

Using STM32CubeIDE

Abstract

This document describes the configuration of TESSY's adaption for the GDB debugger of the STM32CubeIDE. At the time of writing this document version 1.14.1 of the STM32CubeIDE has been used for the adaption. Interactive debugging is only supported by utilizing TESSY's special built-in data binary feature. See chapter 4 for further details.

Important Note: A **functional** STM32CubeIDE project which can build a target binary and successfully launch a debug session is required as a prerequisite!

Please note: The STM32CubeIDE adaption does not support interactive debugging features when executing tests with TESSY. (See 4 to learn how to debug interactively having your test data statically built into the target binary.)

Table of Contents

Abstract	1
1 Introduction.....	2
2 TESSY Environment Settings.....	2
3 Parallel Built and Parallel Execution.....	3
4 Interactive Debugging.....	4
5 Troubleshooting.....	8

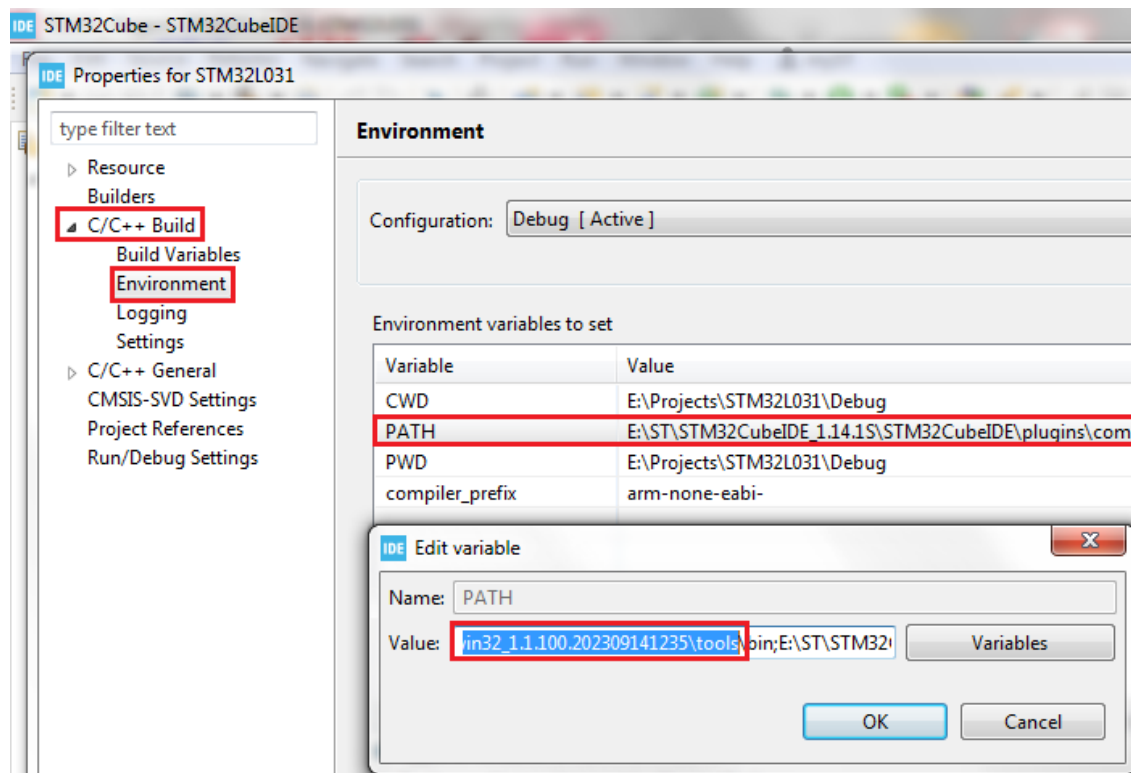
1 Introduction

The communication between TESSY and STM32CubeIDE's GDB server is based on direct commands between the GDB client and the GDB server, i.e. TESSY starts STM32CubeIDE's GDB server, determines the port of the GDB server (which is found in the GDB server output after the match given by attribute **Port Trigger**), starts STM32CubeIDE's GDB client, and sends all commands to the GDB client.

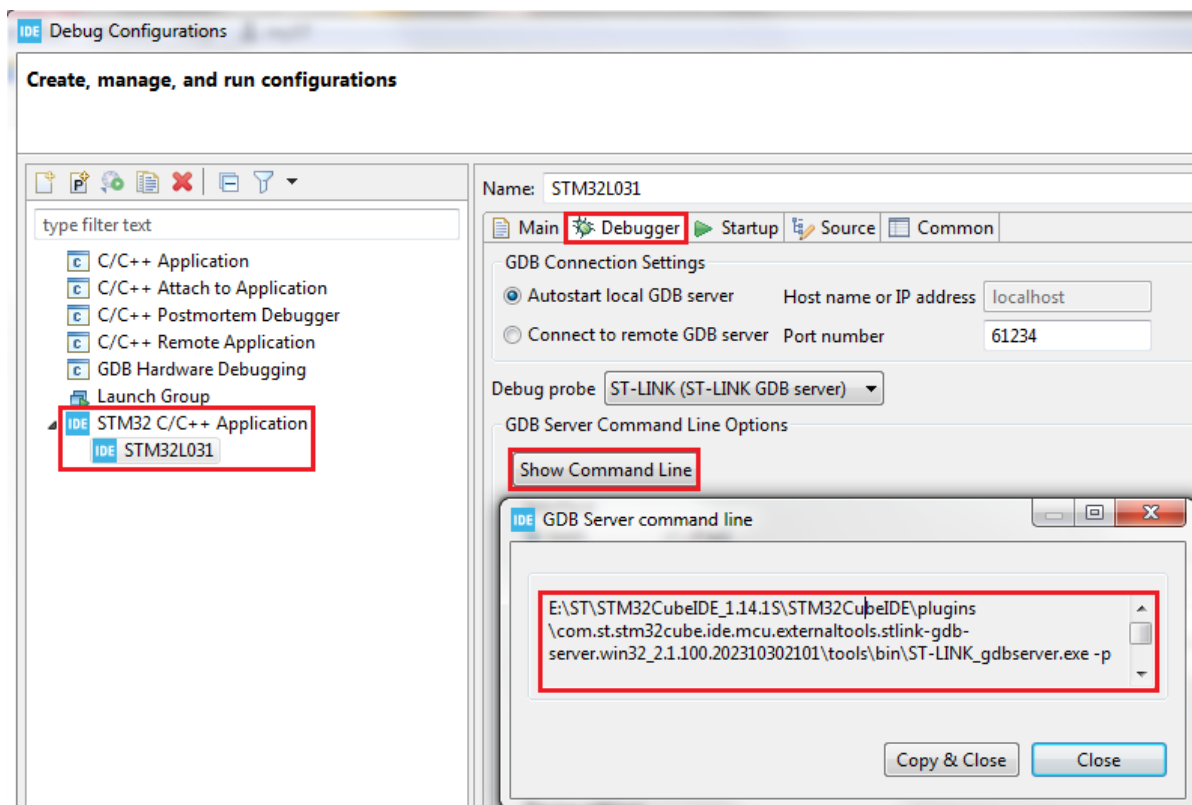
In order to debug the test application interactively with the test case data provided from TDE you need to rebuild the test application in a special mode, i.e. the input values will be compiled into the target binary. Please refer to chapter 4 for further details on how to generate this special target binary.

2 TESSY Environment Settings

At first, set the **Target Install Path** and check which errors remain after toggling the **Show Errors/Warnings** toggle button, which is found in the **Attributes** view's toolbar. Please adjust all remaining unresolved paths being displayed. Especially **Compiler Install Path** and **STM32CubeIDEProgrammer Path** may differ from TESSY's default path, if your version of STM32CubeIDE differs. To find the path to the compiler install path, you may open the STM32CubeIDE project's **Properties** dialog and try the first entry of the `PATH` environment variable as shown below. Or, you may use Microsoft's `procmon` tool and build the STM32CubeIDE project again to determine the path of the `arm-none-eabi-gcc` compiler. The compiler's install path is found one directory above the compiler's binary.



Next, check the TEE attribute **Linker** and fill in the TEE attribute **InitObjDir** to point to a directory which contains your board's startup code and set attribute **Use Board Files** to `true` to let the test driver be built including the startup code. The startup code may be found in folder `Debug->Core->Src` and `Debug->Core->Startup` of your STM32CubeIDE project. Do not use file `main.o` of the directory. TESSY's TEE provides attribute **Init Definition** to add further declarations at the beginning of TESSY's main test driver file, if required, and attribute **Init Code** to let TESSY generate code at the beginning of function `main()` found in that file. If you are not using target core 0, please choose the number by attribute **Target Core**. You should not need to modify attribute **GDB Server Debugger**, unless there are special options required. See below how to find the debugger's command line.



It is strongly recommended not to use option `-p [NUMBER]`, so that the debugger may let the operating system choose a convenient port. Since the debugger outputs its port number, TESSY can parse the debugger's output and can find the port number. Attribute **Port Trigger** should be the unique text pattern that is output in front of the port number. The default value for attribute **Port Trigger** should be fine.

3 Parallel Built and Parallel Execution

The amount of test objects built in parallel is given by **Compiler Concurrency**, which is by default set to 4, while 20 source files per test object are compiled in

parallel. So, no more than 80 processes run by default in parallel for the built. You can disable the parallel built by setting TEE attribute **Compiler Concurrency** to 1 and by clearing the TEE attribute **Make Options**.

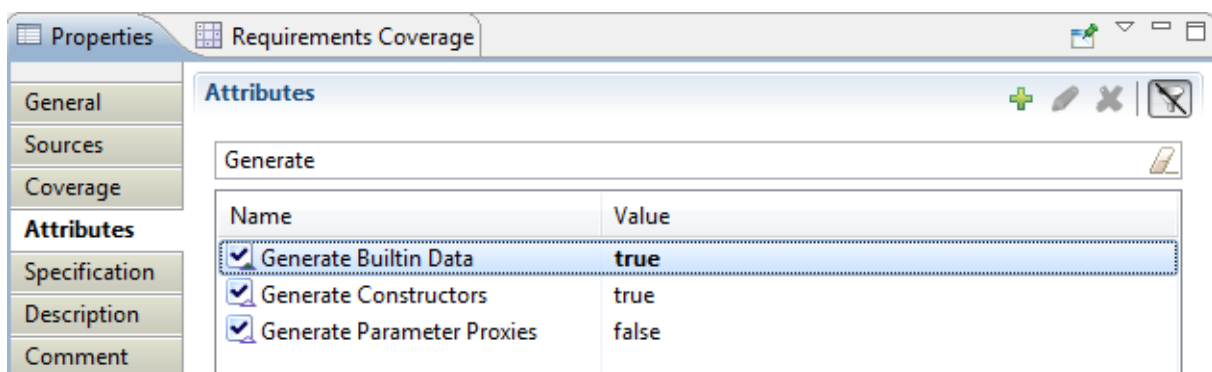
The parallel execution represented by attribute **Target Concurrency** is set to 1. That means one module test at a time by default. If you are using the same target devices, it might be possible to run multiple target tests each on its own target device in parallel. The corresponding master of TESSY supports it. However, we do not know if STM32CubeIDE supports parallel execution of multiple target devices of the same kind at the same time at all and we performed our tests using one target device for the adaption only.

Too many compiler or debugger processes in parallel may slow down the test execution. It depends on your computer system. So, feel free to test different values for **Compiler Concurrency**. You have to **Enable Expert Mode** from the **Attributes** view's toolbar to alter these attributes.

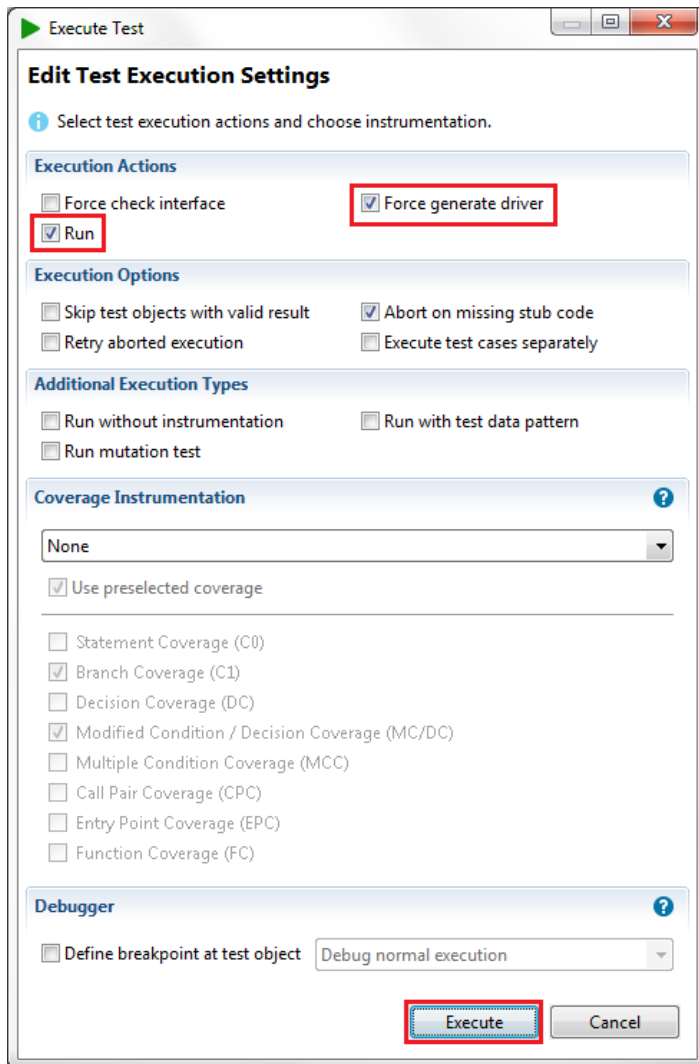
4 Interactive Debugging

Important Note: You do need a **working** project which can successfully launch a debug session!

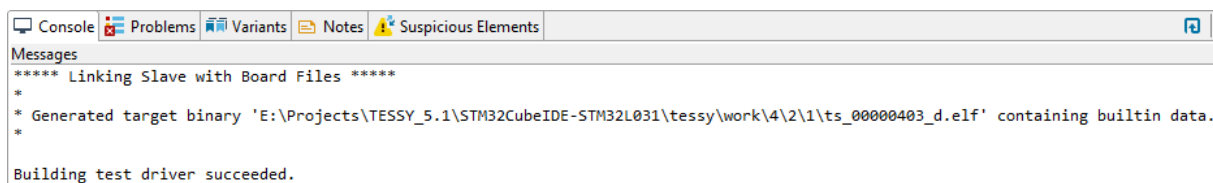
In order to debug the test application interactively with the test case data provided from TDE you need to rebuild the test application in a special mode, i.e. the input data will be compiled into the target binary. TESSY provides the TEE attribute **Generate Builtin Data** to facilitate this task. The attribute is of type `Boolean` and, if set to `true`, TESSY will rebuild your target binary during the next test run having the selected test data built-in, i.e. TESSY will not actually perform the test run but instead create the target binary with test data built-in to it. To disable this feature, you have to set the attribute to `false`.



Open the **Execute Test** dialog and make sure **Force Generate Driver** is selected.



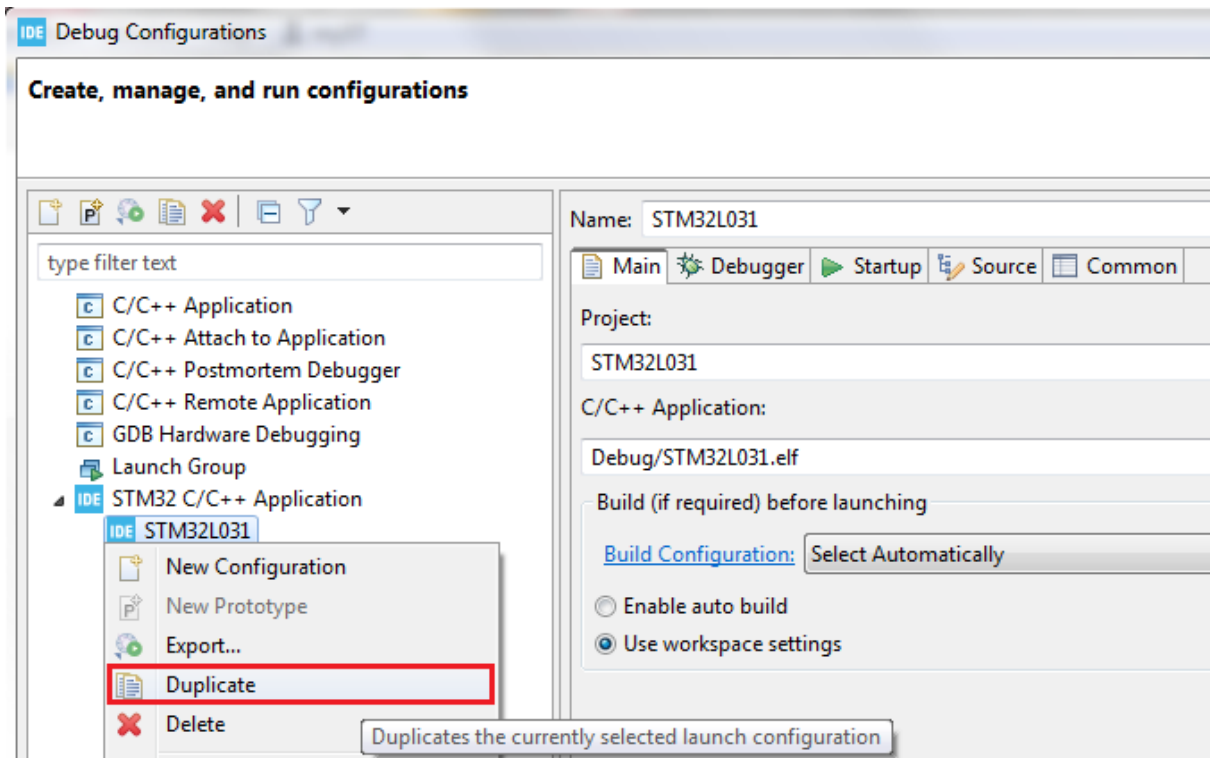
Now execute the test by pressing the **Execute** button. After the built, TESSY displays the path to the generated built-in target binary within its **Console** view.



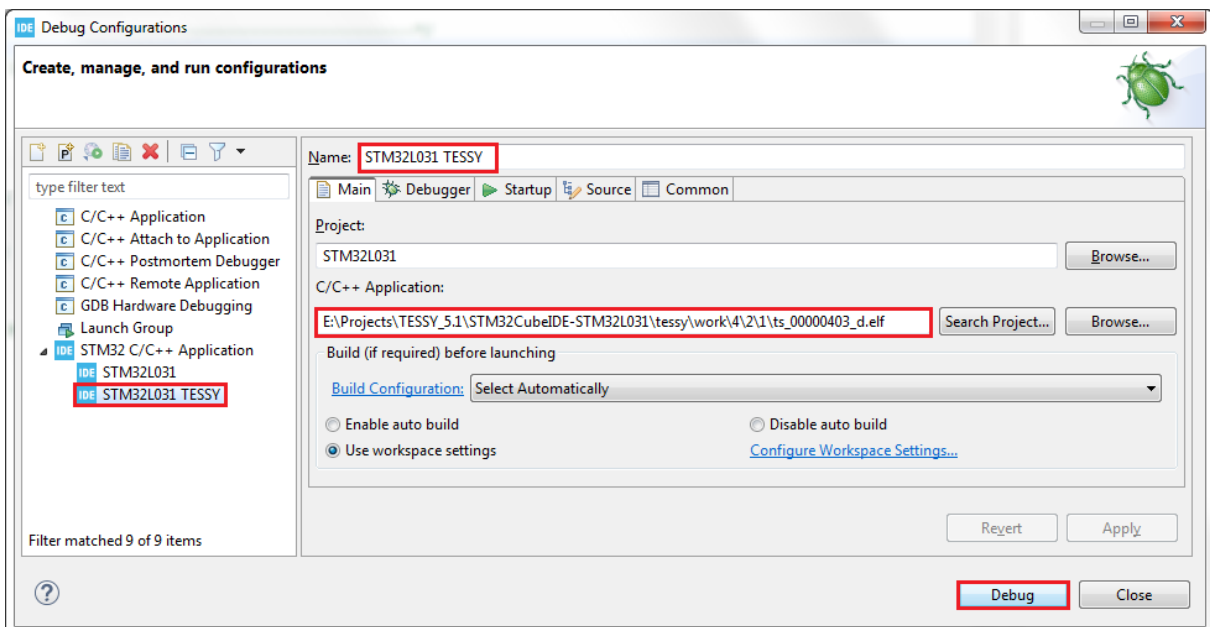
Since a test driver was built containing the test data but a test was not executed, TESSY will complain about it. Please ignore this complain.

Copy the target binary path from the **Console** view, switch to STM32CubeIDE, and open the **Debug Configuration...** from menu **Run**. Duplicate your current launch configuration and rename the (1) to TESSY. So, it will be easier to find this special launch configuration later.

TESSY Application Notes

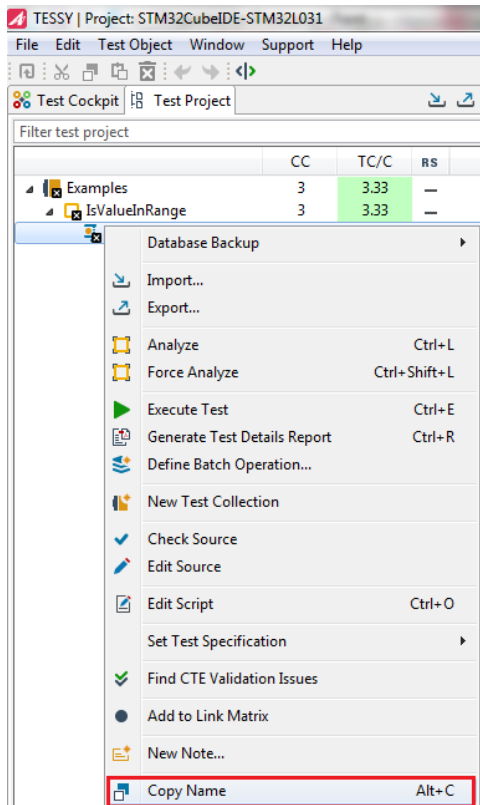


Select the new launch configuration and paste the path copied from TESSY's console into the **C/C++ Application** field as shown below and click **Debug**.

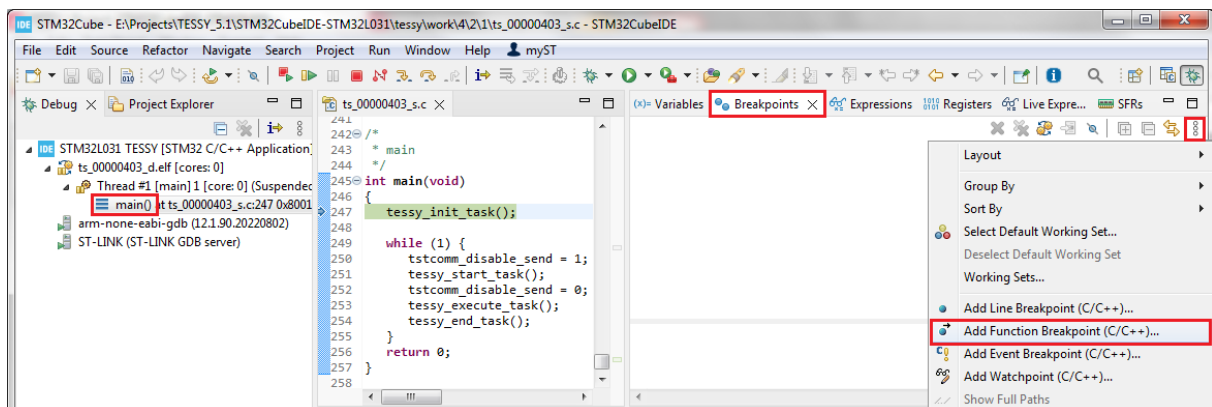


Switch to TESSY, right click the test object to open the context menu, and copy the test object name.

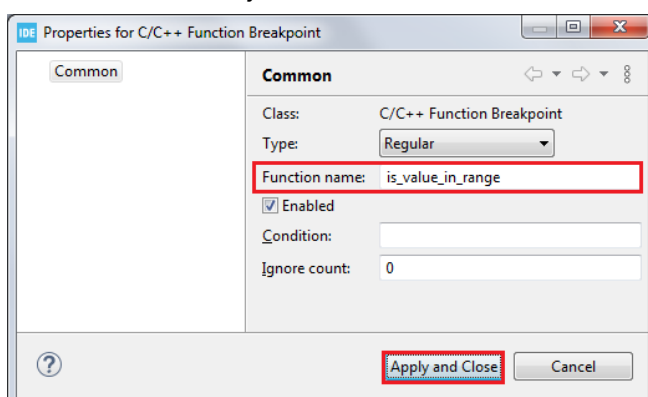
TESSY Application Notes



Switch back to STM32CubeIDE and open the **Properties for C/C++ Function Breakpoint** dialog.

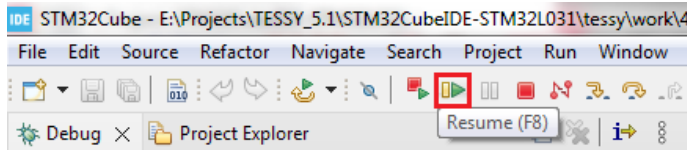


Paste the test object name into the **Function name** field and click **Apply and Close**.

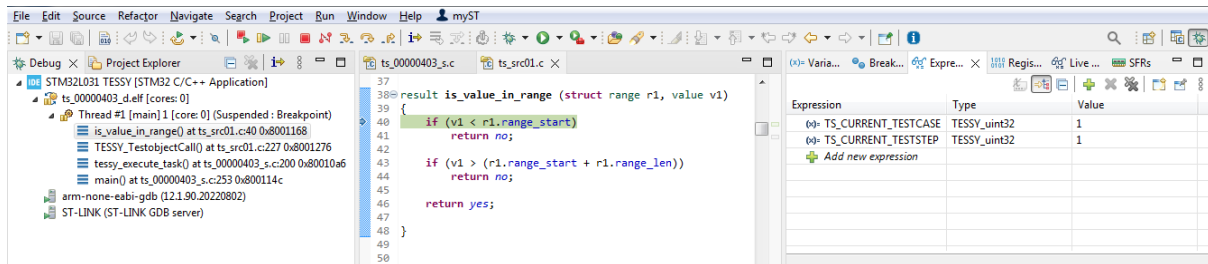


TESSY Application Notes

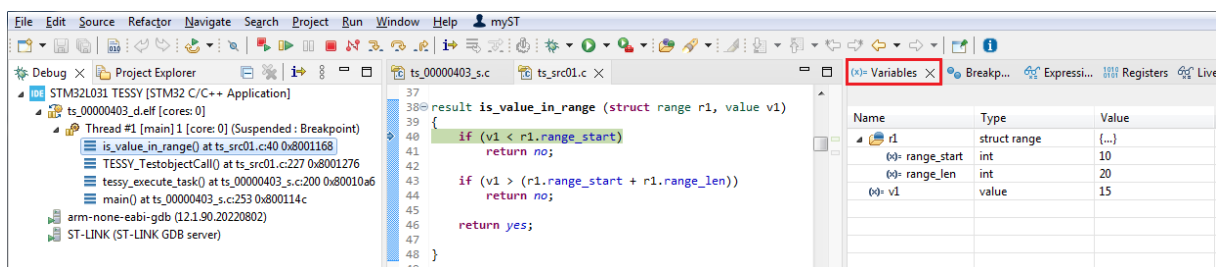
Now, resume the execution.



The debugger will stop at your test object.



It might be helpful to add the global variables `TS_CURRENT_TESTCASE` and `TS_CURRENT_TESTSTEP` to get the current test case number respectively the current test step number. The **Variables** view on the right side of STM32CubeIDE's **Debug** perspective shows the local variables.



5 Troubleshooting

Please contact support@razorcat.com if you encounter any unsolvable problems.