Tutorial: Integrating New SDK Libraries in SES Projects

This tutorial explains how new libraries and drivers are added to an existing Segger Embedded Studio (SES) project. There are essentially two steps involved in the process. First the source files and header files are integrated into the project, secondly the SDK is configured to activate the library and/or driver.

SES Configuration

First ensure that the path to the include file (`.h`) is registered in SES. Enter *Project* → *Edit Options...*. In the applicable build drop-down list, select *Common* to ensure that changes to configurations are applied to both the debug and release build. The drop-down list is highlighted in Figure 1. Then navigate to *Code* → *Preprocessor* in the left pane, and double click on *User Include Directories*, which brings up a list where the paths to header files can be registered.

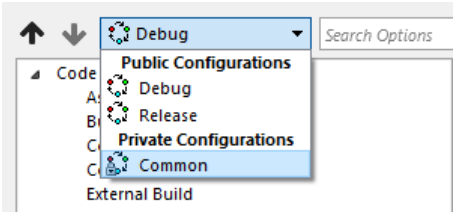

Figure 1: Pulldown to select which builds configuration changes are applied to.

With the header files registered, the source file is imported directly into the left pane folder structure. Either create a new folder in this hierarchy, or use an existing such as *nRF_Libraries*. Right-click and select *Add Existing File...*. This can be seen in Figure 2. Navigate to the implementation file (*.c*) and include it.

At this point all the source code is included, but the compiler will still throw an error if any attempt is made to use the libraries. This is handled in the next section where the static configuration options of the SDK are set.

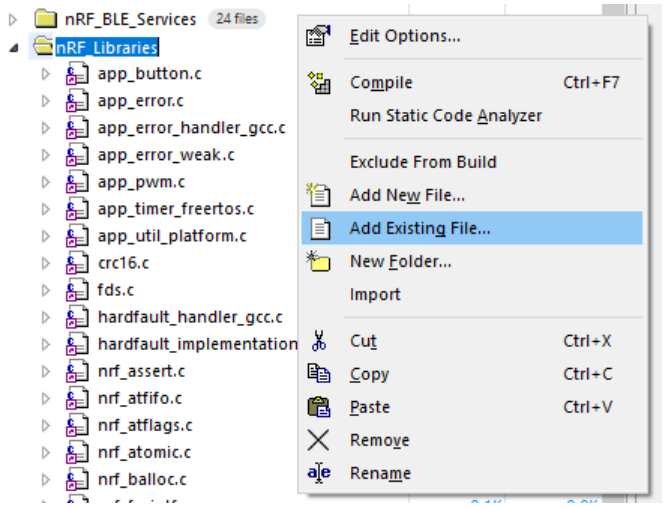

Figure 2: Menu option to include source files into project.

SDK Configuration

Before using the included libraries, the peripherals and library layer must be enabled in the SDK configuration file `sdk_config.h`. These changes can be done directly in the file itself with a text editor, but a better approach is to use the CMSIS Configurator tool. This results in a more structured overview of the configurations, and ensures that correct format is maintained at all times. The tool can be seen in Figure 3.

Note that it is usually not enough to only enable the library for a peripheral. The specific instance has to be enabled as well in the driver configuration. For example, consider the case where the SPI transaction manager is to be enabled. The transaction manager is an ”upper layer” library, and in the SDK configuration its option is located under *nRF_Libraries* → *NRF_SPI_MNGR_ENABLED*. To actually use any SPI peripheral, the instance and drivers will also have to be enabled. These options are located under *nRF_Drivers* → *NRFX_SPI_ENABLED* and *nRF_Drivers* → *SPI_ENABLED*.

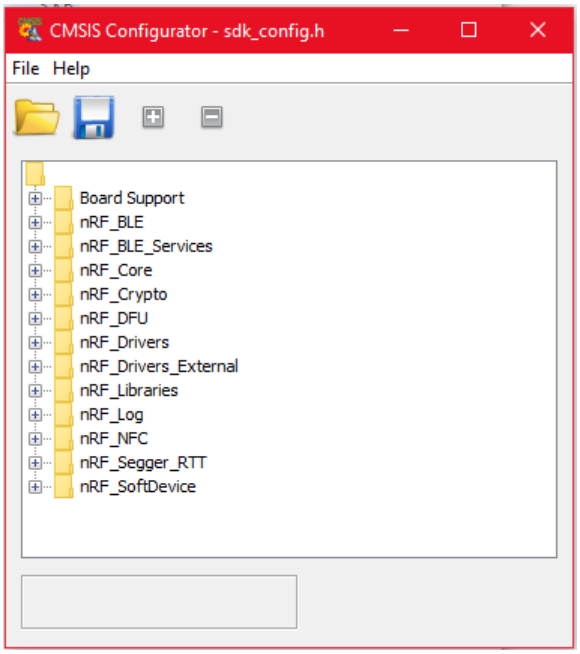

Figure 3: CMSIS Configurator overview.