

Using ABIX HiperSim Debugger

Abstract

This document describes the usage of the ABIX HiperSim debugger as target debugger. At the time of writing this document version 1.0.1 of the debugger was tested. The ABIX HiperSim simulator supports multiple targets. By the time of writing this application, only the Melexis Mlx16 compiler was supported by TESSY.

Important Note: You need a **functional** project which can successfully launch a debug session.

Please note: The ABIX HiperSim debugger 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	2
4 Interactive Debugging.....	2
5 Troubleshooting.....	4

1 Introduction

The communication between TESSY and the ABIX HiperSim debugger is based on `mlx-gdb.exe` from within which the `mlxhipserver.exe` simulator given in TEE attribute **HiperSim Path** in conjunction with the **HiperSim Configuration File** and the **HiperSim ROM File**. The `mlx-gdb.exe` is started by TESSY by executing the command line found in TEE attribute **GDB Client Debugger**.

In order to debug the test application interactively with the test case values provided from within TDE you need to rebuild the test application in a special mode, i.e. the input values will be compiled into the application. You may either load the resulting binary directly into the simulator from a proper command line environment or use an alternative debugger, e.g. the Melexis Interactive Debugger. Please refer to chapter 4 for further details on how to generate the binary.

2 TESSY Environment Settings

At first, set the **Compiler Install Path** and 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. Next, check the TEE attributes **HiperSim ROM Address** and **HiperSim ROM File**. If no HiperSim ROM file is required, please set TEE attribute **GDB HiperSim Mode** to **1**. If the attribute is not visible, enable TEE's expert mode from TEE's toolbar.

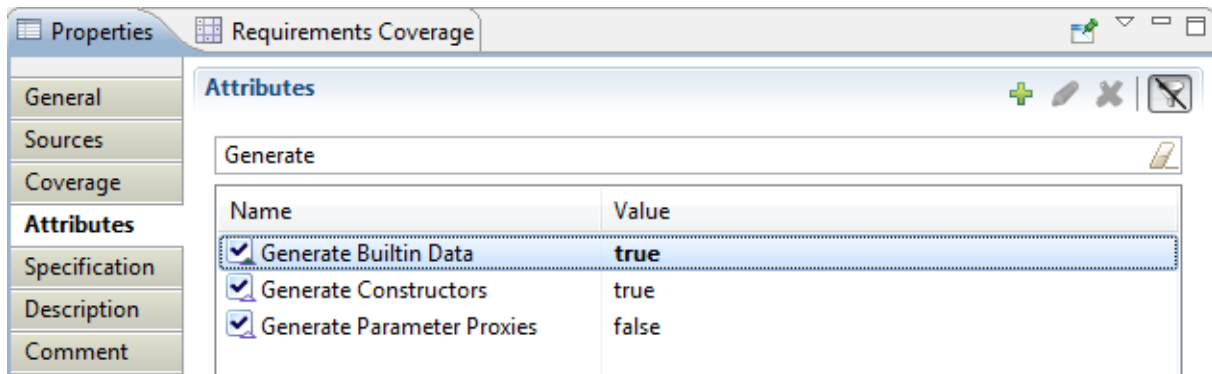
3 Parallel Built and Parallel Execution

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 is set to 10 by default. Too many compiler or debugger processes in parallel may slow down the test execution. It depends on your system. So, feel free to test different values for **Compiler Concurrency** and **Target Concurrency**.

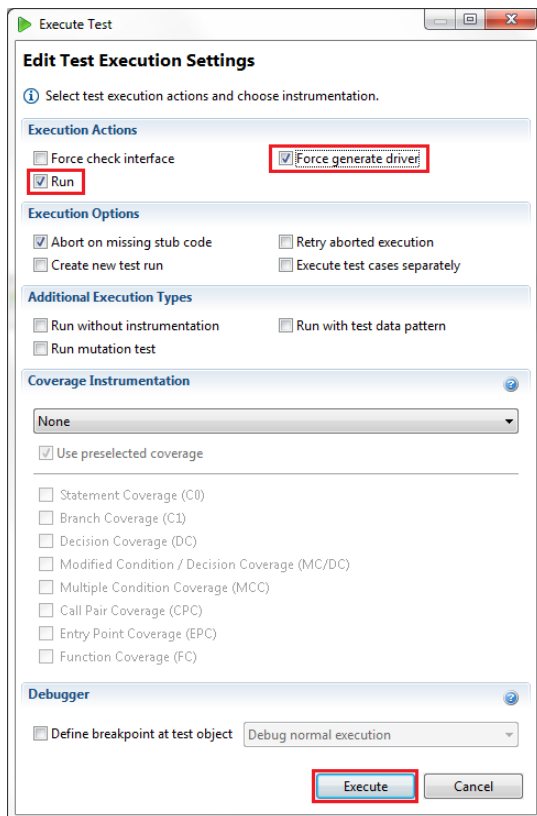
4 Interactive Debugging

In order to debug the test application interactively with the test case values provided from within TDE you need to rebuild the test application in a special mode, i.e. the input values will be compiled into the application. 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 rebuilt 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**.

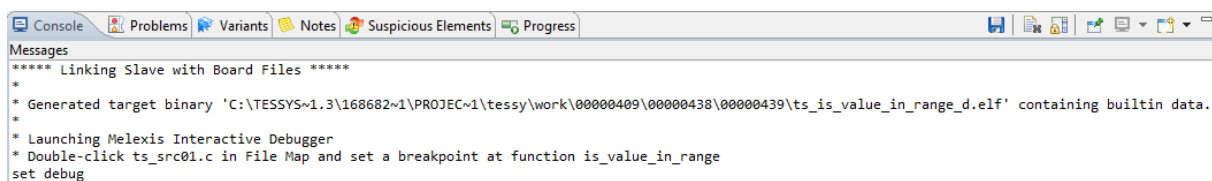
TESSY Application Notes



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



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



If TEE attribute Melexis Interactive Debugger is set to the Melexis MLXDBGW.exe debugger, it will be launched automatically with the generated binary. From the debugger's File Map double click to `ts_src01.c` and add a breakpoint to the function in test. Click go to let the debugger run to the breakpoint.

5 Troubleshooting

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