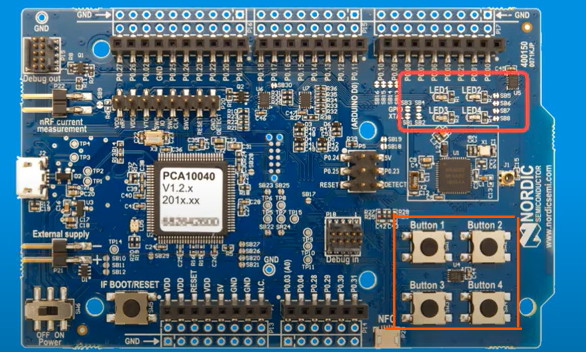
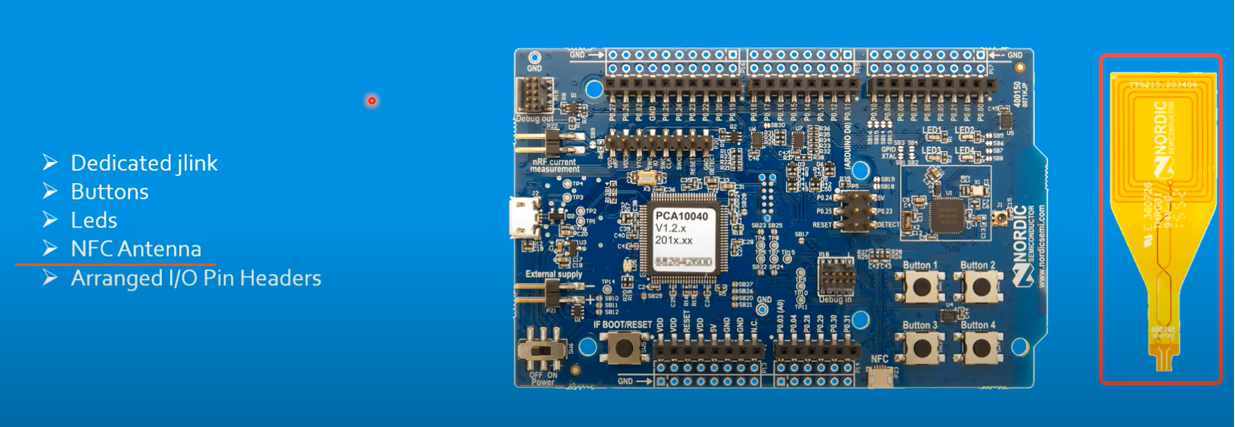
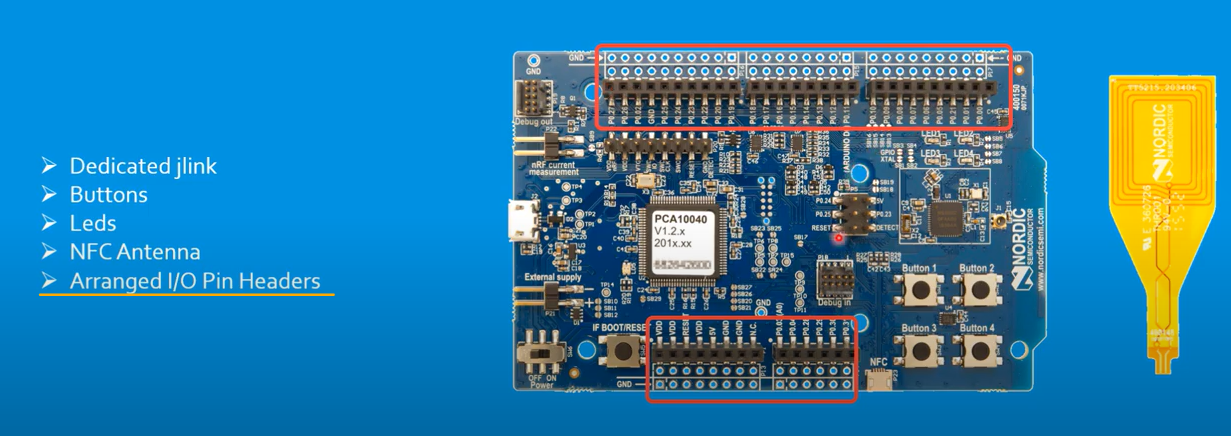


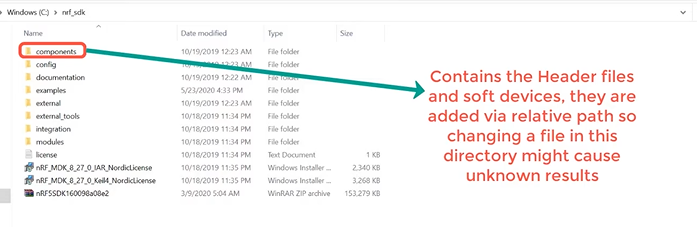
LED and buttons:



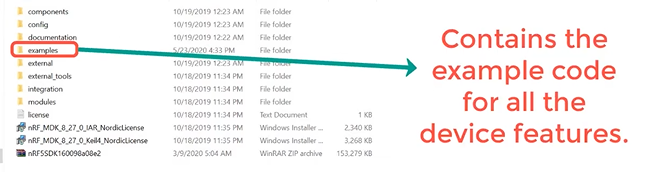


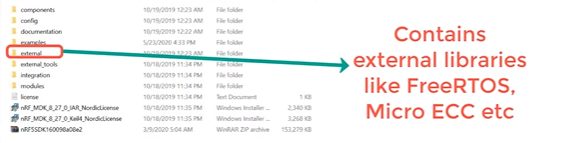


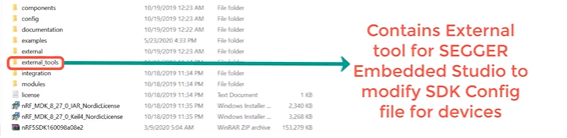
SDK 16.0 Directory Structure Explained

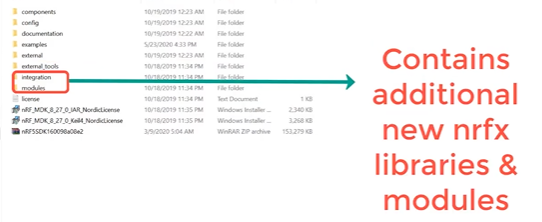




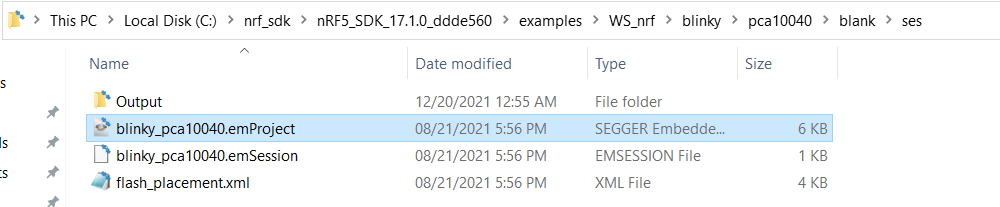






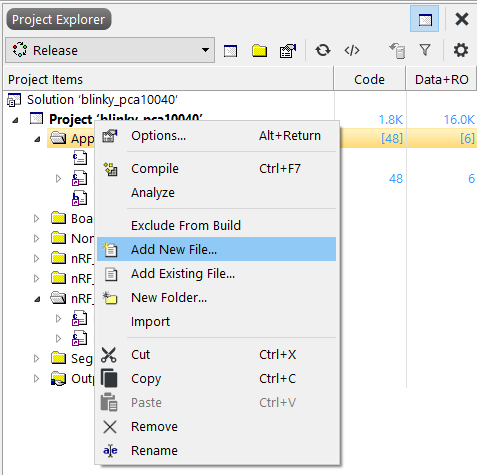


**Creating New Project:**

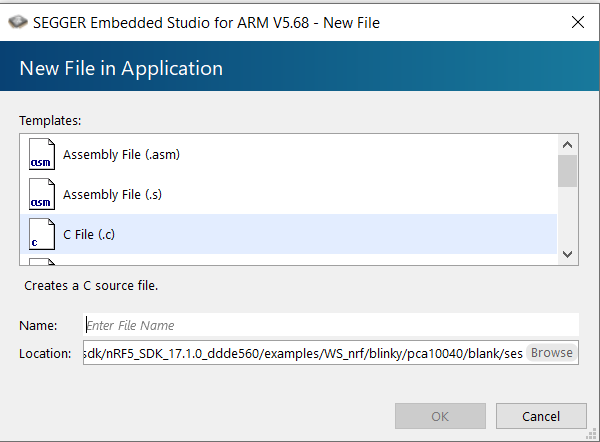
* Tạo 1 folder mới của mk ở trong file examples
* Sau đó, copy nguyên folder cần dùng ở trong folder “examples” sang folder của mk vừa tạo (Lưu ý: sao cho vị trí thứ tự của 2 folder là tương đương nhau)
* Tìm đến file: 
* Định dạng file như trên để mở được trong phần mềm Segger Embedded Studio.
* Clean build -> Build
* Build and Run

# Segger Embedded Studio Explained in Easy Steps

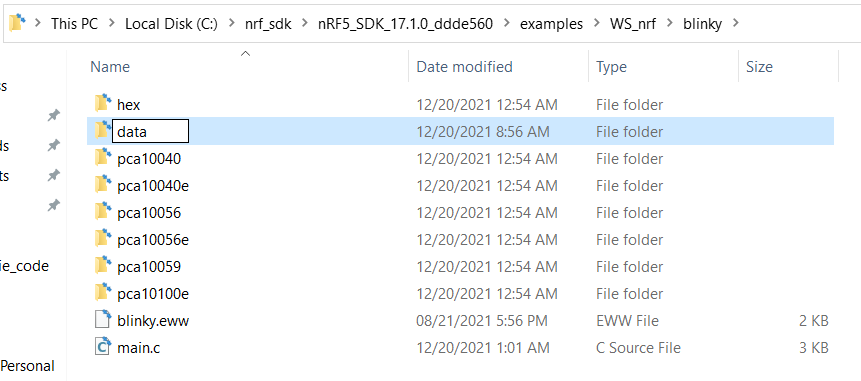
* Add new file:



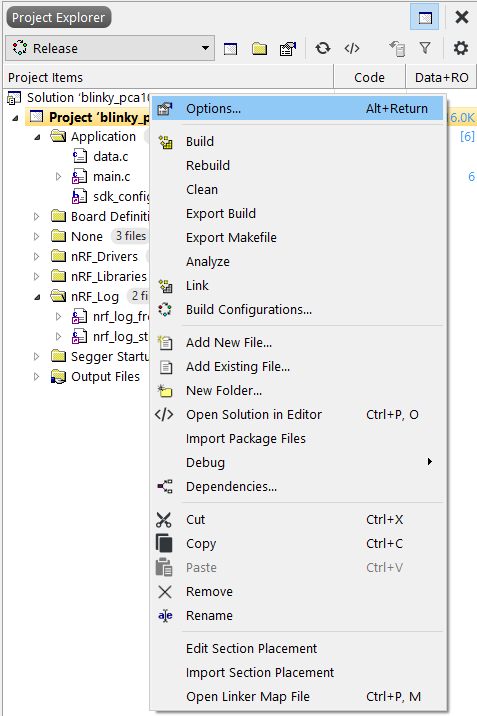
* Chọn đường dẫn -> đặt tên file -> lưu file

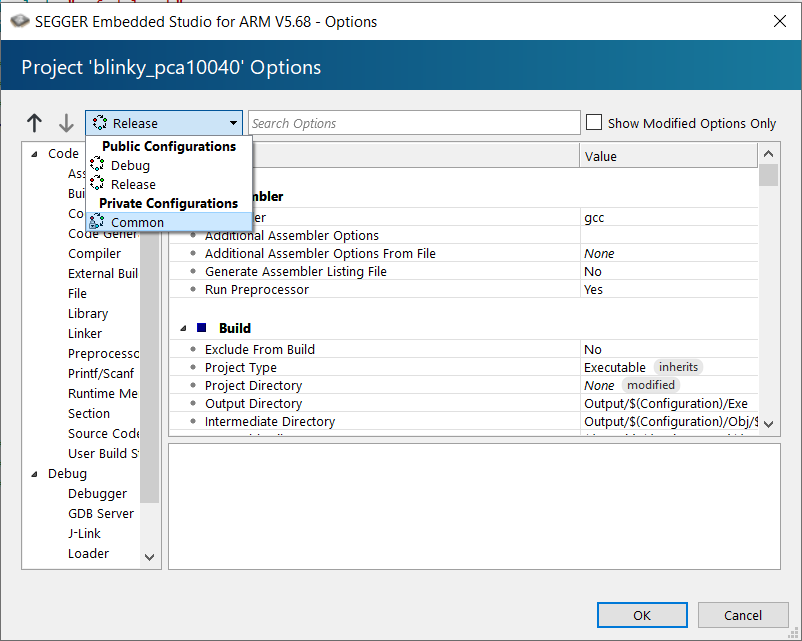


* Tạo 1 folder mới: -> lấy đường dẫn đến folder vừa tạo

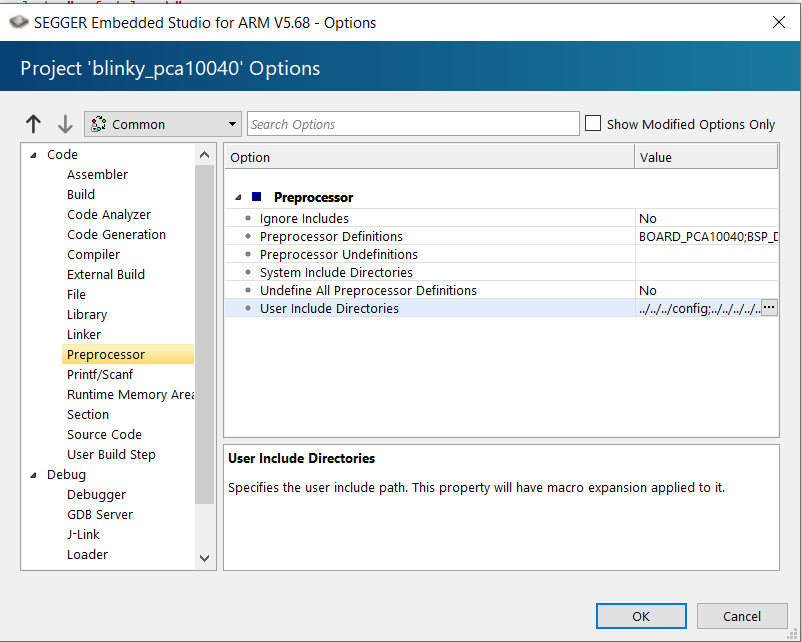


* Chỉnh sửa file path đó từ folder:





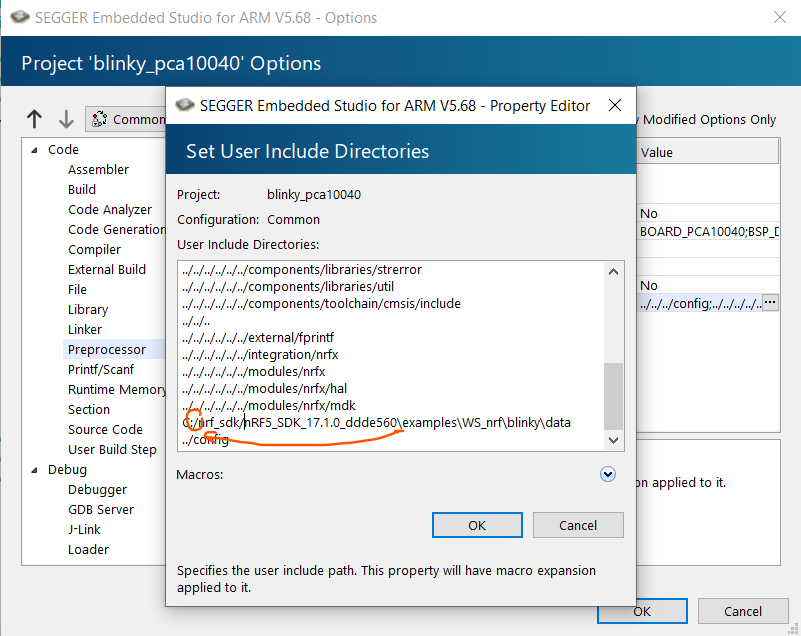
Chọn preprocessor



Chỉnh sửa trong phần users include directories:

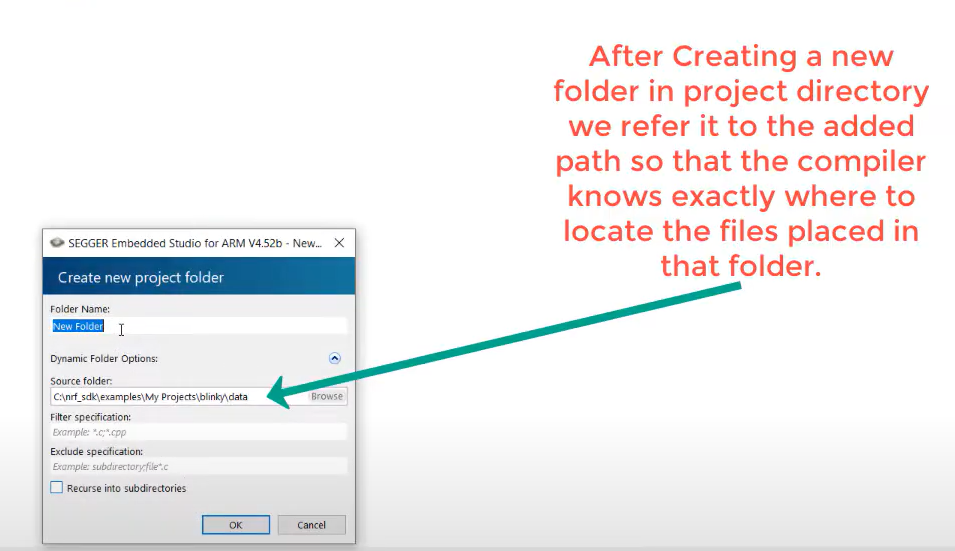
Chọn User include directories:

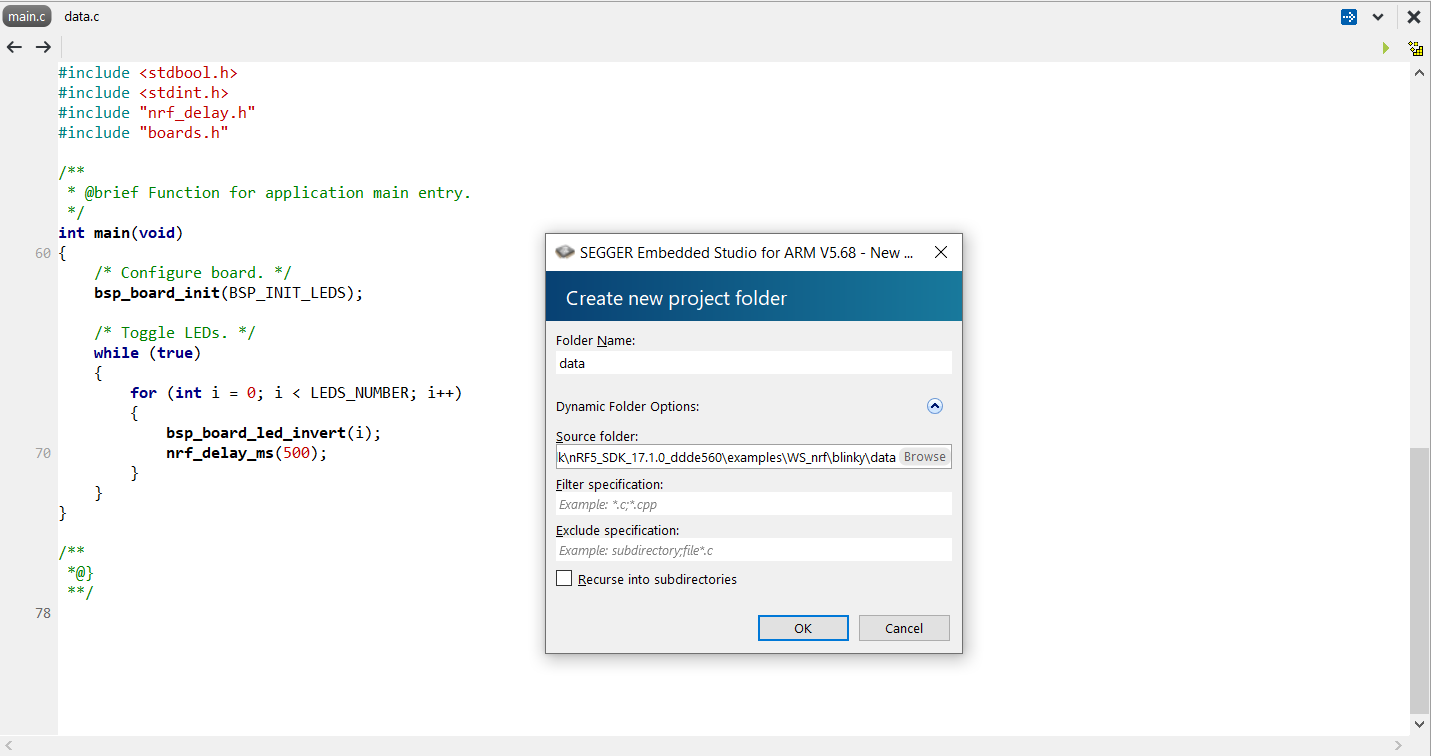
Tạo khoảng trắng, add đường dẫn vào, sau đó thì sửa dấu gach chéo:



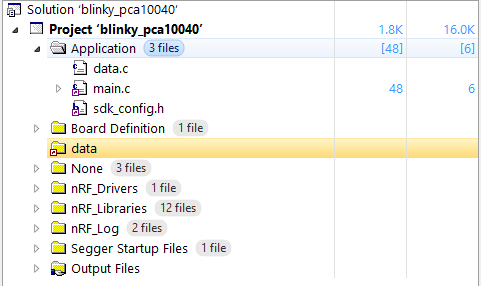
Click OK, OK:

* Add đường dẫn path này vào project
* Tạo 1 new folder mới trong folder vừa tạo:

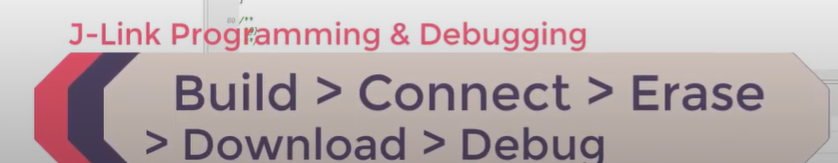


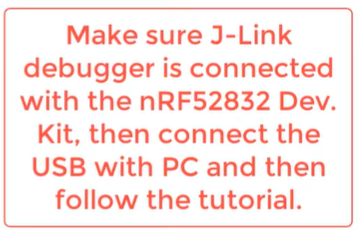


Kết quả sau khi add folder mới :

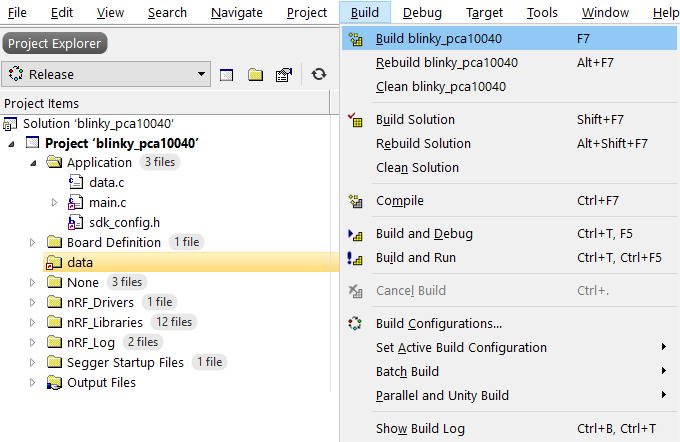


J-Link



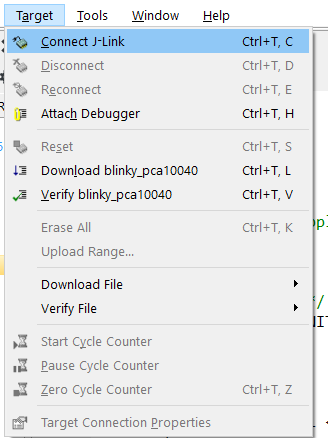


Sau khi hoàn thành code-> build project

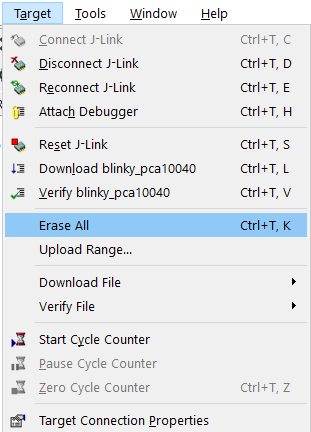


Sau khi build xong, COMPILED nó

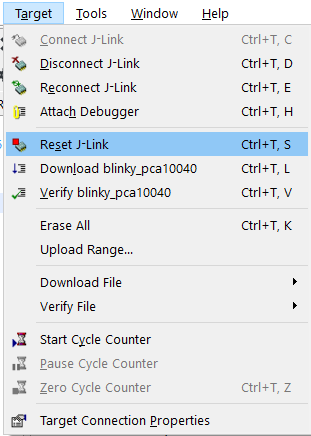
Để upload code to device: -> connect J-link (đảm bảo rằng: J-link được kết nối với cả RF device) cổng USB



Sau đó Erase All -> xoá sạch chương trình cũ trong mạch

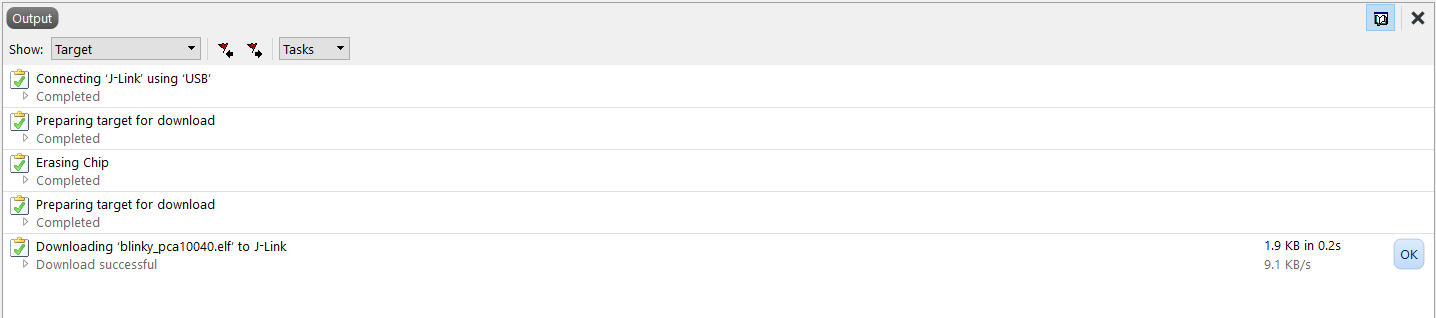


Sau đó, download file code vừa compiled đó xuống thiết bị.

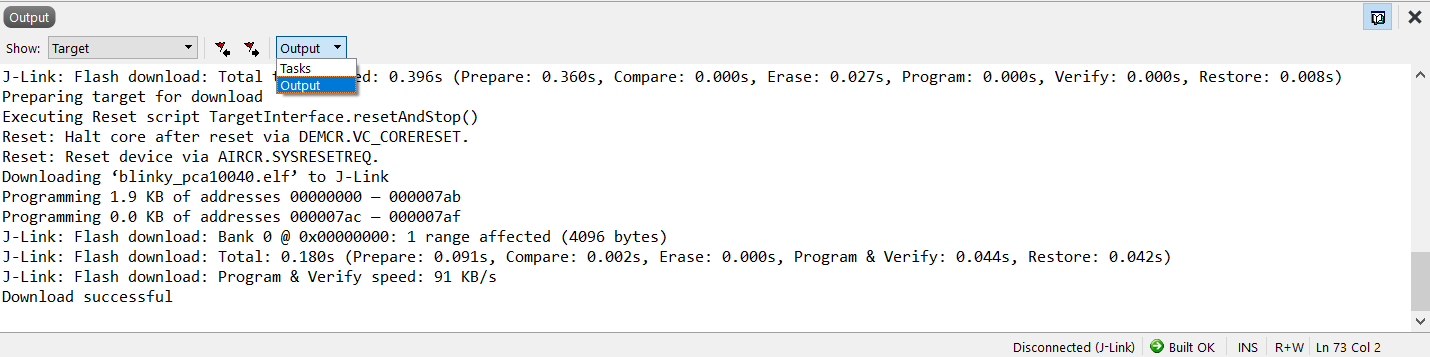


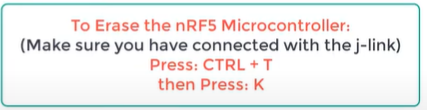
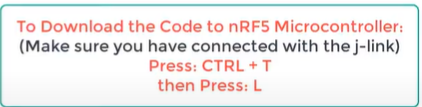
Kết quả sau khi download chương trình xuống:

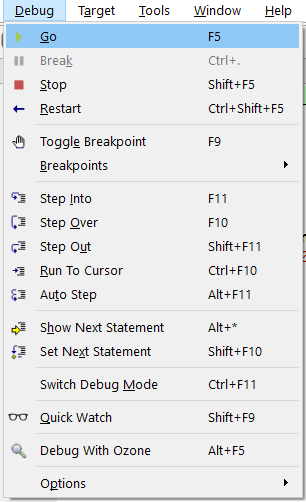
* Task:



* Output:

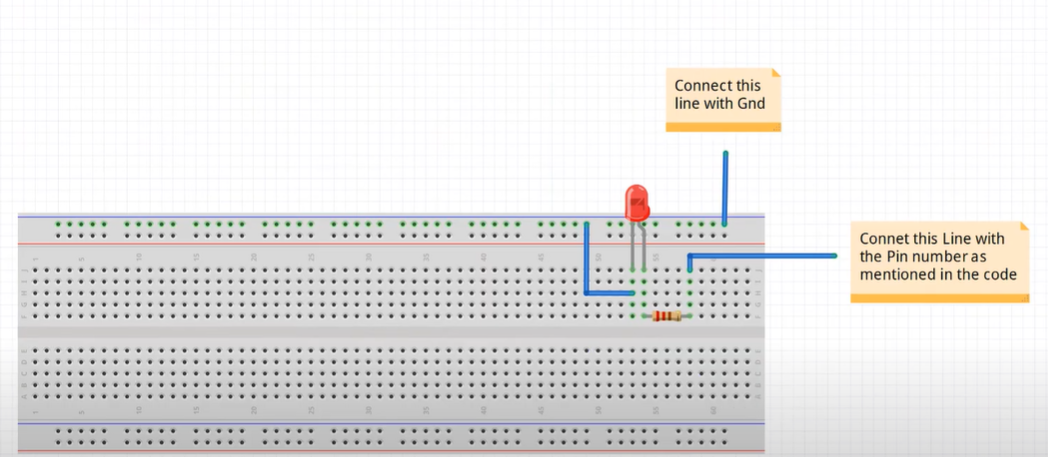


* Các phím tắt:
  + Compiled: F7 
  + Connect to J-link: 
  + Erase all: 
  + Download to device: 
* Bật chế độ Debug:



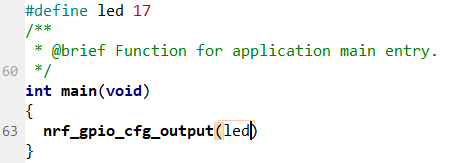
# 5 - Basic OUTPUT (Blinking an LED)

* Circuit:



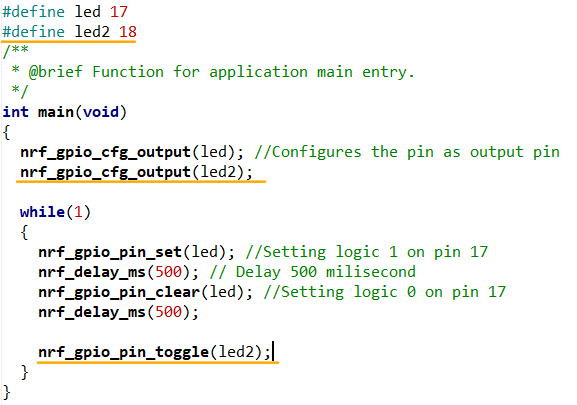
* Import thư viện:



* Khai báo chân config đấu nối với led ( ở đây là khai báo chân số 17) 
* Code blink LED:

|  |
| --- |
| int main(void)  {  nrf\_gpio\_cfg\_output(led); //Configures the pin as output pin  while(1)  {  nrf\_gpio\_pin\_set(led); //Setting logic 1 on pin 17  nrf\_delay\_ms(500); // Delay 500 milisecond  nrf\_gpio\_pin\_clear(led); //Setting logic 0 on pin 17  nrf\_delay\_ms(500);  }  } |

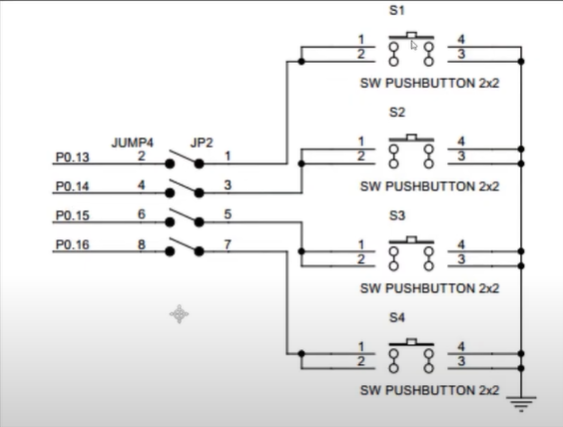
* Code blink 2 led liên tục đan xen nhau:



* + Hàm toggle: tín hiệu ngược với chân led1, không cần phải hàm delay

|  |
| --- |
| #include <stdbool.h>  #include <stdint.h>  #include "nrf\_delay.h"  #include "nrf\_gpio.h"  #define led 17  #define led2 18  /\*\*  \* @brief Function for application main entry.  \*/  int main(void)  {  nrf\_gpio\_cfg\_output(led); //Configures the pin as output pin  nrf\_gpio\_cfg\_output(led2);  while(1)  {  nrf\_gpio\_pin\_set(led); //Setting logic 1 on pin 17  nrf\_delay\_ms(500); // Delay 500 milisecond  nrf\_gpio\_pin\_clear(led); //Setting logic 0 on pin 17  nrf\_delay\_ms(500);  nrf\_gpio\_pin\_toggle(led2);  }  } |

# 6 - Reading Digital Input from Buttons

* Configuring the Pin of Buttons: 
* Code đơn:

|  |
| --- |
| #include <stdbool.h>  #include <stdint.h>  #include "nrf\_delay.h"  #include "nrf\_gpio.h"  #define LED 17  #define Button 13  /\*\*  \* @brief Function for application main entry.  \*/  int main(void)  {  nrf\_gpio\_cfg\_output(LED); // Configures the Led pin as Output pin    nrf\_gpio\_cfg\_input(Button, NRF\_GPIO\_PIN\_PULLUP); // Configure the Button pin as Input pin  nrf\_gpio\_pin\_set(LED); //Turns off the LED  /\* Toggle LEDs. \*/  while (true)  {  if(nrf\_gpio\_pin\_read(Button) == 0)  {  nrf\_gpio\_pin\_clear(LED); //Turn on the LED  while(nrf\_gpio\_pin\_read(Button) == 0); //Stay in this loop until the button is released  nrf\_gpio\_pin\_set(LED); //Turns off the LED  }  }  } |

* Code: nháy 2 led bằng nhấn nút, khi nhấn giữ thì đèn vẫn sáng, code tự viết, khi chưa có vòng lặp hay mảng, đèn sáng tuần tự vì trong cùng 1 vòng loop

|  |
| --- |
| #include <stdbool.h>  #include <stdint.h>  #include "nrf\_delay.h"  #include "nrf\_gpio.h"  #define LED 17  #define LED1 18  #define Button 13  #define ButtonE 16  /\*\*  \* @brief Function for application main entry.  \*/  int main(void)  {  nrf\_gpio\_cfg\_output(LED); // Configures the Led pin as Output pin  nrf\_gpio\_cfg\_output(LED1);  nrf\_gpio\_range\_cfg\_input(13, 16, NRF\_GPIO\_PIN\_PULLUP); // Configure the Button pin as Input pin  nrf\_gpio\_pin\_set(LED); //Turns off the LED  nrf\_gpio\_pin\_set(LED1);  /\* Toggle LEDs. \*/  while (true)  {    if(nrf\_gpio\_pin\_read(13) == 0)  {  nrf\_gpio\_pin\_clear(LED); //Turn on the LED  while(nrf\_gpio\_pin\_read(13) == 0); //Stay in this loop until the button is released  nrf\_gpio\_pin\_set(LED); //Turns off the LED  }  if(nrf\_gpio\_pin\_read(14) == 0)  {  nrf\_gpio\_pin\_clear(LED1); //Turn on the LED  while(nrf\_gpio\_pin\_read(14) == 0); //Stay in this loop until the button is released  nrf\_gpio\_pin\_set(LED1); //Turns off the LED  }  }  }  /\*\*  \*@}  \*\*/ |

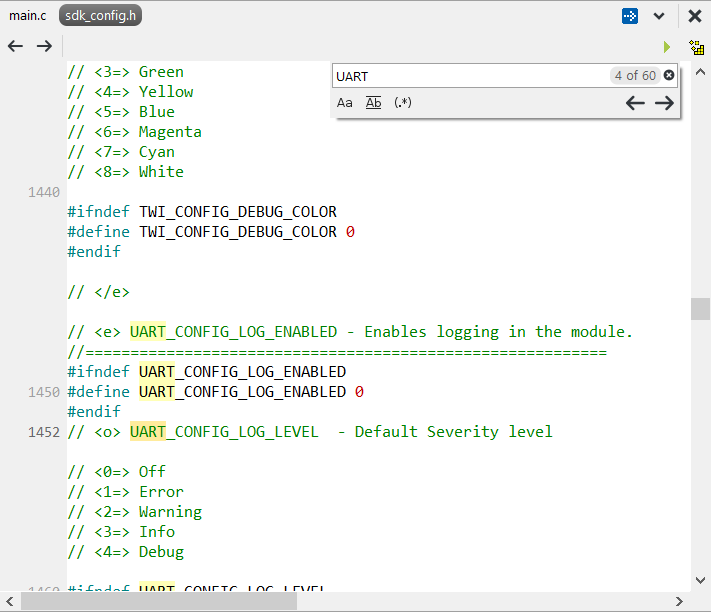
# 7 - Configure sdk\_config.h with CMSIS Configuration wizard

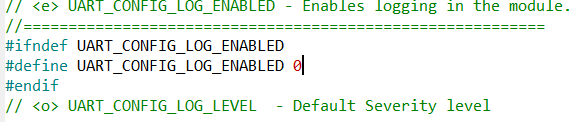
This is the part of by Tutorials for beginners series and in this tutorial i will be explaining the two methods by which we can modify the sdk\_config.h with CMSIS Configuration wizard and also we will see a direct method as well to directly modify the sdk\_config.h file in nrf5 SDK, this is the most important part because if some functions are not activated then they will not work. NRF5 SDK is designed to work with sdk\_config.h file so its really important to learn on how to configure it.

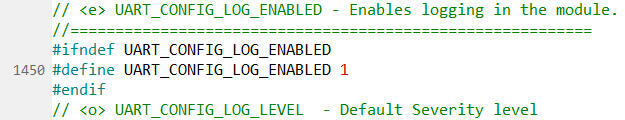


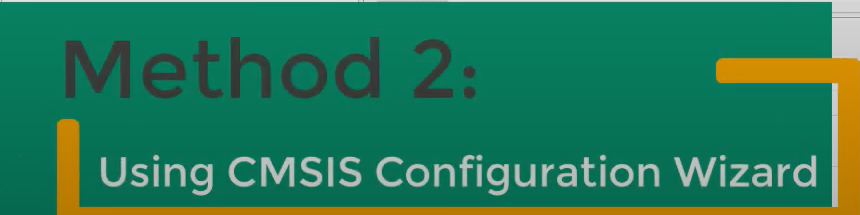
* Mở file sdk\_config.h => nhấn ctrl+F



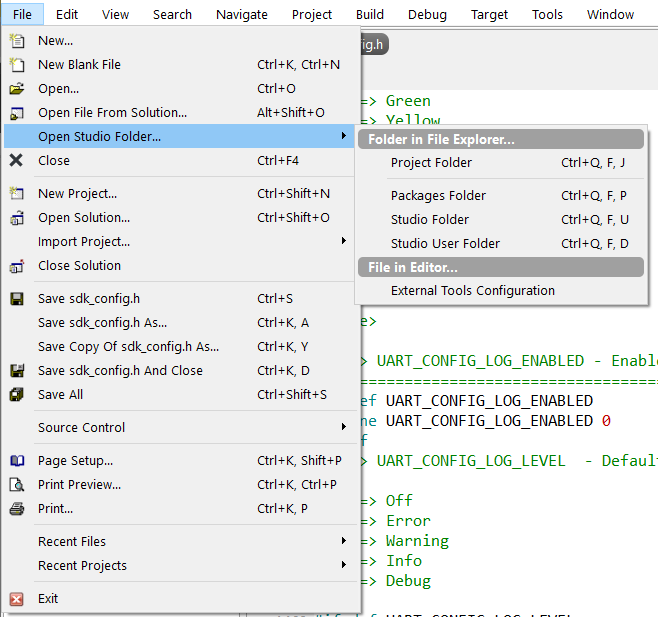
* Ví dụ muốn tìm: 
* Giá trị bằng 0 thì là disable, bằng 1 là **Enable**



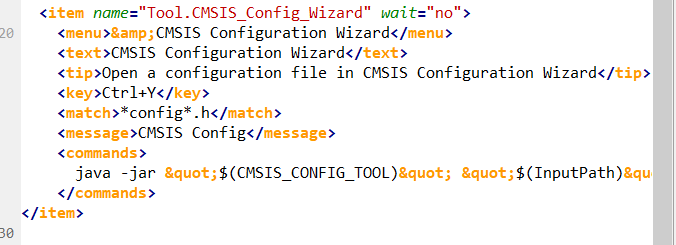
* => 
* Sau khi sửa thì save lại file đó.



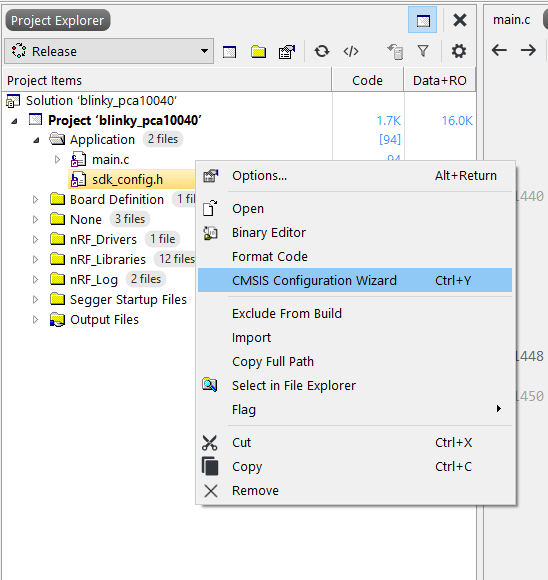
* Cách 2: sử dụng external tool



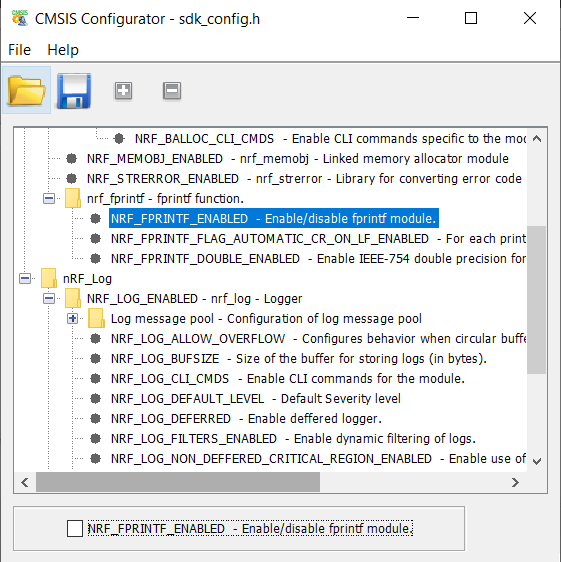
* Mở file sdk\_config.h lên
* Copy file cmsistool.xml vào



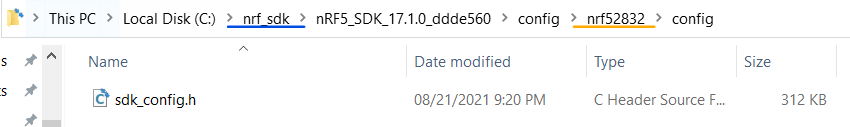
* Sau đó tắt phần mềm và vào lại



* Bỏ chọn để disable cái lựa chọn đó -> Save



* NOTE: file sdk configuration không phải là file chứa tất cả các file configurations.
  + Cách để add thêm configurations vào file sdk\_config:
    - Vào đường dẫn như hình rồi mở file sdk\_config -> tại đây chứa tất cả các phần all configurations:



* + - Mở file đó lên, copy những **configurations** còn thiếu ở trong đó ra file sdk\_config.h của folder cũ mà bị miss ( lưu ý chỉ nên copy không nên modify để giảm trường hợp bị lỗi về sau)

# 8 - Debugging over j-link using Segger RTT Library

how to output the data to debugger console without using any UART port, by directly using the Segger's proprietary library to get the output data over j-link and without using any external resources. Its really handy when we have our UART Port busy in doing some communication, and also we don't have to configure any external terminal to monitor the output on UART Port.

(cách xuất dữ liệu ra bảng điều khiển gỡ lỗi mà không cần sử dụng bất kỳ cổng UART nào, bằng cách sử dụng trực tiếp thư viện độc quyền của Segger để lấy dữ liệu đầu ra qua j-link và không sử dụng bất kỳ tài nguyên bên ngoài nào. Nó thực sự tiện dụng khi chúng ta có Cổng UART bận thực hiện một số giao tiếp và chúng tôi cũng không phải định cấu hình bất kỳ thiết bị đầu cuối bên ngoài nào để giám sát đầu ra trên Cổng UART.)