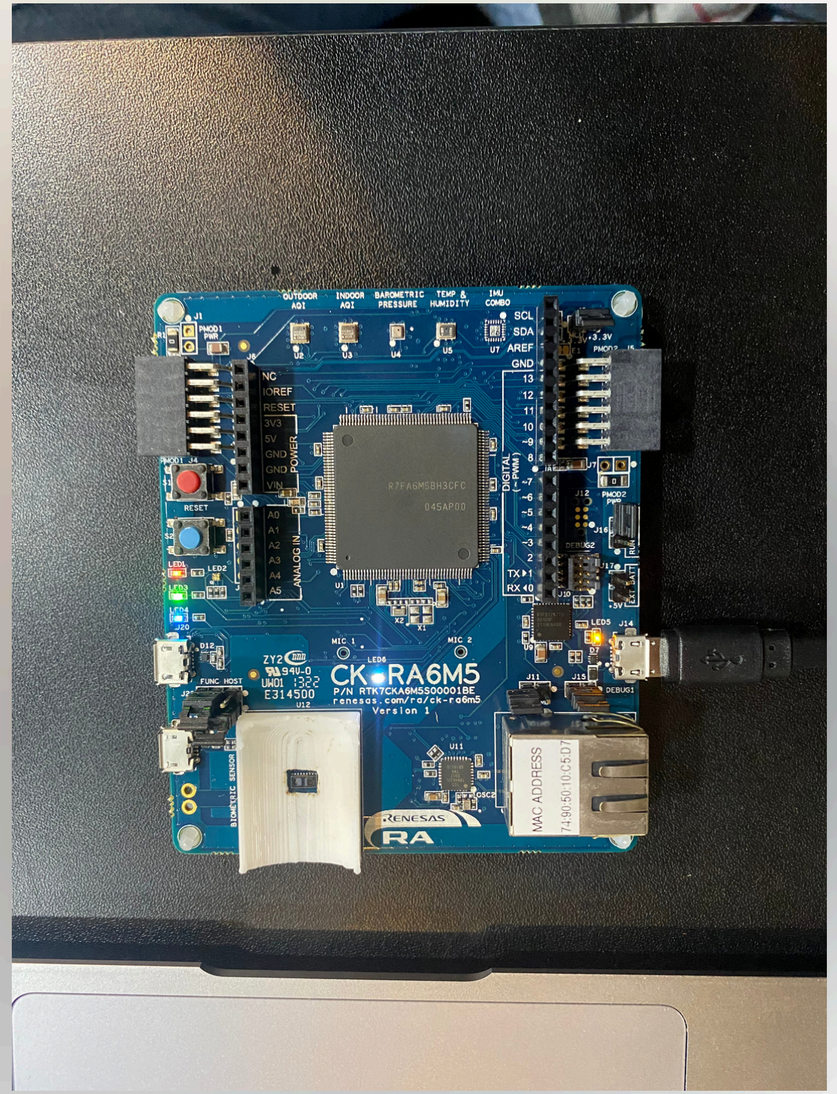



# Blinky LED with Renesas

Nhóm 1:

- Do Minh Quan - 20T2
- Truong Quang Chu - 20T2
- Le Van Dat - 20T2

# Tổng quan

- Giới thiệu
- Hướng dẫn cài đặt
- Giải thích code
- Demo

# I. GIỚI THIỆU

# Giới thiệu về Renesas Electronics Corporation

Renesas Electronics Corporation là một trong những công ty hàng đầu trong lĩnh vực bán dẫn, chuyên về các giải pháp microcontroller (MCU) và microprocessor (MPU).

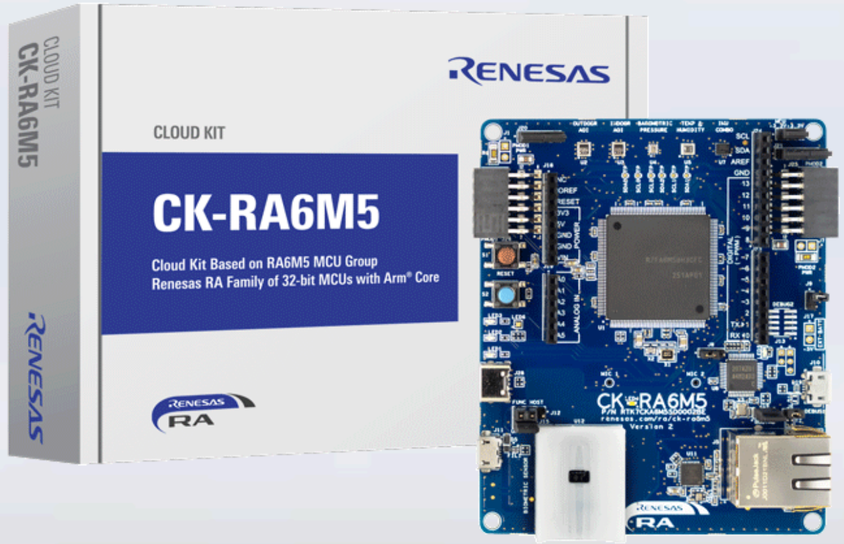
Được thành lập vào năm 2010 sau khi sáp nhập NEC Electronics Corporation và Renesas Technology Corporation.

# Renesas cloud kit CK-RA6M5

RA6M5 được thiết kế cho các ứng dụng IoT, với khả năng bảo mật tích hợp và hiệu suất xử lý cao.

Thông số kỹ thuật:

- Vi điều khiển: Renesas RA6M5 với bộ xử lý ARM Cortex-M33.
- Tần số: 200 MHz.
- Bộ nhớ: 2 MB Code Flash và 512 KB SRAM.
- 176 pins, LQFP package
- Giao tiếp: Hỗ trợ nhiều giao thức như UART, SPI, I2C, CAN, và USB
- Tính năng bảo mật: Tích hợp Trusted Secure IP, hỗ trợ mã hóa phần cứng.

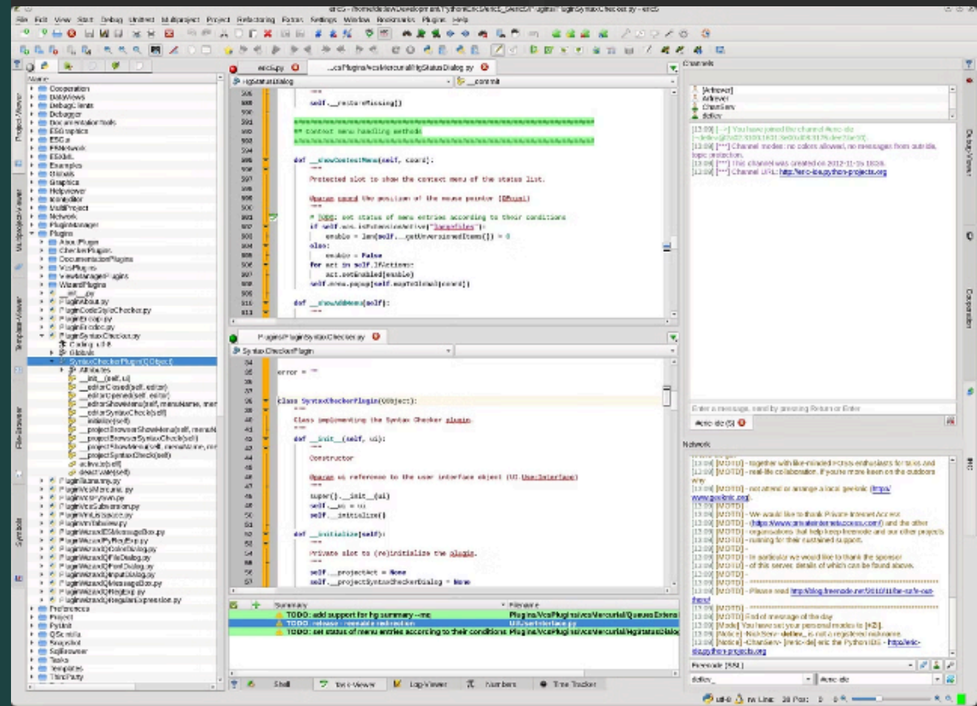


# E2Studio

Hỗ trợ đa nền tảng: Vi điều khiển RA, RX, RL78, RH850.

Công cụ gỡ lỗi mạnh mẽ: JTAG, SWD, Renesas E2 Emulator.

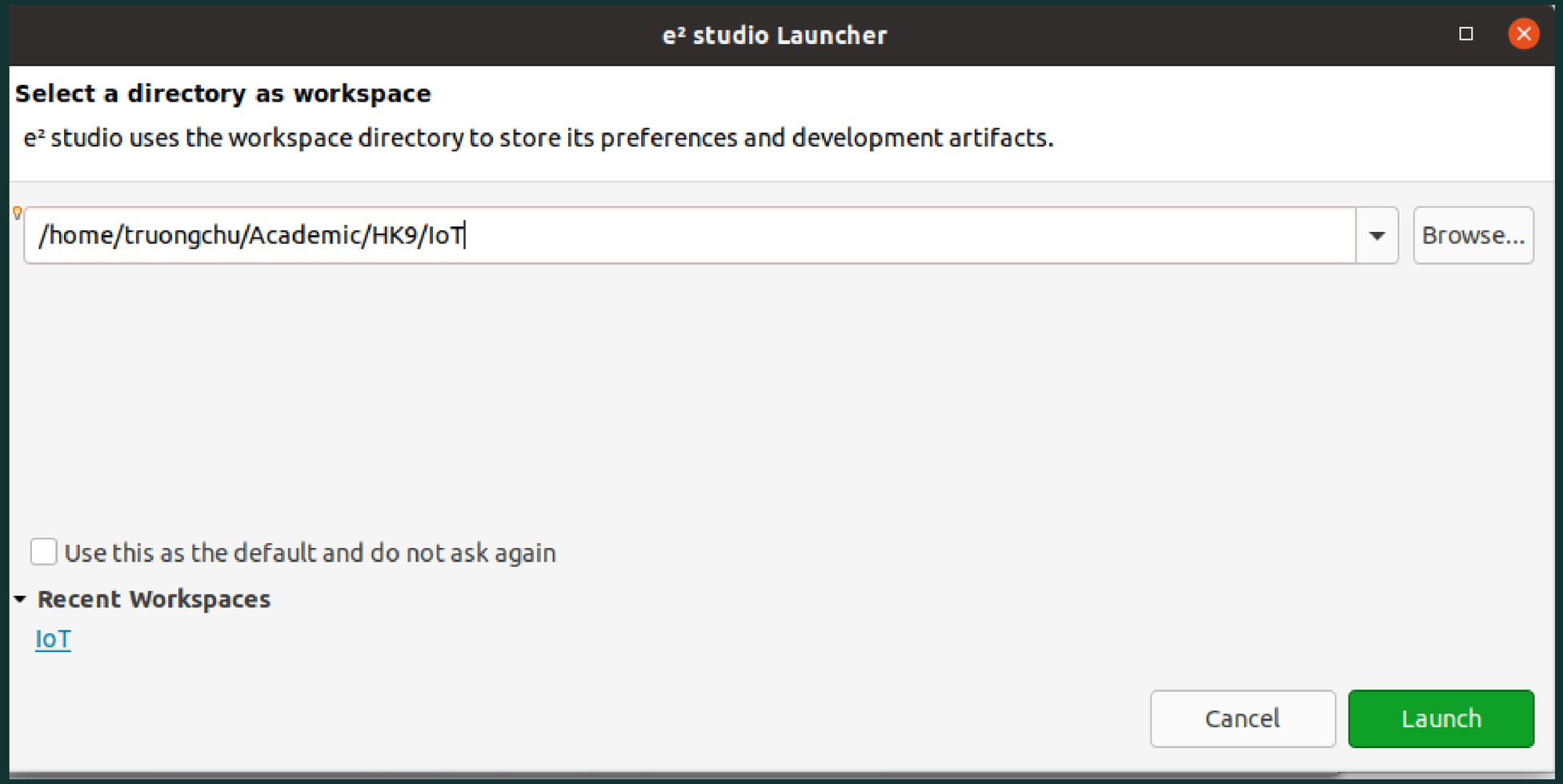
Cài từ https://github.com/renesas/fsp cho RA MCUs



# II. Hướng dẫn cài đặt

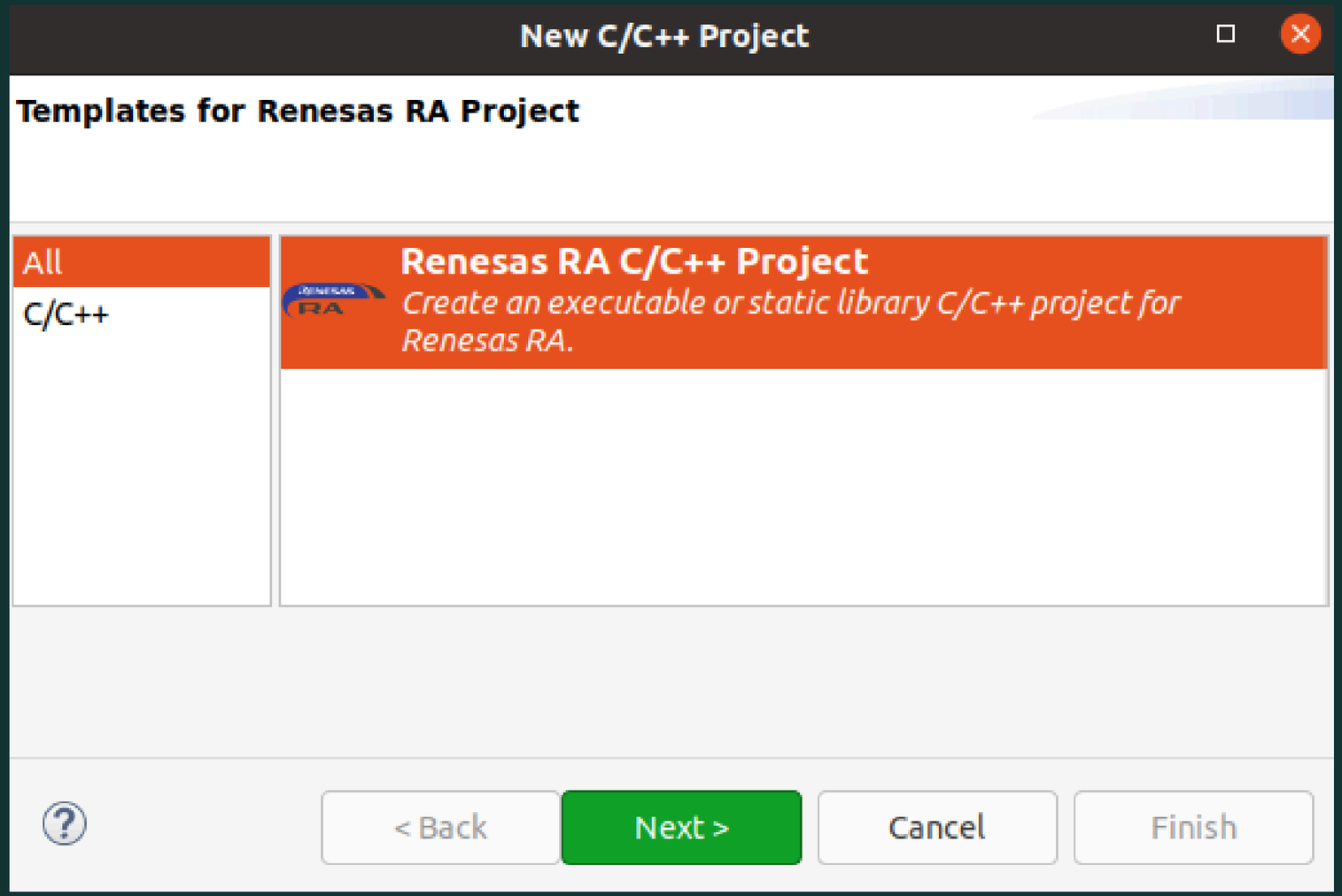
# Hướng dẫn cài đặt chương trình

Bước 1: Mở e2studio và tạo mới hoặc sử dụng 1 workspace có sẵn



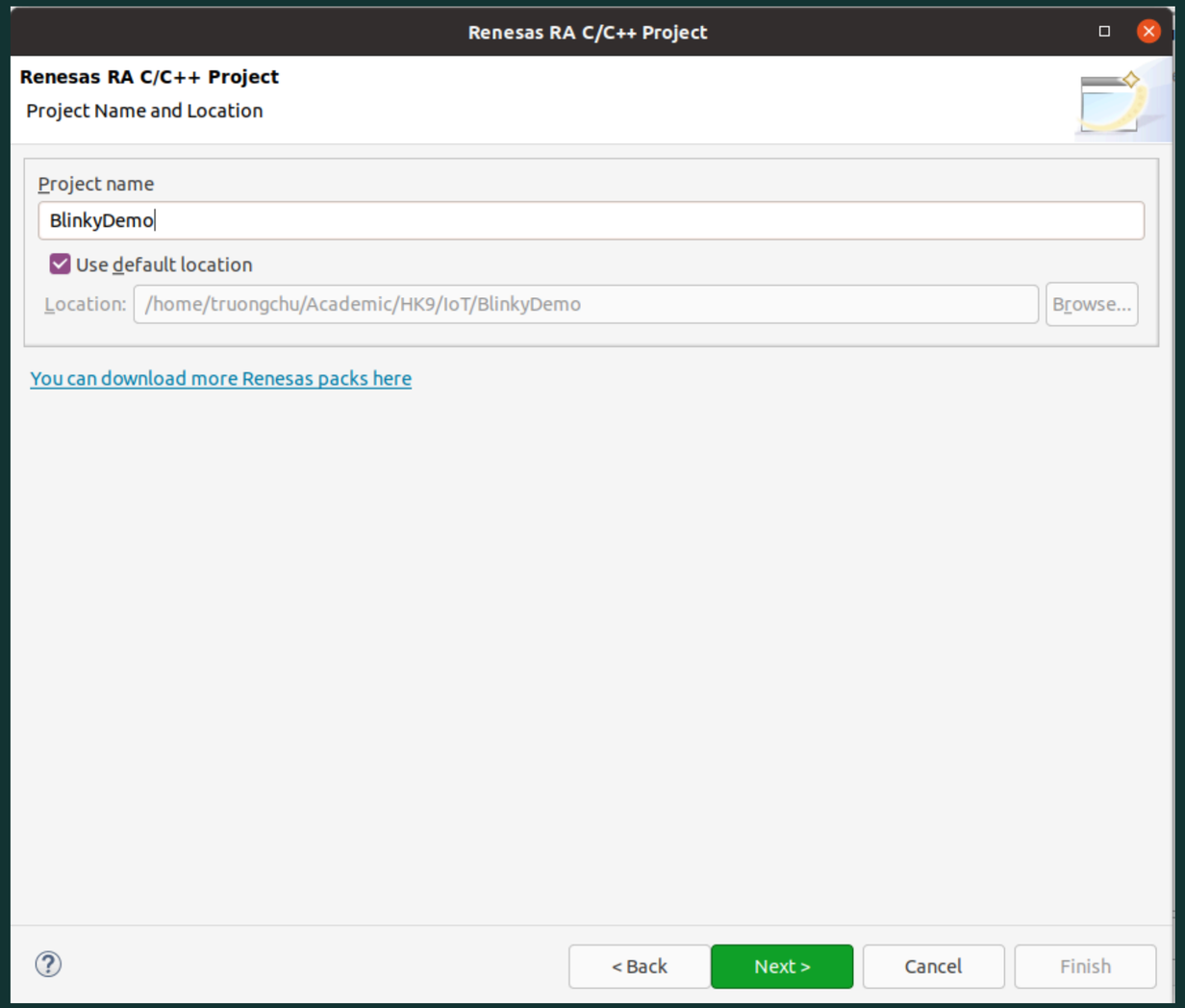
# Hướng dẫn cài đặt chương trình

Bước 2: File -> New -> Renesas C/C++ Project -> Renesas RA (CK RA6M5 MCU)



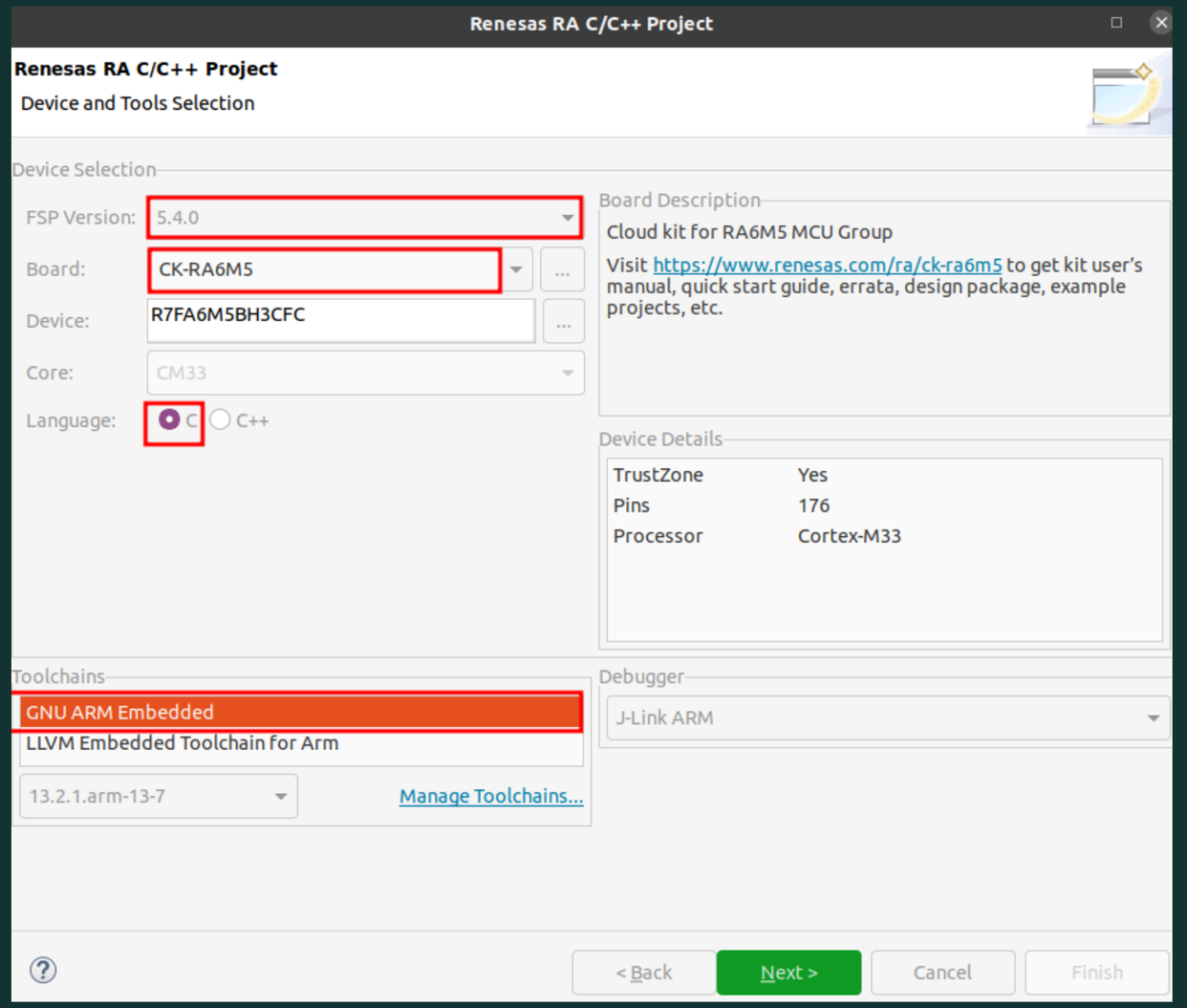
# Hướng dẫn cài đặt chương trình

## Bước 3: Nhập tên cho Project



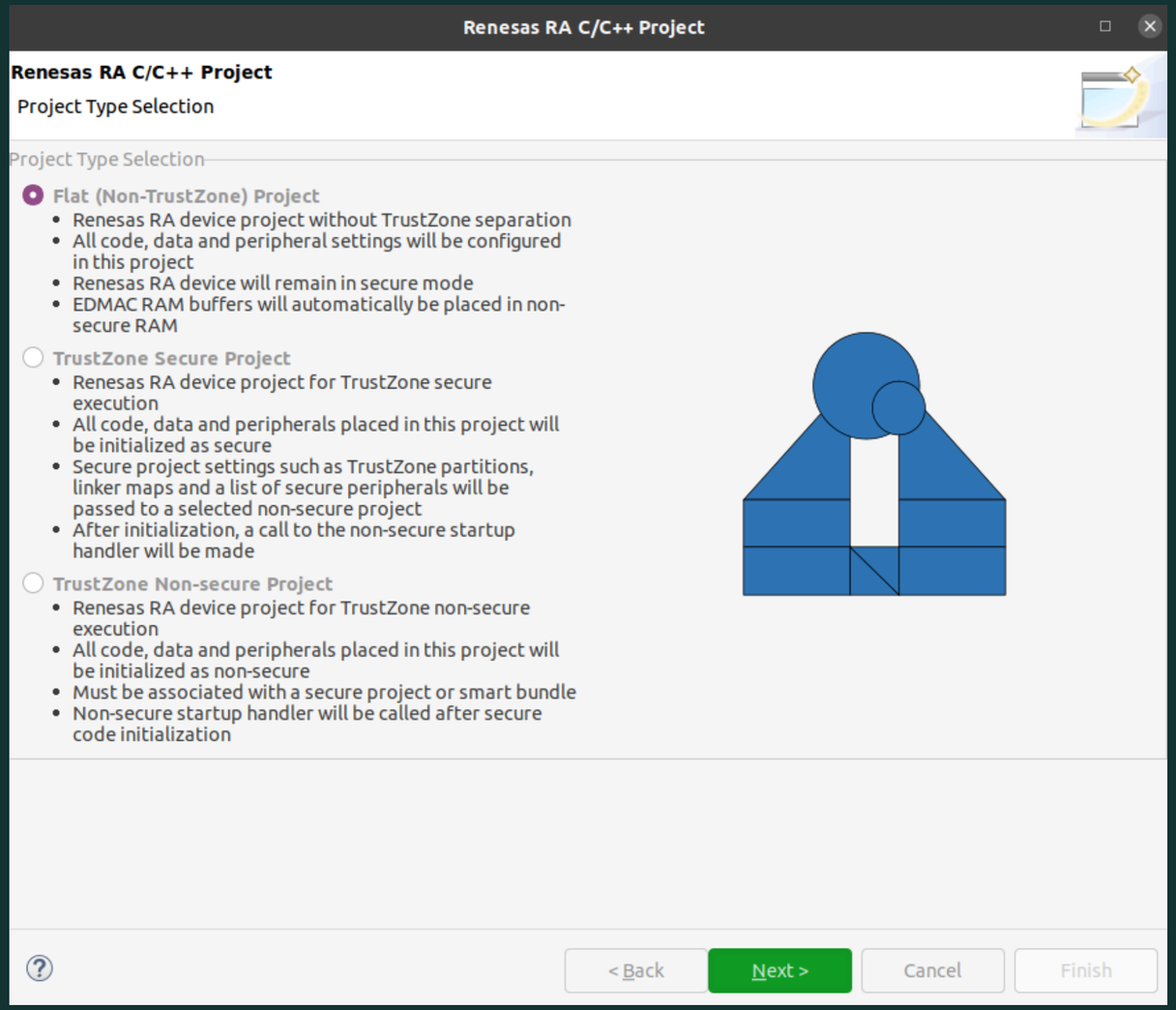
# Hướng dẫn cài đặt chương trình

- Bước 4: Chọn FSP version > Chọn Board > Chọn ngôn ngữ > Next



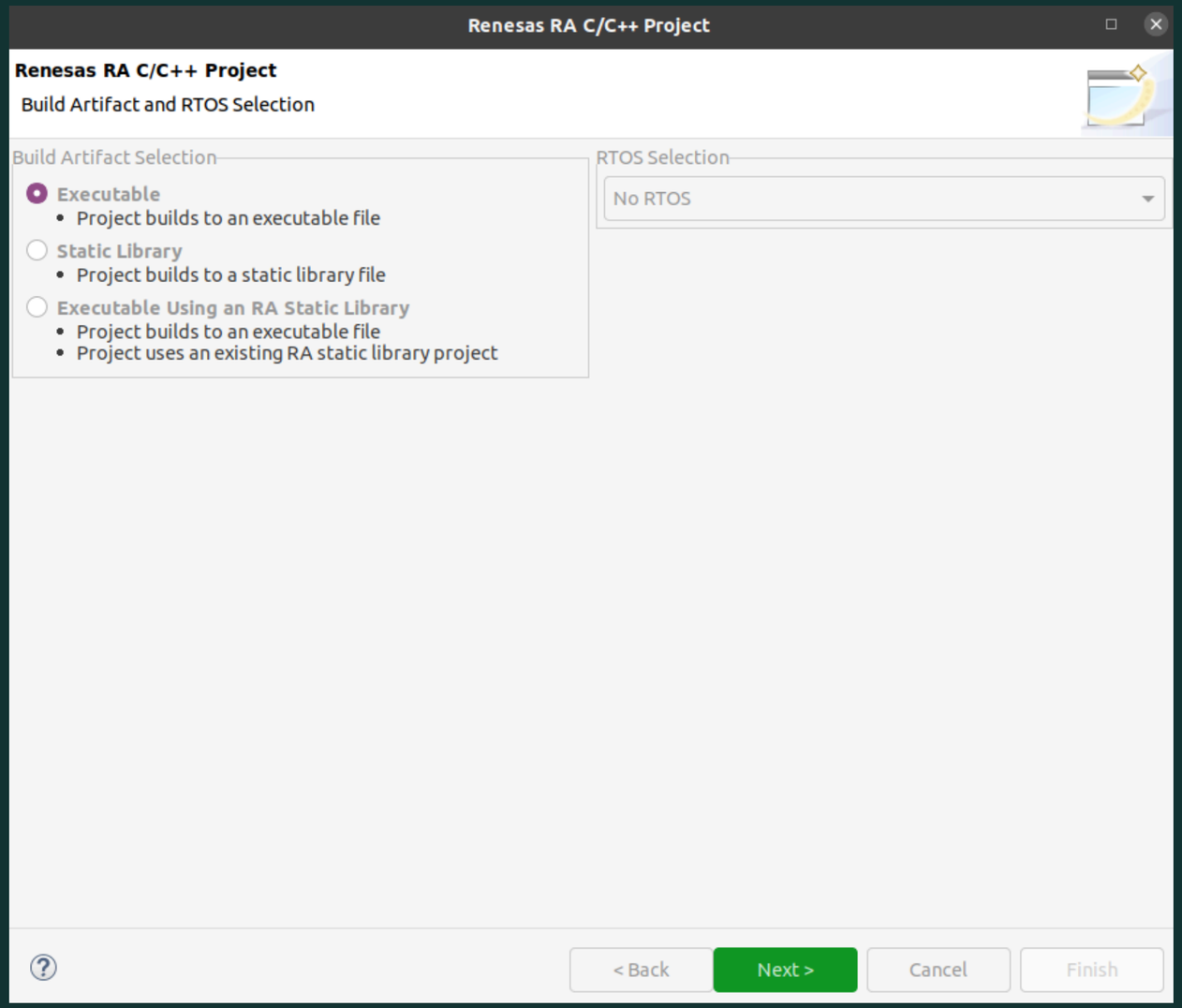
# Hướng dẫn cài đặt chương trình

## Bước 5: Flat (Non-TrustZone) Project => Chọn Next



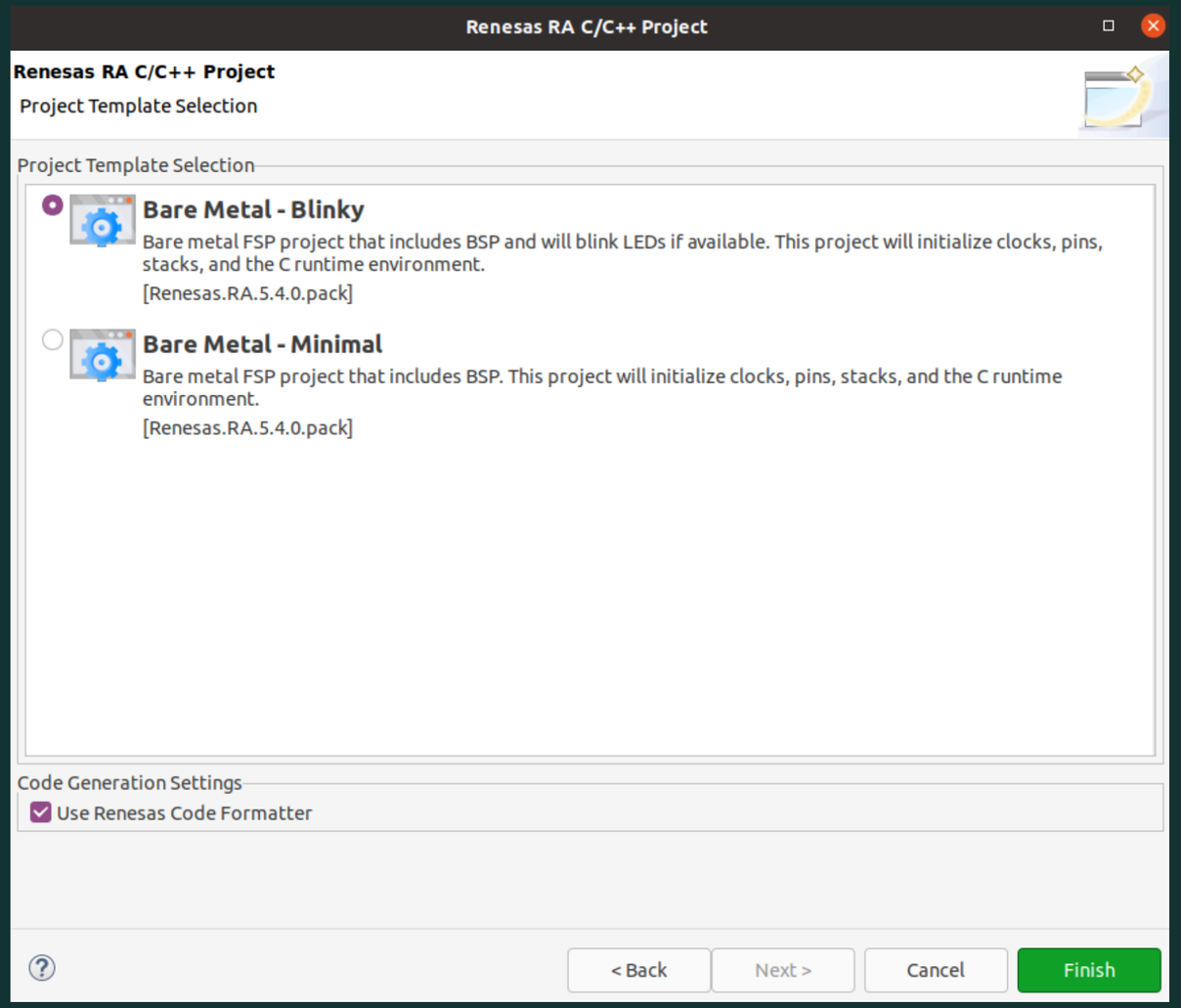
# Hướng dẫn cài đặt chương trình

## Bước 6: Chọn Executable => Next



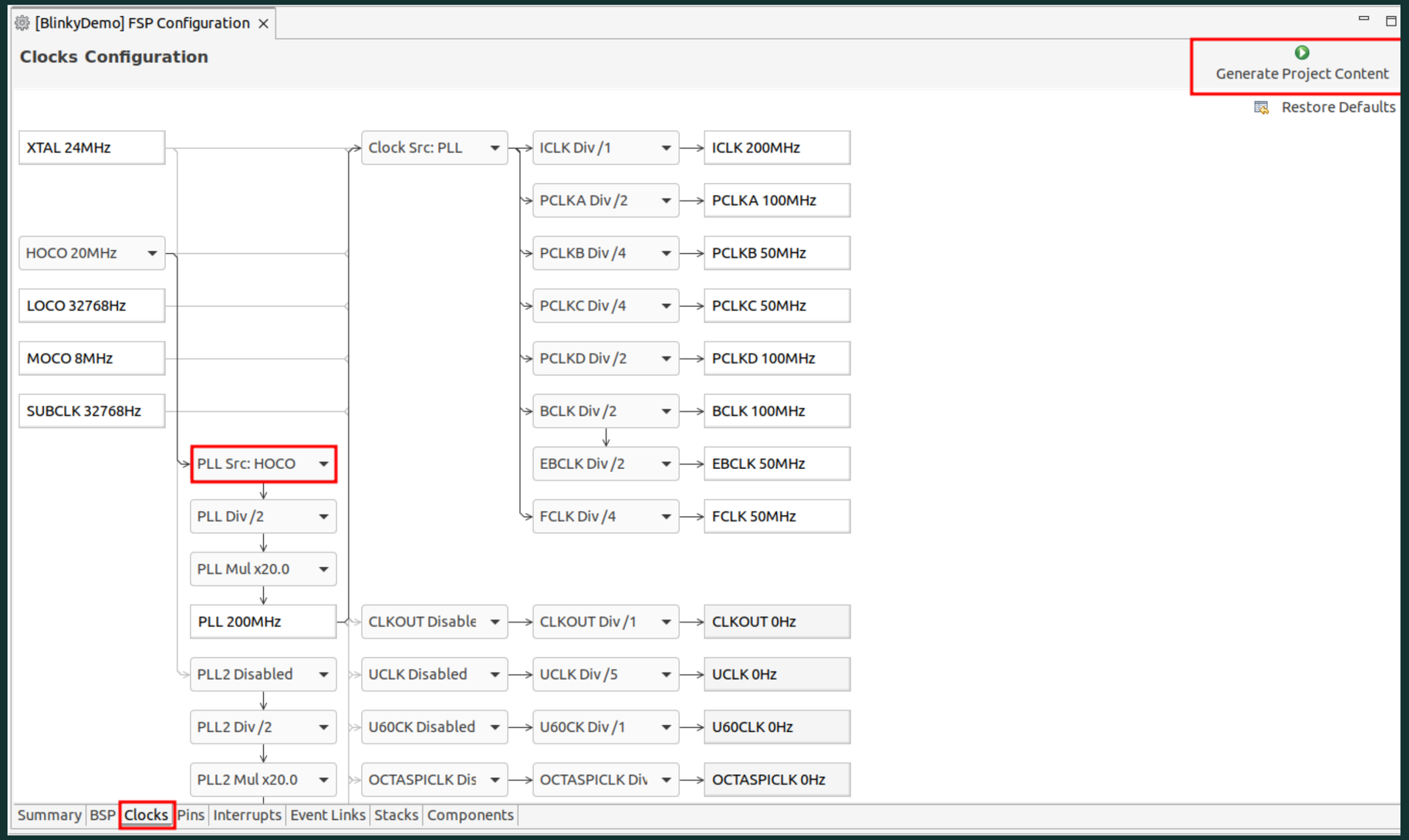
# Hướng dẫn cài đặt chương trình

Bước 7: Chọn  Bare Metal - Blinky => Finish



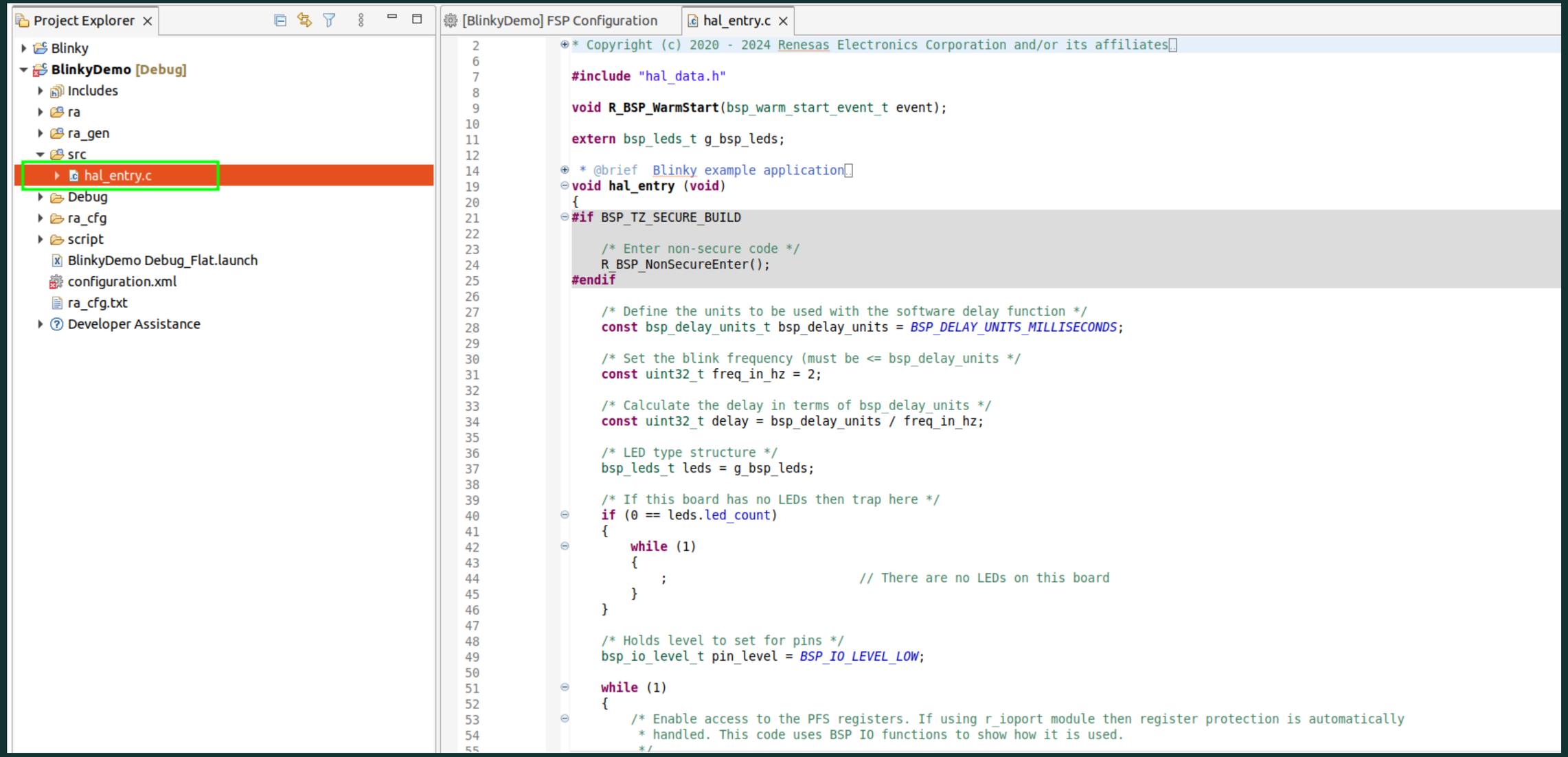
# Hướng dẫn cài đặt chương trình

## Bước 8: Chọn PPL Src: HOCO > Generate Project Content



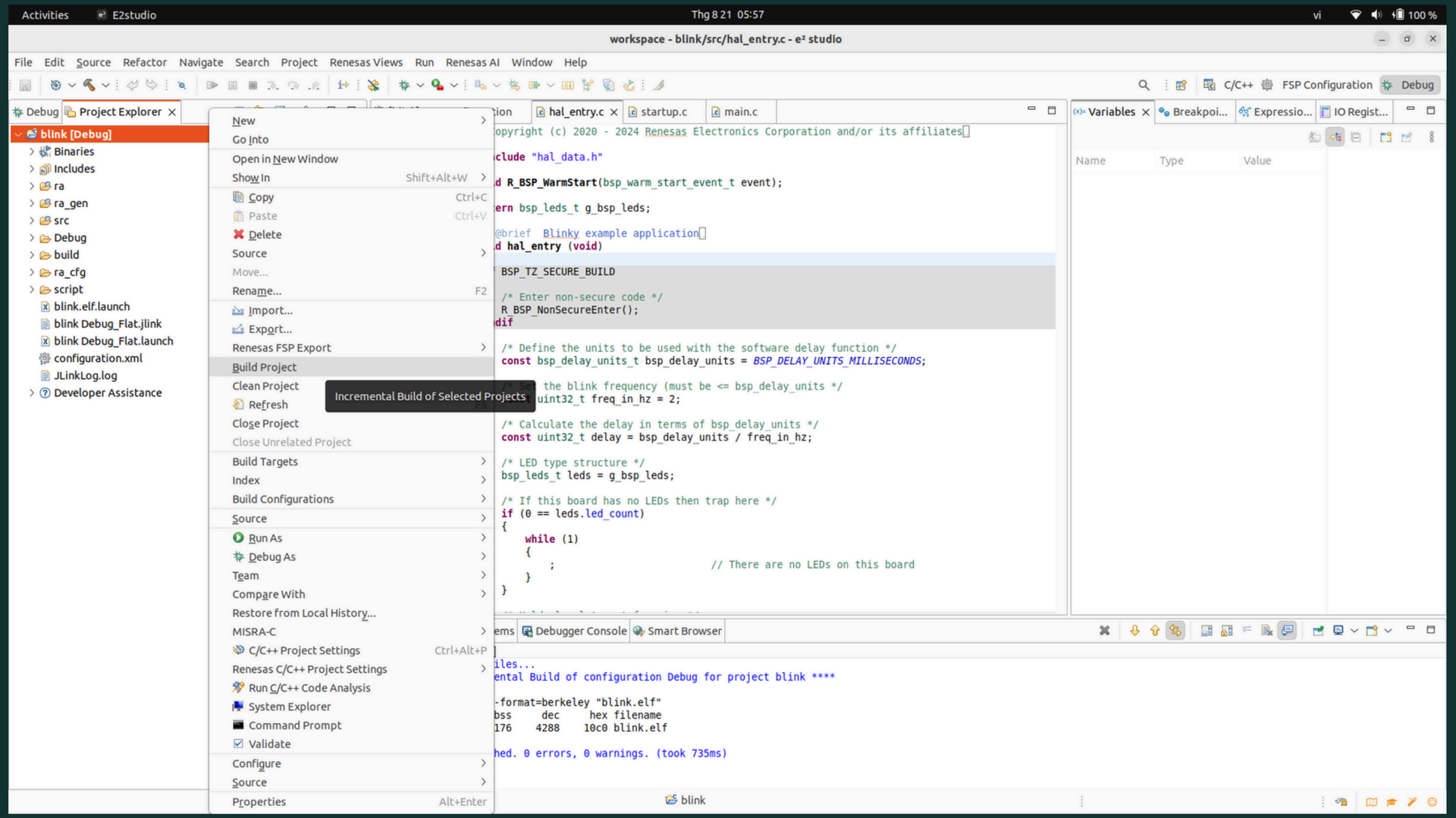
# Hướng dẫn cài đặt chương trình

## Bước 9: File code được gen trong hal_entry.c



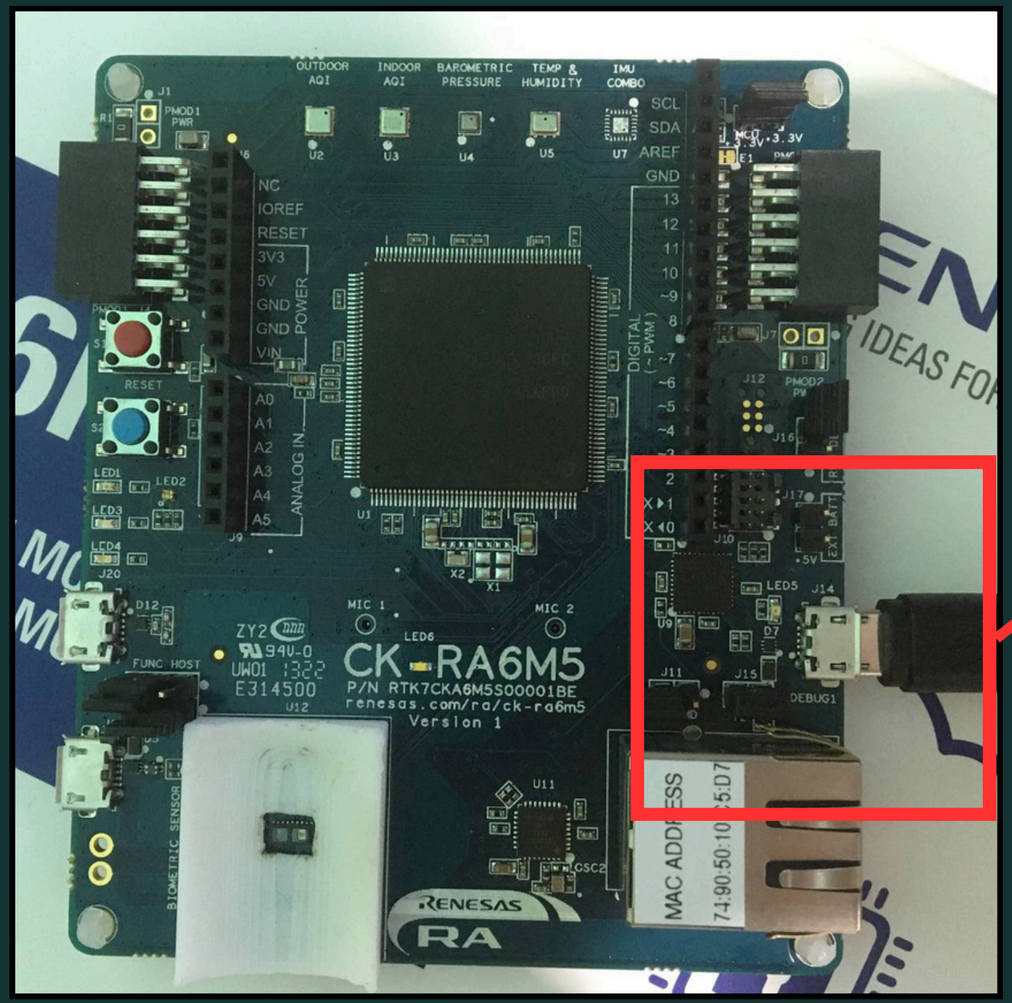
# Hướng dẫn cài đặt chương trình

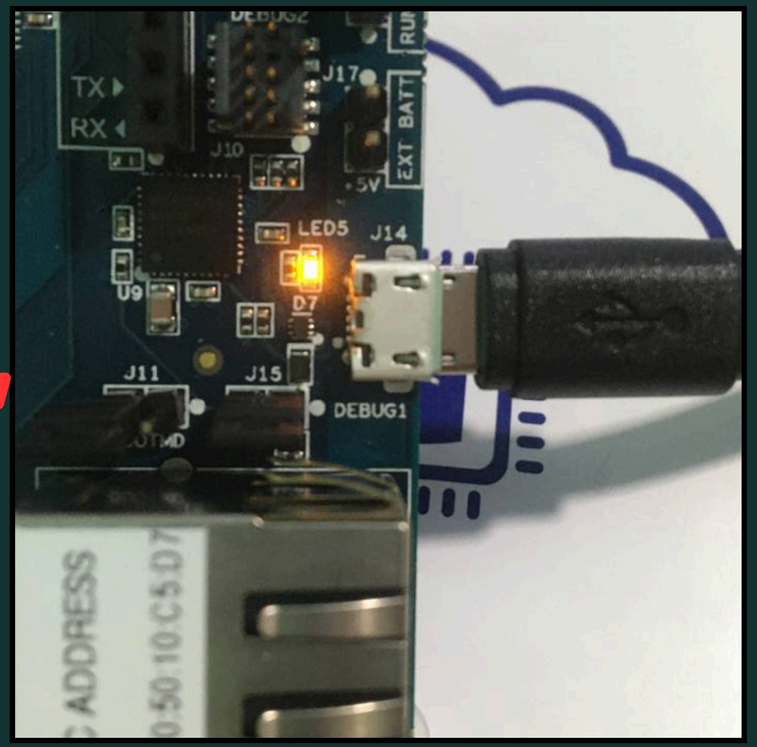
## Bước 10: Nhấn chuột phải vài project => Build Project



# Hướng dẫn cài đặt chương trình

Bước 11: Kết nối dây vào cổng Debug 1

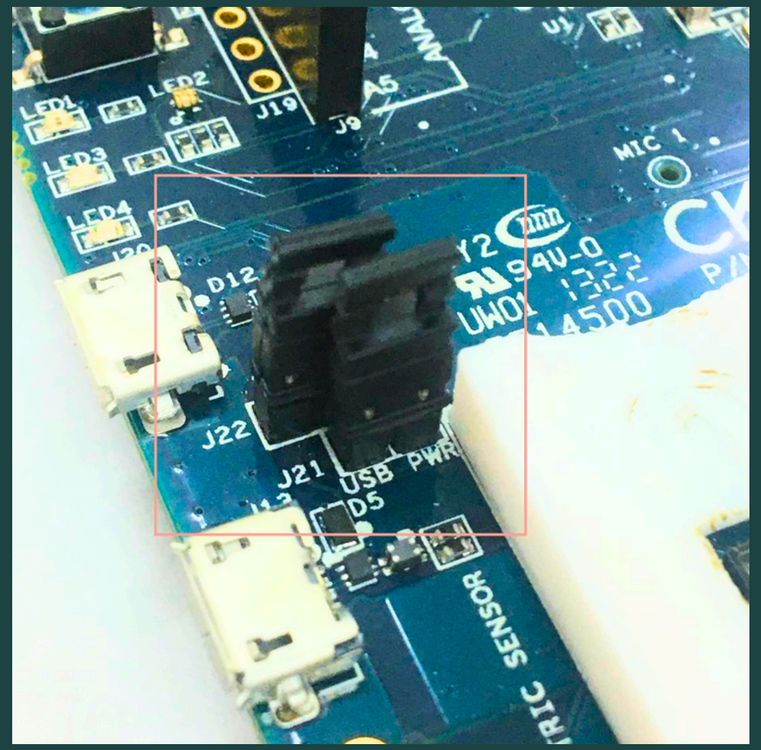


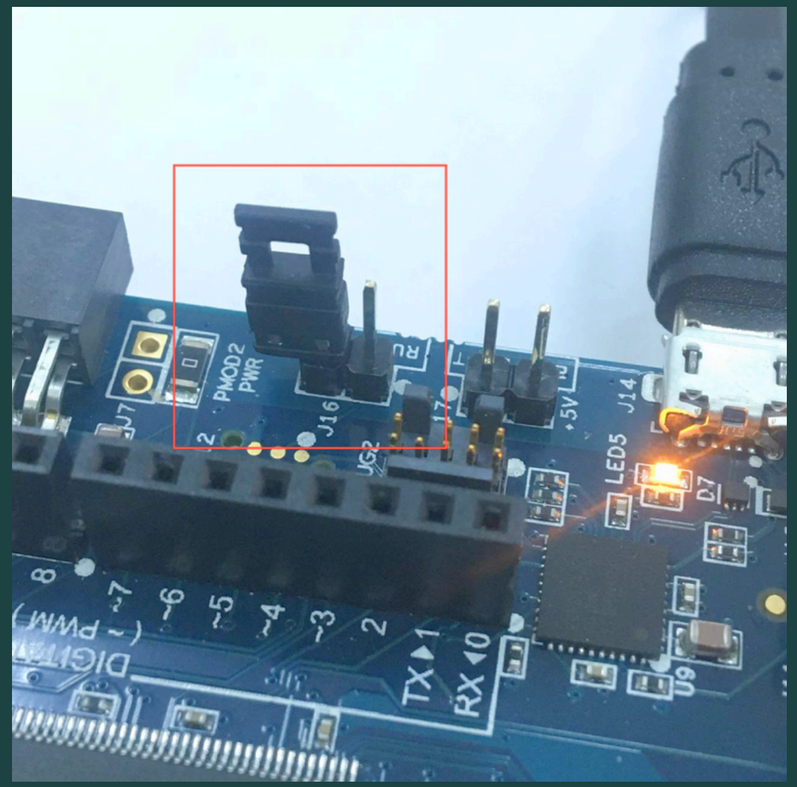



# Hướng dẫn cài đặt chương trình

Lưu ý:

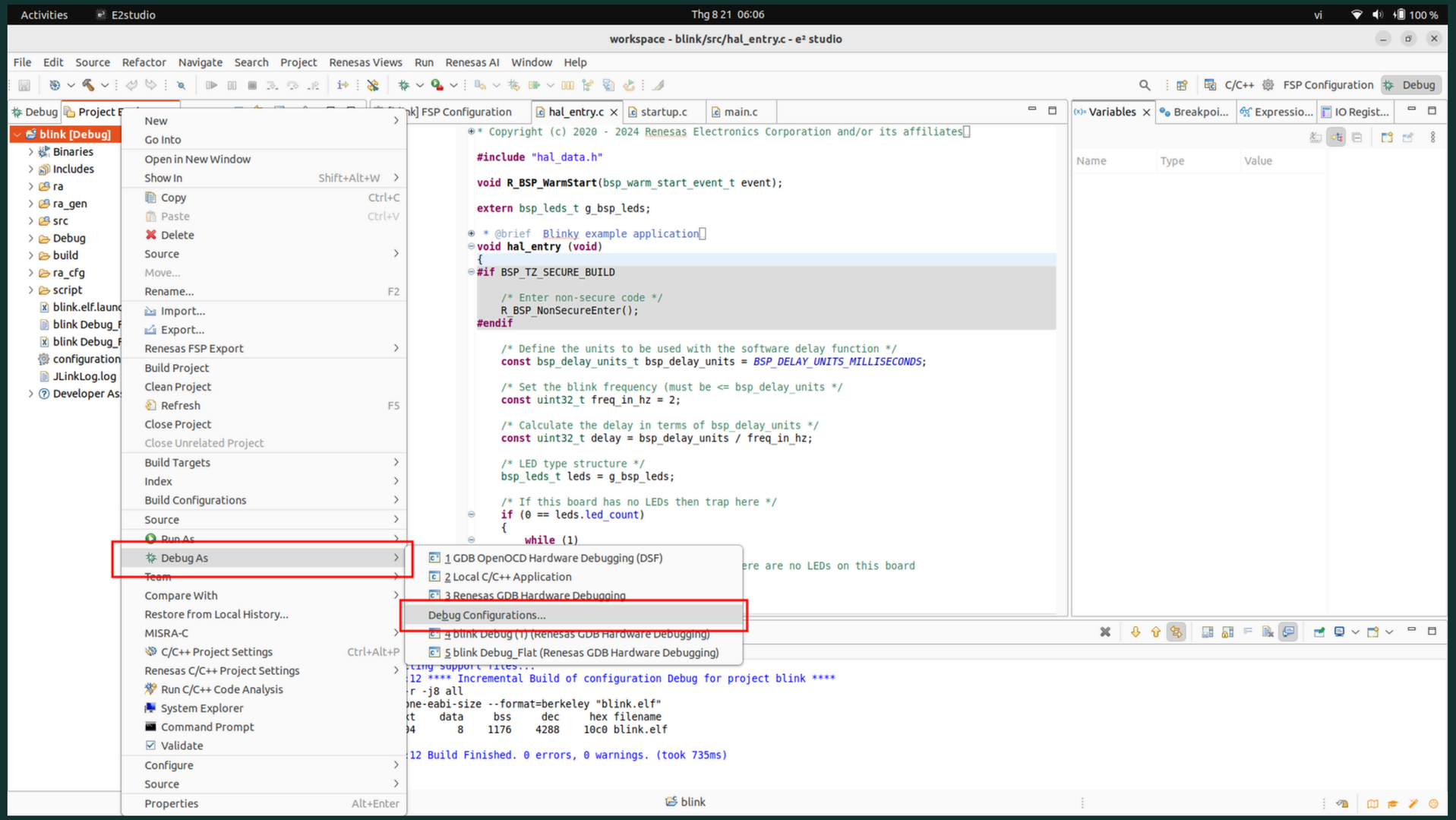
- J22 link pins 2-3
- J21 link đóng
- J16 link mở



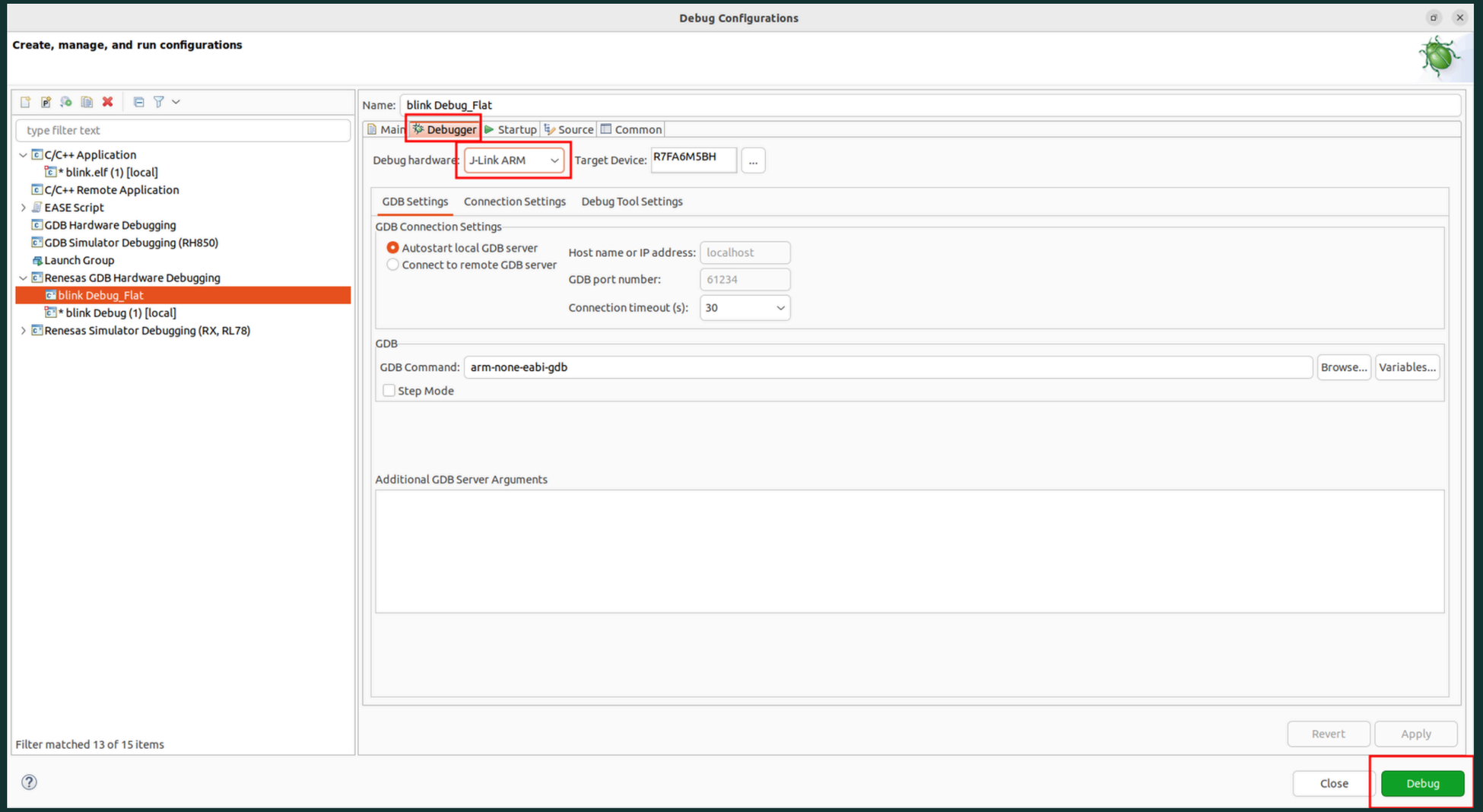
# Hướng dẫn cài đặt chương trình

## Bước 12: Nhấn chuột phải vào project > Debug As > Debug Configurations

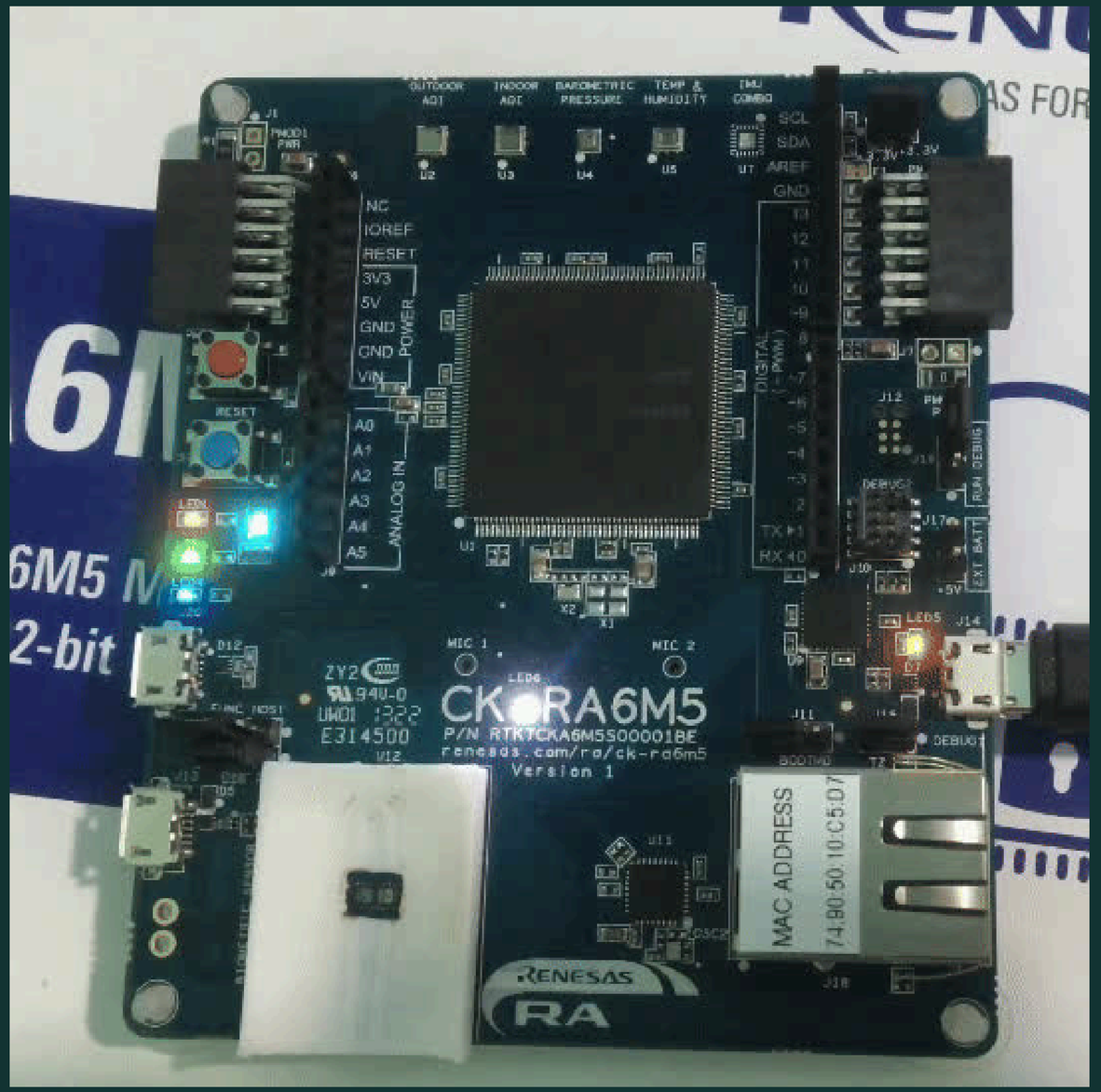


# Hướng dẫn cài đặt chương trình

## Bước 13: Chọn tab Debugger > Chọn J-Link ARM > Debug



# Kết quả


CK-RA6M5
P/N RTKTCKA6M5S00001BE
renesas.com/ra/ck-ra6m5
Version 1
MAC ADDRESS
74:90:50:10:C5:D7
RENESAS
RA

# MacOS

Nếu gặp lỗi: “could not determine gdb version using command: arm-none-eabi-gdb --version”
=> Kiểm tra Python đã được cài đặt trong /Library/Frameworks/Python.framework/3.10 chưa ?
=> Cài đặt python 3.10 theo link sau
*https://www.python.org/ftp/python/3.10.8/python-3.10.8-macos11.pkg*

# III. Giải thích code

# Giải thích code

```
#include "hal_data.h"
void R_BSP_WarmStart(bsp_warm_start_event_t event);
extern bsp_leds_t g_bsp_leds;
```

hardware abstraction layer (HAL)

called at various points during the startup process.

holds information about the LEDs

```
void hal_entry (void)
{
#if BSP_TZ_SECURE_BUILD

    /* Enter non-secure code */
    R_BSP_NonSecureEnter();
#endif

    /* Define the units to be used with the software delay function */
    const bsp_delay_units_t bsp_delay_units = BSP_DELAY_UNITS_MILLISECONDS;

    /* Set the blink frequency (must be <= bsp_delay_units */
    const uint32_t freq_in_hz = 2;

    /* Calculate the delay in terms of bsp_delay_units */
    const uint32_t delay = bsp_delay_units / freq_in_hz;

    /* LED type structure */
    bsp_leds_t leds = g_bsp_leds;

    /* If this board has no LEDs then trap here */
    if (0 == leds.led_count)
    {
        while (1)
        {
            ;                           // There are no LEDs on this board
        }
    }

    /* Holds level to set for pins */
    bsp_io_level_t pin_level = BSP_IO_LEVEL_LOW;
```

TrustZone security

Delay units

2 blinks per second

the delay would be 500 ms

load LED information into leds variable

# Giải thích code

```
/* Holds level to set for pins */
bsp_io_level_t pin_level = BSP_IO_LEVEL_LOW;

while (1)
{
    /* Enable access to the PFS registers. If using r_ioport module then register protection is automatically
     * handled. This code uses BSP IO functions to show how it is used.
     */
    R_BSP_PinAccessEnable();

    /* Update all board LEDs */
    for (uint32_t i = 0; i < leds.led_count; i++)
    {
        /* Get pin to toggle */
        uint32_t pin = leds.p_leds[i];

        /* Write to this pin */
        R_BSP_PinWrite((bsp_io_port_pin_t) pin, pin_level);
    }

    /* Protect PFS registers */
    R_BSP_PinAccessDisable();

    /* Toggle level for next write */
    if (BSP_IO_LEVEL_LOW == pin_level)
    {
        pin_level = BSP_IO_LEVEL_HIGH;
    }
    else
    {
        pin_level = BSP_IO_LEVEL_LOW;
    }

    /* Delay */
    R_BSP_SoftwareDelay(delay, bsp_delay_units);
}
```

init with low

enable pin access

set pin_level for each led

toggle pin level

delay for pin level

# Giải thích code

```
 * This function is called at various points during the startup process.  This implementation uses the event that is
void R_BSP_WarmStart (bsp_warm_start_event_t event)
{
    if (BSP_WARM_START_RESET == event)
    {
#if BSP_FEATURE_FLASH_LP_VERSION != 0

        /* Enable reading from data flash. */
        R_FACI_LP->DFLCTL = 1U;

        /* Would normally have to wait tDSTOP(6us) for data flash recovery. Placing the enable here, before clock and
         * C runtime initialization, should negate the need for a delay since the initialization will typically take more than 6
#endif
    }

    if (BSP_WARM_START_POST_C == event)
    {
        /* C runtime environment and system clocks are setup. */

        /* Configure pins. */
        R_IOPORT_Open(&IOPORT_CFG_CTRL, &IOPORT_CFG_NAME);
    }
}
```

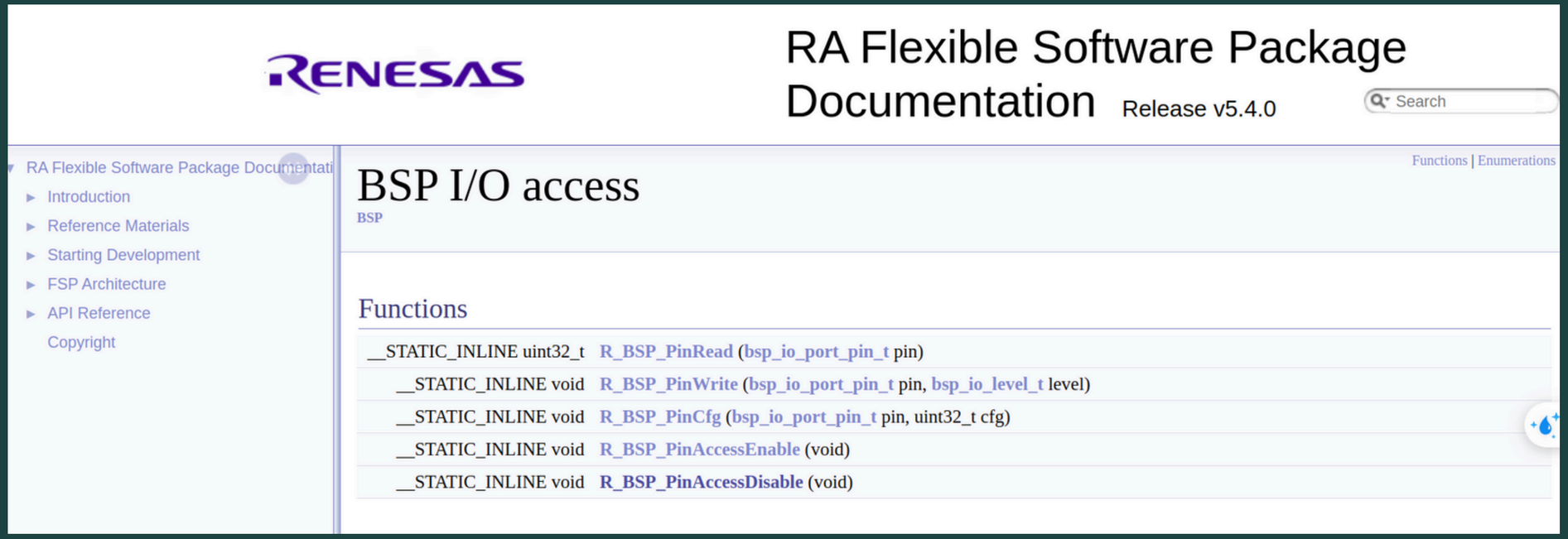
reset event

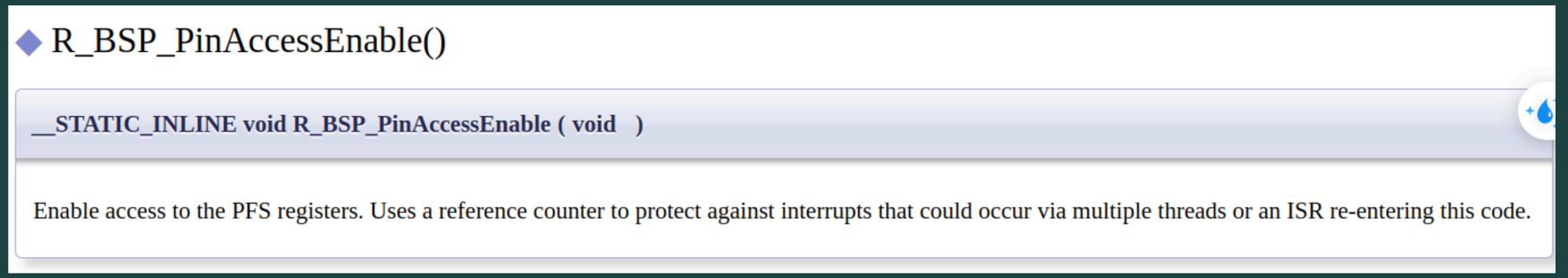
enable data flash

C runtime & clocks are setup

opening the I/O port

# API Docs




https://renesas.github.io/fsp/modules.html

# Bài tập

1. Nhấp nháy nhanh/ chậm hơn
2. Đèn sáng tuần tự

   Các đèn sáng tuần tự từ LED 1 đến LED 2, đến LED 3, LED 4

# DEMO