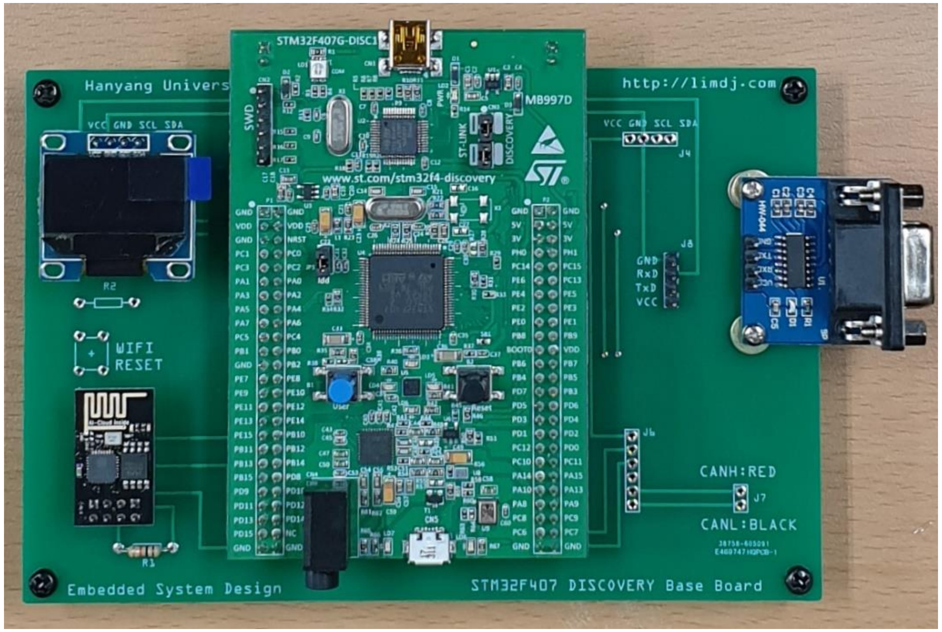
Embedded System Software

Cortex-M Lab 1

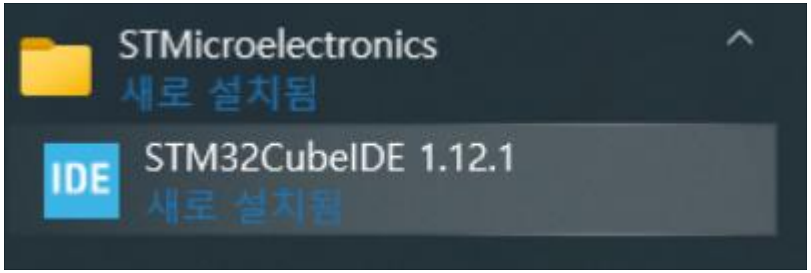
Programming Cortex-M4 STM32F407 Microcontroller

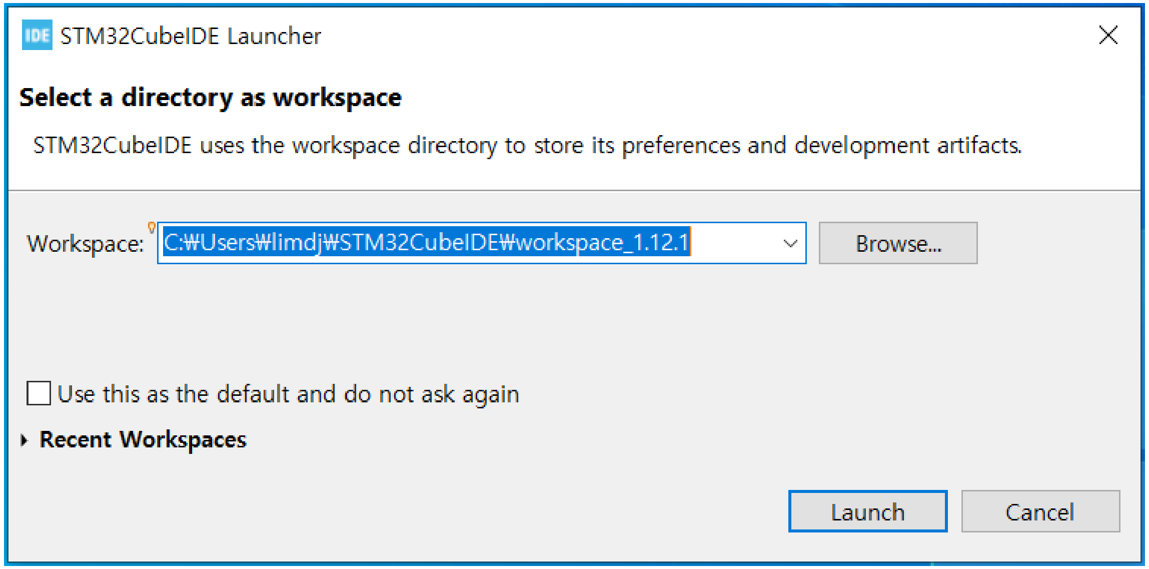
Cortex-M4 Board

- STM32F407 Discovery Board
  - RS232C
  - Serial WIFI
  - 0.96 inch OLED graphic display
  - On board ST-LINK JTAG debugging interface



Start STM32CubeIDE





STM32CubeIDE Launcher

Select a directory as workspace

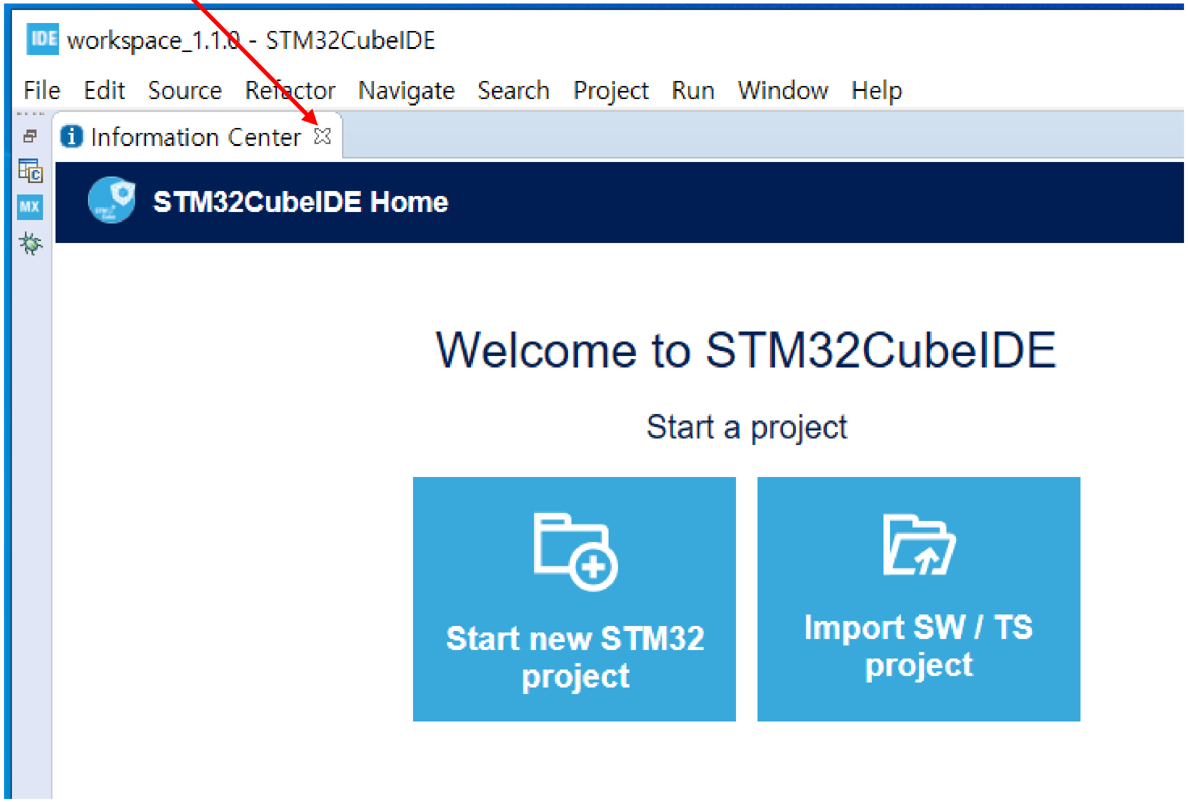
STM32CubeIDE uses the workspace directory to store its preferences and development artifacts.

Workspace: C:₩Users₩limdj₩STM32CubeIDE₩workspace_1.12.1

□ Use this as the default and do not ask again

Recent Workspaces

- Click X to close



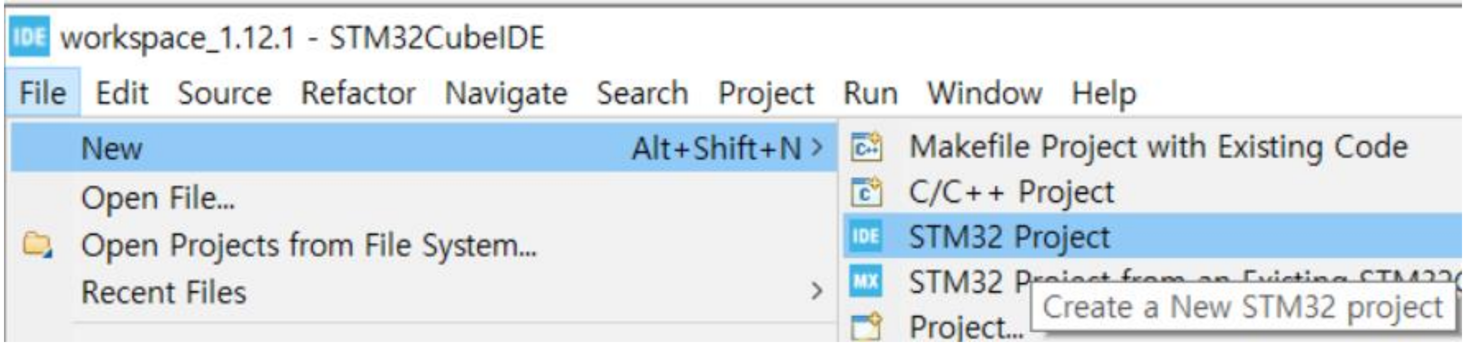
workspace_1.1.0 - STM32CubeIDE

File Edit Source Refactor Navigate Search Project Run Window Help
Information Center
STM32CubeIDE Home

Welcome to STM32CubeIDE

Start a project

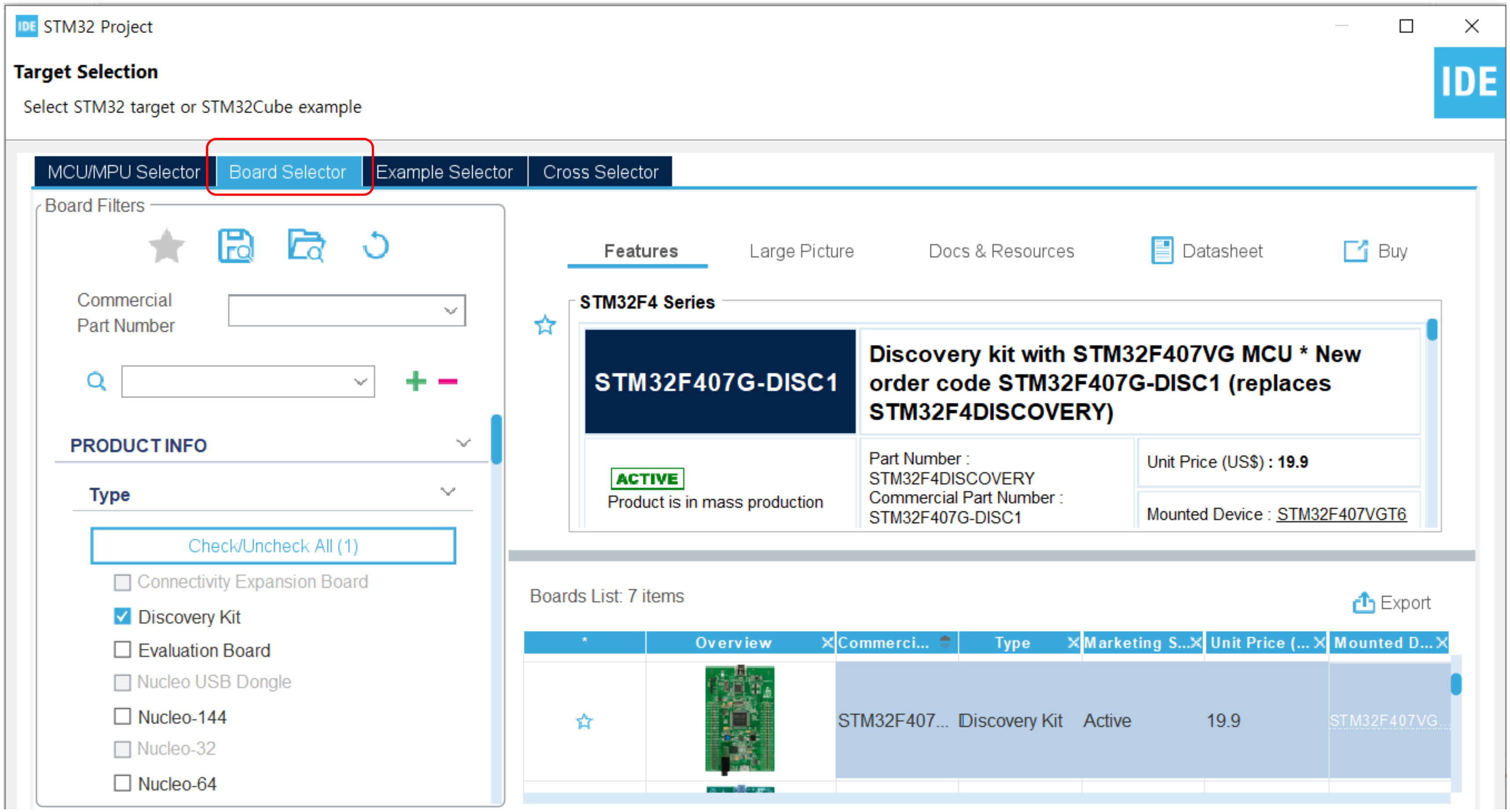
New STM32 Project



workspace_1.12.1 - STM32CubeIDE
File
Edit
Source
Refactor
Navigate
Search
Project
Run
Window
Help
New Alt+Shift+N
Open File...
Open Projects from File System...
Recent Files
Makefile Project with Existing Code
C/C++ Project
STM32 Project
STM32 Project from an Existing STM32(
Project...
Create a New STM32 project

Select Board (Not MCU)

- Select Board Selector



Target Selection

Select STM32 target or STM32Cube example

□ Connectivity Expansion Board
☑ Discovery Kit
□ Evaluation Board
□ Nucleo USB Dongle
□ Nucleo-144
□ Nucleo-32
□ Nucleo-64

STM32F4 Series

STM32F407G-DISC1

Discovery kit with STM32F407VG MCU * New order code STM32F407G-DISC1 (replaces STM32F4DISCOVERY)

ACTIVE
Product is in mass production

Part Number : STM32F4DISCOVERY
Commercial Part Number : STM32F407G-DISC1

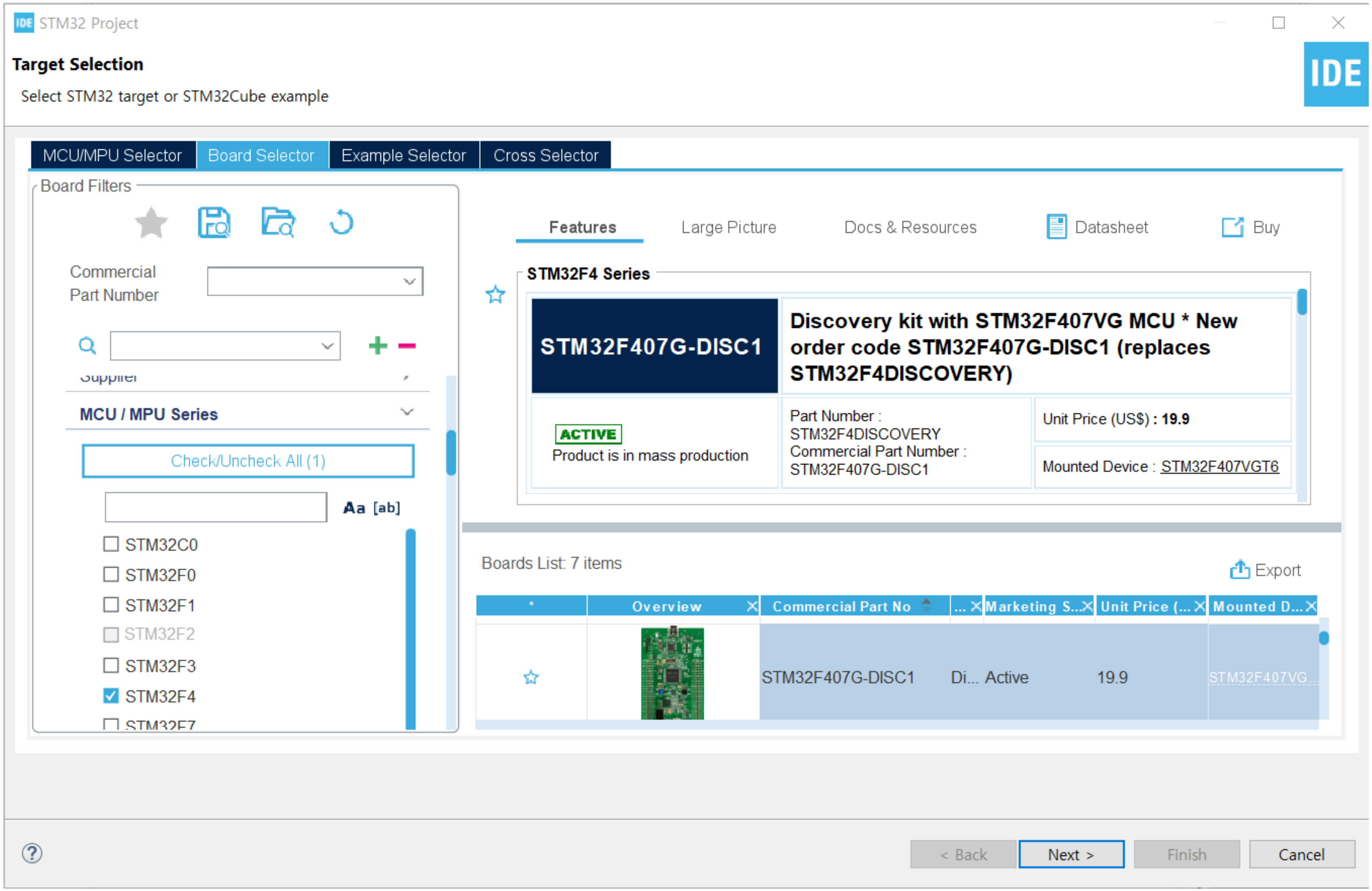
Unit Price (US$) : **19.9**

Mounted Device : STM32F407VGT6

Boards List: 7 items

*	Overview	Commerci...	Type	Marketing S...	Unit Price (...	Mounted D...
☆		STM32F407...	Discovery Kit	Active	19.9	STM32F407VG...

- Select STM32F407G-DISC1 and click Next



STM32 Project

Target Selection

Select STM32 target or STM32Cube example

Commercial Part Number

MCU / MPU Series

□ STM32C0
□ STM32F0
□ STM32F1
□ STM32F2
□ STM32F3
☑ STM32F4
□ STM32F7

STM32F4 Series

STM32F407G-DISC1

Discovery kit with STM32F407VG MCU * New order code STM32F407G-DISC1 (replaces STM32F4DISCOVERY)

ACTIVE
Product is in mass production

Part Number : STM32F4DISCOVERY
Commercial Part Number : STM32F407G-DISC1

Unit Price (US$) : **19.9**

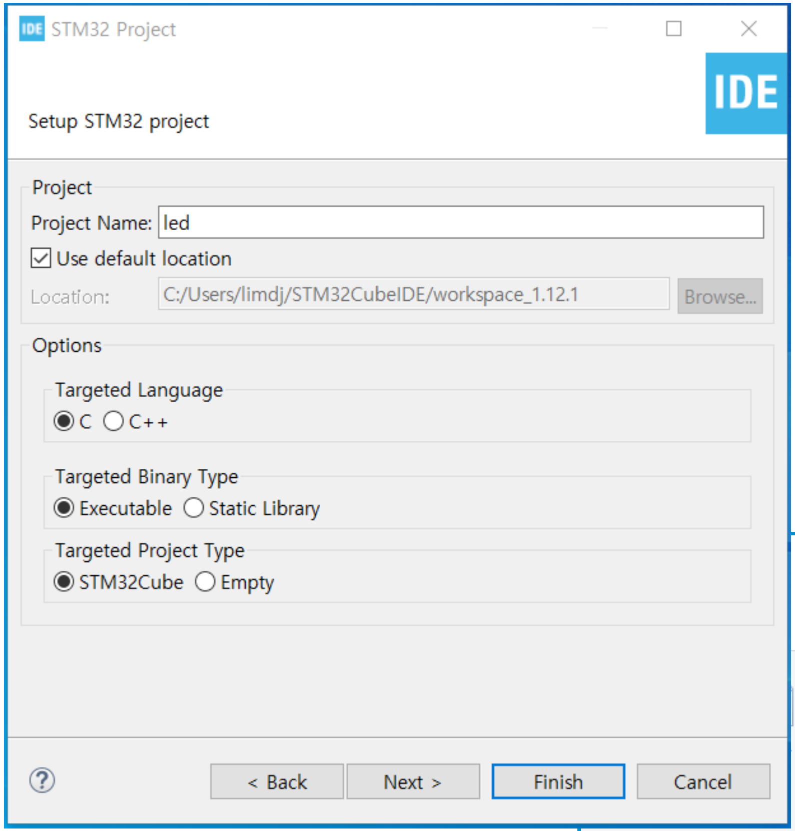
Mounted Device : STM32F407VGT6

Boards List: 7 items

*	Overview	Commercial Part No	...	Marketing S...	Unit Price (...	Mounted D...
☆		STM32F407G-DISC1	Di...	Active	19.9	STM32F407VG...

Project Name

- 이 화면에서 Next를 누르면 다음 화면이 나오고, Finish를 누르면 설정이 끝남



STM32 Project

Setup STM32 project

Project

Project Name: led

☑ Use default location

Location: C:/Users/limdj/STM32CubeIDE/workspace_1.12.1

Options

Targeted Language

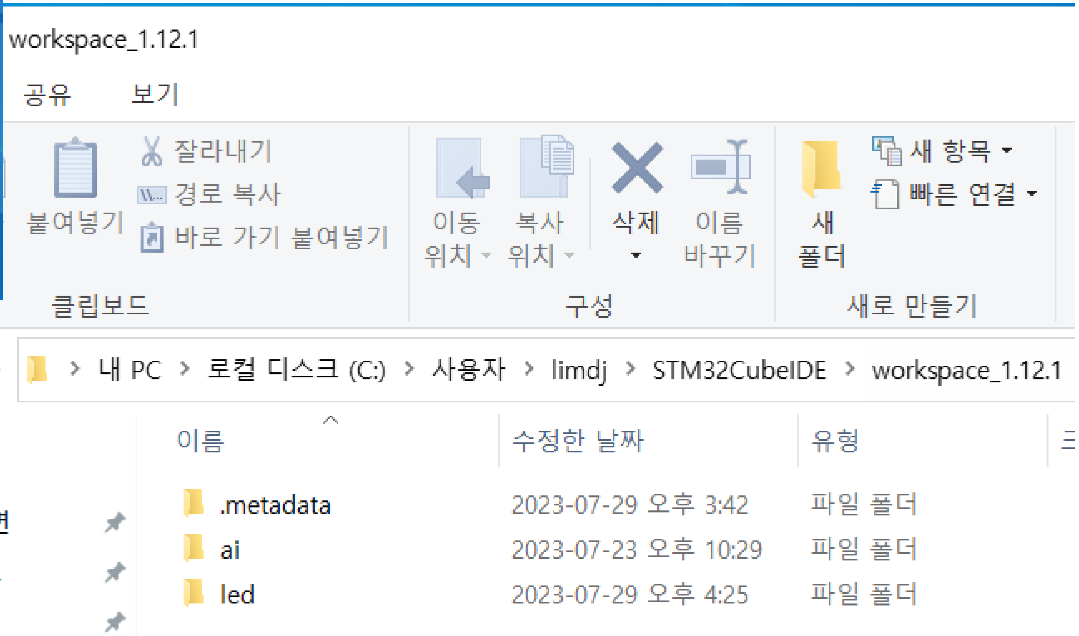
● C ○ C++

Targeted Binary Type

● Executable ○ Static Library

Targeted Project Type

● STM32Cube ○ Empty

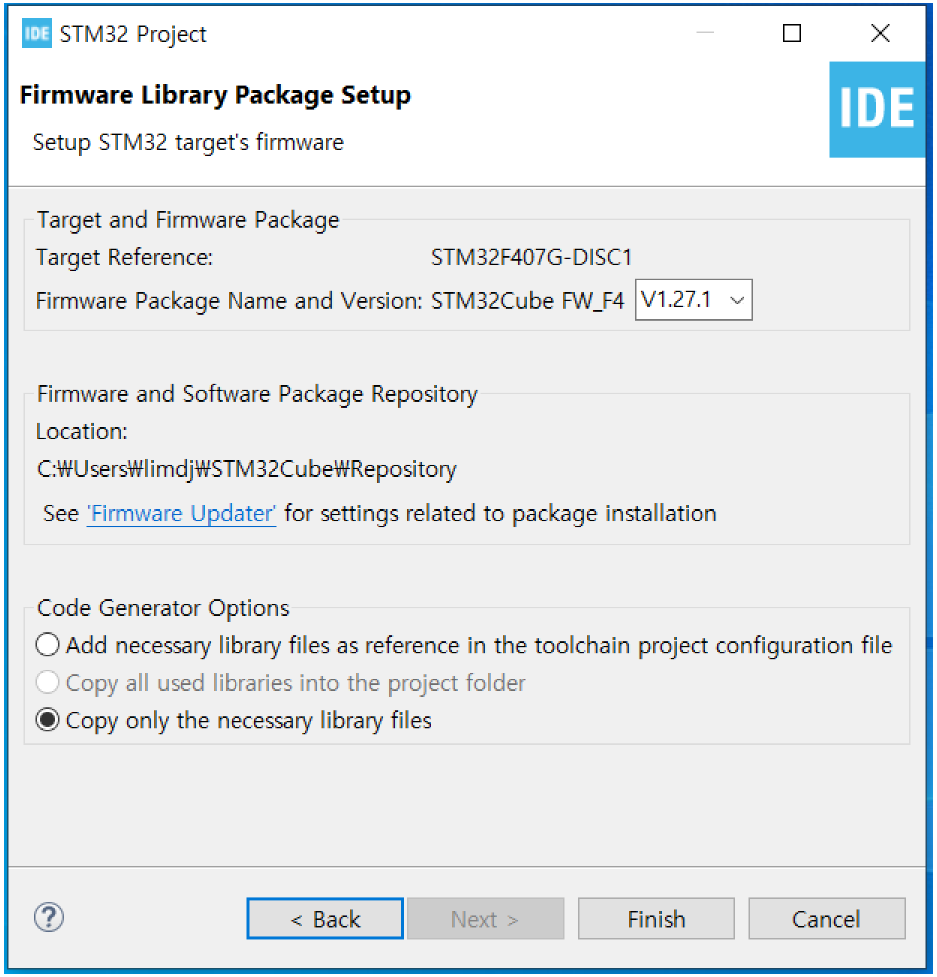


workspace_1.12.1

내 PC > 로컬 디스크 (C:) > 사용자 > limdj > STM32CubeIDE > workspace_1.12.1

이름	수정한 날짜	유형
.metadata	2023-07-29 오후 3:42	파일 폴더
ai	2023-07-23 오후 10:29	파일 폴더
led	2023-07-29 오후 4:25	파일 폴더

- 이전 화면에서 Next를 누르면 이 화면이 나옴
- Firmware Package Version check를 위해서 필요함



STM32 Project

Firmware Library Package Setup

Setup STM32 target's firmware

Target and Firmware Package

Target Reference: STM32F407G-DISC1

Firmware Package Name and Version: STM32Cube FW_F4 V1.27.1

Firmware and Software Package Repository

Location:

C:₩Users₩limdj₩STM32Cube₩Repository

See 'Firmware Updater' for settings related to package installation

Code Generator Options

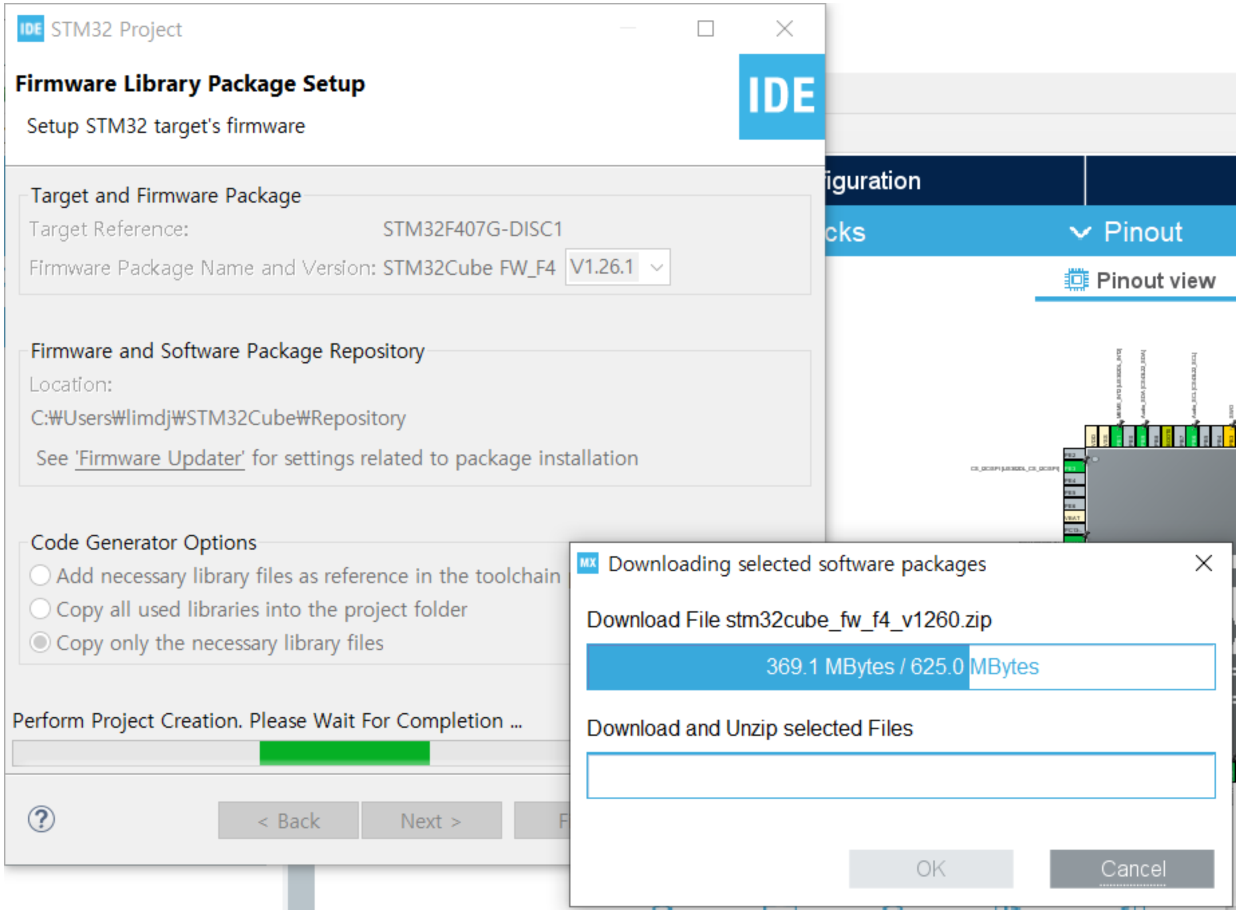
○ Add necessary library files as reference in the toolchain project configuration file

○ Copy all used libraries into the project folder

● Copy only the necessary library files

필요 없음

- 필요한 Firmware Package가 없으면 다운 로드가 진행됨



STM32 Project

Firmware Library Package Setup

Setup STM32 target's firmware

Target and Firmware Package

Target Reference: STM32F407G-DISC1

Firmware Package Name and Version: STM32Cube FW_F4 V1.26.1

Firmware and Software Package Repository

Location:

C:₩Users₩limdj₩STM32Cube₩Repository

See 'Firmware Updater' for settings related to package installation

Code Generator Options

○ Add necessary library files as reference in the toolchain

○ Copy all used libraries into the project folder

● Copy only the necessary library files

Perform Project Creation. Please Wait For Completion ...

Downloading selected software packages

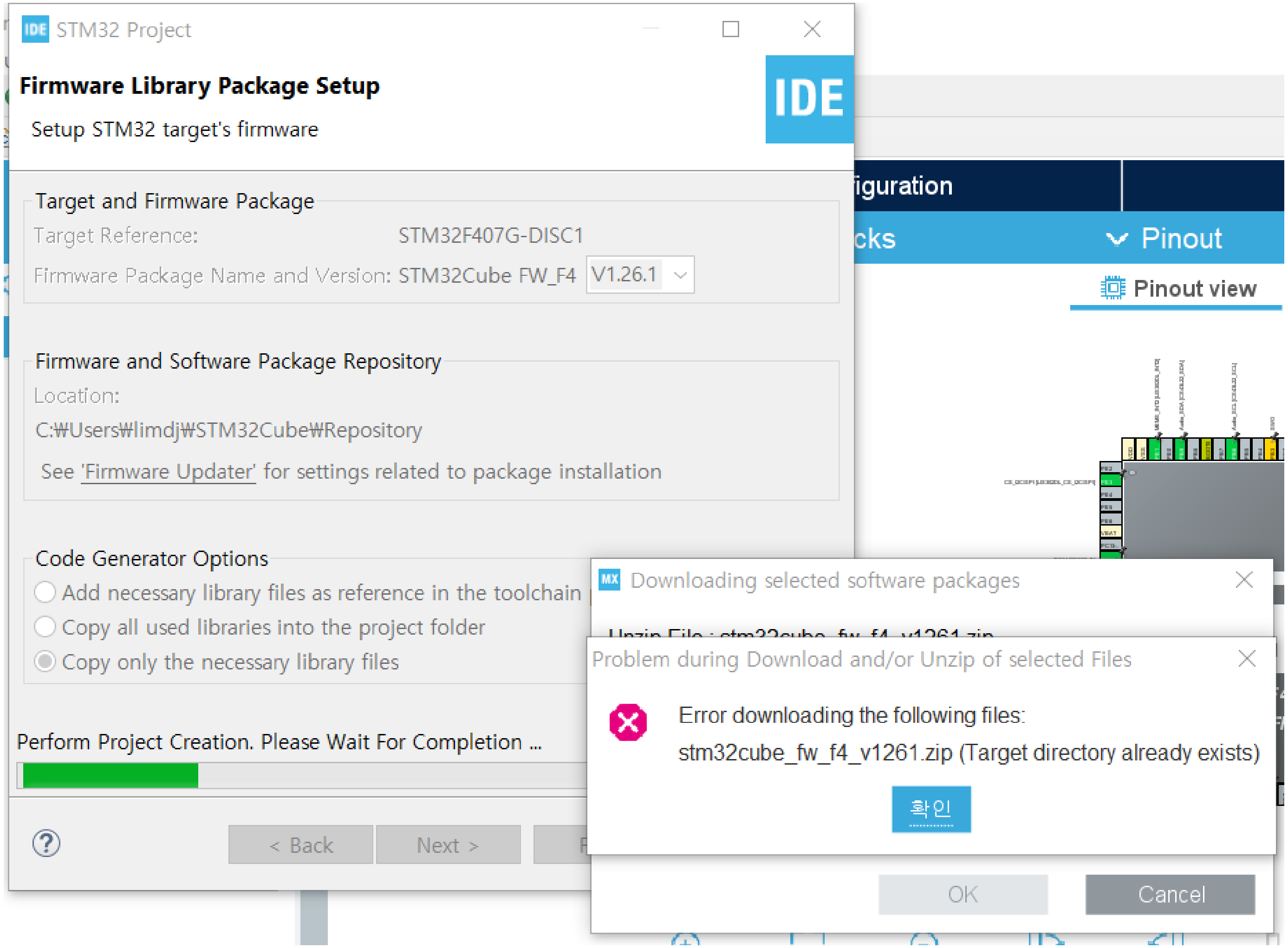
Download File stm32cube_fw_f4_v1260.zip

369.1 MBytes / 625.0 MBytes

Download and Unzip selected Files

필요 없음

- 종종 아래와 같은 에러가 발생할 수 있음



STM32 Project

Firmware Library Package Setup

Setup STM32 target's firmware

Target and Firmware Package

Target Reference: STM32F407G-DISC1

Firmware Package Name and Version: STM32Cube FW_F4 V1.26.1

Firmware and Software Package Repository

Location:

C:₩Users₩limdj₩STM32Cube₩Repository

See 'Firmware Updater' for settings related to package installation

Code Generator Options

○ Add necessary library files as reference in the toolchain

○ Copy all used libraries into the project folder

● Copy only the necessary library files

Perform Project Creation. Please Wait For Completion ...

Downloading selected software packages

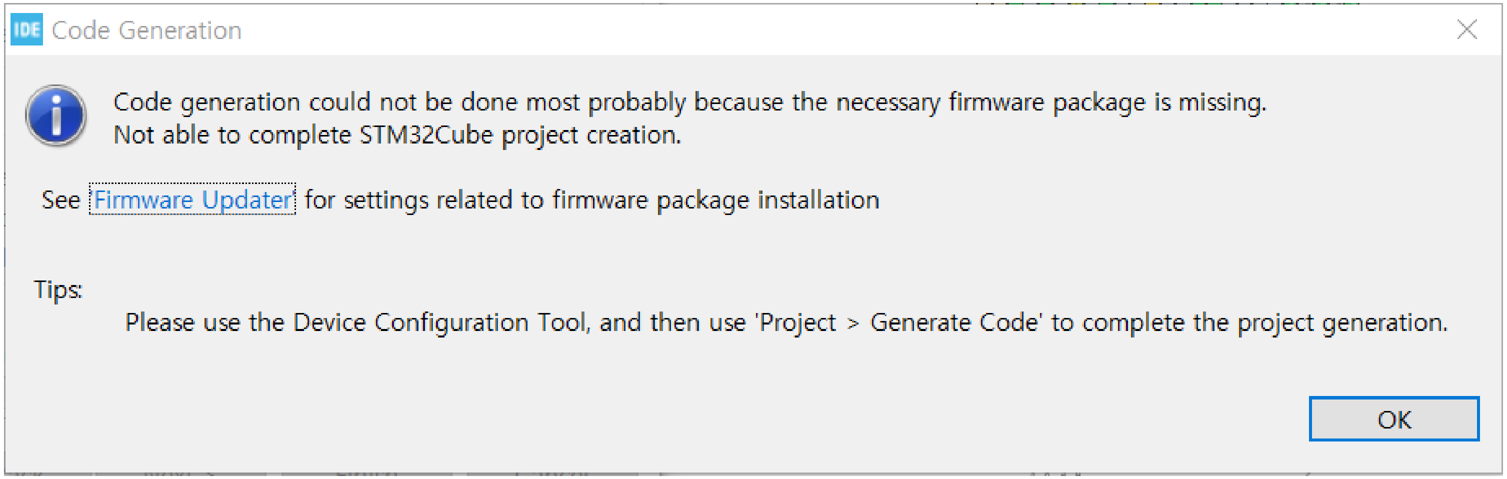
Unzip File : stm32cube_fw_f4_v1261.zip

Problem during Download and/or Unzip of selected Files

Error downloading the following files:
stm32cube_fw_f4_v1261.zip (Target directory already exists)

필요 없음

- 앞의 화면과 같이 다운 로드에서 에러가 발생할 경우 OK를 누르고 수동으로 설치를 진행해야 함



Code generation could not be done most probably because the necessary firmware package is missing.
Not able to complete STM32Cube project creation.

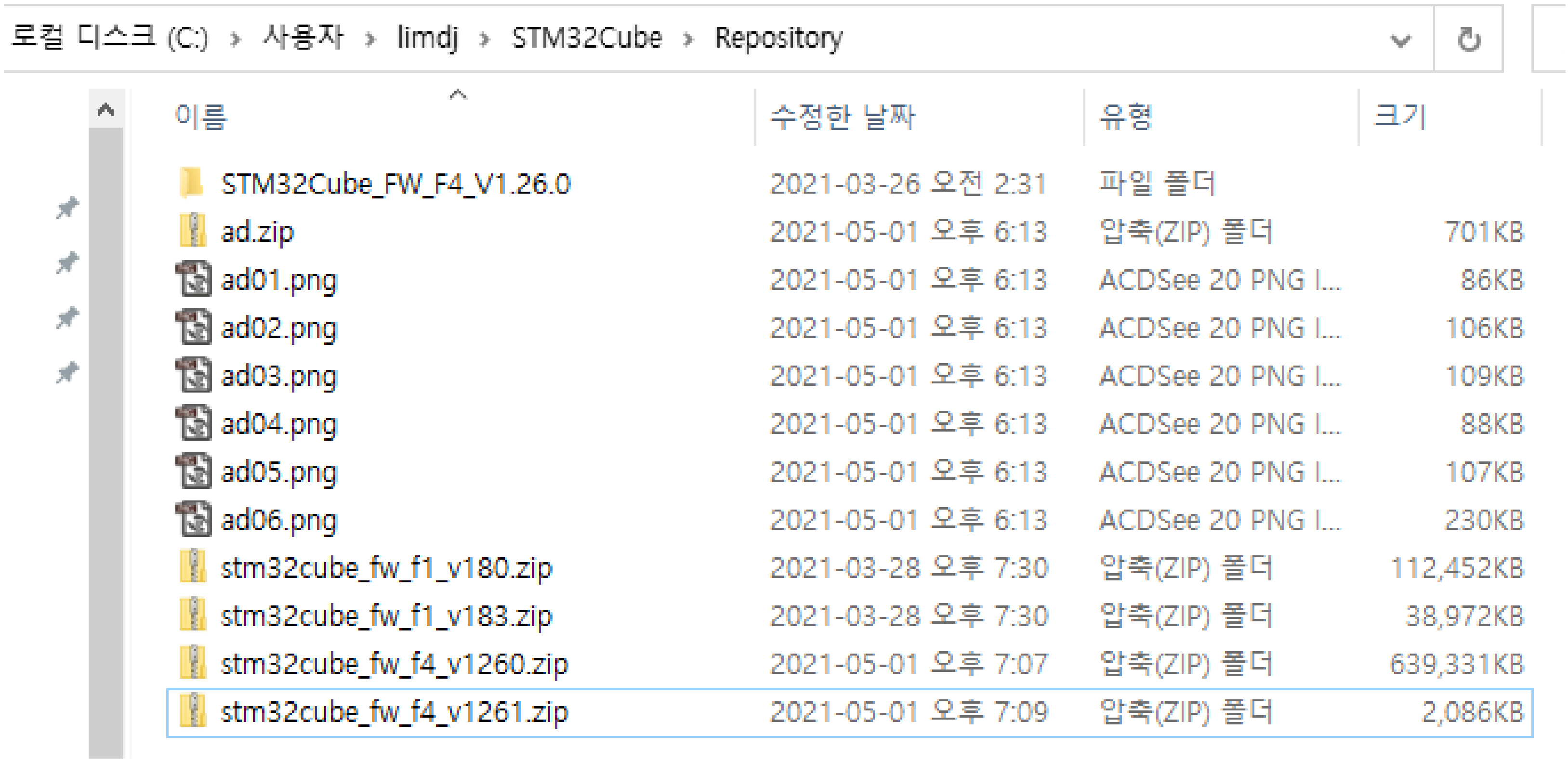
See Firmware Updater for settings related to firmware package installation

Tips:

Please use the Device Configuration Tool, and then use 'Project > Generate Code' to complete the project generation.

필요 없음

- 아래의 디렉토리에서 필요한 Firmware Package 디렉토리가 존재하는지 확인하고 없으면 압축을 풀어야 함

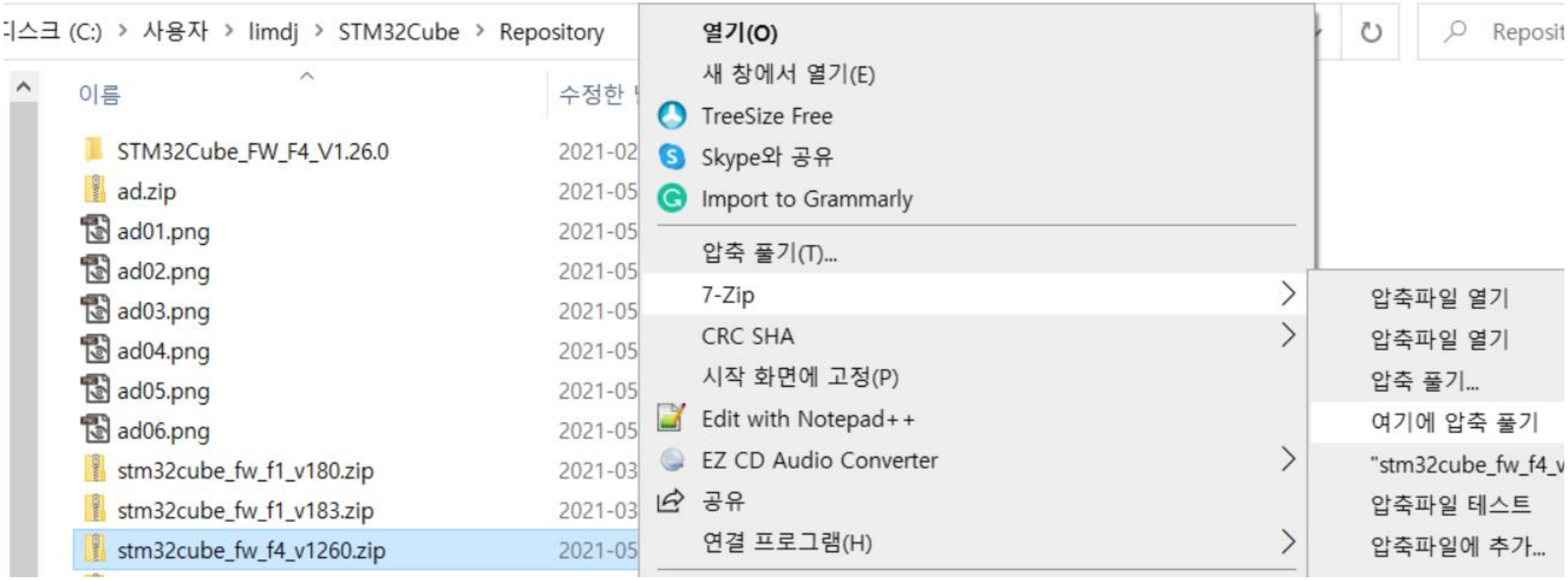


로컬 디스크 (C:) › 사용자 › limdj › STM32Cube › Repository

이름	수정한 날짜	유형	크기
STM32Cube_FW_F4_V1.26.0	2021-03-26 오전 2:31	파일 폴더	
ad.zip	2021-05-01 오후 6:13	압축(ZIP) 폴더	701KB
ad01.png	2021-05-01 오후 6:13	ACDSee 20 PNG I...	86KB
ad02.png	2021-05-01 오후 6:13	ACDSee 20 PNG I...	106KB
ad03.png	2021-05-01 오후 6:13	ACDSee 20 PNG I...	109KB
ad04.png	2021-05-01 오후 6:13	ACDSee 20 PNG I...	88KB
ad05.png	2021-05-01 오후 6:13	ACDSee 20 PNG I...	107KB
ad06.png	2021-05-01 오후 6:13	ACDSee 20 PNG I...	230KB
stm32cube_fw_f1_v180.zip	2021-03-28 오후 7:30	압축(ZIP) 폴더	112,452KB
stm32cube_fw_f1_v183.zip	2021-03-28 오후 7:30	압축(ZIP) 폴더	38,972KB
stm32cube_fw_f4_v1260.zip	2021-05-01 오후 7:07	압축(ZIP) 폴더	639,331KB
stm32cube_fw_f4_v1261.zip	2021-05-01 오후 7:09	압축(ZIP) 폴더	2,086KB

필요 없음

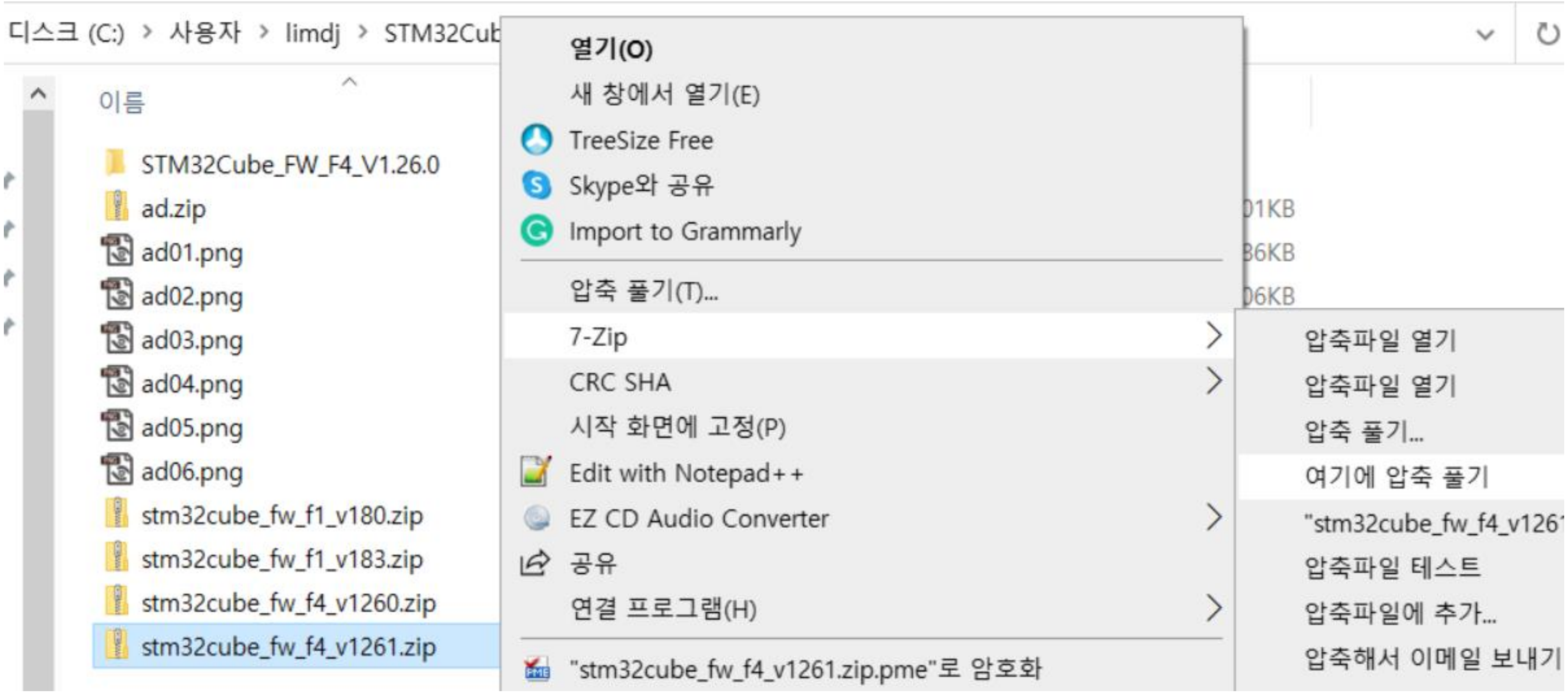
- 여기에 압축 풀기

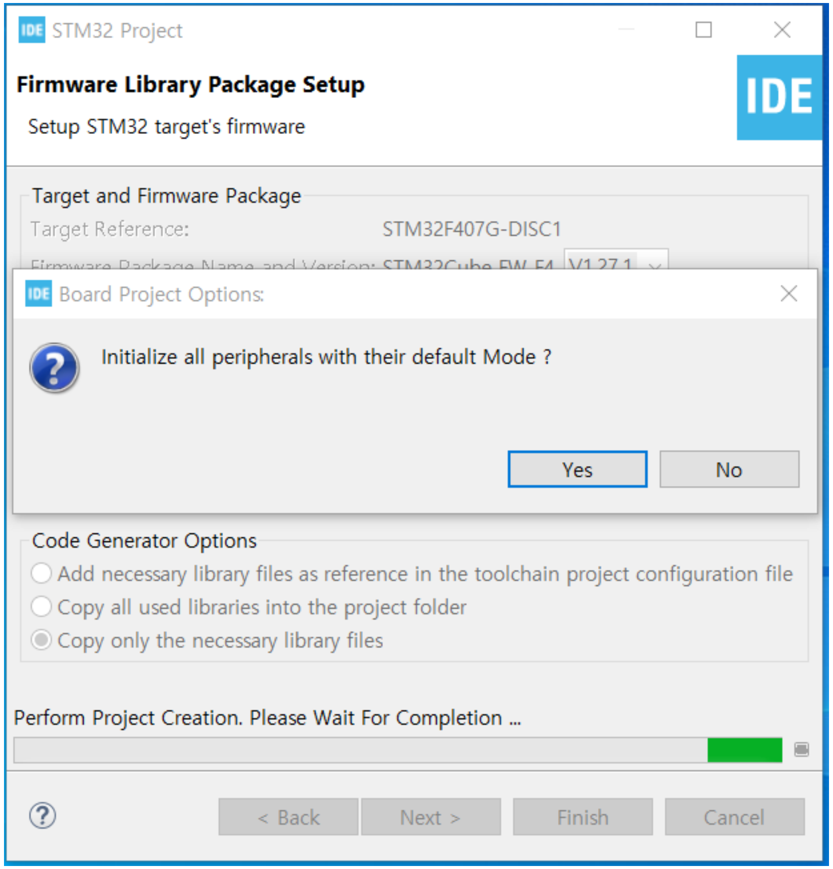


이름	수정한
STM32Cube_FW_F4_V1.26.0	2021-02
ad.zip	2021-05
ad01.png	2021-05
ad02.png	2021-05
ad03.png	2021-05
ad04.png	2021-05
ad05.png	2021-05
ad06.png	2021-05
stm32cube_fw_f1_v180.zip	2021-03
stm32cube_fw_f1_v183.zip	2021-03
stm32cube_fw_f4_v1260.zip	2021-05

필요 없음

- 만약 마이너 버전 업데이트가 있으면 아래와 같이 업데이트 파일의 압축을 풀며, 이때 이전 설치 디렉토리에 덮어쓰게 되므로 **모두 덮어쓰기**를 선택해서 업데이트를 함





STM32 Project

Firmware Library Package Setup

Setup STM32 target's firmware

Target and Firmware Package

Target Reference: STM32F407G-DISC1

Firmware Package Name and Version: STM32Cube FW_F4 V1.27.1

Board Project Options:

Initialize all peripherals with their default Mode ?

Code Generator Options

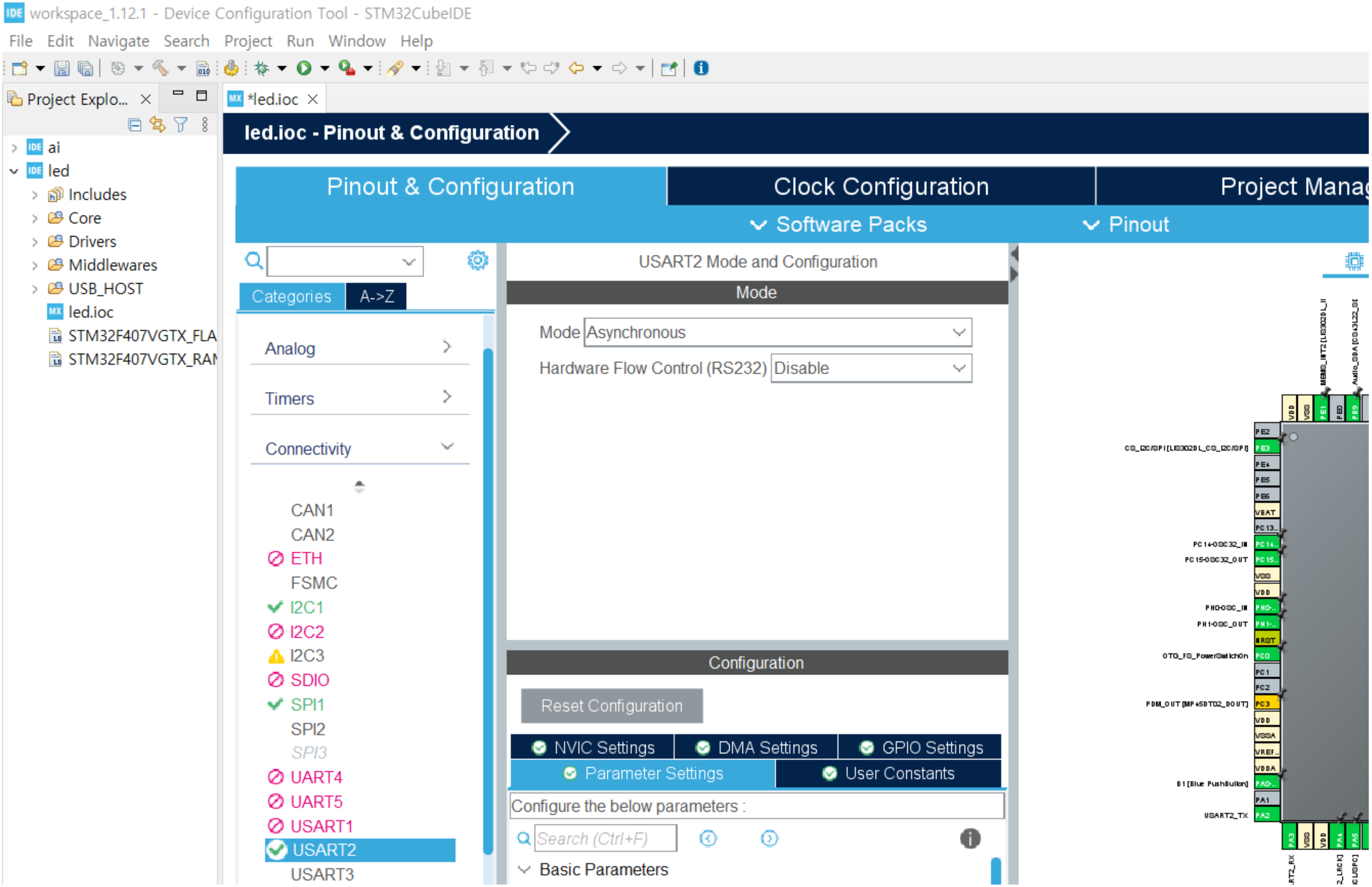
○ Add necessary library files as reference in the toolchain project configuration file

○ Copy all used libraries into the project folder

● Copy only the necessary library files

Perform Project Creation. Please Wait For Completion ...

- Enable USART2 (Mode: Asynchronous)



workspace_1.12.1 - Device Configuration Tool - STM32CubeIDE

led.ioc - Pinout & Configuration

USART2 Mode and Configuration

Mode

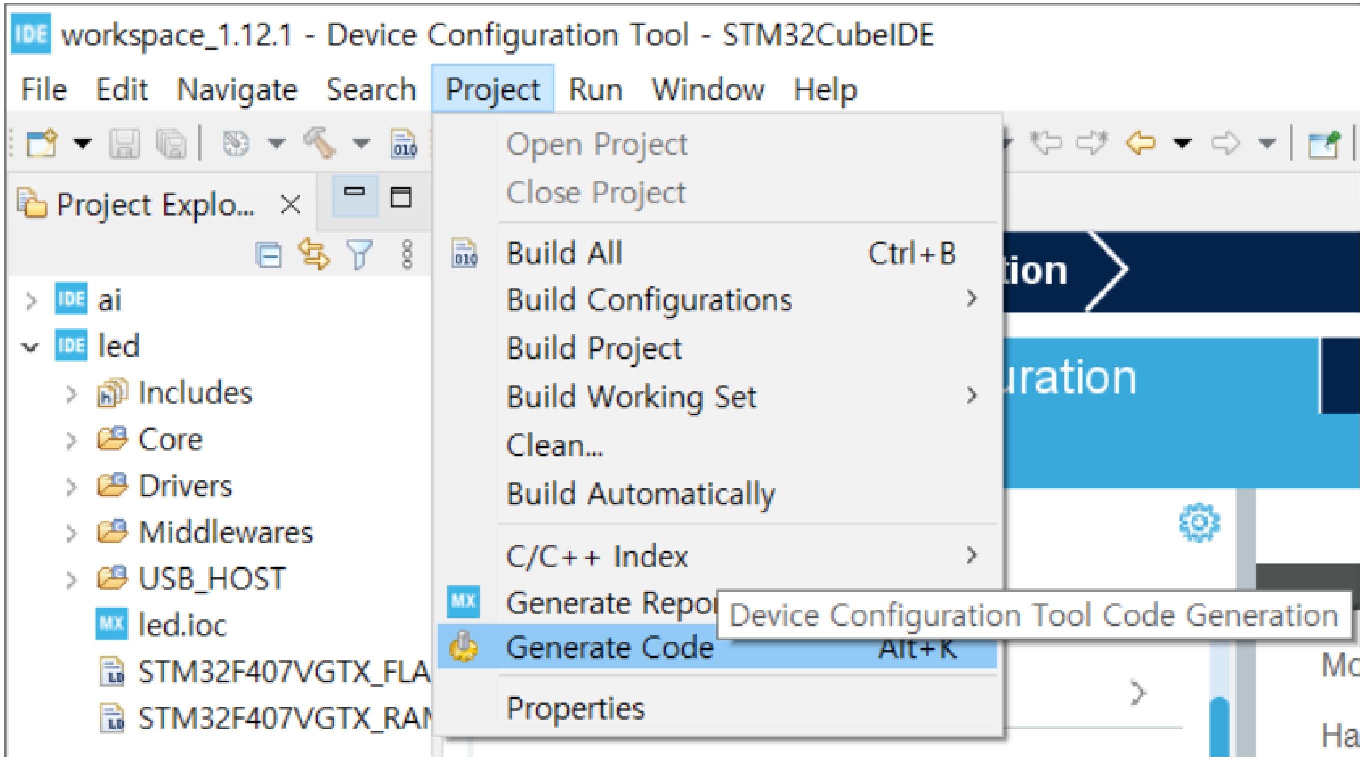
Mode: Asynchronous

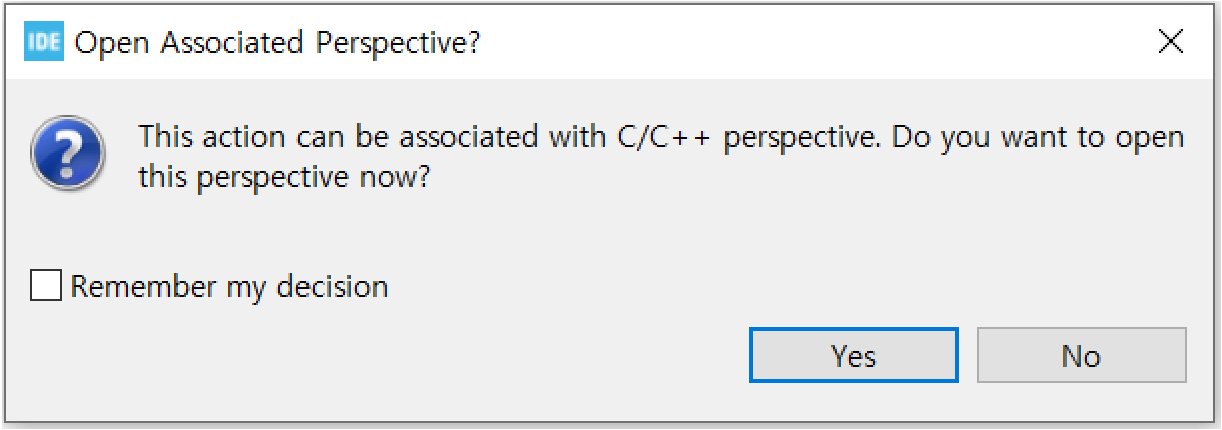
Hardware Flow Control (RS232): Disable

Configuration

Configure the below parameters :

Generate Code

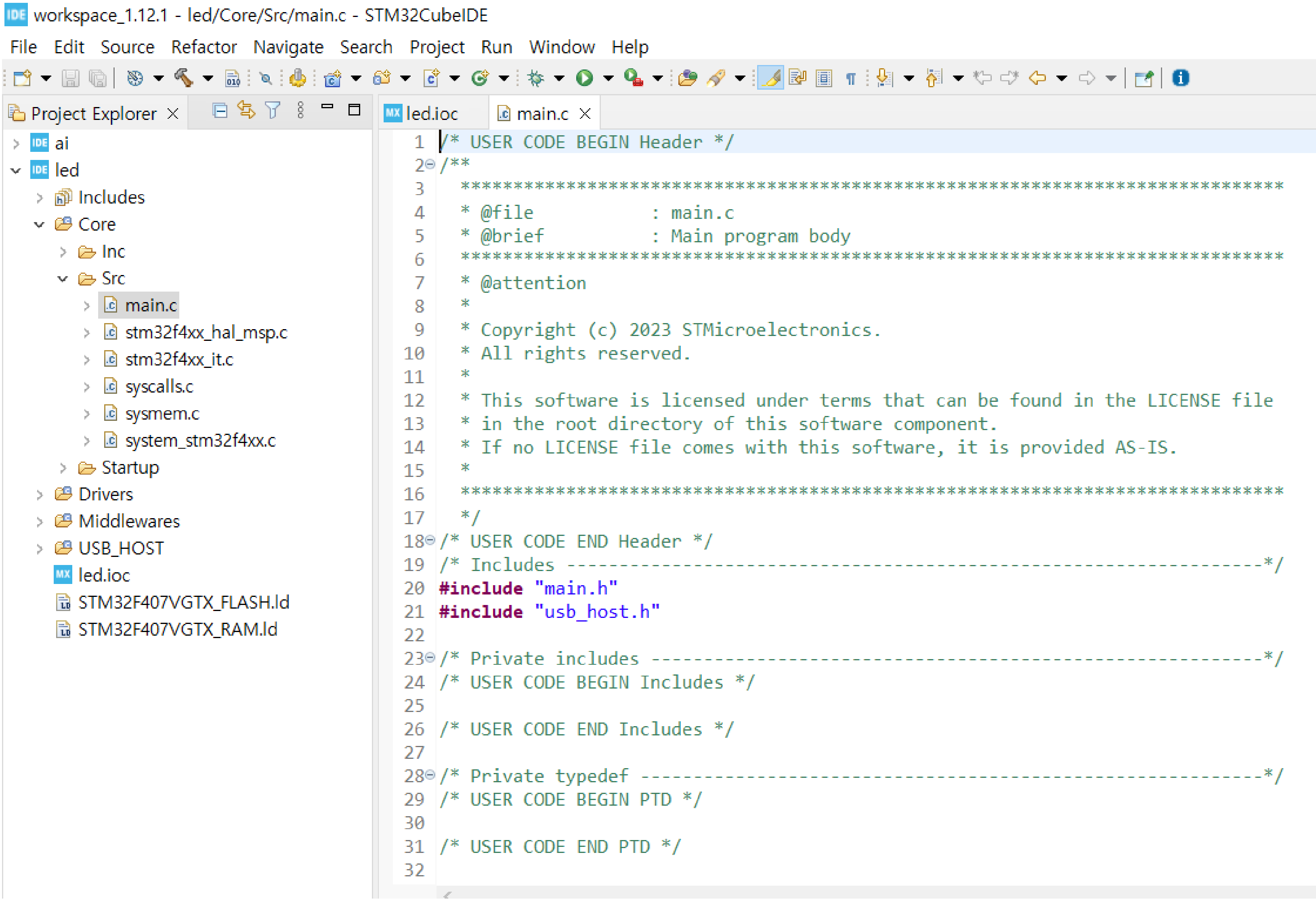




Open Associated Perspective?

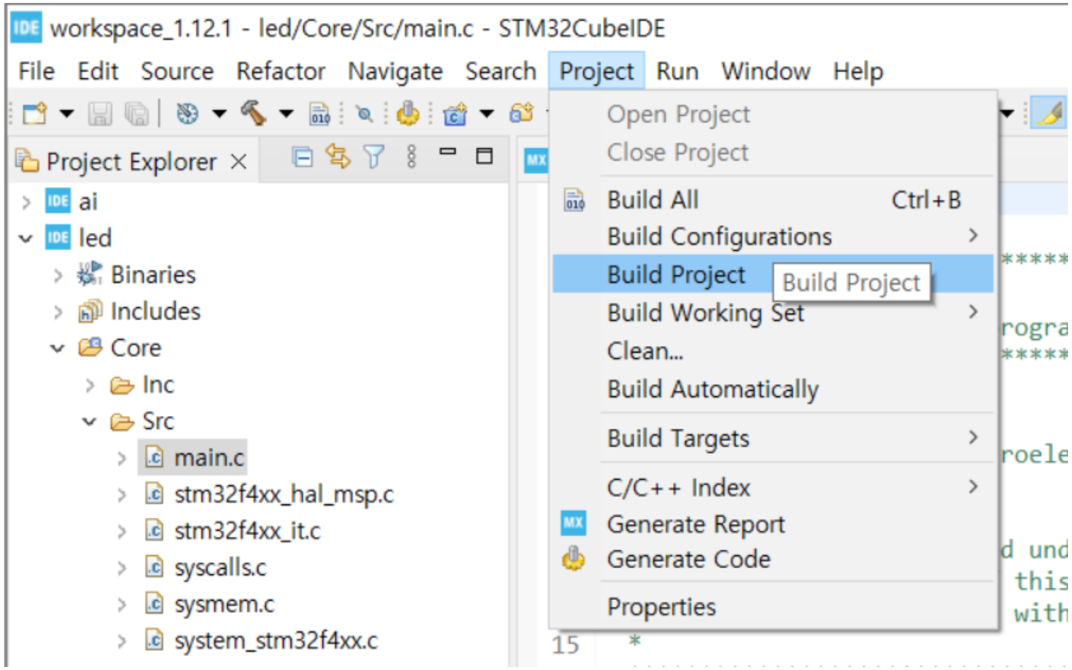
This action can be associated with C/C++ perspective. Do you want to open this perspective now?

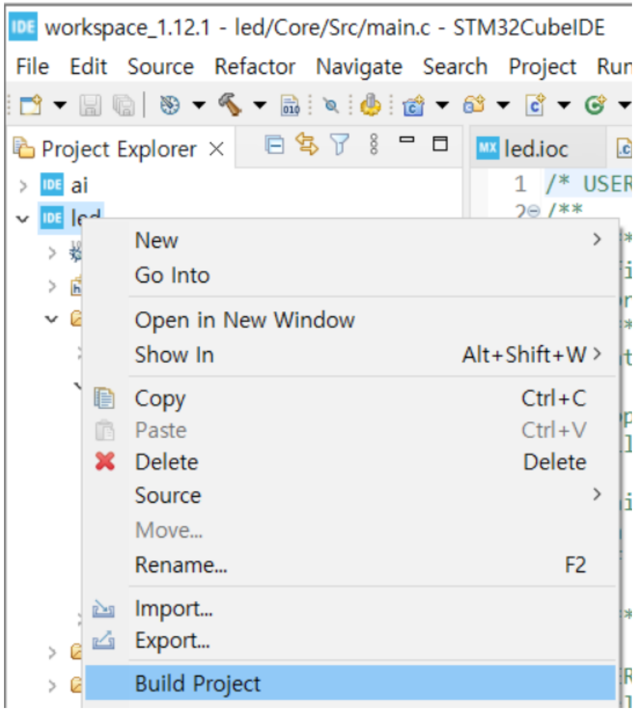
- Open main.c

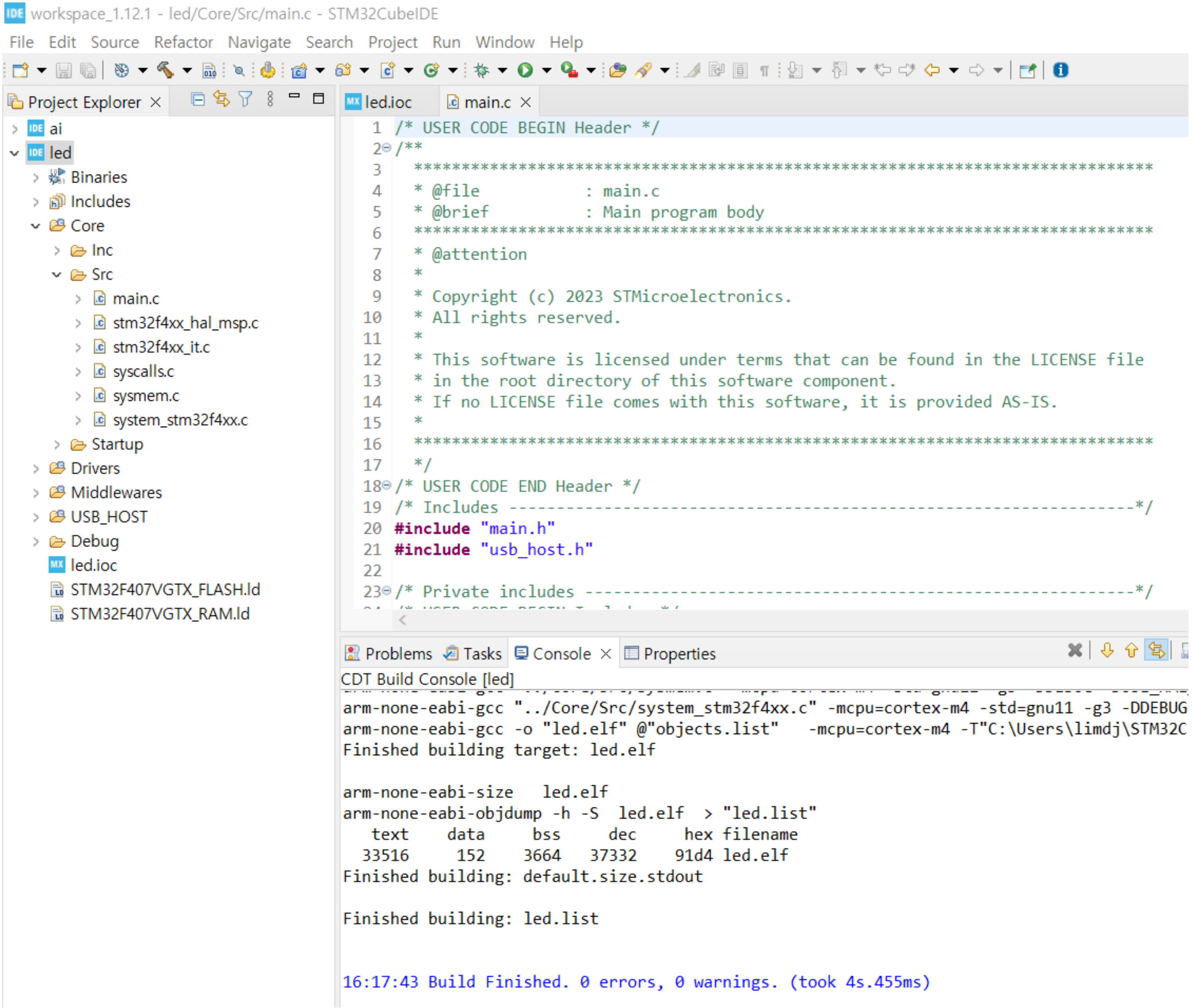


```c
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2023 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 comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "usb_host.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

```

Build Project





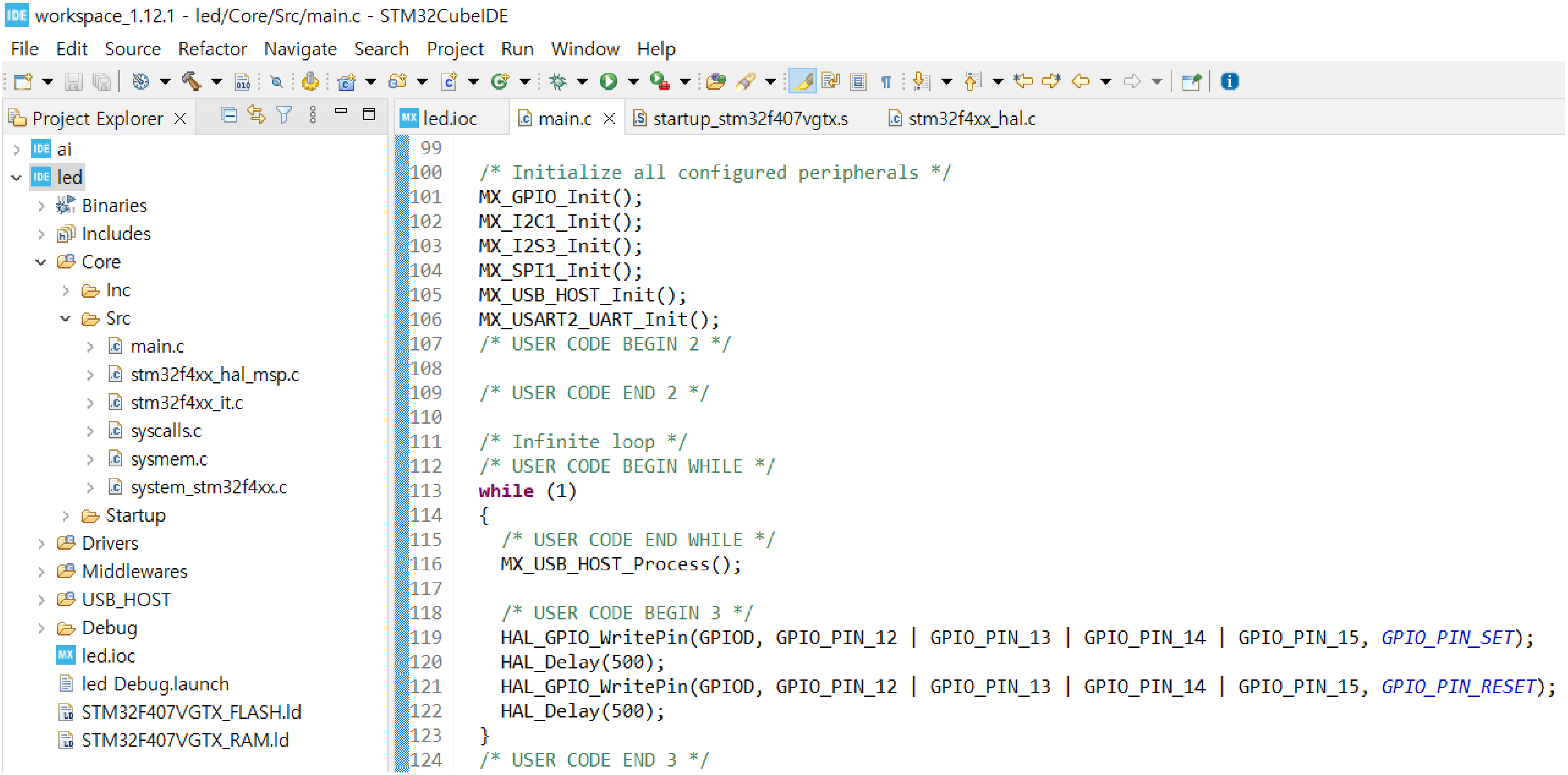


```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2023 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 comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "usb_host.h"

/* Private includes ----------------------------------------------------------*/
```

```
arm-none-eabi-gcc "../Core/Src/system_stm32f4xx.c" -mcpu=cortex-m4 -std=gnu11 -g3 -DDEBUG
arm-none-eabi-gcc -o "led.elf" @"objects.list"   -mcpu=cortex-m4 -T"C:\Users\limdj\STM32C
Finished building target: led.elf

arm-none-eabi-size   led.elf
arm-none-eabi-objdump -h -S  led.elf  > "led.list"
   text	   data	    bss	    dec	    hex	filename
  33516	    152	   3664	  37332	   91d4	led.elf
Finished building: default.size.stdout

Finished building: led.list


16:17:43 Build Finished. 0 errors, 0 warnings. (took 4s.455ms)
```

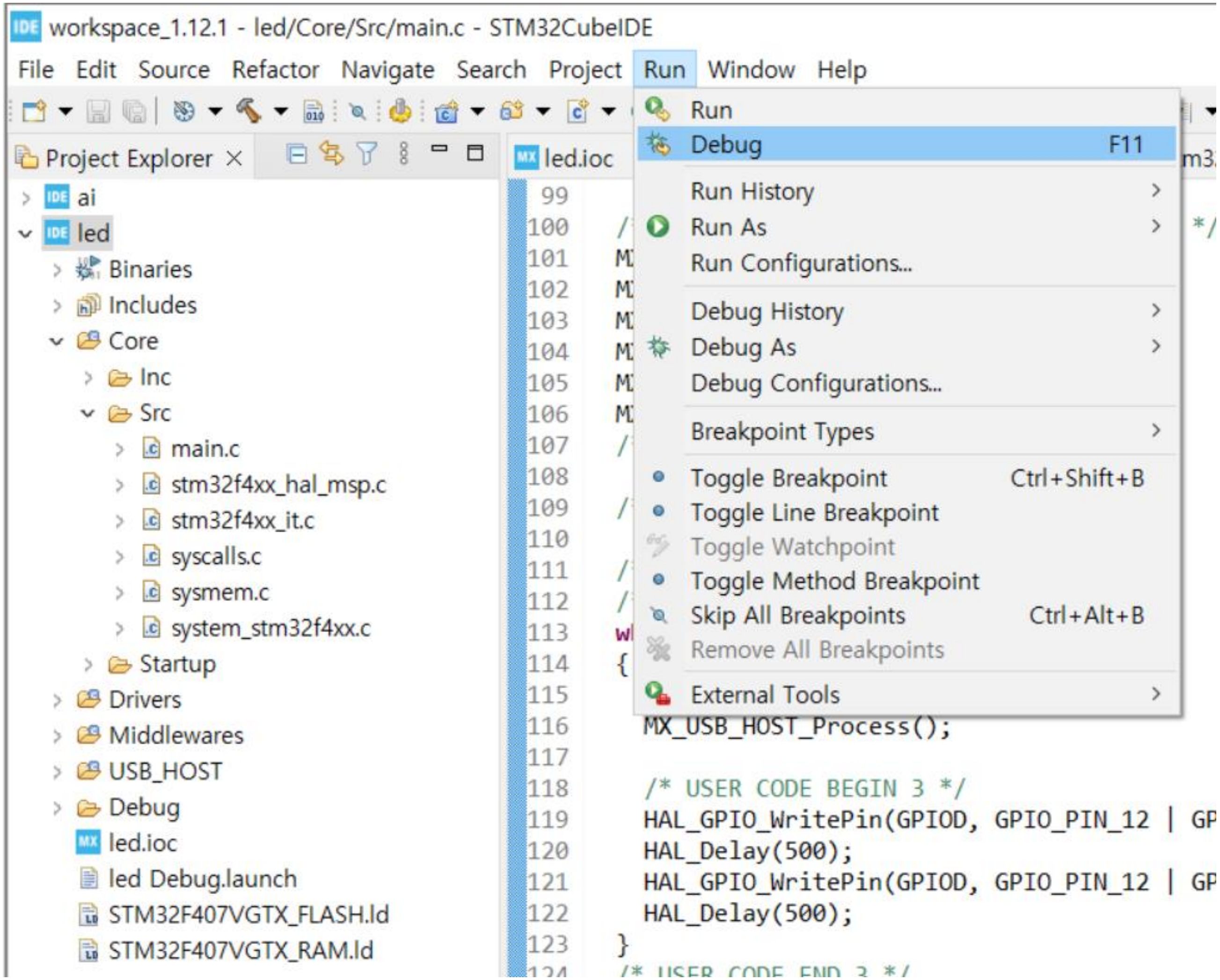
Source code main.c 에 입력

```
/* USER CODE BEGIN 3 */
   HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15, GPIO_PIN_SET);
   HAL_Delay(500);
   HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15, GPIO_PIN_RESET);
   HAL_Delay(500);
 }
 /* USER CODE END 3 */
```



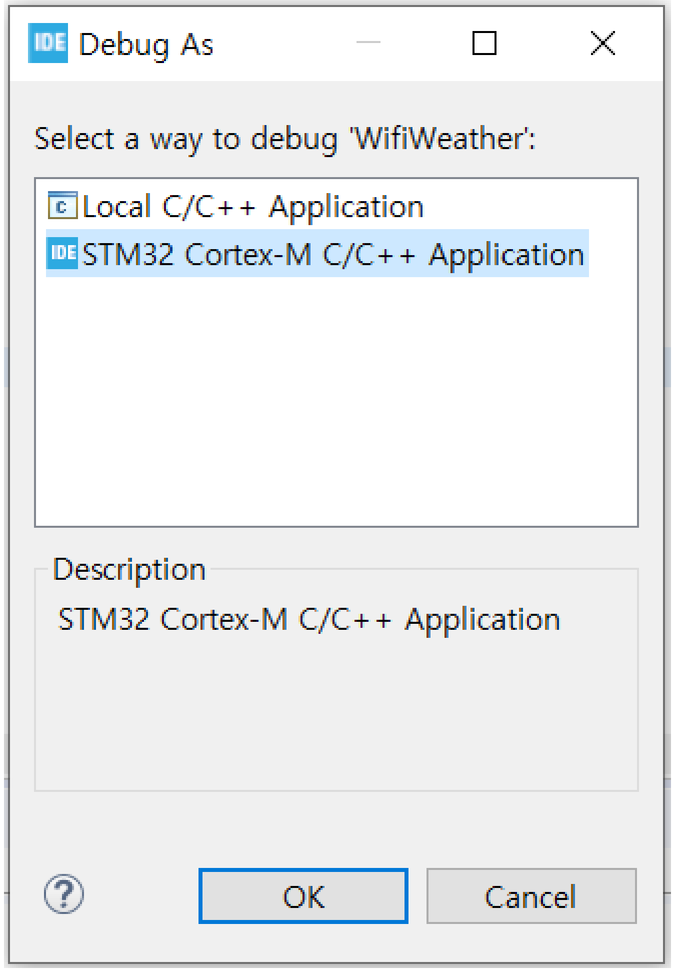
```c

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_I2C1_Init();
  MX_I2S3_Init();
  MX_SPI1_Init();
  MX_USB_HOST_Init();
  MX_USART2_UART_Init();
  /* USER CODE BEGIN 2 */

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
    MX_USB_HOST_Process();

    /* USER CODE BEGIN 3 */
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15, GPIO_PIN_SET);
    HAL_Delay(500);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15, GPIO_PIN_RESET);
    HAL_Delay(500);
  }
  /* USER CODE END 3 */
```

Run Debug



```
    MX_USB_HOST_Process();

    /* USER CODE BEGIN 3 */
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GP
    HAL_Delay(500);
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GP
    HAL_Delay(500);
  }
  /* USER CODE END 3 */
```

- 이 화면이 나올 경우도 있고 안 나와도 상관 없음



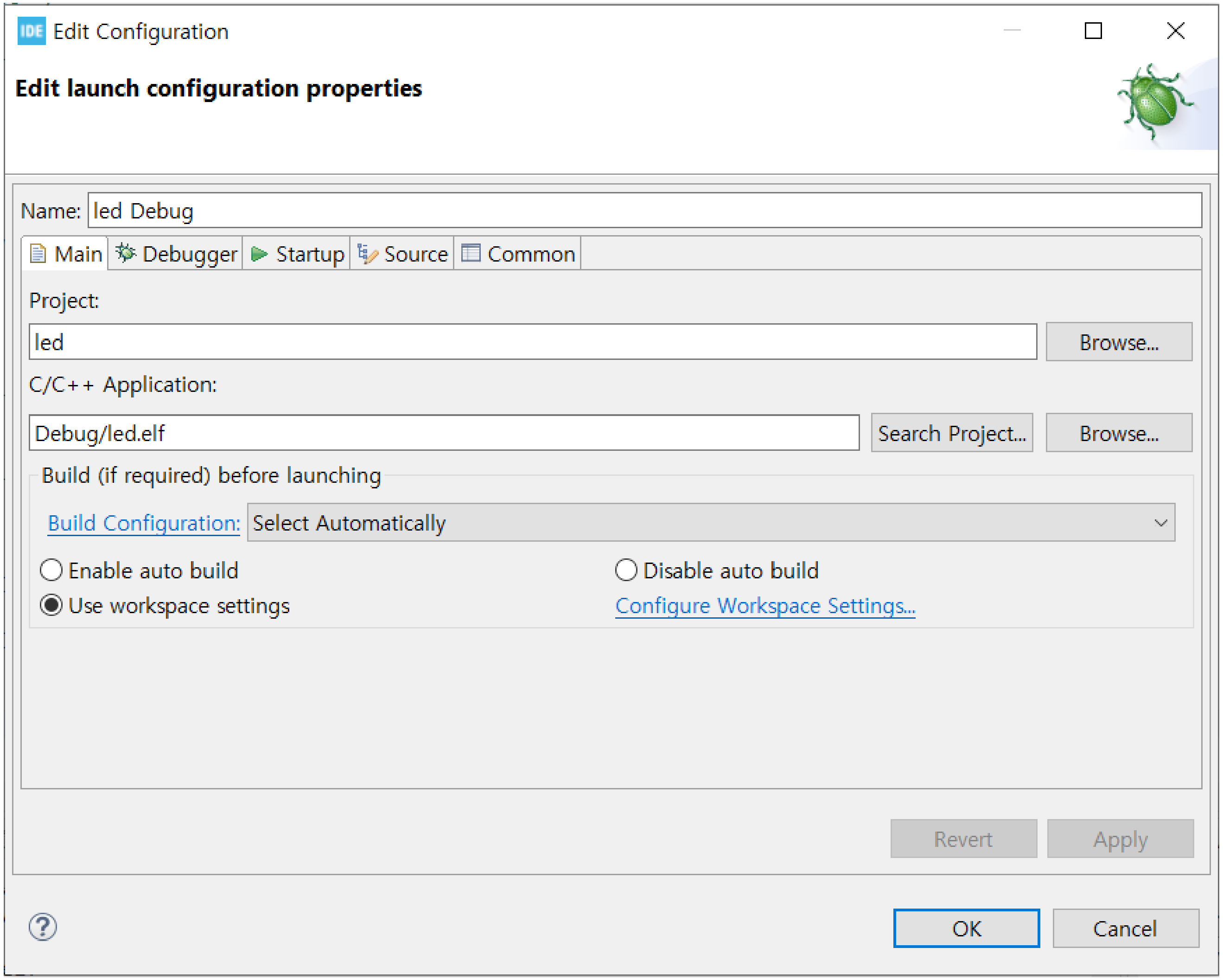
Select a way to debug 'WifiWeather':

Local C/C++ Application

STM32 Cortex-M C/C++ Application

Description

STM32 Cortex-M C/C++ Application



Edit Configuration

Edit launch configuration properties

Name: led Debug

Project:

led

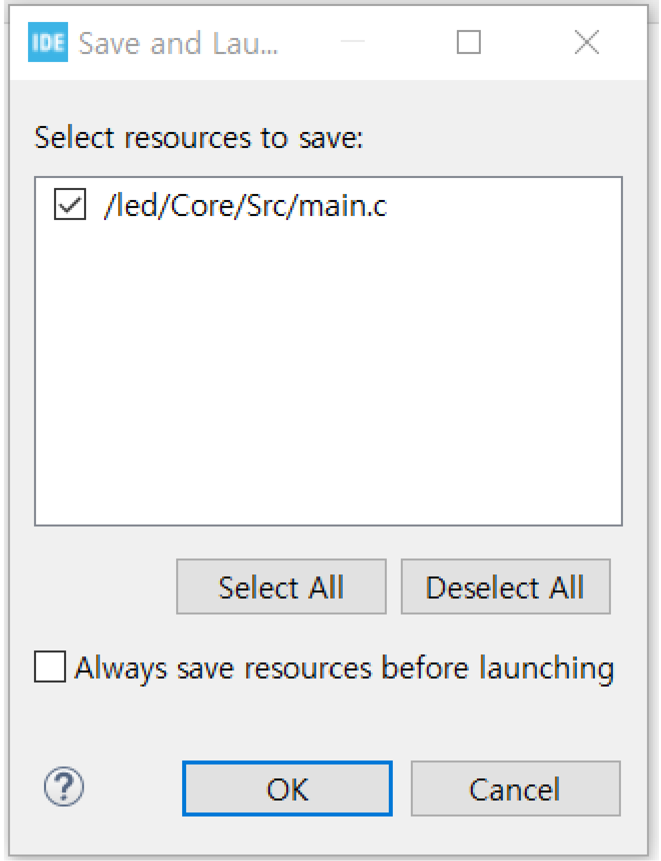
C/C++ Application:

Debug/led.elf

Build (if required) before launching

Build Configuration: Select Automatically

○ Enable auto build
○ Disable auto build
● Use workspace settings

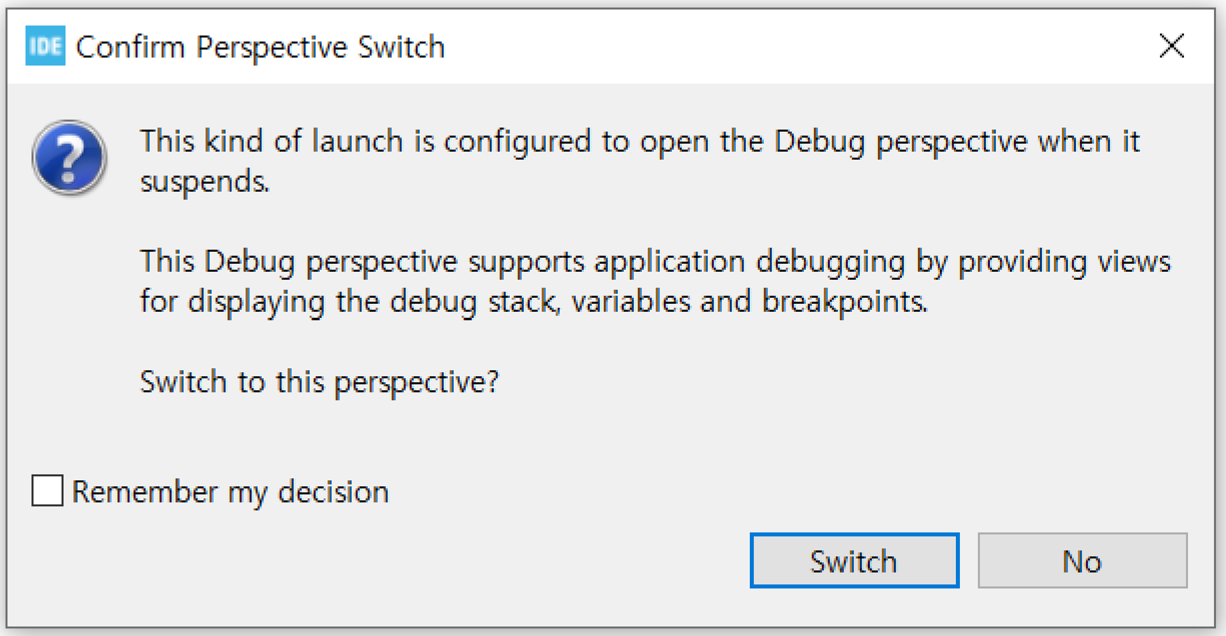


Select resources to save:

☑ /led/Core/Src/main.c

□ Always save resources before launching

- Remember my decision(이 박스를 체크하면 다시 안 물어 봄)
- Switch



Confirm Perspective Switch

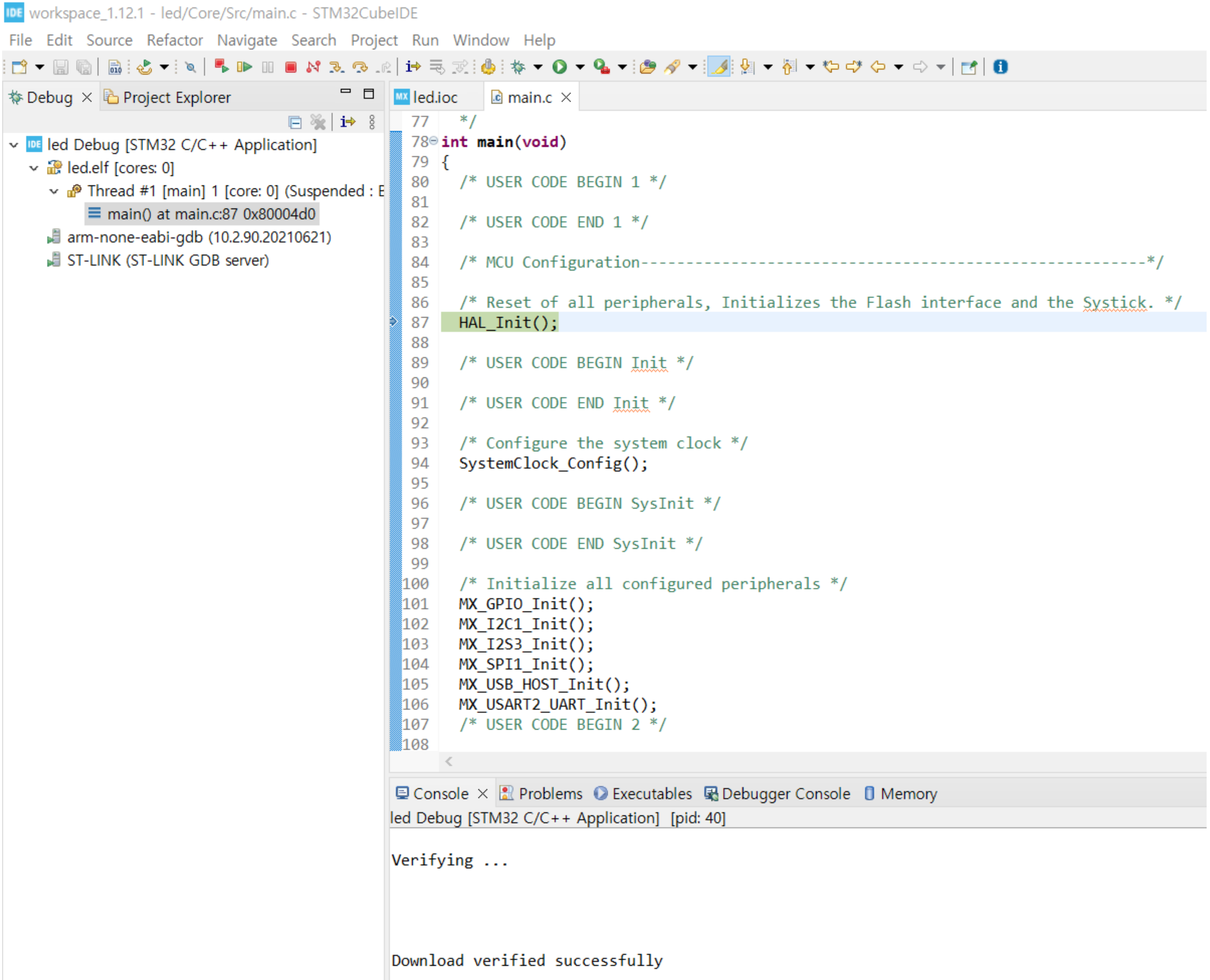
This kind of launch is configured to open the Debug perspective when it suspends.

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

Switch to this perspective?

☐ Remember my decision

Debugger screen 으로 전환됨



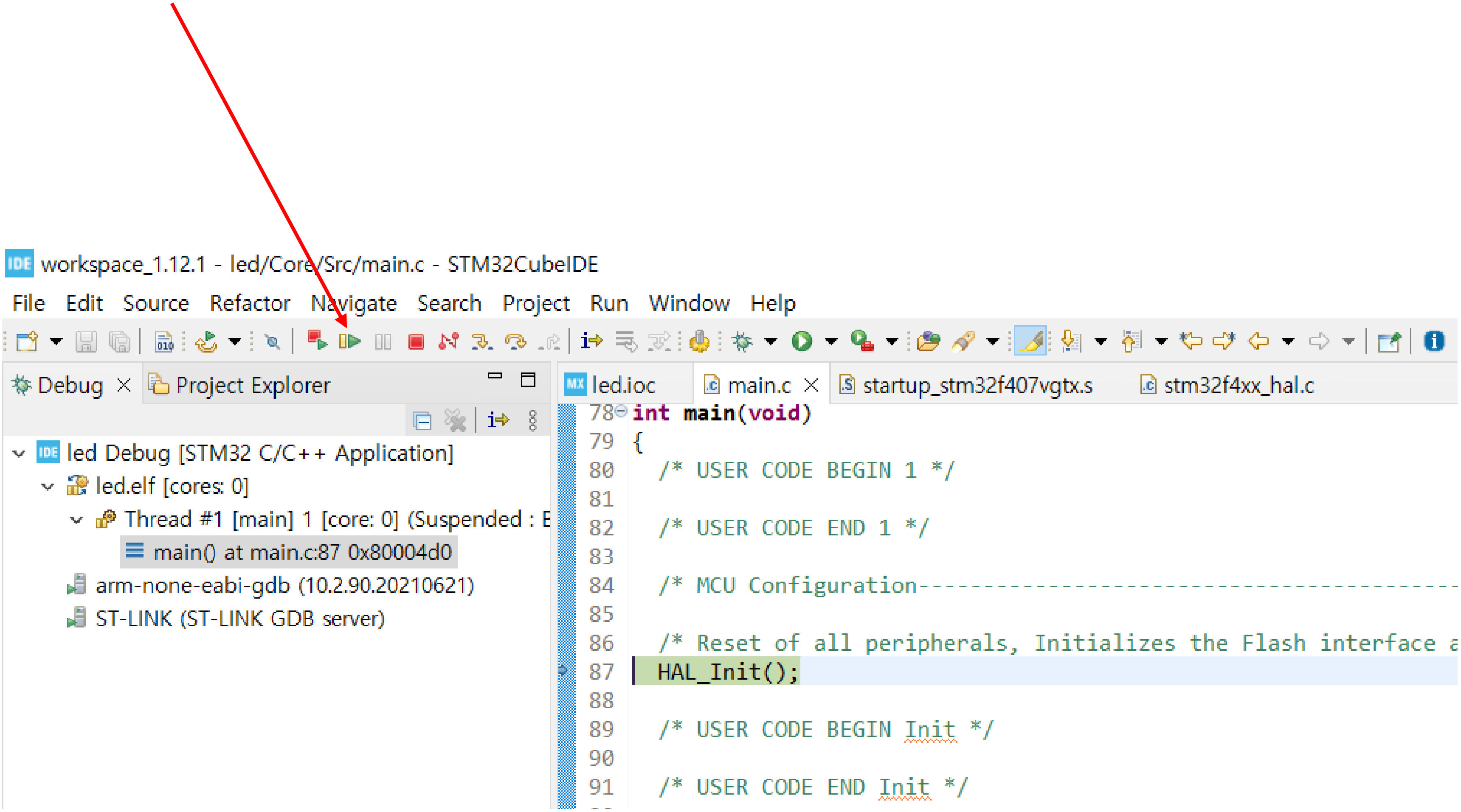
```
led Debug [STM32 C/C++ Application]
  led.elf [cores: 0]
    Thread #1 [main] 1 [core: 0] (Suspended : E
      main() at main.c:87 0x80004d0
  arm-none-eabi-gdb (10.2.90.20210621)
  ST-LINK (ST-LINK GDB server)
```

```c
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_I2C1_Init();
  MX_I2S3_Init();
  MX_SPI1_Init();
  MX_USB_HOST_Init();
  MX_USART2_UART_Init();
  /* USER CODE BEGIN 2 */

```

```
led Debug [STM32 C/C++ Application]  [pid: 40]

Verifying ...



Download verified successfully
```

Debugger screen에서 실행

- Resume

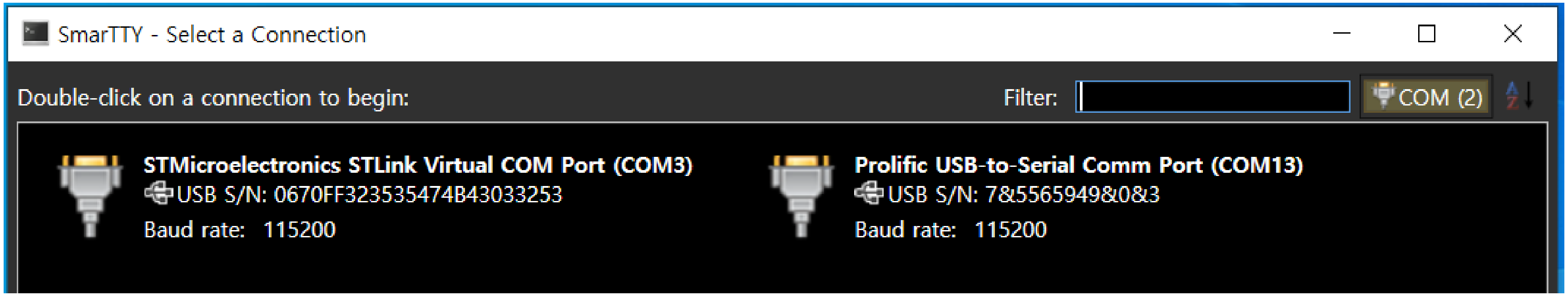


led Debug [STM32 C/C++ Application]
led.elf [cores: 0]
Thread #1 [main] 1 [core: 0] (Suspended : E
main() at main.c:87 0x80004d0
arm-none-eabi-gdb (10.2.90.20210621)
ST-LINK (ST-LINK GDB server)

```
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration------------------------------------------

  /* Reset of all peripherals, Initializes the Flash interface a
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */
```

Exercise 1

- 터미널 프로그램인 SmarTTY를 열어서 usb-to-serial에 할당된 com port를 연 후, 아래와 같이 메시지를 프린트 하는 코드를 작성하고 실행한다.

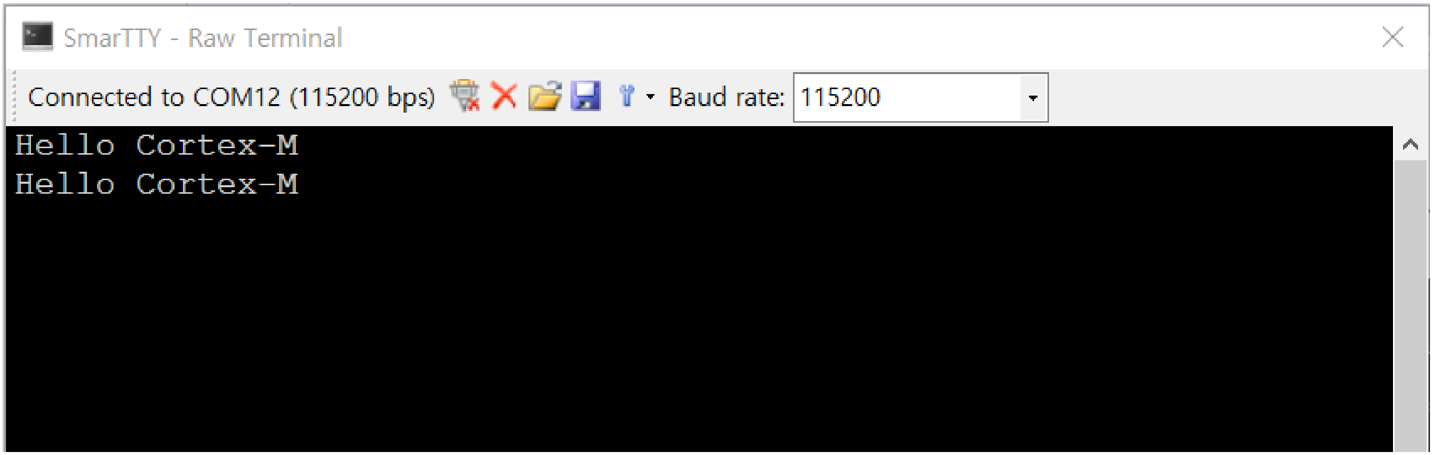


SmarTTY - Select a Connection

Double-click on a connection to begin:

STMicroelectronics STLink Virtual COM Port (COM3)
USB S/N: 0670FF323535474B43033253
Baud rate: 115200

Prolific USB-to-Serial Comm Port (COM13)
USB S/N: 7&5565949&0&3
Baud rate: 115200



SmarTTY - Raw Terminal

Connected to COM12 (115200 bps)

Baud rate: 115200

```
Hello Cortex-M
Hello Cortex-M
```

Source code main.c 에 입력

```
/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "string.h"
/* USER CODE END Includes */
```

```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
void PrintString(uint8_t * string)
{
   HAL_UART_Transmit(&huart2, (uint8_t *)string, strlen((char *)string), 0xffff);
}
/* USER CODE END 0 */
```

```
 /* USER CODE BEGIN 2 */
    PrintString((uint8_t *)"Hello Cortex-M\n\r");
 /* USER CODE END 2 */
```

Exercise 2

- 시리얼 터미널이 연결된 상태에서 스페이스바를 한 번 누를 때 마다 4개의 led가 한 개 씩 교대로 켜지는 프로그램을 작성하고 시험해 본다.

Source code main.c 에 입력

```
/* USER CODE BEGIN 1 */
 uint8_t buffer[10];
 uint8_t state=0;
 /* USER CODE END 1 */


/* USER CODE BEGIN 3 */
   HAL_UART_Receive(&huart2, (uint8_t *)buffer, 1, 10);
   if (buffer[0]==' ') { state++; if (state > 3) {state = 0;}  buffer[0]=0;}
   if (state==0){
     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12, GPIO_PIN_SET);
     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13 | GPIO_PIN_14 | GPIO_PIN_15, GPIO_PIN_RESET);
   }
   if (state==1){
     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GPIO_PIN_14 | GPIO_PIN_15, GPIO_PIN_RESET);
   }
   if (state==2){
     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_SET);
     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GPIO_PIN_13 | GPIO_PIN_15, GPIO_PIN_RESET);
   }
   if (state==3){
     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_15, GPIO_PIN_SET);
     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12 | GPIO_PIN_13 | GPIO_PIN_14, GPIO_PIN_RESET);
   }
 }
 /* USER CODE END 3 */
```

Exercise 3

- Exercise 2로 프로그램된 Cortex-M 보드의 시리얼 포트로 500msec 간격으로 스페이스 문자(‘ ‘)를 주기적으로 보내는 파이썬 프로그램을 작성해서 시험해 본다. 즉, 4개의 led가 500msec 간격으로 켜지는지 확인한다.

- 주의 사항: 터미널 프로그램이 열려 있으면 파이썬 프로그램에서 시리얼 포트를 열지 못하므로 터미널 프로그램을 반드시 닫는다.

- Anaconda prompt를 열어서 시리얼 통신 패키지를 설치한다.

```
pip install pyserial
```

Python Code

```
import serial
import time

port = "COM13"
baud = 115200

ser = serial.Serial(port, baud, timeout=1)
    # open the serial port
if ser.isOpen():
     print(ser.name + ' is open...')

for i in range(30):
  ser.write(bytes(' ',encoding='ascii'))
  time.sleep(0.5)

ser.close()
```

주의: 위의 코드에서 COM번호는 PC마다 다를 수 있으므로 SmarTTY 프로그램에서 COM번호를 확인해서 변경해야 한다. 이 프로그램을 Control-C 를 이용해서 강제 종료할 경우 시리얼 포트가 열린 채로 종료 되므로, 다음에 다시 실행할 때 시리얼 포트가 열리지 않는다. 그런 경우에는 USB 케이블을 뺐다가 다시 연결한다.

Exercise 4

- 첫 예제(Exercise 1의 앞에 있는 LED blinking code)의 코드를 다음과 같이 수정한다.
- LED의 깜빡이는 속도를 2배 빠르게 조정한다.
- PC의 터미널의 입력을 받아들이도록 수정하여 아래와 같은 동작을 구현한다.
  - 프로그램이 처음 시작되면 모든 LED를 끈 상태에서 대기한다.
  - PC의 터미널 프로그램에서 스페이스바를 누르면 깜빡이는 동작을 시작한다.
  - 다시 스페이스바를 누르면 깜빡이는 동작을 정지한다.
  - 위의 동작을 무한 반복한다.