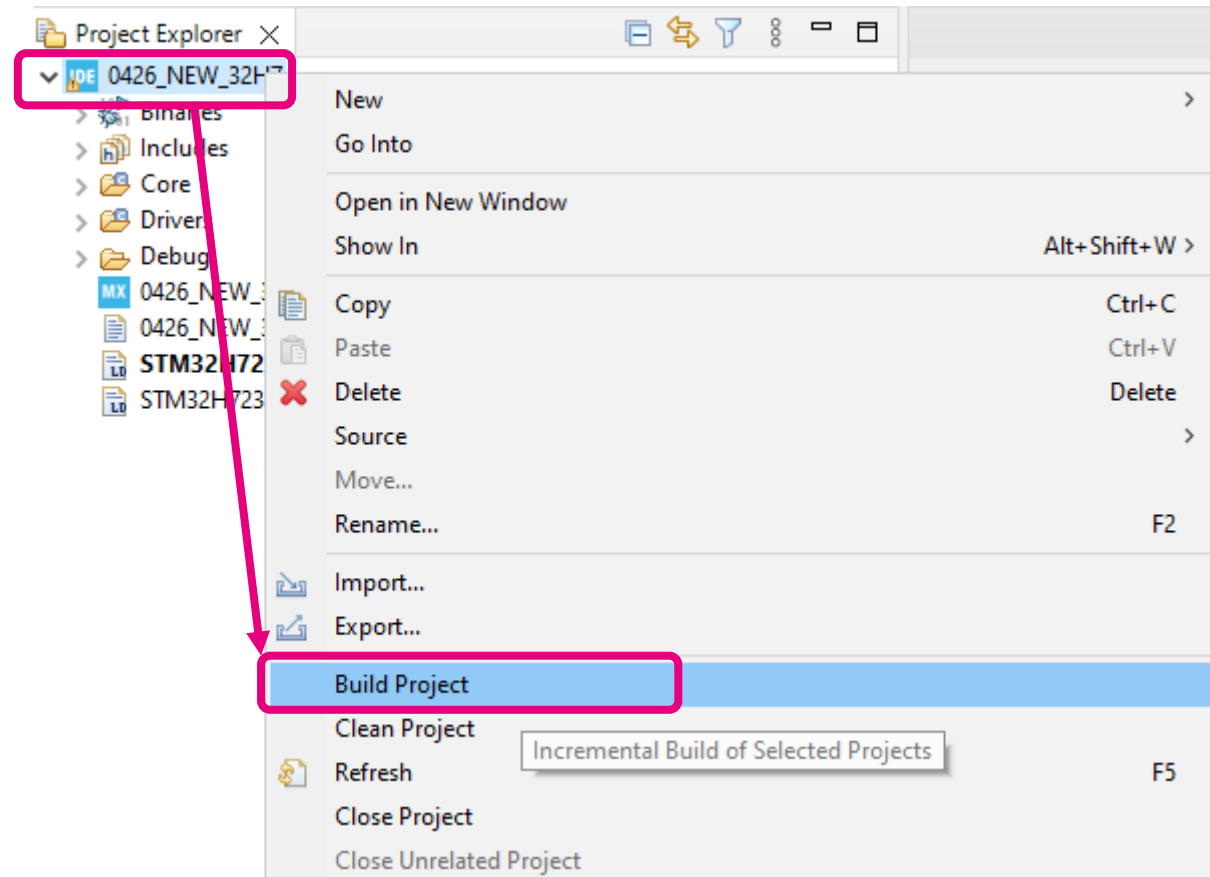
```
/* USER CODE END 3 */
```



CubeIDE コード追記

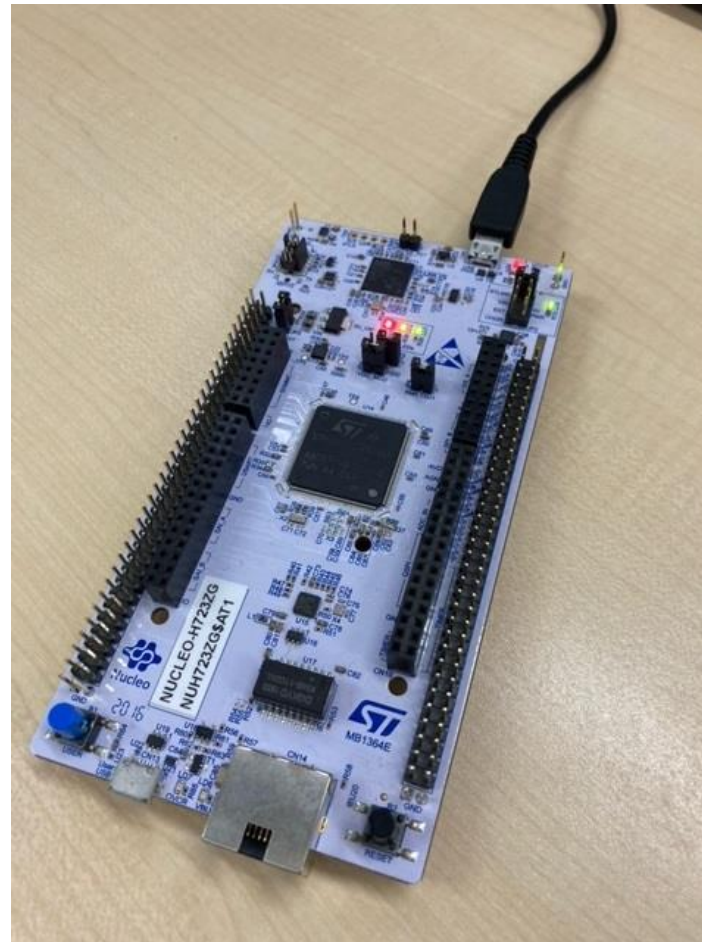
追記が終わりましたらビルドします。
プロジェクトを右クリック → **Build Project**

右クリック



プログラム動作

- プログラムをロードするとLEDが点滅します。



おまけ：Nucleoボードでのボード選択例

STM32CubeMXを起動した後に

評価ボードを選択することでプロジェクトを生成する事も出来ます。

The screenshot illustrates the board selection process in STM32CubeMX. On the left, the 'New Project' dialog is shown with three options: 'Start My project from MCU', 'Start My project from STBoard', and 'Start My project from Cross Sel...'. The 'ACCESS TO BOARD SELECTOR' button is highlighted with a red box. The main interface shows the 'PRODUCT INFO' section with 'Type' set to 'Nucleo-144' and 'MCU / MPU Series' set to 'STM32H7'. A table below shows a list of boards, with 'NUCLEO-H723ZG' highlighted. The table has a 'Commercial Part No' column.

Commercial Part No
NUCLEO-H723ZG
NUCLEO-H743ZI

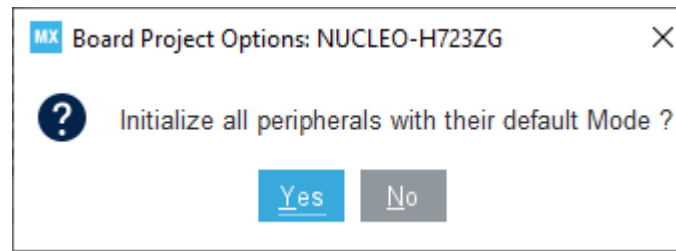
ACCESS TO BOARD SELECTORから

Type : **Nucleo-144** MCU Series : **STM32H7**

を選択し、リストから**NUCLEO-H723ZG**をダブルクリックします。

おまけ：Nucleoボードでのボード選択例

以下のダイアログではYesを押します。
これにより、ボードを使うときの標準的な設定が自動的に適用されます。



參考資料



STM32CubeH7の展開場所

事前準備に以下の指示表示あり (STM32CubeMXセットアップ資料)

3. ユーザ名が日本語の方

ユーザ名が英数字+空白の方はこの作業は不要です。

メニューのHelp → Updater Settings のUpdater Settings タブで、Repository Folder に「C:\ST」を設定します。

ユーザ名が日本語の環境の場合

C:\ST\

ユーザ名が英数字+空白の環境の場合 :

C:\Users*<user name>*\STM32Cube\Repository\

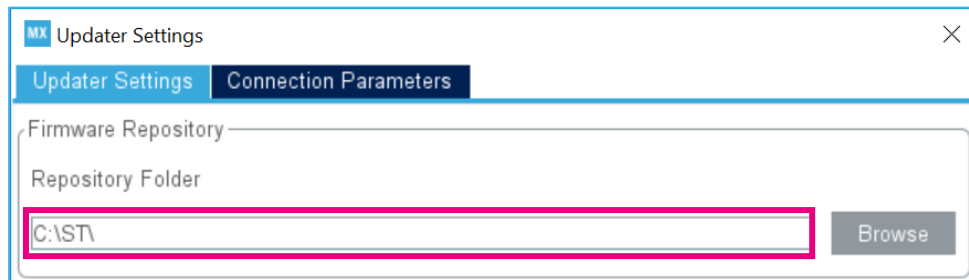


STM32CubeH7の展開場所

- STM32CubeMXを起動
- メニューより、Help → Updater Settings を選択
- Repository Folder 欄を確認

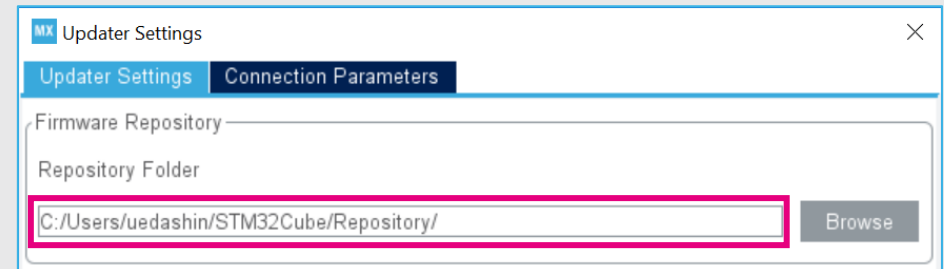
ユーザ名が日本語の環境の場合

C:\STI



ユーザ名が英数字+空白の環境の場合 :

C:\Users\<user name>\STM32Cube\Repository



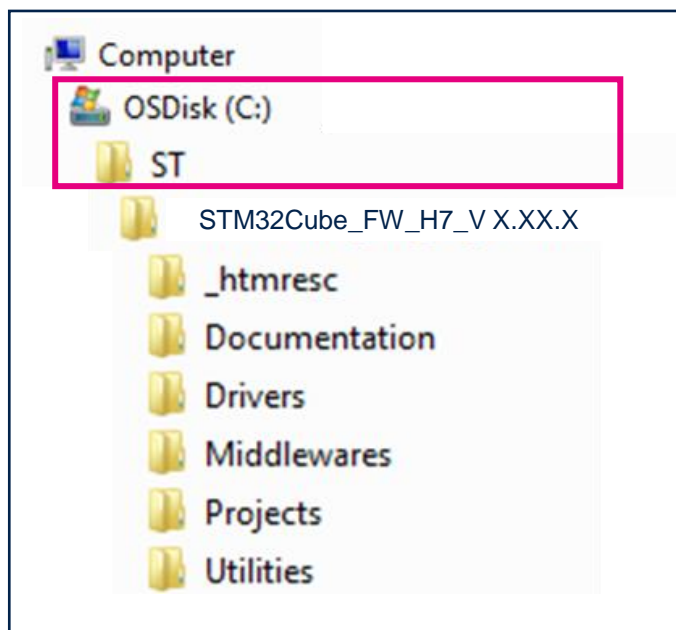


STM32CubeH7の展開場所

- STM32CubeH7の展開場所を確認

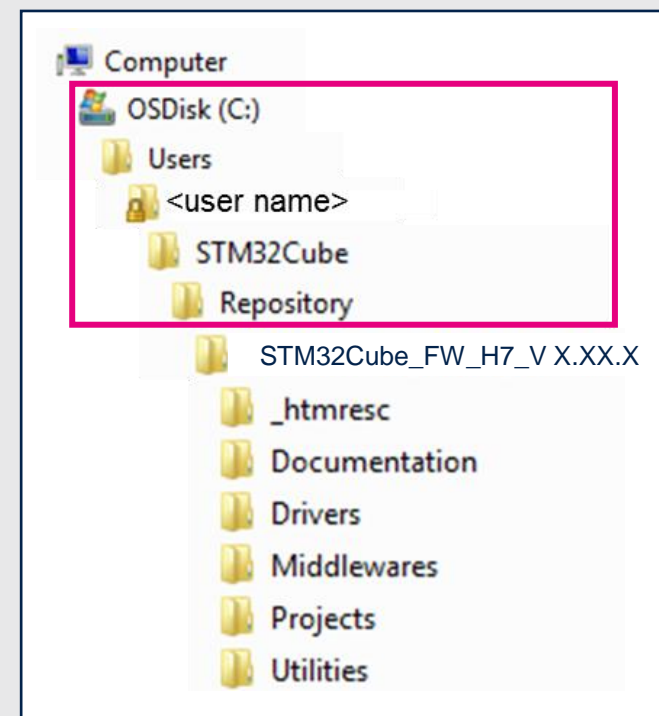
ユーザ名が日本語の環境の場合

C:\STI



ユーザ名が英数字+空白の環境の場合 :

C:\Users\\STM32Cube\Repository



Thank you

© STMicroelectronics - All rights reserved.

The STMicroelectronics corporate logo is a registered trademark of the STMicroelectronics group of companies. All other names are the property of their respective owners.



life.augmented