

STM32CubeIDE: How to debug STM32H7Rx/Sx project without flashing boot every time

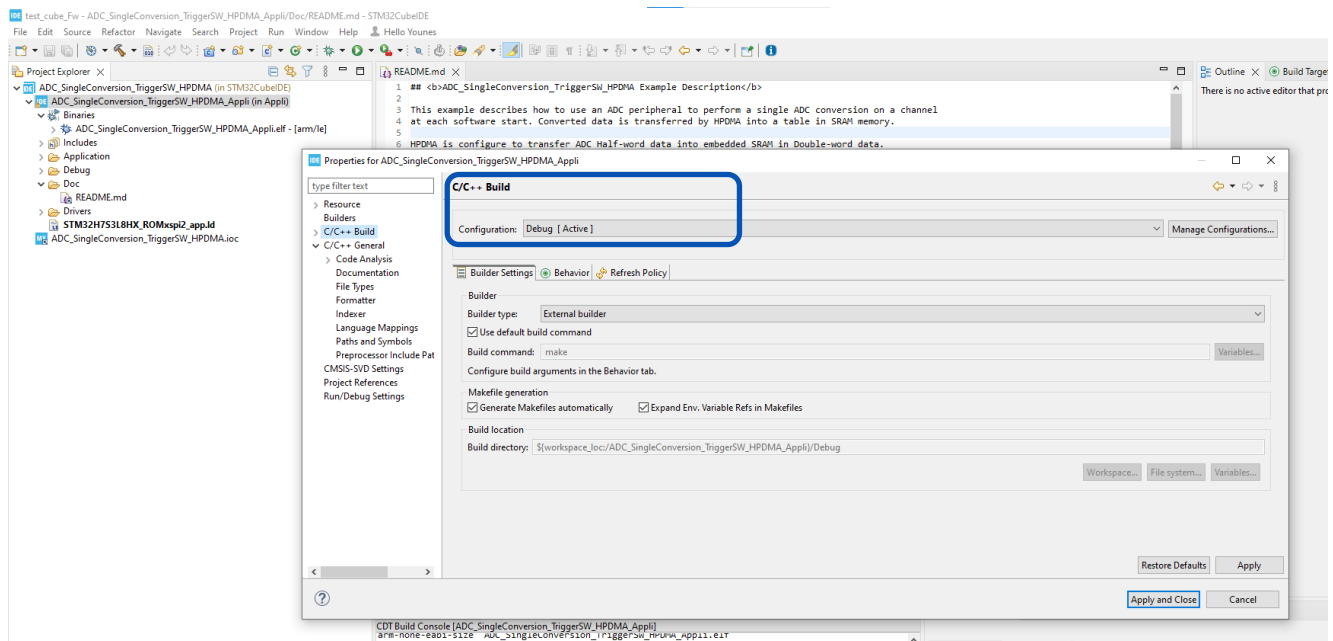
Introduction

The STM32H7R/S devices come with 64k bytes of boot flash allowing applications to run from external memory. The STM32CubeH7RS Firmware package comes with a rich set of examples running on STM32H7S or STM32H7R boards. Each example requires at least 2 subprojects: boot and appli.

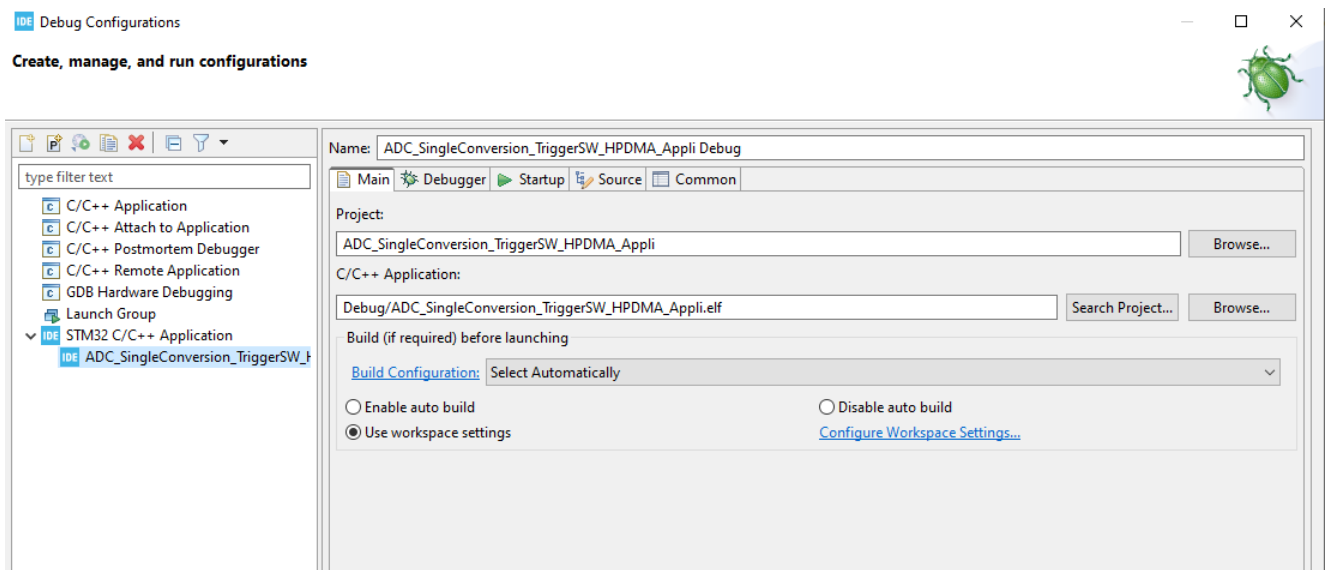
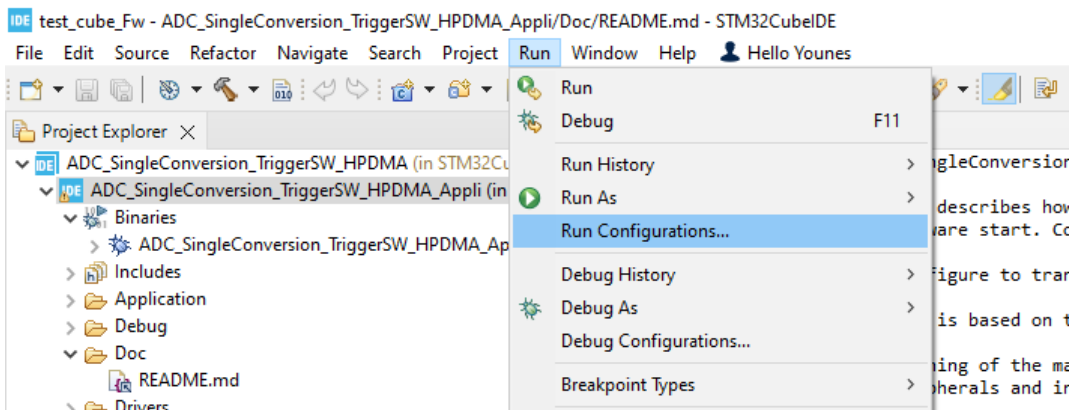
The readme file describes how to configure the multi-projects for the following toolchains: IAR, MDK-ARM and STM32CubeIDE. However, STM32CubeIDE multi-project seems a bit complicated for non-familiar users. To better facilitate the steps mentioned in the readme file, this document illustrates each step with snapshot images for more clarifications.

STM32CubeIDE multi-project configuration

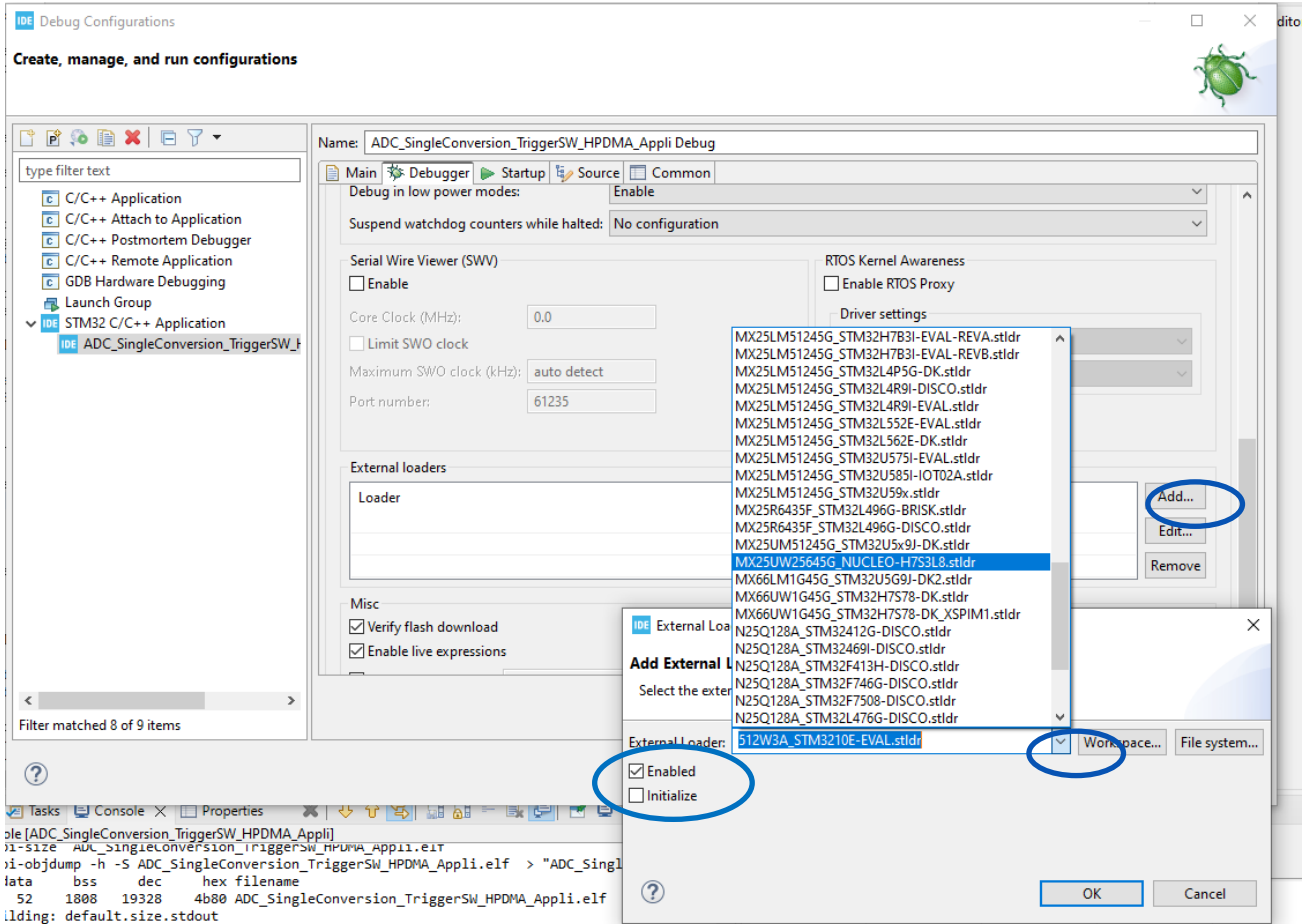
- Compile the example/application.
 - the elf file is required to configure the debug profile (the “active configuration” must be “debug”, else only assembly debug is available)



- Open the menu [Run]->[Debug configuration] and double click on [STM32 C/C++ Application]
 - It creates a default debug configuration for the current project selected



- In [Debugger] tab, section “External loaders” add the external loader corresponding to your Board/Memory as described below:
 - In “External loaders” section, click on [Add]
 - Select the loader among the available list (**MX25UW25645G_NUCLEO-H7S3L8.stldr** or **MX66UW1G45G_STM32H7S78-DK.stldr**)



IDE Debug Configurations

Create, manage, and run configurations

Name: ADC_SingleConversion_TriggerSW_HPDMa_Appli Debug

Debug in low power modes: Enable

Suspend watchdog counters while halted: No configuration

Serial Wire Viewer (SWV)

Enable

Core Clock (MHz): 0.0

Limit SWO clock

Maximum SWO clock (kHz): auto detect

Port number: 61235

External loaders

Loader

Misc

Verify flash download

Enable live expressions

RTOS Kernel Awareness

Enable RTOS Proxy

Driver settings

Driver: ThreadX

Port: cortex_m0

Port number: 60000

External Loader: MX25UW25645G_NUCLEO-H7S3L8.stldr

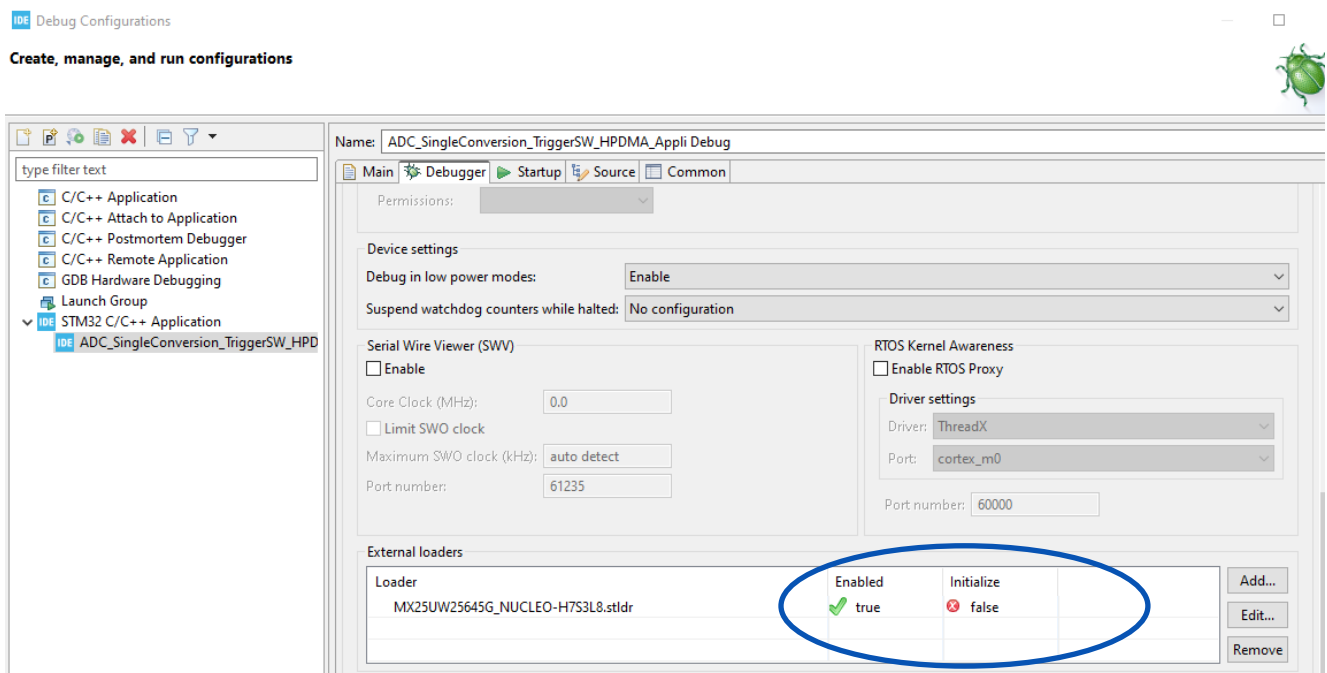
Enabled Initialize

OK Cancel

```

i-size ADC_SingleConversion_TriggerSW_HPDMa_Appli.elf
i-objdump -h -S ADC_SingleConversion_TriggerSW_HPDMa_Appli.elf > "ADC_Singl
lata bss dec hex filename
52 1808 19328 4b80 ADC_SingleConversion_TriggerSW_HPDMa_Appli.elf
lding: default.size.stdout
  
```

- Option “Enabled” checked and Option “Initialize” unchecked.
 - If not double click to change the settings



IDE Debug Configurations

Create, manage, and run configurations

Name: ADC_SingleConversion_TriggerSW_HPDMa_Appli Debug

Permissions: [dropdown]

Device settings

Debug in low power modes: Enable

Suspend watchdog counters while halted: No configuration

Serial Wire Viewer (SWV)

Enable

Core Clock (MHz): 0.0

Limit SWO clock

Maximum SWO clock (kHz): auto detect

Port number: 61235

RTOS Kernel Awareness

Enable RTOS Proxy

Driver settings

Driver: ThreadX

Port: cortex_m0

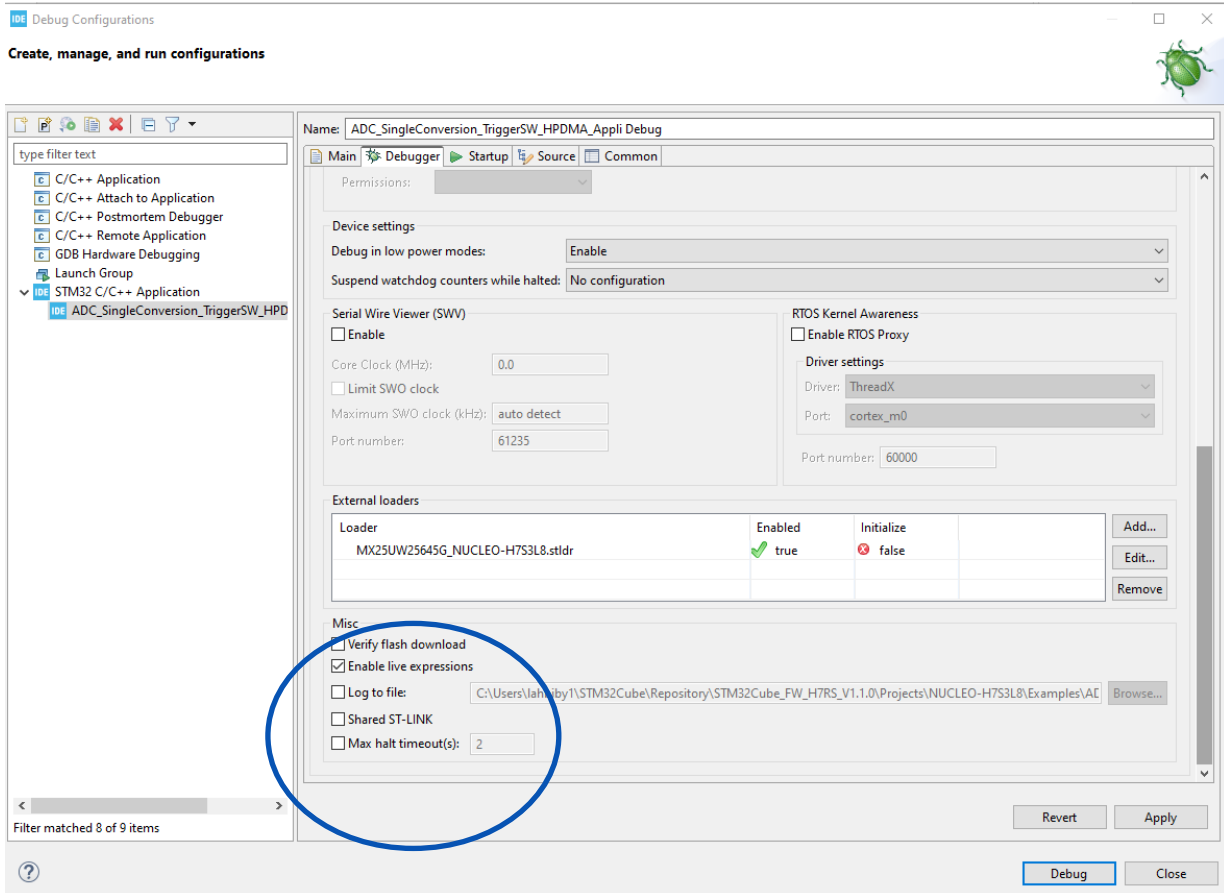
Port number: 60000

External loaders

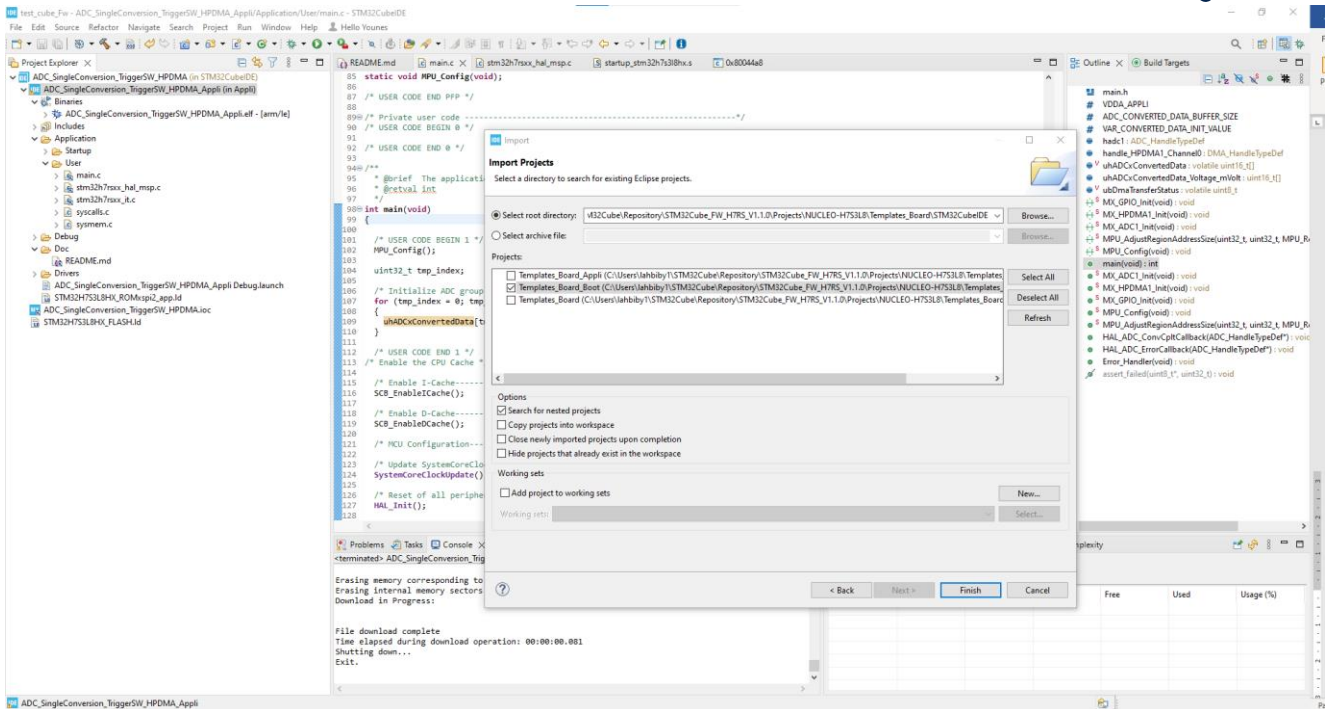
Loader	Enabled	Initialize
MX25UW25645G_NUCLEO-H7S3L8.stldr	<input checked="" type="checkbox"/> true	<input type="checkbox"/> false

Add... Edit... Remove

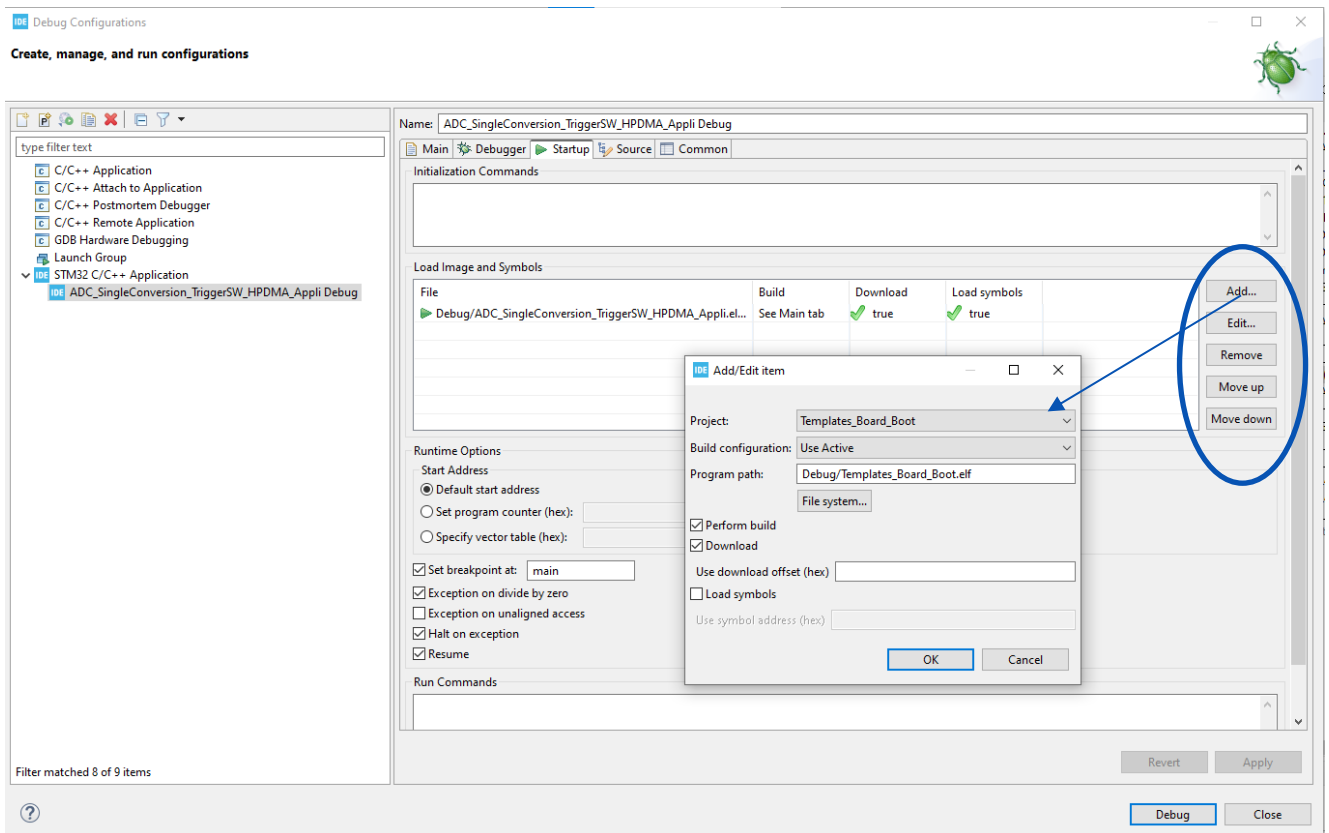
- In “Misc” section, uncheck the option “Verify flash download.”

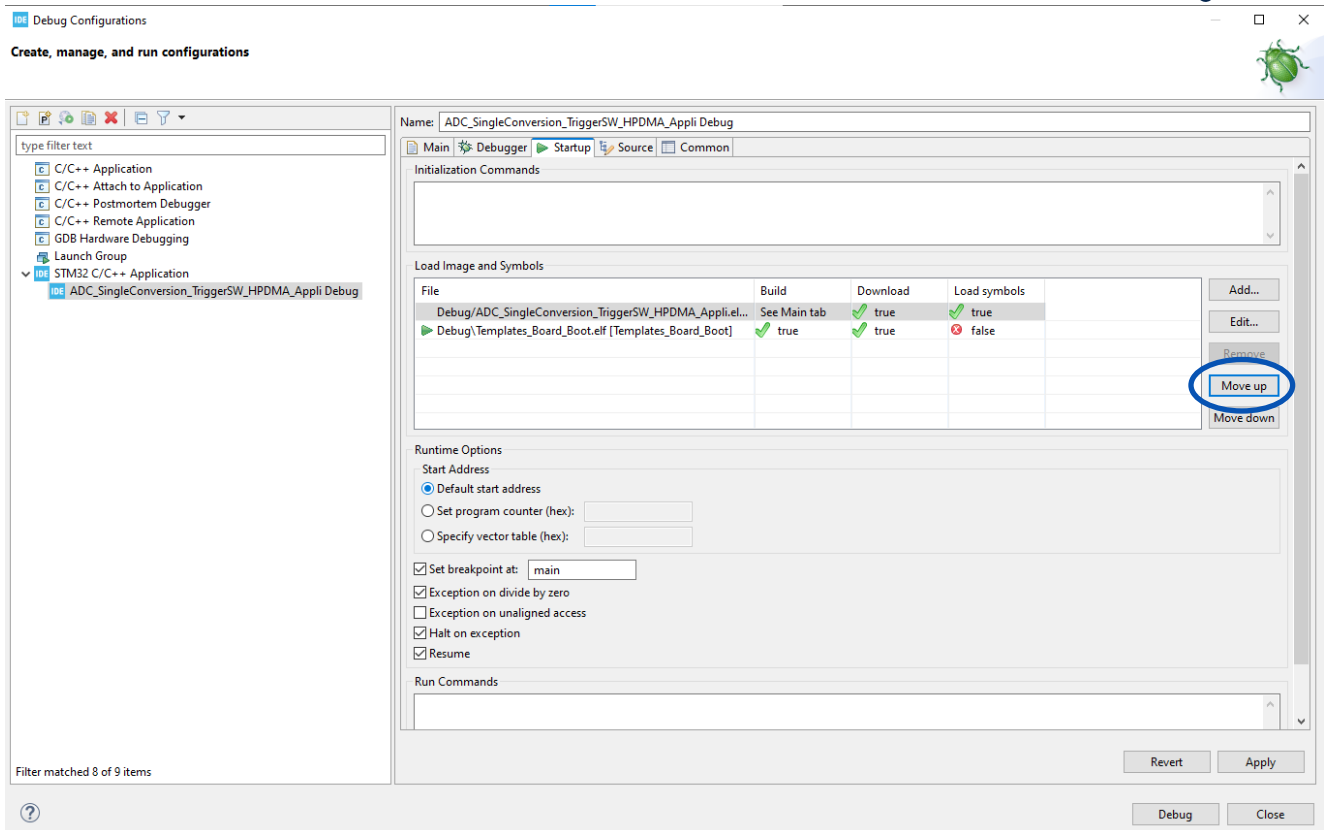


- In [Startup] tab, section “Load Image and Symbols”:
 - Click on [Add]
 - If your project contains a boot project:
 - click on “Project” and then select the boot project.
 - click on Build configuration and select “Use active”.
 - then select the following options:
 - “Perform build” checked.
 - “Download” checked.
 - “Load symbols” unchecked.
 - For example, you can import the boot project into workspace as follow.



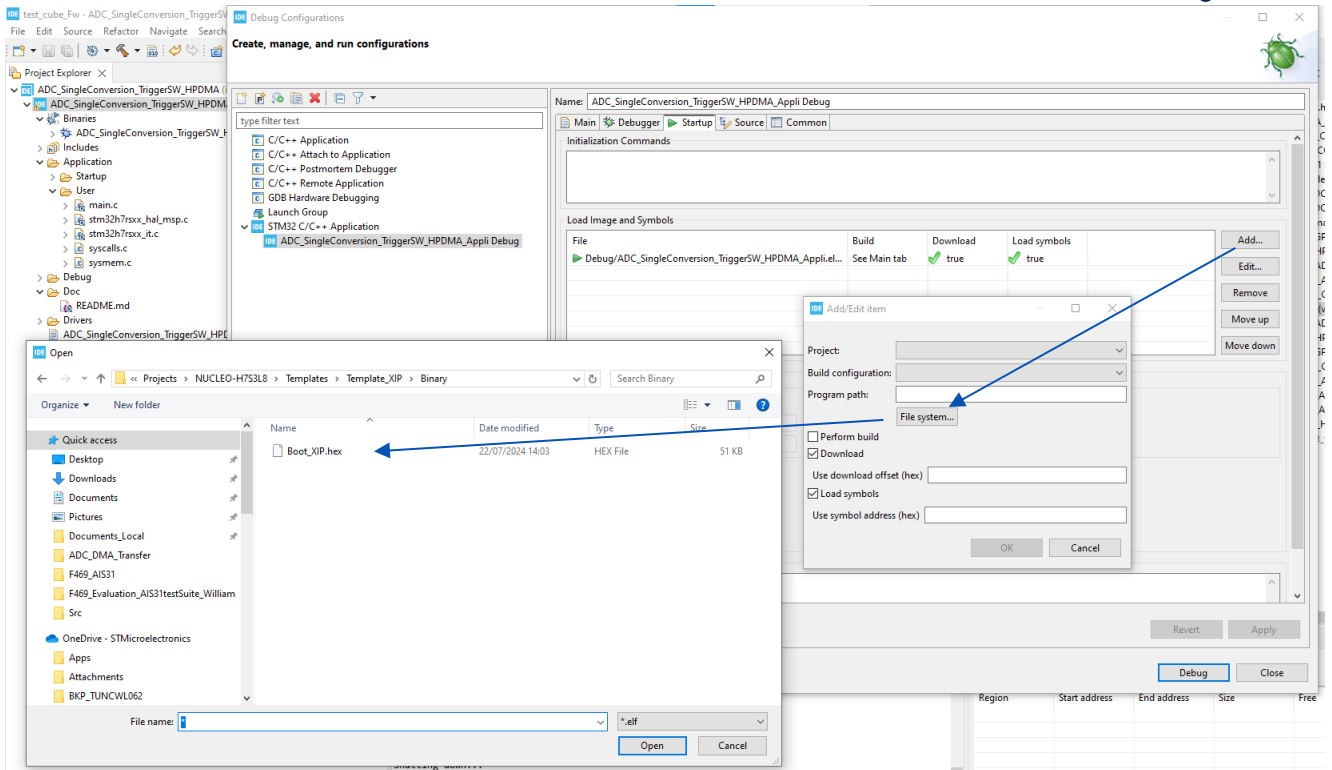
- Select then the added boot project and build/compile it first.
- Then click on add button and proceed as follow:



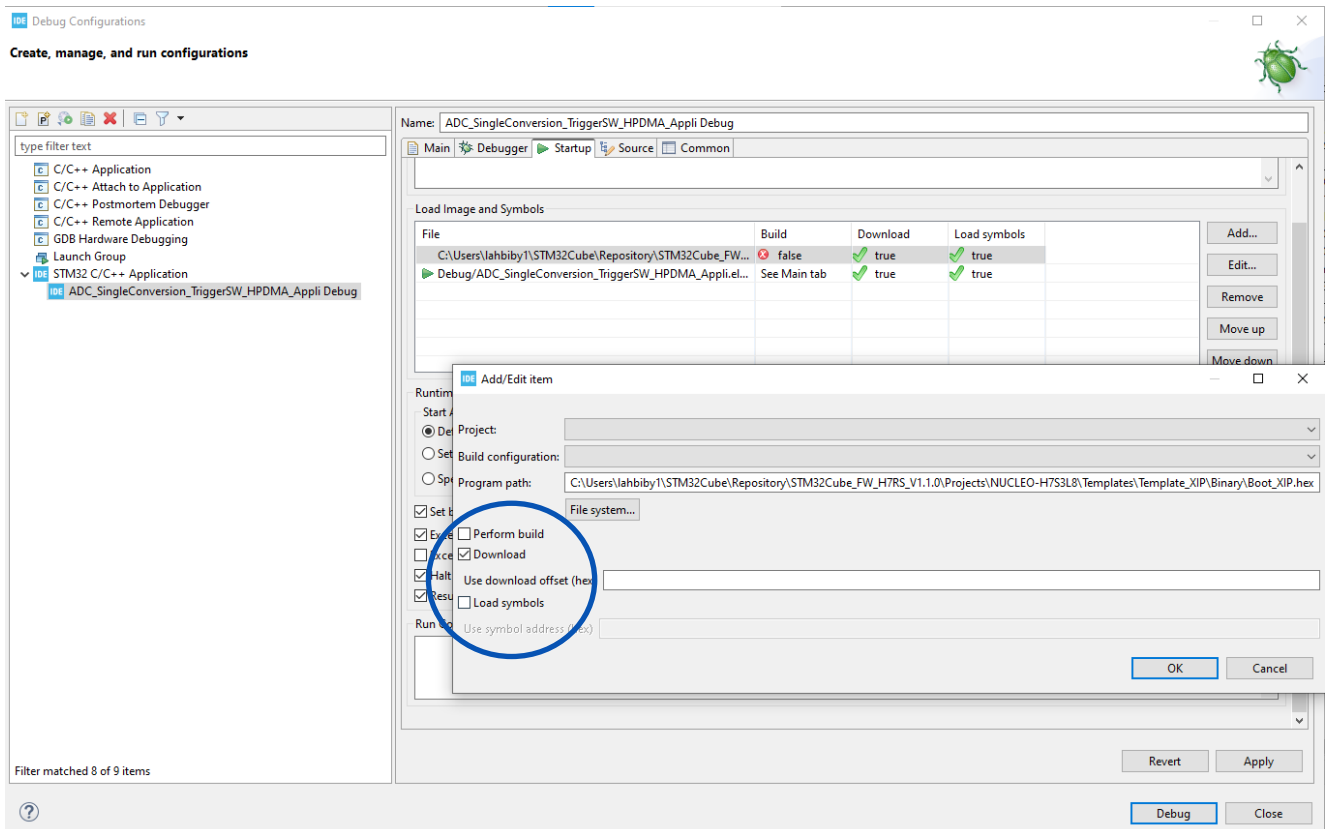


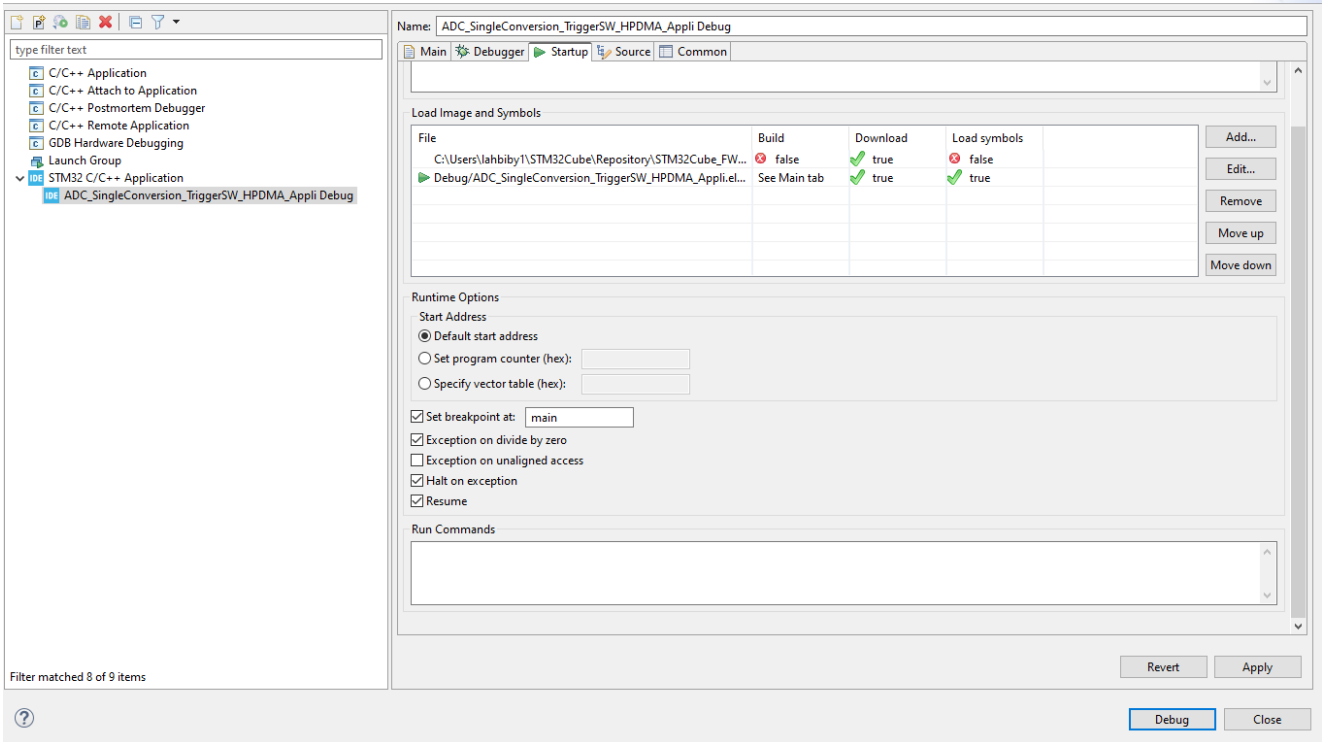
- If your project doesn't contain a boot project:
 - click on [File System] and select the Boot HEX file corresponding to your board

Boot_XIP.hex can be found in folder [Binary] on each Template_XIP project. You may need to force the capability to select a .hex file by typing " * " + pressing the "Enter" key in the file name dialog



- then select the following options by double clicking on the hex file to change the options
 - “Download” checked.
 - “Load symbols” unchecked.
 - Click Ok

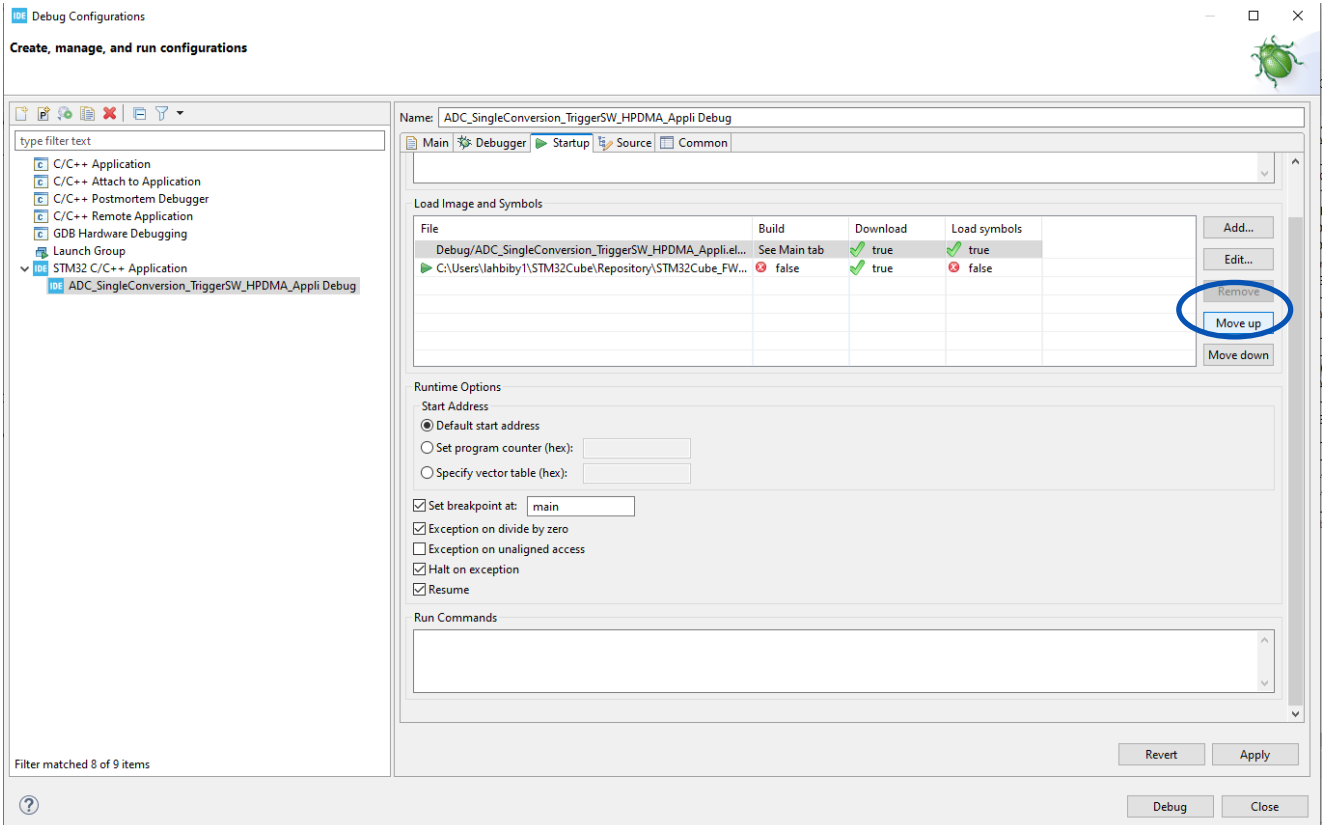




File	Build	Download	Load symbols
C:\Users\lahbiby1\STM32Cube\Repository\STM32Cube_FW... Debug/ADC_SingleConversion_TriggerSW_HPDM..._Appli.e...	See Main tab	✓ true	✓ true
C:\Users\lahbiby1\STM32Cube\Repository\STM32Cube_FW...	✗ false	✓ true	✗ false

- Back in the in the [Startup] tab, move down the boot project for it to be in second position.

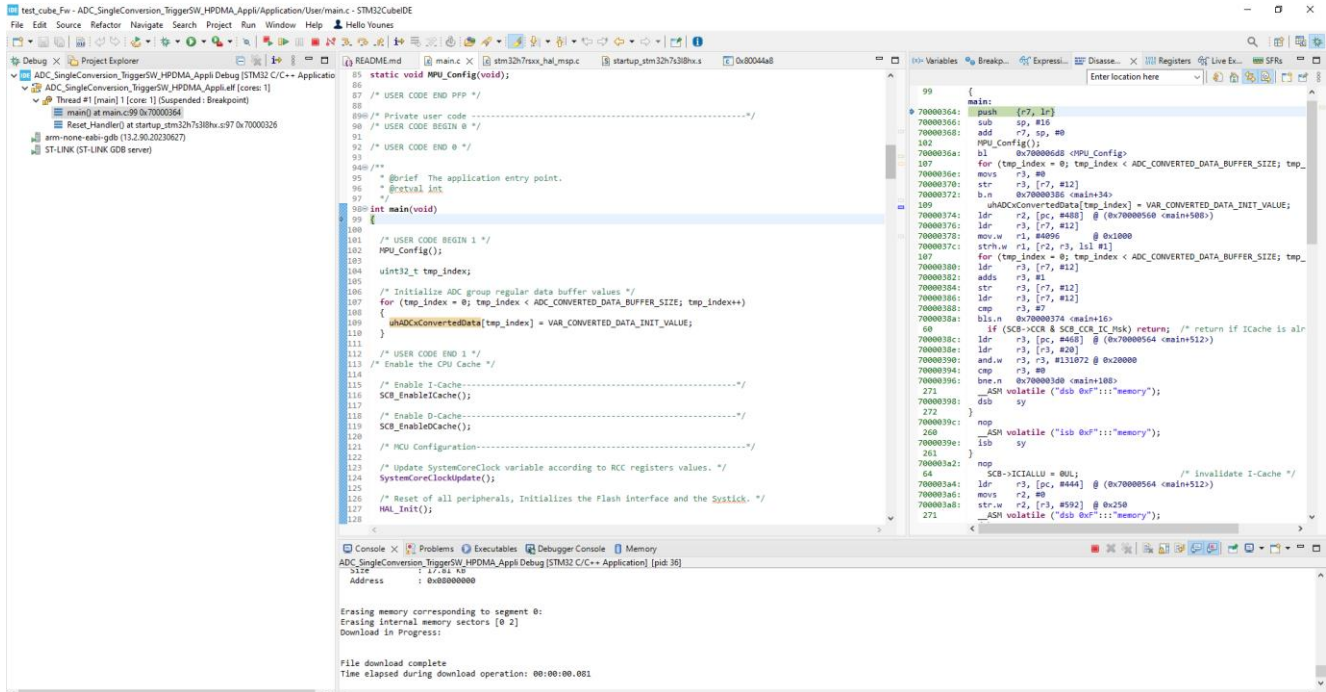
You can do this by selecting the Appli elf file and click on Move up button.



File	Build	Download	Load symbols
Debug/ADC_SingleConversion_TriggerSW_HPDM..._Appli.e...	See Main tab	✓ true	✓ true
C:\Users\lahbiby1\STM32Cube\Repository\STM32Cube_FW...	✗ false	✓ true	✗ false

- Our debug configuration is ready to be used.

Application is executed from the flash external memory @0x70000000



The screenshot displays an IDE interface for a project named 'ADC_SingleConversion_TriggerSW_HPDMA_Appli'. The main window shows the C source code for 'MPU_Config(void)' and 'main(void)'. The 'main' function includes comments for enabling I-Cache, D-Cache, and RCU Configuration, and updating SystemCoreClock. The assembly window on the right shows the compiled code for 'main', starting with a push instruction for 'r7, lr' and followed by various register operations and memory accesses. The console window at the bottom shows the process of erasing memory and downloading the application to the target device, with a message indicating 'File download complete' and a time elapsed of 00:00:00.001.