

# PLS Universal Debug Engine (UDE)

## 1 Abstract

This document provides important information for using the UdeDesktop debugger from PLS as target system with TESSY.

## Table of Contents

1	Abstract .....	1
2	TESSY Environment Settings.....	2
2.1	UDE Configuration File.....	2
2.2	Providing a Workspace File.....	2
2.3	Specifying the Startup Code (e.g. for TASKING compiler) .....	4
2.4	Locator Settings .....	4
3	PLS UDE .....	4
3.1	Workspace File Settings.....	4
3.1.1	Connect target after load.....	5
3.1.2	Ignore program from workspace .....	6
3.1.3	Automatically flash the binary.....	6
3.2	Configuration File Settings .....	7
3.2.1	Disable Watchdog Timer .....	7
3.2.2	Using a specific UDE version .....	8
4	Debugging the Test Object.....	9
5	Visibility of the UDE .....	10
6	Troubleshooting.....	10
6.1	The UDE Starts Up Very Slowly .....	10
6.2	TASKING C166, Version 8.0 .....	11
6.3	TASKING Tricore VX, Version 2.0 and later.....	11
6.4	Flashing Problems.....	11
6.4.1	TC26xx/TC27xx .....	11

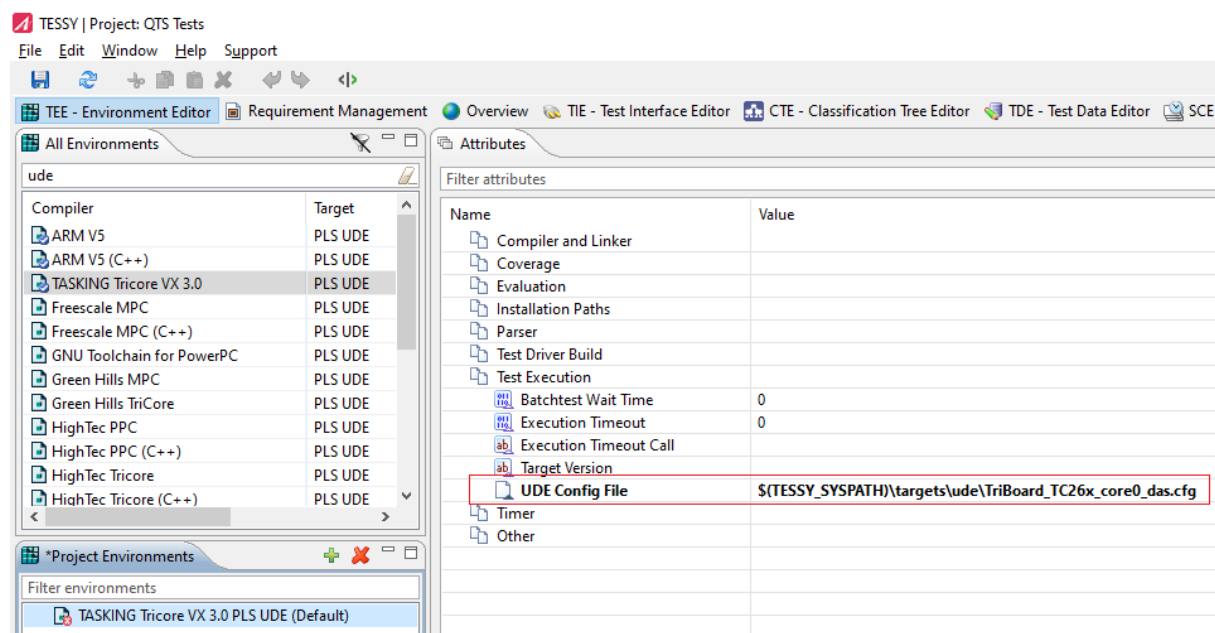
## 2 TESSY Environment Settings

For specific hardware targets you need to select the correct UDE configuration file and provide the necessary startup code and memory layout. This information should be available from your development environment settings of your UDE workspace.

Refer to the TASKING compiler application note for details on how to configure the TESSY module properties.

### 2.1 UDE Configuration File

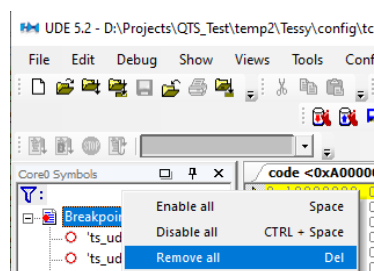
The **UDE Config File** attribute from the TESSY Environment Editor (TEE) points to the desired UDE target configuration file as shown below. The configuration file is required for the initialization of the COM interface of UDE. Please select the target configuration file that you are actually using for your PLS UDE development project.



The **UDE Config File** attribute's content is used if the **Workspace File** attribute is missing or empty (see 2.2). Otherwise the workspace file's configuration file is used.

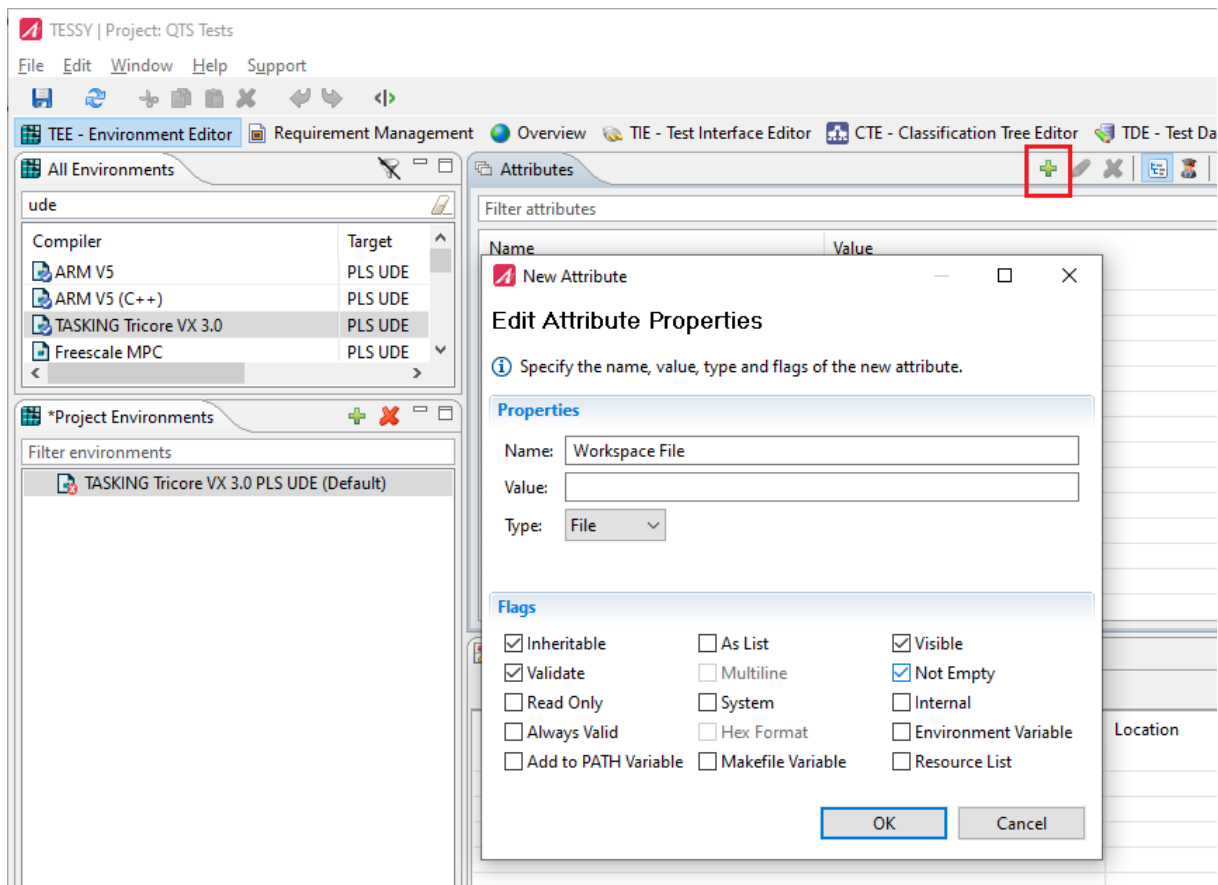
### 2.2 Providing a Workspace File

For each test run TESSY creates a copy of the given UDE workspace file. If you really need to use your own UDE workspace file, which is not recommended, it is strongly recommended to remove all breakpoints within this UDE workspace file. Open the breakpoint view from within the UDE desktop (**Views->Symbols**). If there are any, remove them by selecting **Remove all** from the context menu.

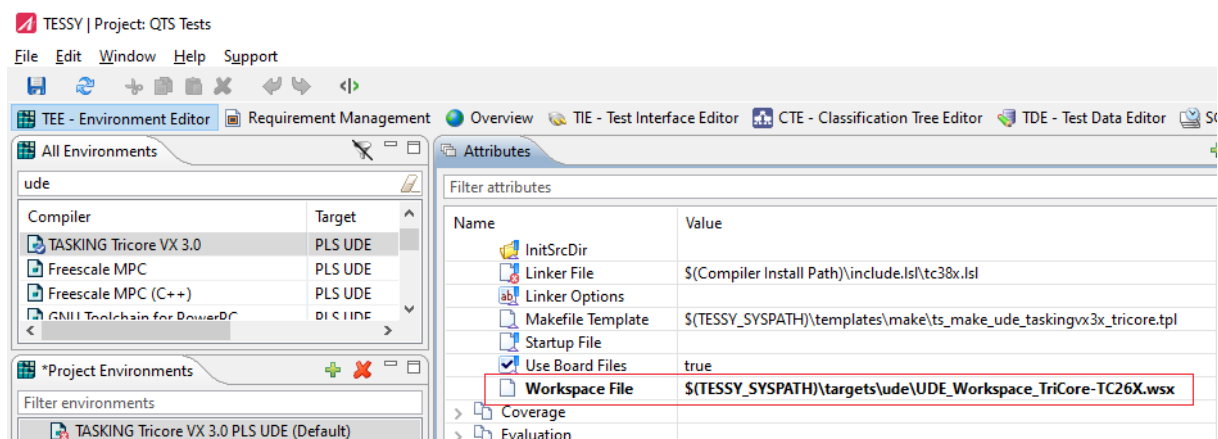


If no UDE workspace file is given, the UDE will create one by default based on the given UDE configuration file. *This is the strongly recommended procedure!*

But if you want to use your own UDE development workspace file with TESSY, you need to provide the UDE workspace file within the TEE attribute **Workspace File**. If the TEE attribute is missing within TEE, create a new attribute within the **PLS UDE** section using the context menu entry **New Attribute** as shown below.



Make sure the type and flags for your new attribute are correct. Click **OK** and select the respective workspace file by double clicking within the **Value** column of the **Attributes** pane or click **Edit Attribute Value** from the attribute's context menu.



Since TESSY version 4.2.11 the communication runs via a separate daemon process. The daemon is run by the **Expert Mode** (*Options->Expert Mode*) attribute **TESSYTHCmd**. If you need to use a UDE workspace file you will have to add the attribute to the end of the command as shown below.

```
ab TESSYTHCmd $(TESSY_SYSPATH)\targets\ude\ts_uded.exe $(Target Install Path) $(UDE Config File) $(Workspace File) ←
```

**Please note:** If the **Workspace File** attribute points to a valid workspace file path the contents of the **UDE Config File** attribute is ignored, but has to be given anyway.

## 2.3 Specifying the Startup Code (e.g. for TASKING compiler)

If you are using specific target hardware, you will probably also need to provide special startup code (`cstart.obj`) which is needed to initialize your target hardware properly for use with TESSY. To avoid all dependencies to other modules of your project, make a copy of your `cstart.obj` and strip it off to the basic initialization needed to initialize the communication with UDE and **disable all watchdog timers**. TESSY will not service any watchdog timers during the test run.

In case of the TASKING compiler, you may configure all required settings within the TASKING EDE, generate a startup code file, and let the TEE attribute **Startup File** point to its location. Please refer to the application note of the TASKING compiler for details.

## 2.4 Locator Settings

It may also be necessary to adapt the memory location settings within the Makefile template or within the linker file. Please review the settings and change the template accordingly.

In case of the TASKING compiler, you may configure all required linker and locator settings within the TASKING EDE and provide the respective LSL file within the TEE attribute **Linker File**. Please refer to the application note of the TASKING compiler for details.

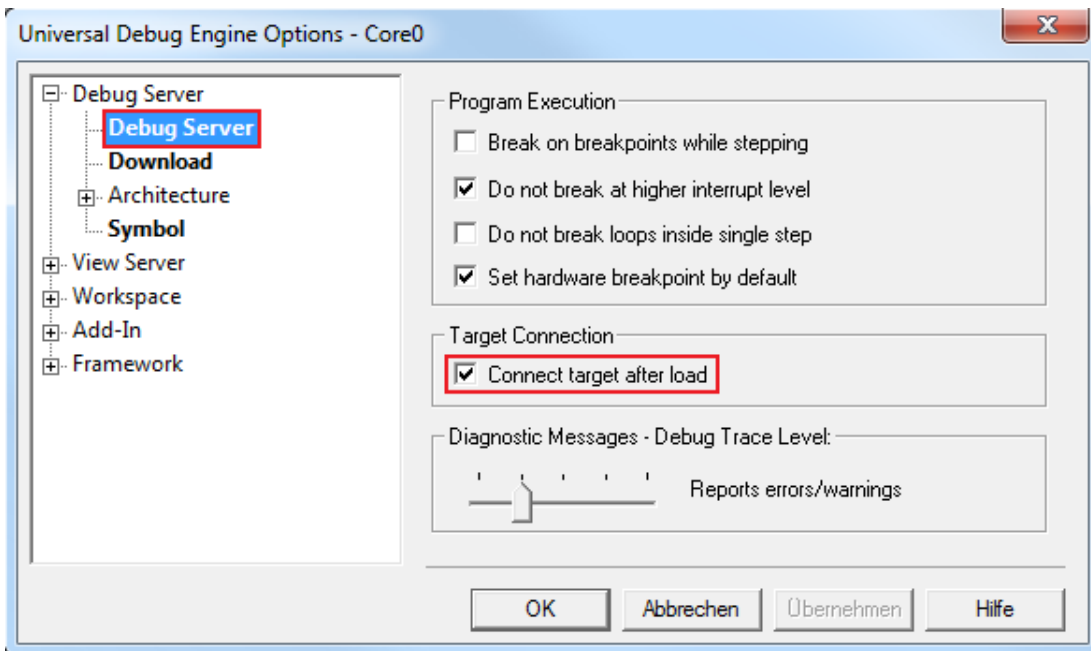
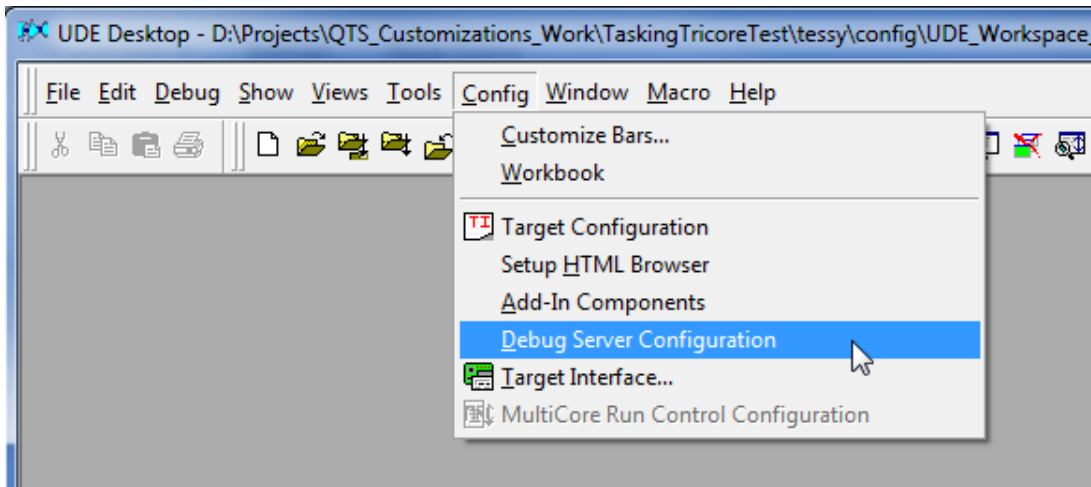
# 3 PLS UDE

The communication between TESSY and UDE is based on the COM interface of UDE. If the **Workspace File** attribute is omitted or empty, UDE will automatically create a workspace file and connect to the target configuration using the file from the **UDE Config File** attribute (see 2.1 on how to change this setting). TESSY starts the UDE automatically for each test run. After the test run TESSY shuts down the UDE debugger automatically.

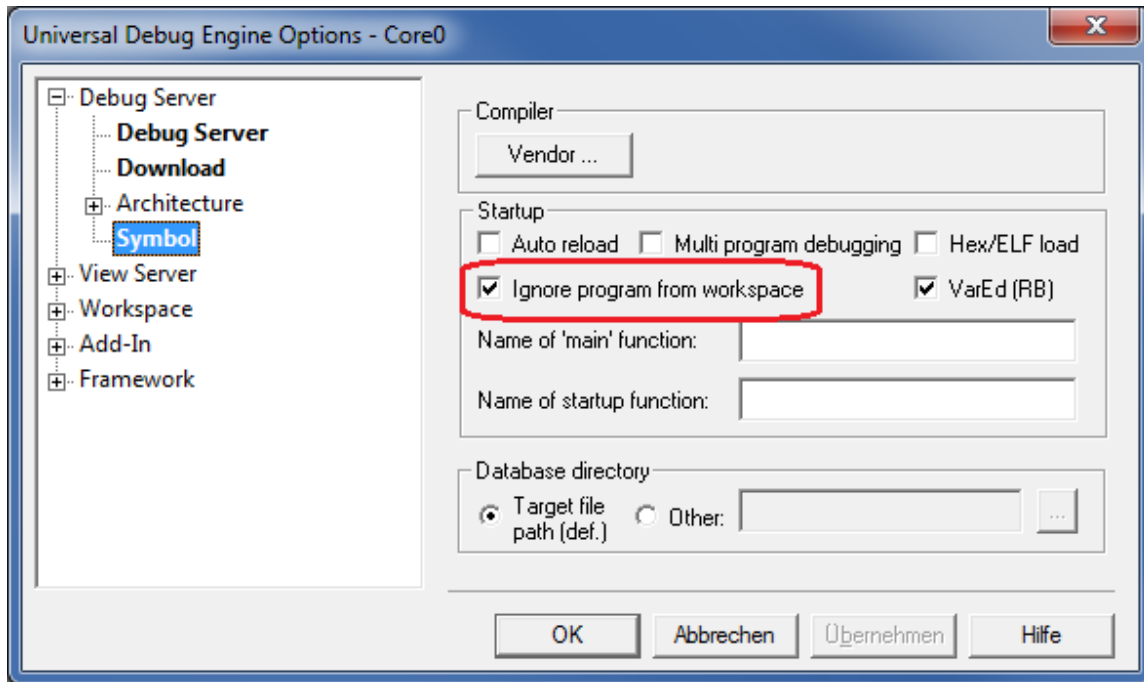
## 3.1 Workspace File Settings

If you really need to use a UDE workspace file for TESSY, it is recommended to save this file into your TESSY project's configuration folder. The following settings are required for the preselected workspace (see 2.2):

### 3.1.1 Connect target after load

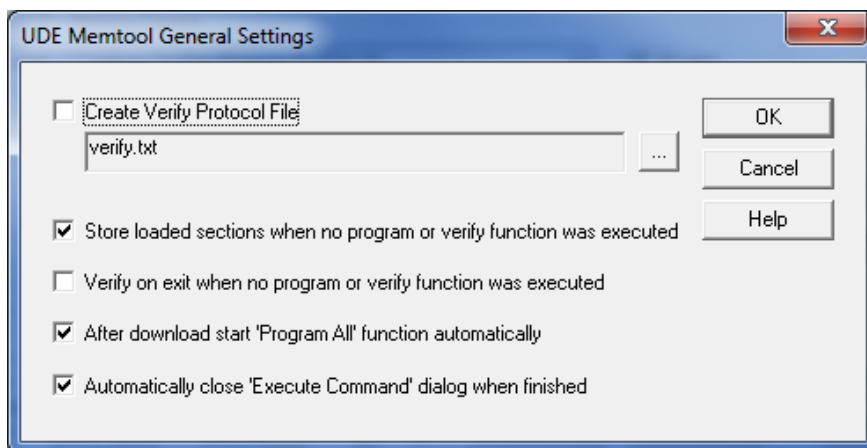
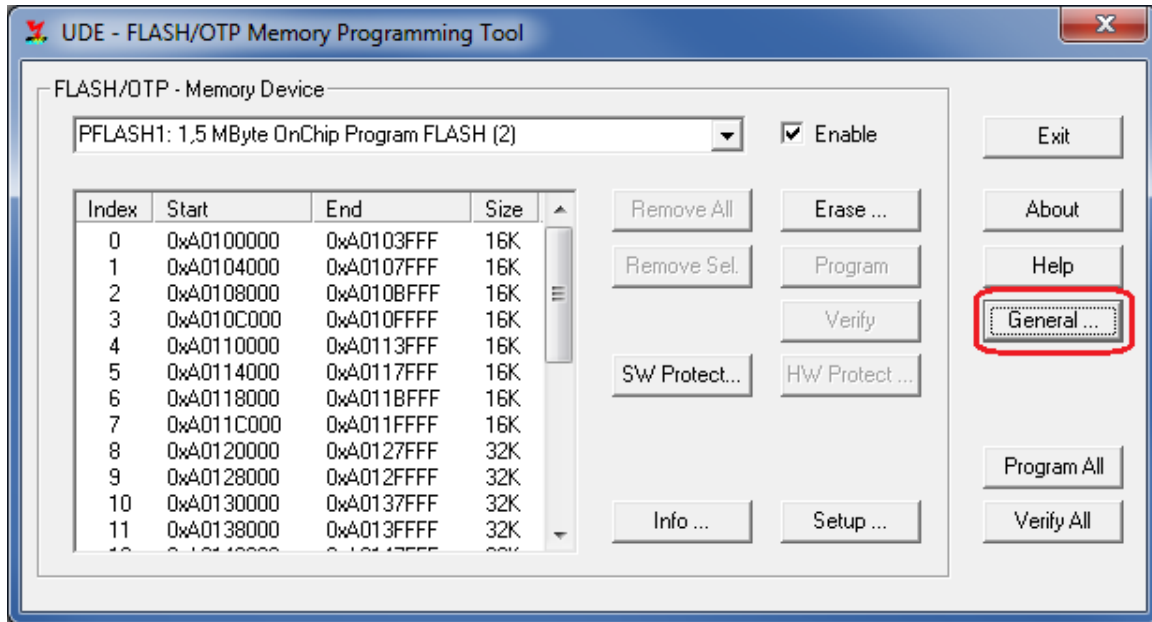


### 3.1.2 Ignore program from workspace



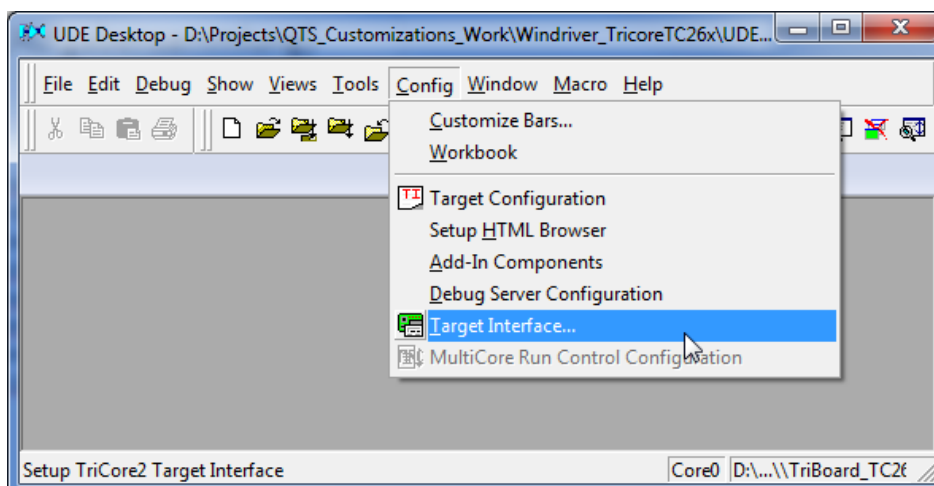
### 3.1.3 Automatically flash the binary

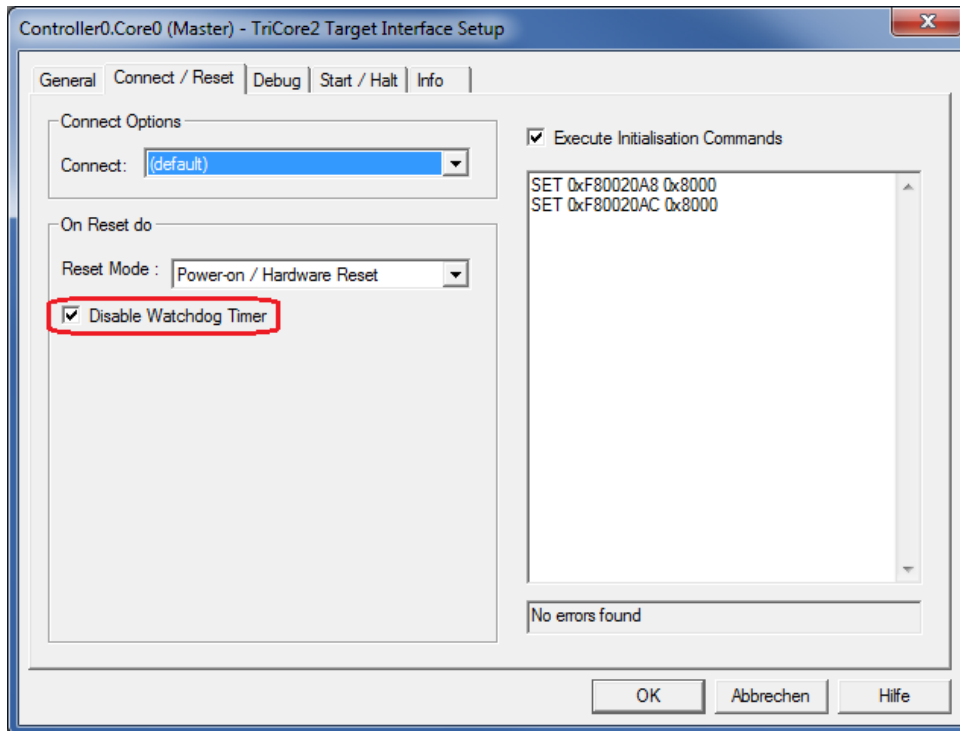




## 3.2 Configuration File Settings

### 3.2.1 Disable Watchdog Timer





It is also possible to add the corresponding connect option **DiswdtOnReset** to *suspend watchdog timers permanently* to your PLS UDE configuration file, e.g.

```
[Controller0.Core0.Tc2CoreTargIntf]
DiswdtOnReset=1
```

### 3.2.2 Using a specific UDE version

Assuming you have multiple version of UDE installed on your PC, it is possible to choose a specific version by copying file

```
TESSY_INSTALL_PATH\sys\targets\ude\AtsUdeInterface.ini
```

into folder `C:\Users\Public\Documents\pls\AtsUdeInterface.ini` and adapting the following line.

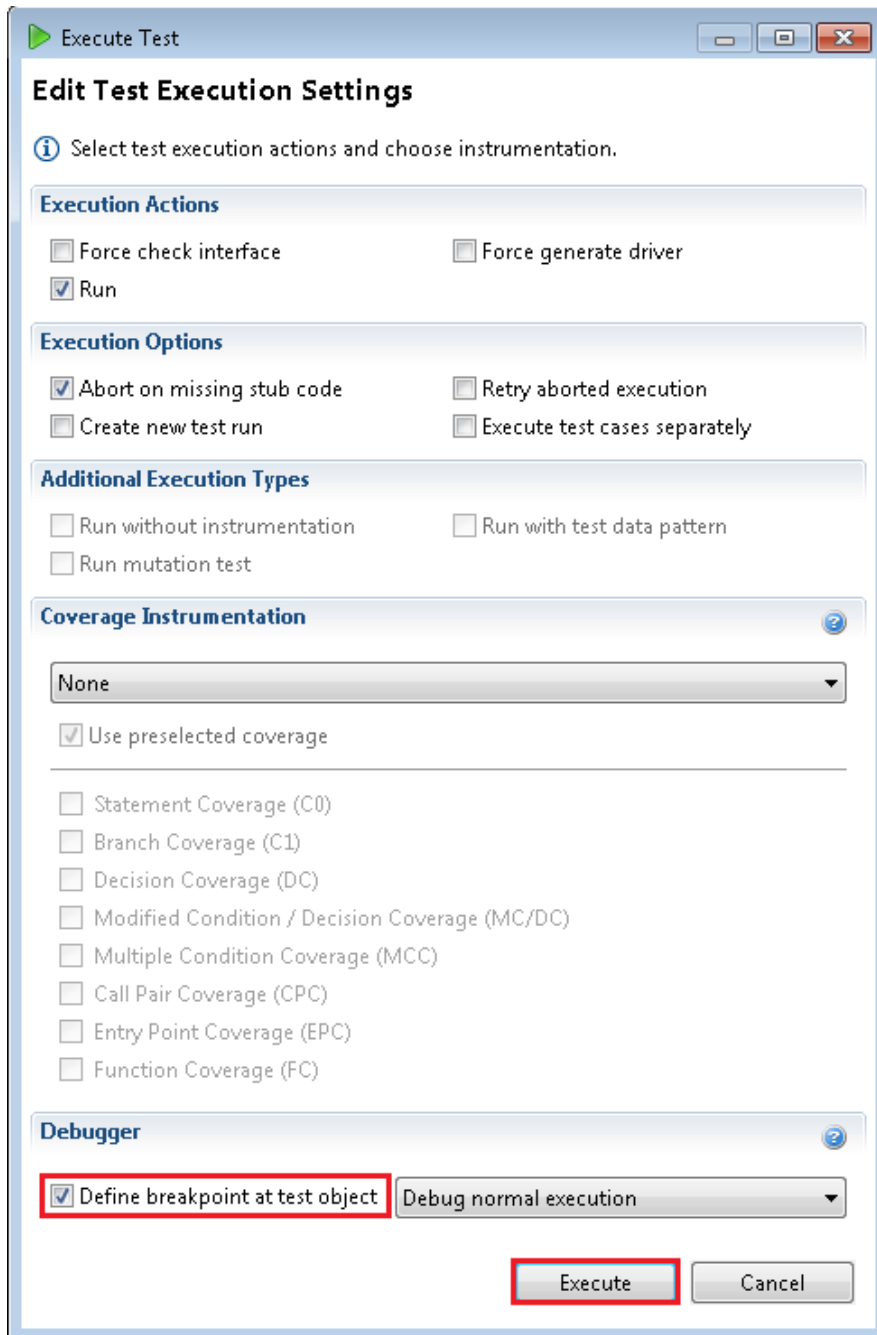
```
;VersionToStart = "2022"
```

to match the desired version for, e.g.

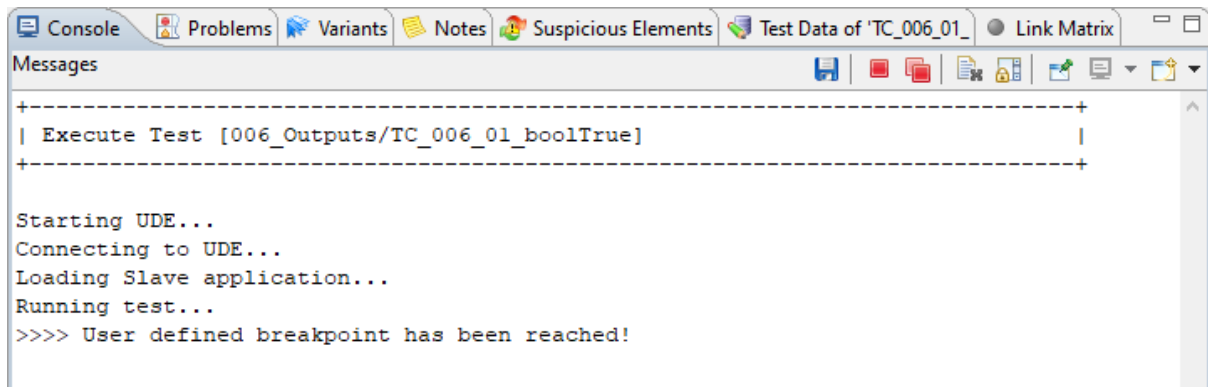
```
VersionToStart = "5.2"
```

## 4 Debugging the Test Object

If you want to step through the test object code with the test data provided by TESSY, you may add a breakpoint at the start of the test run.



When the test object breakpoint has been reached, the following message will be displayed within the TESSY message window:



You may now switch to the UDE desktop and step through your test object. Press the **Start Program** or **F5** button within UDE to resume execution (or run to the next test step, i.e. the next call of the test object).

## 5 Visibility of the UDE

By default the UDE is started by TESSY's `ts_uded` daemon. The daemon starts the UDE always hidden. So the UDE will not be visible during test execution. In case of interactive debugging the master starts the UDE on its own in visible mode automatically.

If you need to watch the UDE during normal test runs, you will need to set the **Expert Mode** attribute **Hide IDE** to *false* and set **Expert Mode** attribute **Execution Mode** to `0x71`. Now the master starts the UDE in visible mode for each test object being tested. Since this prolongs the execution time significantly, it is not recommended to use it by default. For normal test runs, i.e. **Execution Mode** is set to `0x61`, **Hide IDE** is ignored.

## 6 Troubleshooting

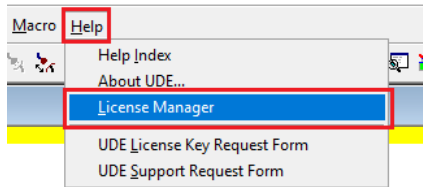
If the memory layout does not fit to your target requirements, or the wrong startup code was provided, the test execution within UDE may stop due to certain hardware traps of the microcontroller. In this case you will probably see the same printout as for the reached test object, even though you did not select the "Define Breakpoint" toggle in the test execution dialog.

In this case, close the UDE and make sure, that no more UDE process is left running (using the task manager). If the UDE could not be closed down properly, any new test run will fail until the old process is completely stopped.

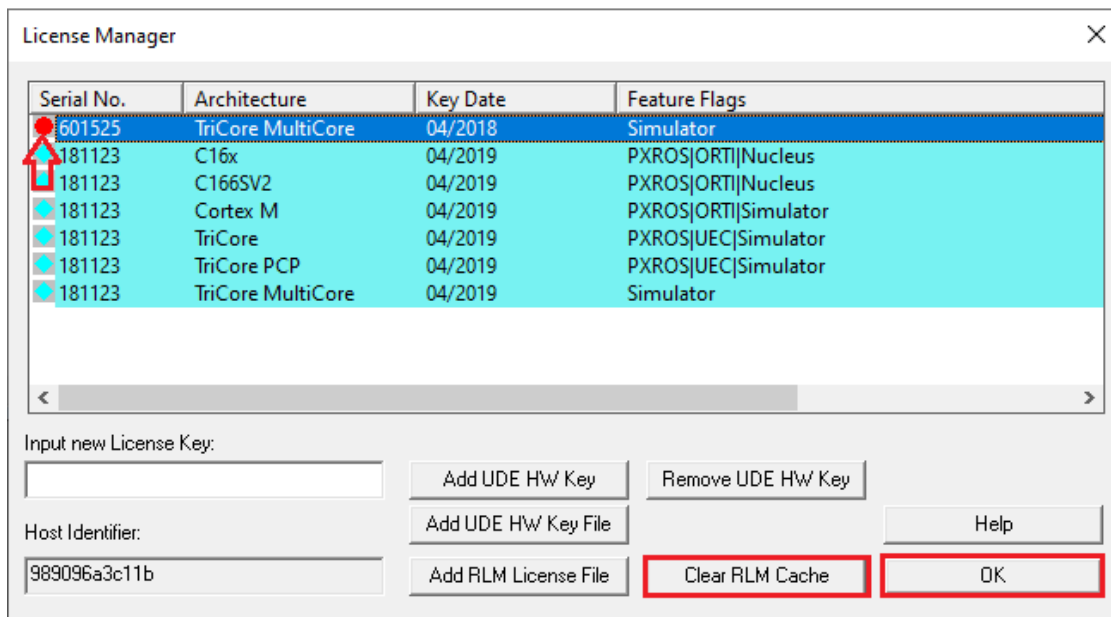
### 6.1 The UDE Starts Up Very Slowly

You may have to remove all breakpoints from your UDE workspace file (See 2.2).

Please make sure that there are no outdated licenses listed within the PLS UDE License Manager.



Red dots denote an outdated or invalid entry. Click **Clear RLM Cache** to remove those entries automatically and click **OK**.



## 6.2 TASKING C166, Version 8.0

The default cstart code provided with the libraries will not work correctly with the PLS target communication boards. Please create and use your own cstart.obj as described above.

## 6.3 TASKING Tricore VX, Version 2.0 and later

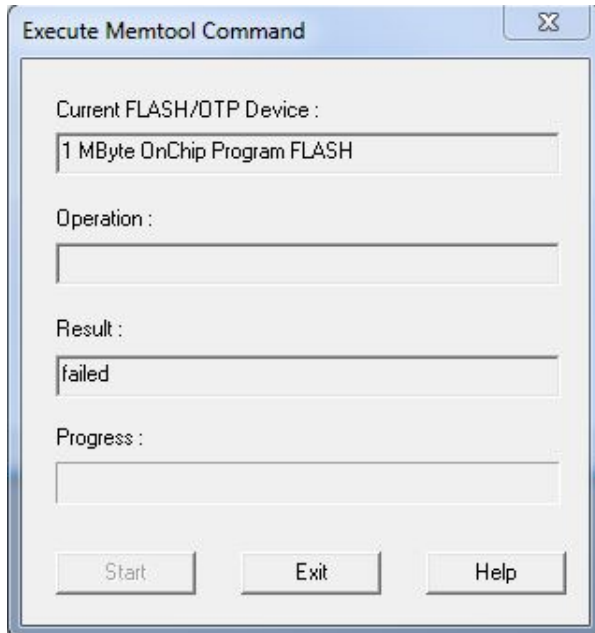
The TASKING VX compiler tool chain allows to configure the startup configuration and memory layouts by using the TASKING EDE. To change any settings for a specific Tricore target, modify the sample TESSY TASKING VX Tricore project accordingly or refer to the application note on the TASKING compiler.

## 6.4 Flashing Problems

The flashing may sometimes cause several error messages.

### 6.4.1 TC26xx/TC27xx

If you encounter the following error message



you might have to erase the flash memory as shown below. Open the FLASH/OTP dialog and select **Erase** for the flash memory sections PFLASH0 and PFLASH1.

