CPE3500: Embedded Digital Signal Processing

Lab Exercise 2: Discrete-Time Signals

Objective:

This lab focuses on the basics of discrete-time signals and their generation on the Nucleo board. The main objective is to make students familiar with digital signals and their transformation.

Introduction:

STM32CubeIDE is a comprehensive and user-friendly platform for developing and debugging STM32 microcontroller applications. This will prepare the student for subsequent labs involving digital signal processing exercises that can run on Nucleo-L476 microcontroller board.

In this exercise students will:

- Learn how to include CMSIS-DSP library and required settings to be able use Math and DSP functions.
- Create various types of basic discrete time signals and their transformations.
- Export data from STM32 microcontroller memory in debug mode and plot discrete time signals in Matlab.

Required Equipment:

Personal computer (Windows/Mac OS)

Nucleo-L476RG Board (64-pin)

USB Type-A (or Type-C) to Mini-B cable

Procedure:

1. Create a new project

Using the information provided in Lab-1, create a new project on STM32CubeIDE.

2. Including CMSIS-DSP Library and Header Files

After creating the project, open STM32CubeL4 repository page:

https://github.com/STMicroelectronics/STM32CubeL4/tree/master

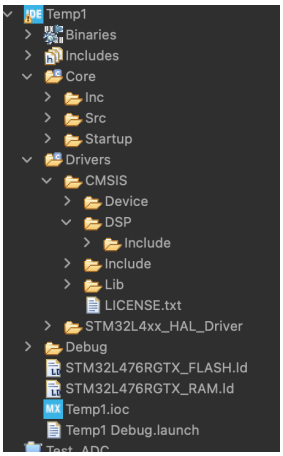
Click on the “Code” button and download the repository as a zip file. Extract the zip file in your Downloads folder.

Using another Windows Explorer (or Finder window for Mac), find your project folder (inside the workspace path) and go to /Drivers/CMSIS/. Create a DSP folder inside CMSIS folder.

Copy the include folder which is inside the DSP folder in /STM32CubeL4-master/Drivers/CMSIS/DSP/ to the DSP folder that you have just created in your workspace.

Copy the lib folder inside the Repository DSP folder into the workspace CMSIS folder (not inside DSP folder as you did in the precious step!).

After completing these steps, you see your project content inside the project explorer as below:



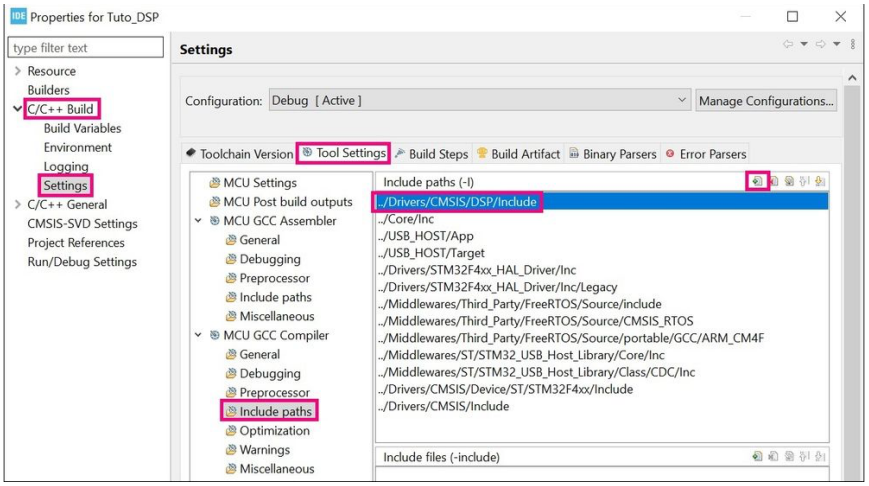
3. Including Paths and Updating Libraries

Select your project from project explorer.

- From Project menu or File menu, go to Project properties > C/C++ Build >

Settings > Tool Settings > MCU GCC Compiler > Include paths.

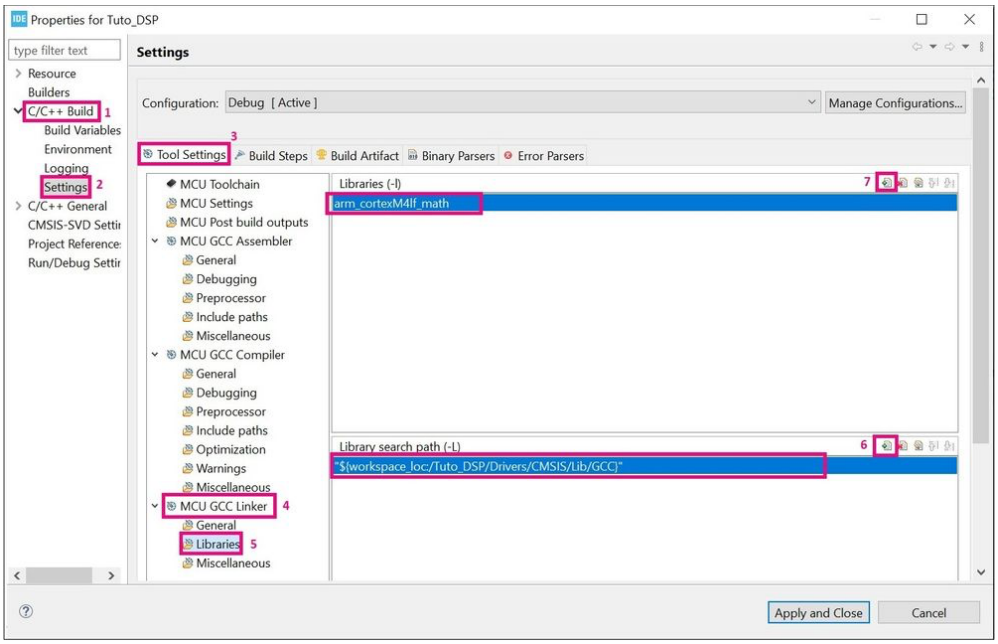
- Click on “Add” to include the new paths.
- Add your ../Drivers/CMSIS/DSP/Include path to here.



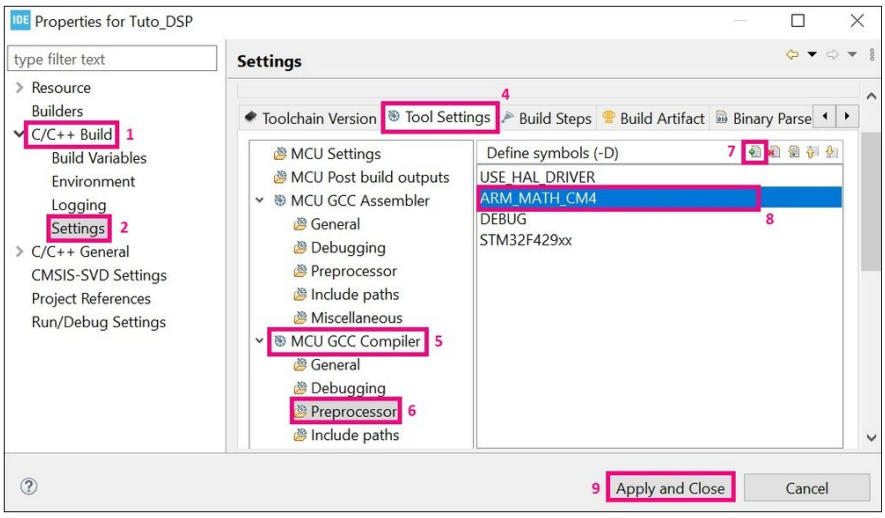
From MCU GCC Linker menu, select Libraries and add library path:

Select the GCC library present in the workspace path: ../Drivers/CMSIS/Lib/GCC.

In the Libraries (-l) section, add and insert he following library: "arm_cortexM4lf_math".



The final step, is to add the “ARM_MATH_CM4” symbol to the project, as presented in the figure below:



Now you can use the DSP libraries in your project, after including the required header file.

4. Creating Discrete-Time Signals

In this part, we will define and create basic discrete time signals. A sample C code given below can create unit impulse, unit step, rectangular, exponential, and sinusoidal signals.

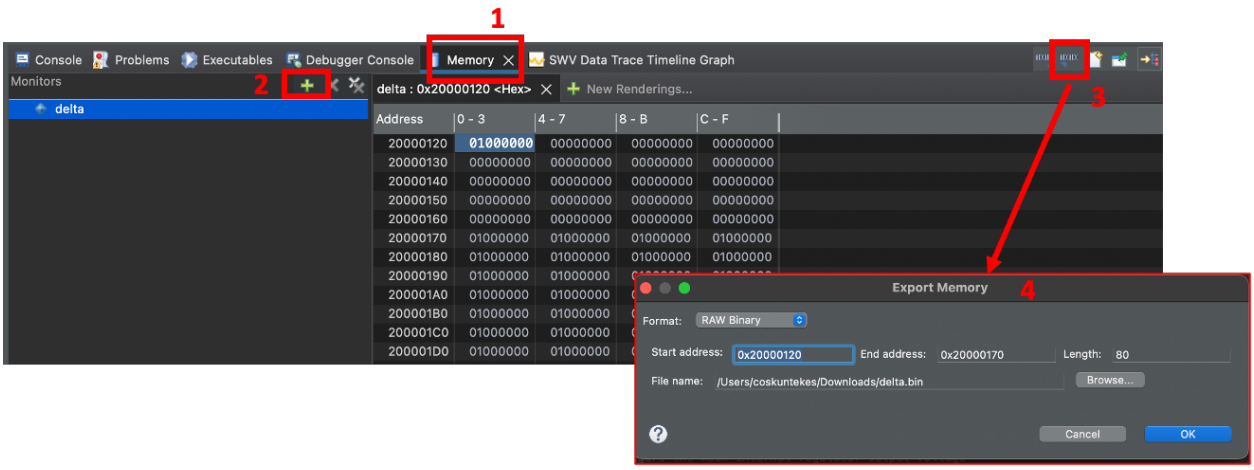
```c
#include "math.h"
#include "arm_math.h"

#define N 20

int n;
int delta[N], step[N], rect[N];
float expon[N], sinus[N];
float a=0.8, w0=PI/4;

int main()
{
  // unit impulse signal
  for(n=0;n<N;n++)
      if (n==0) delta[n]=1;
          else delta[n]=0;

  //unit step signal
  for(n=0;n<N;n++)
      step[n]=1;

   //rectangular signal between n=0 and 5
  for(n=0;n<N;n++)
      if ((n>=0) & (n<6))
          rect[n]=1;
      else
          rect[n]=0;

  //exponential signal
  for(n=0;n<N;n++)
      expon[n]=pow(a,(float)n);

  //sinusoidal signal
  for(n=0;n<N;n++)
      sinus[n]=sin(w0*(float)n);

   return 0;
}
```

Place the sections (includes, definitions, variables, etc.) of the code in your main.c of the project.

In the debug mode, insert a breakpoint on while(1) line of the main function and run your code up to this point.

You will then export the variables (discrete signals) into binary files to plot these signals in Matlab as described below:

1- In the console window, first click on Memory tab.
2- Click + sign and enter the variable name that you want to see the memory content
3- Click on the export button. In the pop-up window, select Raw Binary option, put appropriate filename (XXX.bin) and place 80 (N=20 and there is 4 byte allocation since both integer and float types are 4-byte long) for the length box.



Task-1:

Save all the discrete time signals in different files in binary format. Read these files in Matlab and plot them for each signal. To read binary files and plot signals use **readBinFileAndPlotSignal.m** file provided to you.

Task-2:

Modify the main.c code on STM2CubeIDE to generate another sinusoidal signal with ω=pi/8. Add these two sin signals to obtain the final composite signal. Save this signal and plot it in Matlab. Comment on this new signal compared with the original sine signal.

Task-3:

Apply time shifting (to the right only) of different amounts to the step, exponential and sinusoidal signal. Use the sample code below as reference which shifts unit impulse signal 3 units to the right.

Save and plot these shifted signals in Matlab.

```
//shifted unit pulse signal
for(n=0;n<N;n++)
    if (n<3)
        x1[n]=0;
    else
        x1[n]=delta[n-3];
```

Lab Exercise 2 Report:

Prepare a lab report (as single pdf file) consisting of the followings and upload it to the D2L dropbox.

- Cover Sheet
- Task-1: The main.c code screenshots (variable definitions and main function only), all the plots with proper labeling and title.
- Task-2: Modified code section, plot, and the comment.
- Task-3: Modified code section and plot.
- Conclusion (1-paragraph)