

How to integrate CMSIS-DSP libraries on a STM32 project



KDJEM.1
ST Employee

on 2024-06-03 05:00 AM - edited on 2024-06-24 01:36 AM by **ADMIN** Laurids_PETERSEN

Introduction

Since CMSIS V 5.8.0, the CMSIS-DSP libraries are supplied as an individual package. The advantage is to decouple the release cycles of DSP from the CMSIS-Core stuff. As a side-effect, the DSP libraries structure was changed and the steps described in the article: [Configuring DSP libraries on STM32CubeIDE](#) cannot be applied.

For that the aim of this article is to describe step by step: How to integrate this new CMSIS-DSP on an STM32 project.

Before you get started, open the ARM.CMSIS.pdsc file in \STM32Cube_FW_xx_Vx.x.x\Drivers\CMSIS folder with notepad to know which CMSIS version is included in the package.

```
<package schemaVersion="1.7.7" xmlns:xs="http://www.w3.org/2001/XMLSchema-instance" xs:noNamespaces>
  <name>CMSIS</name>
  <description>CMSIS (Common Microcontroller Software Interface Standard)</description>
  <vendor>ARM</vendor>
  <!-- <license>license.txt</license> -->
  <url>http://www.keil.com/pack/</url>

  <releases>
    <release version="5.9.0" date="2022-05-02">
      CMSIS-Core (M) : 5.6.0
      - Arm Cortex-M85 cpu support
      - Arm China STAR-MC1 cpu support
      - Updated system_ARMCM55.c
      CMSIS-DSP: 1.10.0 (see revision history for details)
      CMSIS-NN: 3.1.0 (see revision history for details)
      - Support for int16 convolution and fully connected for reference implementation
      - Support for DSP extension optimization for int16 convolution and fully connected
```

Software prerequisite

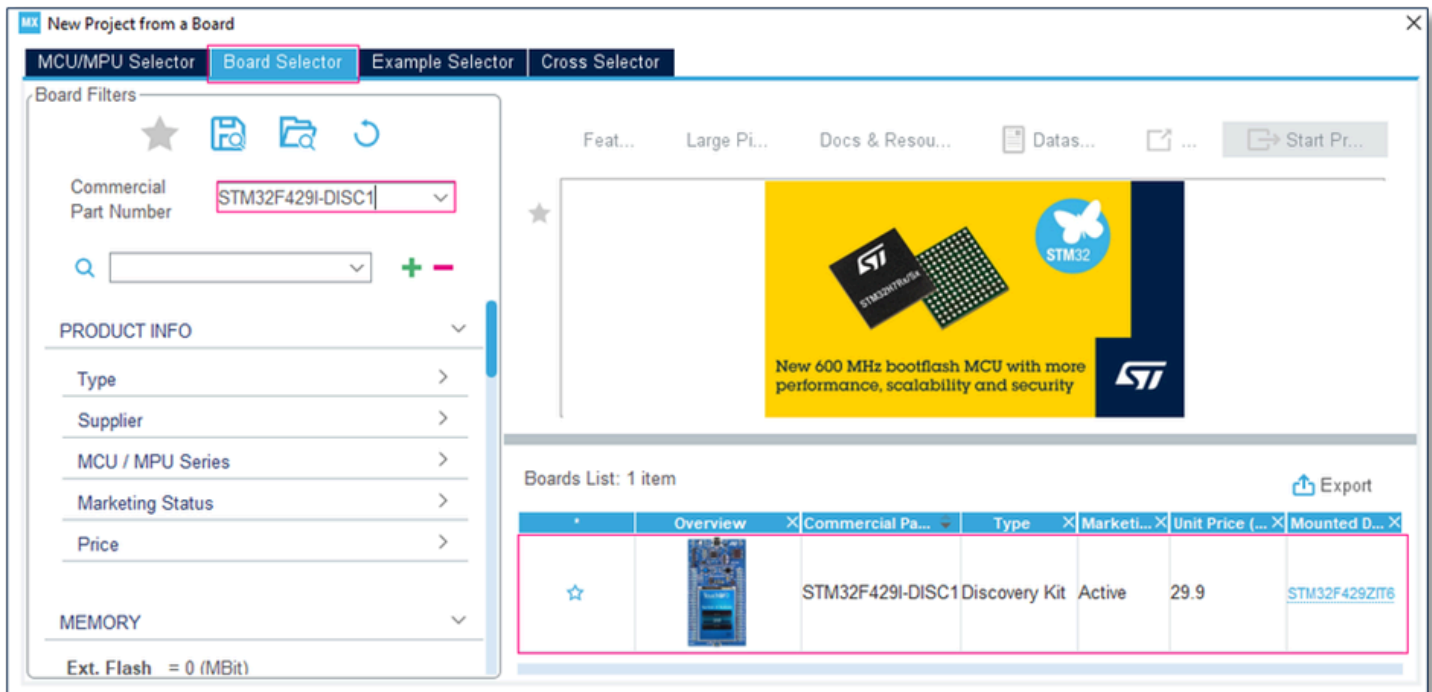
The versions used in this tutorial are:

- 1.15.0 for [STM32CubeIDE](#)
- 6.11.0 for [STM32CubeMX](#)
- 1.28.0 for [STM32CubeF4](#)

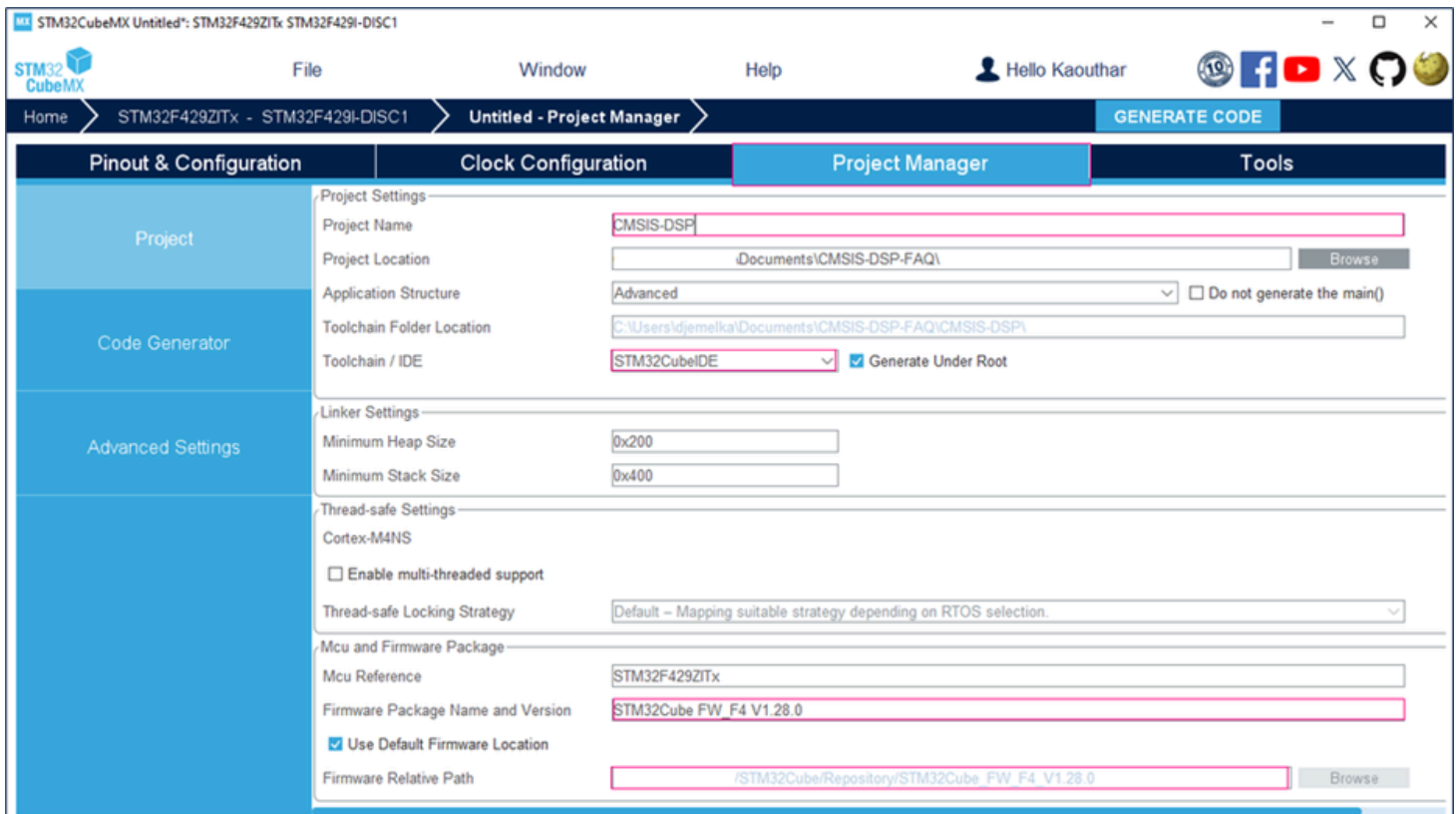
Step-by-step instructions

The following steps show how to integrate DSP libraries in your project when using STM32CubeIDE toolchain. For this demonstration, an example based on the STM32CubeF4 package version 1.28.0 is used.

1. Create a new project using STM32CubeMX and [STM32F429-DISC1](#) board.



2. After finishing the settings, generate the code with the STM32CubeIDE tool chain and STM32CubeF4 1.28.0



3. In the generated project, create a folder under `..\Drivers` named "CMSIS_DSP"






4. Copy these folders in the created folder:

<STM32Cube_Repository>\STM32Cube_FW_F4_V1.28.0\Drivers\CMSIS\DSP\Include

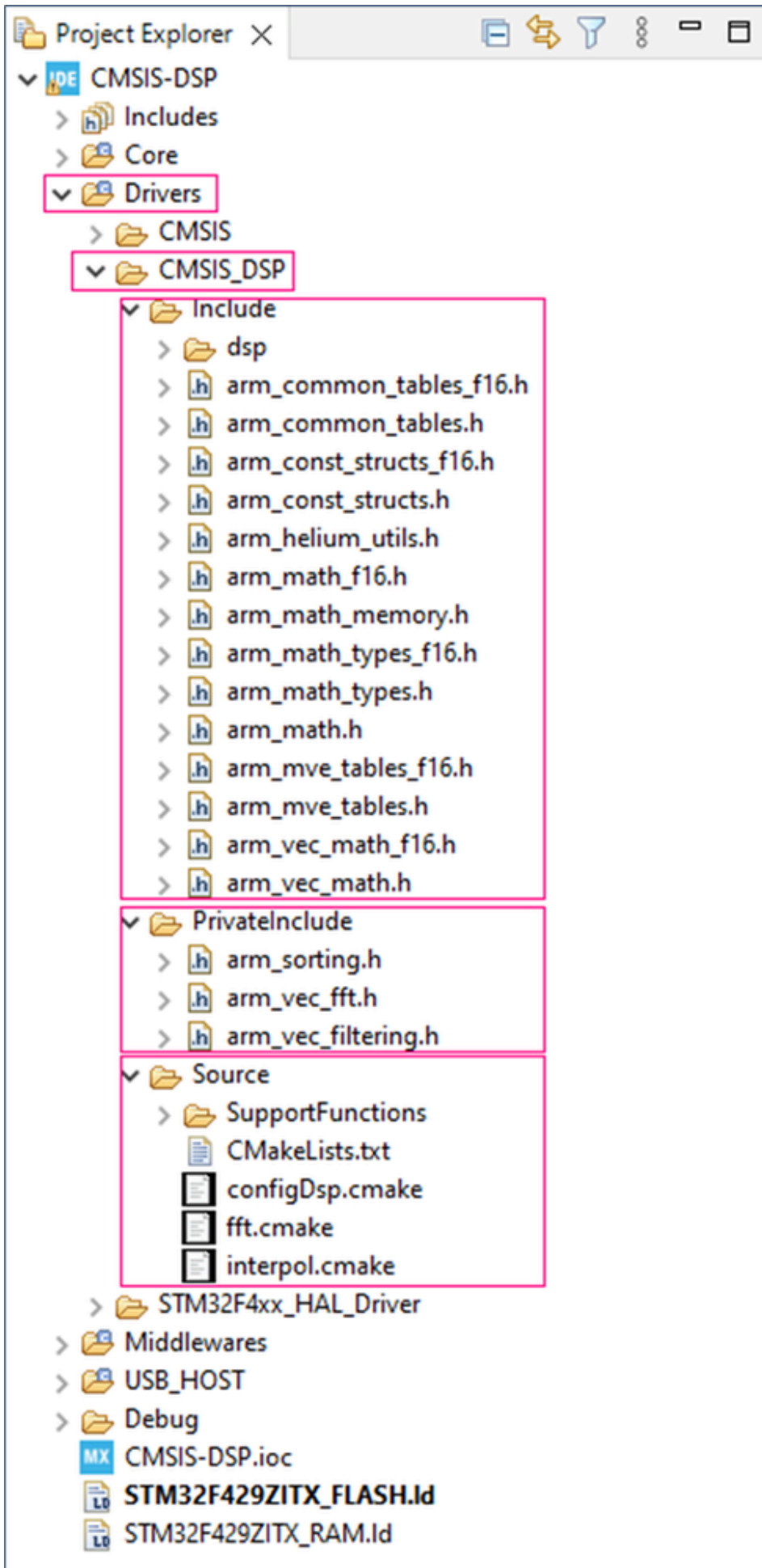
<STM32Cube_Repository>\STM32Cube_FW_F4_V1.28.0\Drivers\CMSIS\DSP\Private\Include

<STM32Cube_Repository>\STM32Cube_FW_F4_V1.28.0\Drivers\CMSIS\DSP\Source

5. For this example, only the SupportFunctions are used. For that, open the "Source" folder under "..\Drivers\CMSIS_DSP" and keep only the SupportFunctions (other folders can be removed).

uments > CMSIS-DSP-FAQ > CMSIS-DSP > Drivers > CMSIS_DSP > Source				
Name		Date modified	Type	Size
 SupportFunctions		26/04/2024 15:56	File folder	
 CMakeLists.txt		28/11/2023 08:13	Text Document	14 KB
 configDsp.cmake		28/11/2023 08:13	CMake Source File	2 KB
 fft.cmake		28/11/2023 08:13	CMake Source File	35 KB
 interpol.cmake		28/11/2023 08:13	CMake Source File	2 KB

6. Make sure that you have these folders in your project with this structure

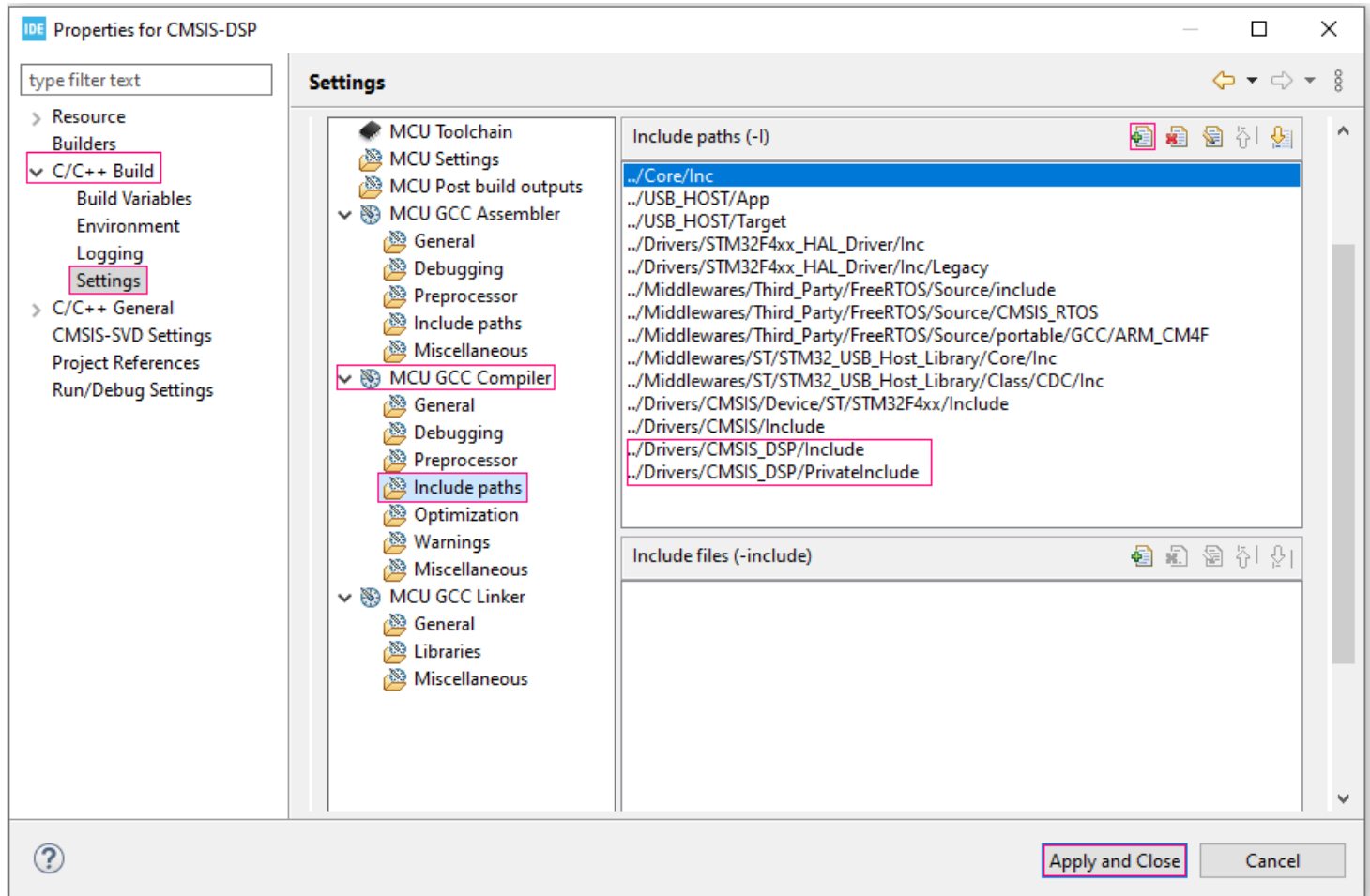


7. Update include paths:

Include paths: Select project properties from the [Project Explorer] section

From the [Project menu or [File menu], go to [Project properties] > [C/C++ Build] > [Settings] > [Tool Settings] > [MCU GCC Compiler] > [Include paths]

- Click on [Add] to include the new paths.
- Add the `../Drivers/CMSIS_DSP/Include` path and the `../Drivers/CMSIS_DSP/PrivateInclude` path



8. Open SupportFunctions.c and SupportFunctionsF16.c files and comment these lines of code:

File name	Source	Comment code lines
SupportFunctions.c	Drivers\CMSIS_DSP\Source\ SupportFunctions	<pre> #include "arm_barycenter_f32.c" #include "arm_bitonic_sort_f32.c" #include "arm_bubble_sort_f32.c" #include "arm_copy_f32.c" #include "arm_copy_f64.c" #include "arm_copy_q15.c" #include "arm_copy_q31.c" #include "arm_copy_q7.c" #include "arm_fill_f32.c" #include "arm_fill_f64.c" </pre>

```
#include "arm_fill_q15.c"
#include "arm_fill_q31.c"
#include "arm_fill_q7.c"
#include "arm_heap_sort_f32.c"
#include "arm_insertion_sort_f32.c"
#include "arm_merge_sort_f32.c"
#include "arm_merge_sort_init_f32.c"
#include "arm_quick_sort_f32.c"
#include "arm_selection_sort_f32.c"
#include "arm_sort_f32.c"
#include "arm_sort_init_f32.c"
#include "arm_weighted_sum_f32.c"

#include "arm_float_to_q15.c"
#include "arm_float_to_q31.c"
#include "arm_float_to_q7.c"
#include "arm_q15_to_float.c"
#include "arm_q15_to_q31.c"
#include "arm_q15_to_q7.c"
#include "arm_q31_to_float.c"
#include "arm_q31_to_q15.c"
#include "arm_q31_to_q7.c"
#include "arm_q7_to_float.c"
#include "arm_q7_to_q15.c"
#include "arm_q7_to_q31.c"
```

SupportFunctionsF16.c

Drivers\CMSIS_DSP\Source\
SupportFunctions

```
#include "arm_copy_f16.c"

#include "arm_fill_f16.c"

#include "arm_f16_to_q15.c"

#include "arm_f16_to_float.c"

#include "arm_q15_to_f16.c"

#include "arm_float_to_f16.c"

#include "arm_weighted_sum_f16.c"

#include "arm_barycenter_f16.c"
```

9. Now, you can use the DSP libraries in your project. For that open main.c and add the required header file "**arm_math.h**" and declare the used variables:

```
#include "arm_math.h"
```

```
/* USER CODE BEGIN PD */  
#define FFT_Length 1024  
/* USER CODE END PD */
```

```
/* USER CODE BEGIN PV */  
float32_t FFT_Input_Q15_f[50];  
float32_t aFFT_Input_Q15[50];  
/* USER CODE END PV */
```

10. Let us take the example of using the "arm_float_to_q15" function:

```
/* USER CODE BEGIN 1 */  
    arm_float_to_q15((float32_t *)&FFT_Input_Q15_f[0], (q15_t *)&aFFT_Input_Q15[0],  
/* USER CODE END 1 */
```

Note:

- If you use the 5.8.0 version of CMSIS or higher, follow the steps shared in this article.
- If you use a version of CMSIS less than 5.8.0, refer to [Configuring DSP libraries on STM32CubeIDE - STMicroelectronics Community](#) to add the DSP libraries in your project.

Conclusion

In conclusion, after following the steps outlined in this article, you should be able to add DSP libraries to a STM32 project. If you encounter any issues, we suggest that you create a post in the ST community for further assistance.