

# Using Freescale CodeWarrior

## Abstract

This document describes the usage of the Freescale/Metrowerks **CodeWarrior® Legacy Suites** (CW) for HC(S)08/HC(S)12/S12X/DSP568E . For newer versions of CodeWarrior see application note *Using Freescale CodeWarrior (Eclipse based version)*.

The default settings delivered with the TESSY installation works with the CodeWarrior standard simulator. If you want to run your tests on target hardware, please follow the steps described in this document (i.e. create a CW project und use these settings for the test).

Please note the setting for the remote connection to be used to avoid startup dialogs within CodeWarrior when executing the test (in case of DSP5xx only).

## Table of contents

Abstract .....	1
1 Introduction.....	3
2 Emulator/debugger setup for HC(S)08/HC(S)12/S12X .....	3
2.1 Sample TESSY project.....	3
2.2 Sample CodeWarrior project .....	5
2.3 Startup code.....	6
2.4 Using the linker file (PRM).....	7
2.5 Adaptation of the makefile template .....	9
2.6 Target connection settings .....	10
2.6.1 CW project path .....	10
2.6.2 CW debugger project file.....	10
2.6.3 Target interface .....	11
2.6.4 Adapting the slave call .....	12
3 Simulator execution for HC(S)08/HC(S)12/S12X .....	13
3.1 Setting the target interface .....	13
3.2 Setting the CPU derivative .....	13
3.3 Troubleshooting.....	15
4 DSP5xx .....	15
4.1 Setting the remote connection.....	16
4.2 Restrictions of the DSP568E controller .....	18

## 1 Introduction

The Freescale/Metrowerks CodeWarrior (CW) simulator debugger does not require a special setup in order to execute tests with TESSY (except one setting for DSP5xx like described below). If you are using the standard TESSY installation settings, the standard CW simulator will be used for test execution.

TESSY controls the test execution using a generated script file that will be passed on the command line when starting the CW debugger (`hiwave.exe`). The debugger executes this script during the test session. The default script is appropriate for use with the simulator.

In order to run tests on target hardware, you need to prepare a CW project and make some changes to the default TESSY setup. For this purpose, there are TESSY sample projects available as zip files, which may be used as starting point for adaptations to the CW target. Please follow the steps described in chapter 2 in order to setup a running TESSY project with the CW debugger.

## 2 Emulator/debugger setup for HC(S)08/HC(S)12/S12X

The following steps are required to create a working setup for test execution on target hardware connected to an emulator or debugger. You should be familiar with the CW IDE and debugger settings.

### 2.1 Sample TESSY project

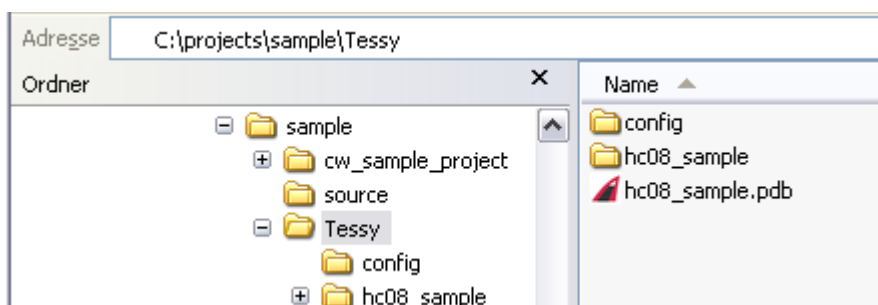
There are sample projects available (as zip files) for various controllers and also one for HC08 and another for the HC12 family of microcontrollers on the following web site:

<https://www.razorcat.com/en/tessy-sample-projects.html>

We refer to the HC08 project in the rest of this chapter.

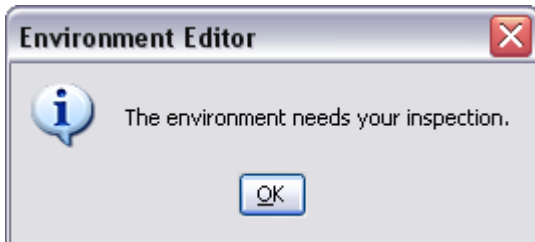
Please unzip one of the sample project files to any location on your local disk (e.g. `C:\projects\sample`). The sample project has the following structure:

- a folder with the CW sample project
- a folder with the source file(s)
- a folder with the TESSY sample project

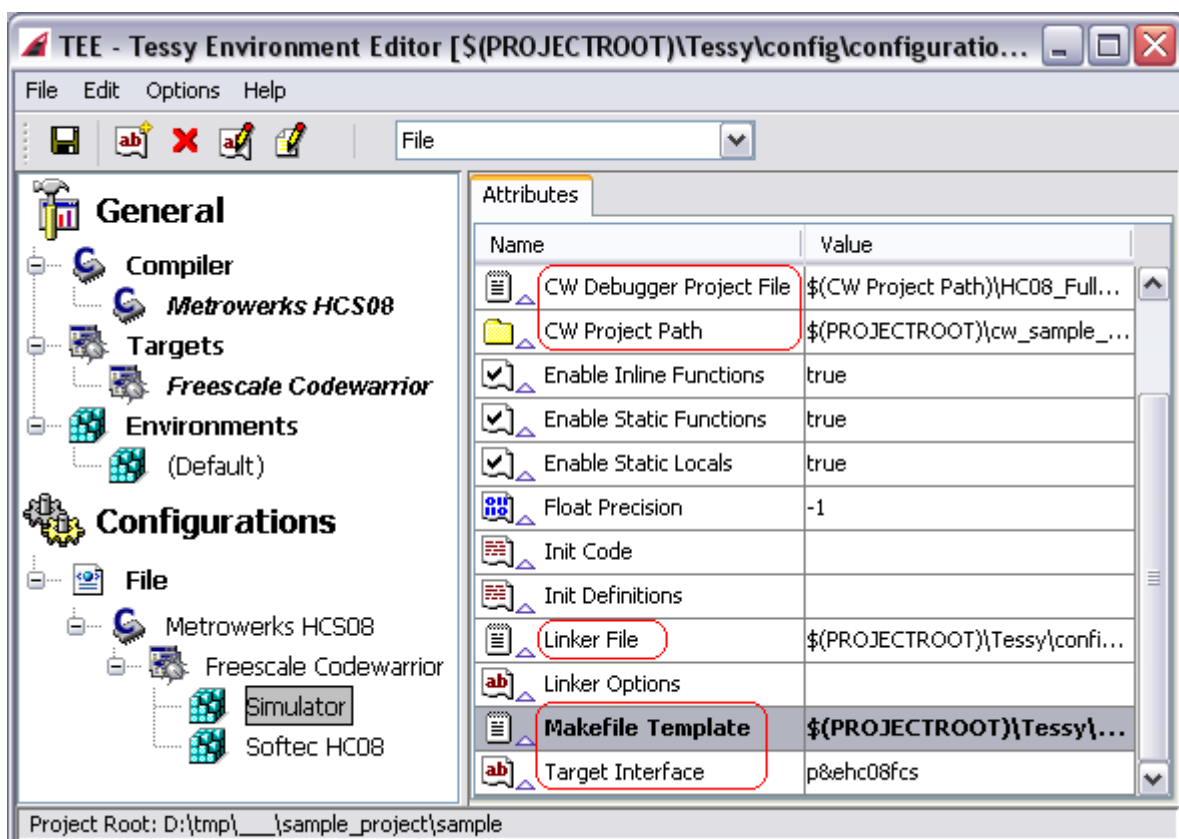


Please open the PDBX file (e.g. `C:\projects\sample\Tessy\hc08_sample.pdbx`) with TESSY.

Afterwards TESSY may ask you to inspect the TEE settings (e.g. if the installation path of CodeWarrior is wrong or missing).



Please add all necessary settings, save and exit TEE. The sample TESSY project uses a configuration file and you should do any changes within the **File** section of TEE. The picture below shows the attributes that are added or changed compared to the default configuration:

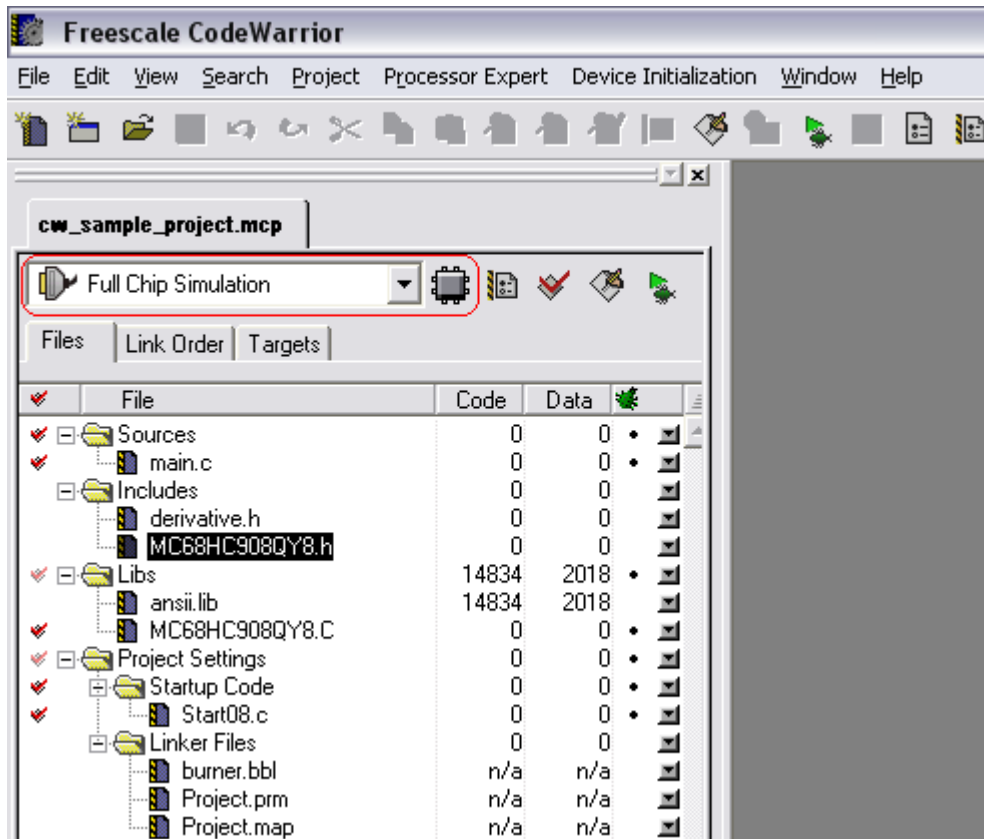


These attributes are related to the sample CW project included in this sample setup. Refer to the following sections for details about the required settings.

## 2.2 Sample CodeWarrior project

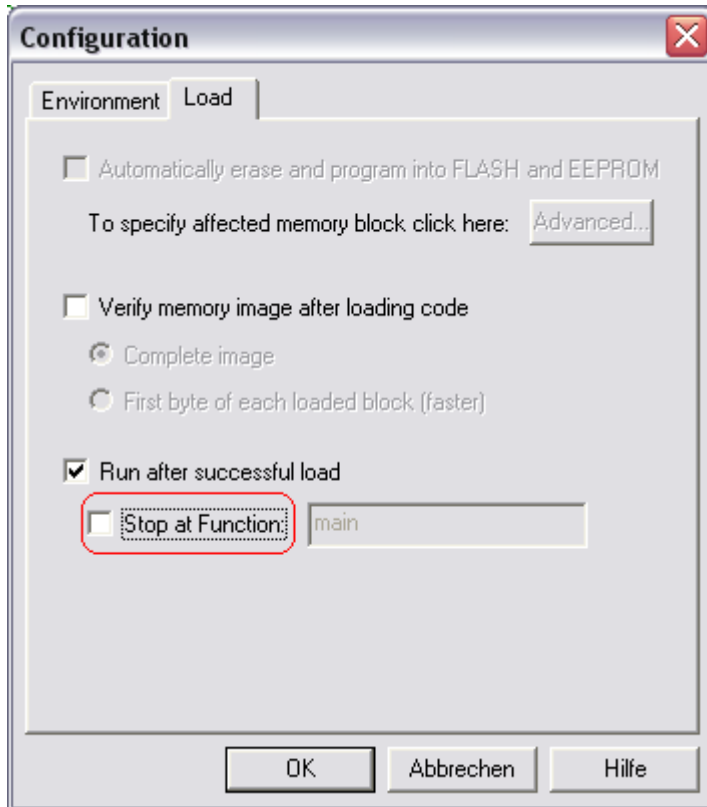
The sample project also contains a sample CW project. You may either use this CW sample project or create a new CW project using the CW project wizard. Just make sure to create any new project as a subfolder of the PROJECTROOT folder.

We will continue by opening the existing sample CW project. The picture below shows the sample CW project within CodeWarrior:



Choose the appropriate MCU and target connection (marked in red) and build the sample CW project in order to generate the PRM and object files used later within the TESSY sample project.

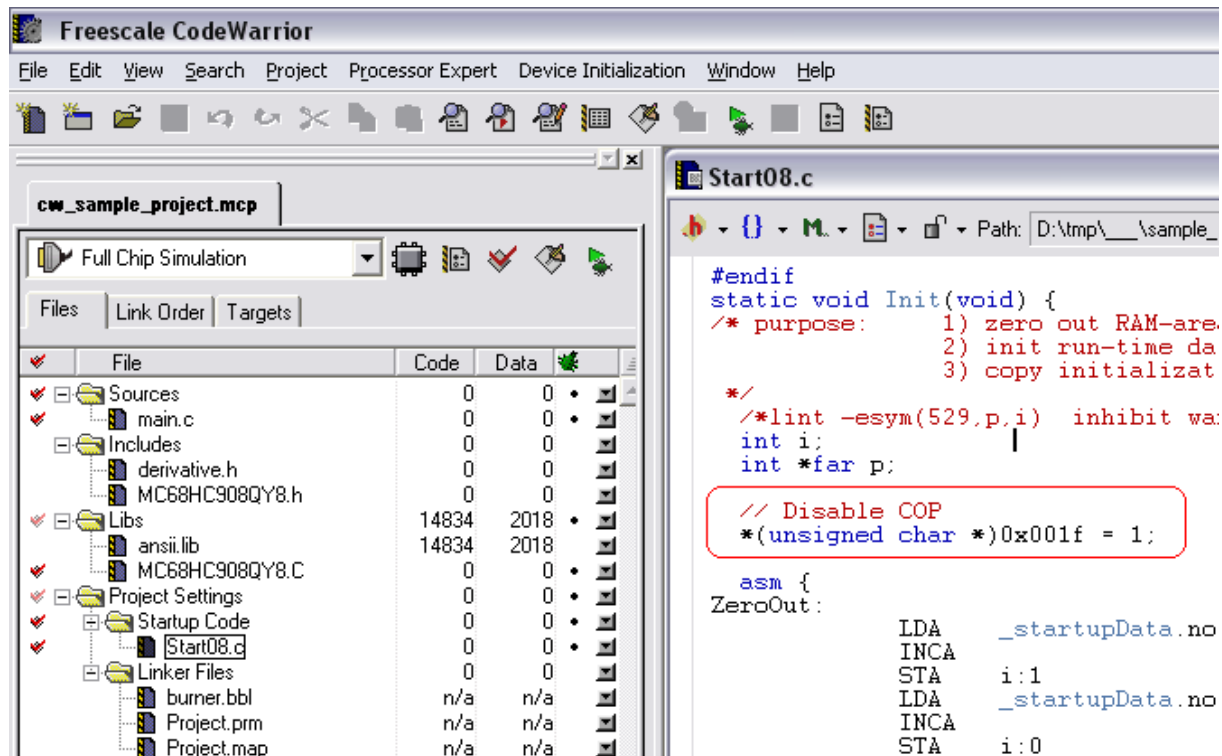
If everything builds correctly, please start the debugger from within CodeWarrior by pressing the **Debug** button (F5). This will start the Hiwave debugger and you should be able to step through the code. There is one more setting required for automated tests by TESSY: Select **Configuration** from the **File** menu of Hiwave. This will show the **Configuration** dialog:



Please make sure that the **Stop at Function** toggle within the **Load** tab of the dialog is not selected like shown above. Click on the **OK** button and exit Hiwave.

### 2.3 Startup code

By default, TESSY uses the standard startup code provided within the CodeWarrior libraries. If you need to have another startup code (e.g. to disable the watchdog like described below or in order to initialize processor I/O registers), you need to change the TESSY makefile template like described in section 2.5.

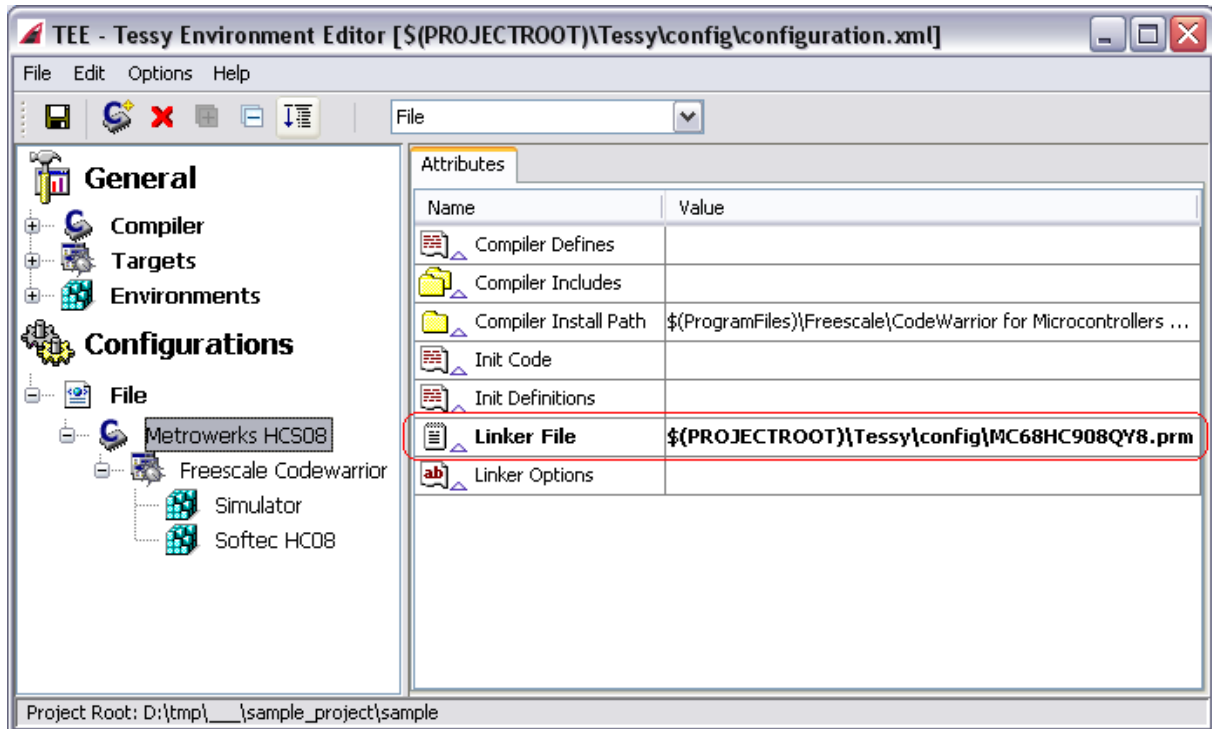


The red marked program code within the `start08.c` is necessary for the HC08 derivatives to disable the watchdog timer (COP). Please refer to the respective microcontroller specific header file or CW documentation for further information about resetting the watchdog (e.g. the location of the CONFIG1 register is different for the HCS08 devices).

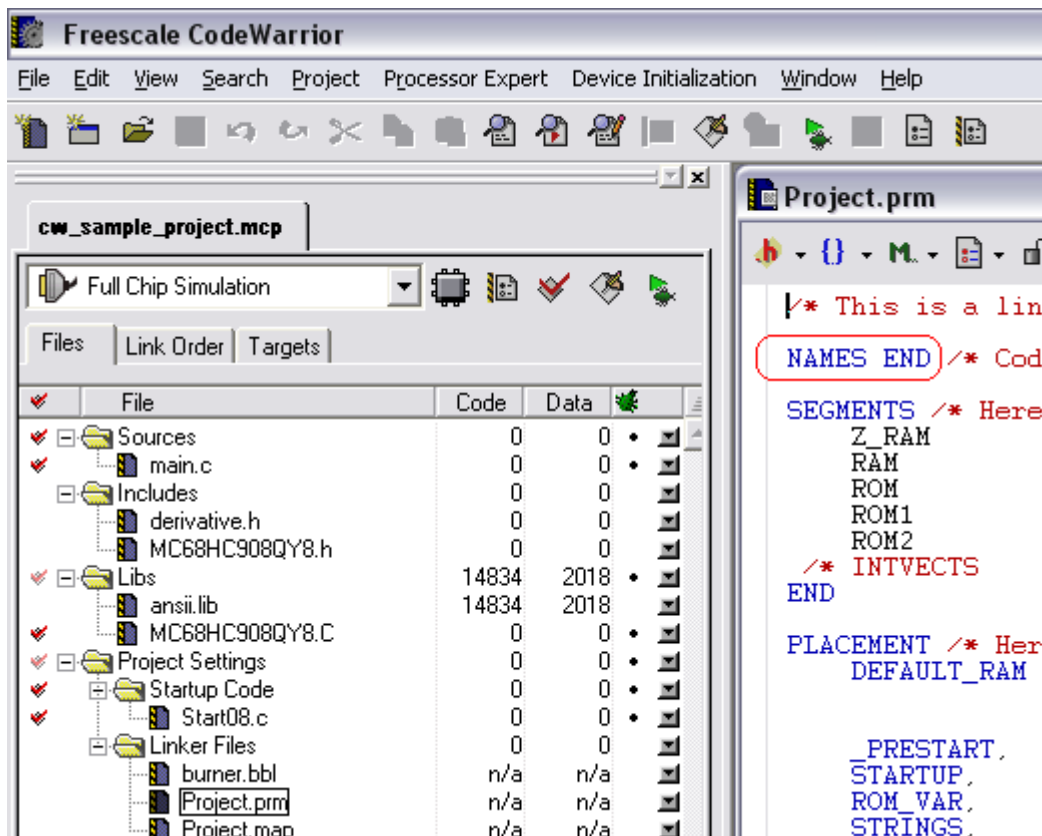
The device selected here has such a short COP timeout, that it is required to disable the COP as soon as possible. Doing this within the `main()` program is too late for this device! For other devices, you may use the **Init Code** attribute within TEE to specify a code fragment to disable the watchdog.

## 2.4 Using the linker file (PRM)

You may use the generated linker file (`project.prm`) from the CW project for testing. Please copy this file into another location and reference the copied file within the TEE attribute **Linker File**. The sample TESSY project already contains the copied and renamed PRM file from the sample CW project within the `config` subdirectory as shown within TEE:



Please note, that you need to remove the NAMES ... END entry within the CW generated PRM file as shown below (because TESSY generates this entry itself):



Within the TESSY sample project, this line is already commented out.

Another important setting is the stack size. You should increase the default size (you may use e.g. 0x200) to make sure there is enough stack space available:

```
34 STACKSIZE 0x200
35
36 VECTOR 0 _Startup /* Reset vector: this is the default entry point for an application. */
37
```

## 2.5 Adaptation of the makefile template

The TESSY makefile template may need adaptation in the following cases:

- using another startup code
- changing the memory model
- using other C libraries (e.g. due to memory model changes)

In the TESSY sample project, the startup code object file has been changed in order to use the compiled `start08.c` of the sample CW project (to disable the COP watchdog timer) as shown below:

```
117 @echo *
118 @echo ***** Linking Slave *****
119 @echo LINK $@ > $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
120 @echo NAMES >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
121 @echo $(MODULE_PATH_DOS)\ts_$(TESTOBJECT)_s$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJEC
122 @sh -c 'for f in $(S_SRC_OBJECTS); do echo "$$f" >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm; done
123 @echo $(S_UC_OBJECT) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
124 @echo $(S_STUB_OBJECT) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
125 @sh -c 'for f in $(S_C1_OBJECTS); do echo "$$f" >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm; done
126 @echo $(MODULE_PATH_DOS)\tslows_$(IDE_NAME)$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJEC
127 @echo $(MODULE_PATH_DOS)\tstcomm$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
128 @echo $(MODULE_PATH_DOS)\tscoml$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
129 ifndef S_COM_OBJECTS
130 @echo $(MODULE_PATH_DOS)\check_driver_init$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJEC
131 @echo $(MODULE_PATH_DOS)\check_driver_layers$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJE
132 @echo $(MODULE_PATH_DOS)\check_driver_layer_one$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TESTO
133 @echo $(MODULE_PATH_DOS)\check_driver_layer_two$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TESTO
134 @echo $(MODULE_PATH_DOS)\check_driver_layer_three$(OBJECT_POSTFIX) >> $(MODULE_PATH_DOS)\ts_$(TE
135 endif
136 @echo $(METROWERKS_HOME)\lib\hc08c\lib\ansi.lib >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
137 @echo $(PROJECTROOT)\cw sample project\cw sample project Data\Standard\ObjectCode\start08.c.o >>
138 @echo END >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
139 @type $(LINKER_FILE) >> $(MODULE_PATH_DOS)\ts_$(TESTOBJECT).prm
```

Other C libraries could be used by changing the line above the reference to the startup code.

The desired memory model may be configured within the compiler options at the beginning of the makefile template:

```

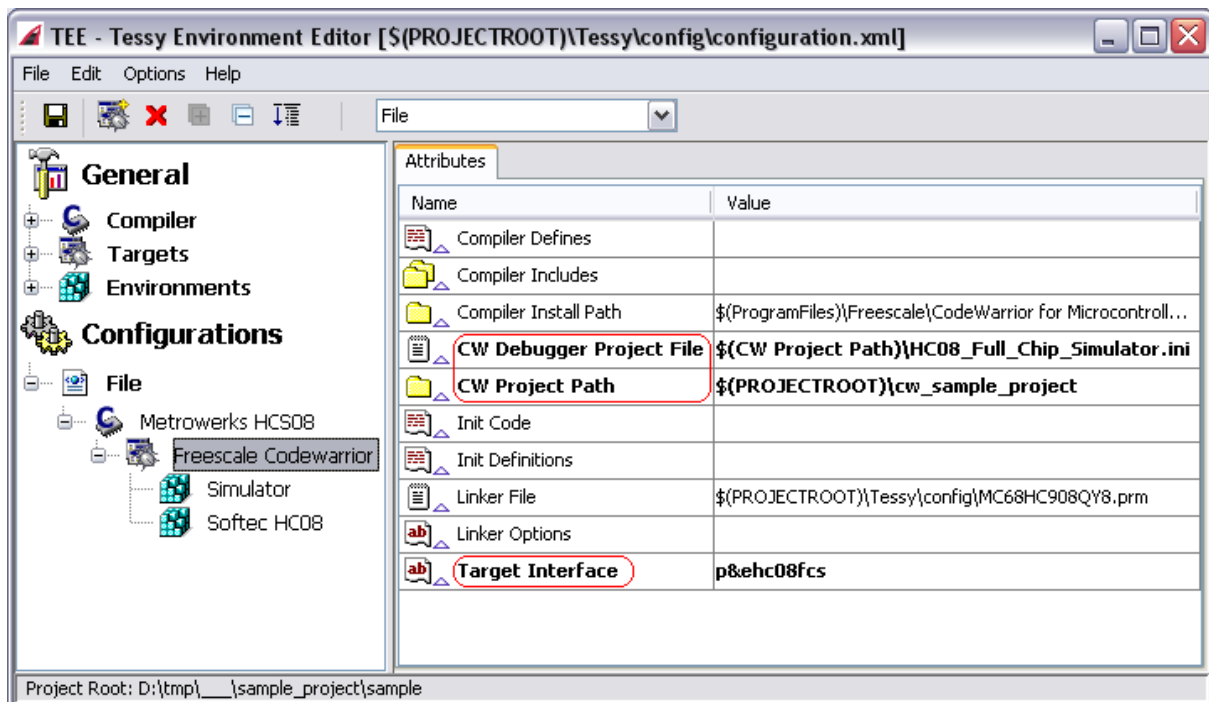
46 #
47 # SLAVE
48 #
49 S_CC           := $(METROWERKS_HOME)\prog\chc08.exe
50 S_LINK        := $(METROWERKS_HOME)\prog\linker.exe
51 S_INCLUDES    := -I$(TESSY_SYS_DOS)\include\tessy\comm -I$(MODULE_PATH_DOS)
52 S_STUB_OBJECT := $(MODULE_PATH_DOS)\ts_$(TESTOBJECT)_stubs$(OBJECT_POSTFIX)
53 S_UC_OBJECT   := $(MODULE_PATH_DOS)\ts_$(TESTOBJECT)_usr$(OBJECT_POSTFIX)
54 S_COMP_OPTIONS := -W2 -WmsgNu=abcde -Cppc -F2 -Ms -DTESSY -DTS_SLAVE -DTS_HC
55 S_LINK_OPTIONS := -W1 -WmsgNu=abcde -M

```

The default memory model for the HC08 devices is the small model.

## 2.6 Target connection settings

Within the TESSY sample project, there are some additional attributes that refer to the sample CW project location. Please make sure that the following attributes refer to the appropriate directory, file and connection name:



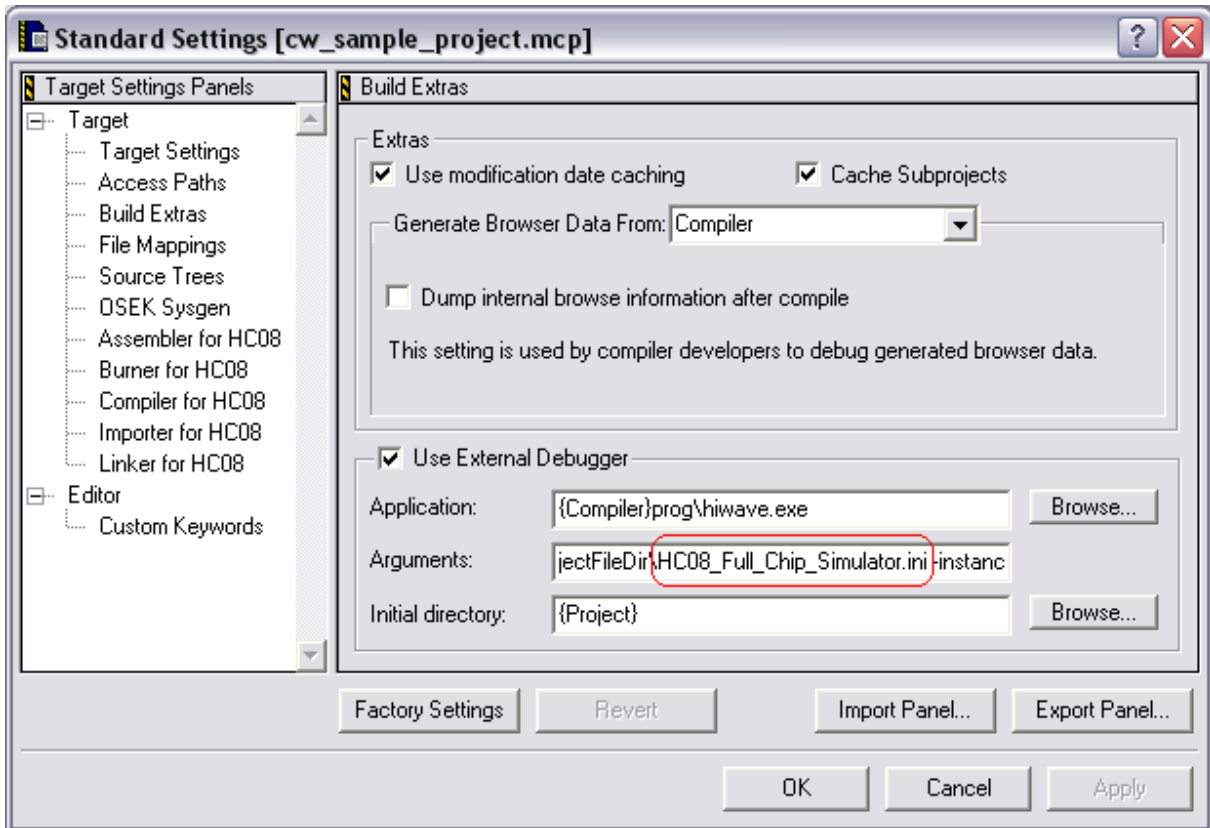
### 2.6.1 CW project path

This is the path to the sample CW project. It should reside within the PROJECTROOT folder in order to facilitate keeping all TESSY project settings together.

### 2.6.2 CW debugger project file

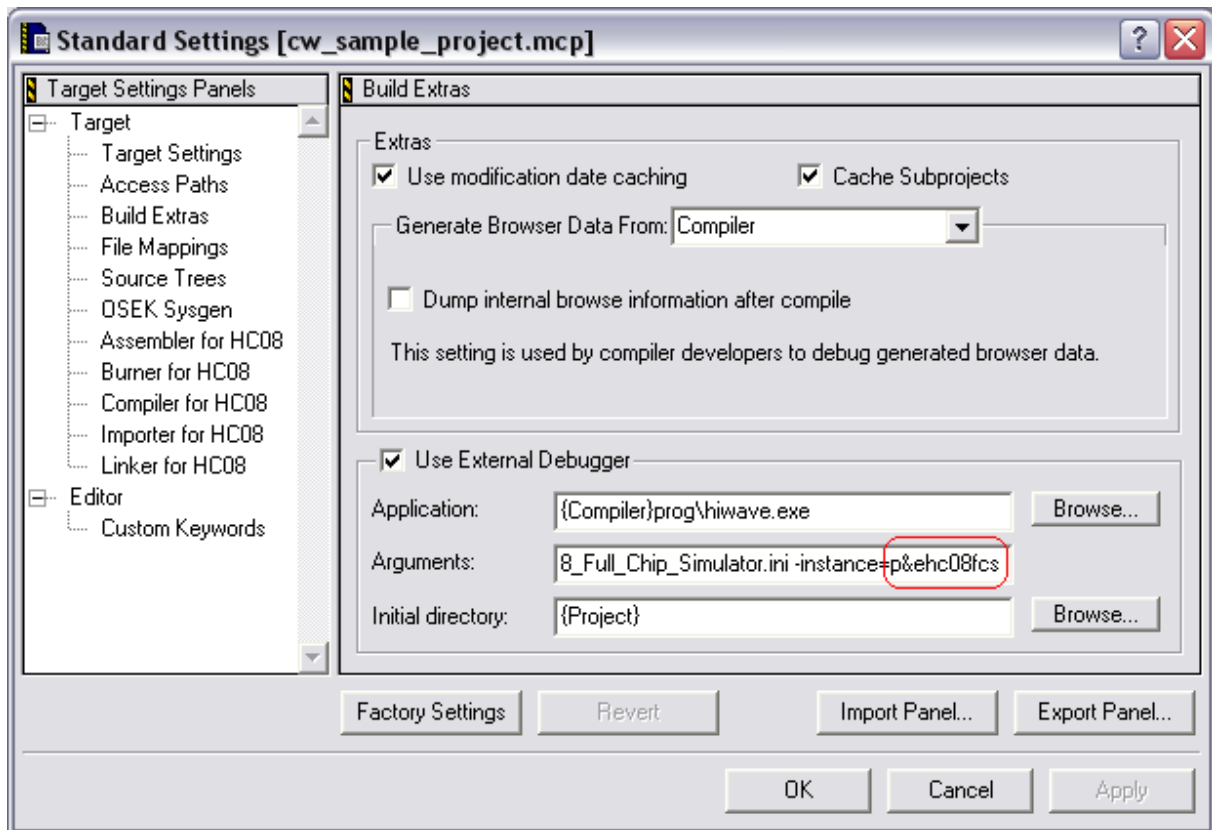
This file refers to the CW INI file for the desired target connection. It is located within the CW project path and the name depends on the target you are using with CodeWarrior. You can find the name of the file within the **Standard Settings** dialog of CodeWarrior (this page shows the arguments passed to CodeWarrior. TESSY will

use the same command line arguments when starting hiwave.exe, refer to section 2.6.4):



### 2.6.3 Target interface

You can find this setting also within the **Standard Settings** dialog of CodeWarrior:



## 2.6.4 Adapting the slave call

The **Slave Call** attribute needs adjustment, if you are using the full chip simulation or any attached hardware. The arguments to the CodeWarrior program (`hiwave.exe`) shown within the **Standard Settings** dialog need to be used for the **Slave Call** also (i.e. the `-Prod` and `-instance` arguments). Below is the CW debugger argument list for the full chip simulation as an example:

```
%targetFilePath -Prod=%projectFileDir\HC08_Full_Chip_Simulator.ini -instance=p&ehc08fcs
```

The respective **Slave Call** attribute for this example would be like follows (all within one line, copy the whole line and paste it into TEE):

```
$(CW Project Path):@:$(Target Install Path)\prog\hiwave.exe  
$(MODULEPATH)\ts_$(TESTOBJECT)_s.abs -Prod=$(CW Debugger Project File) -instance=$(Target  
Interface) -c $(TESSY_TESTAREA)\testarea\start.cmd
```

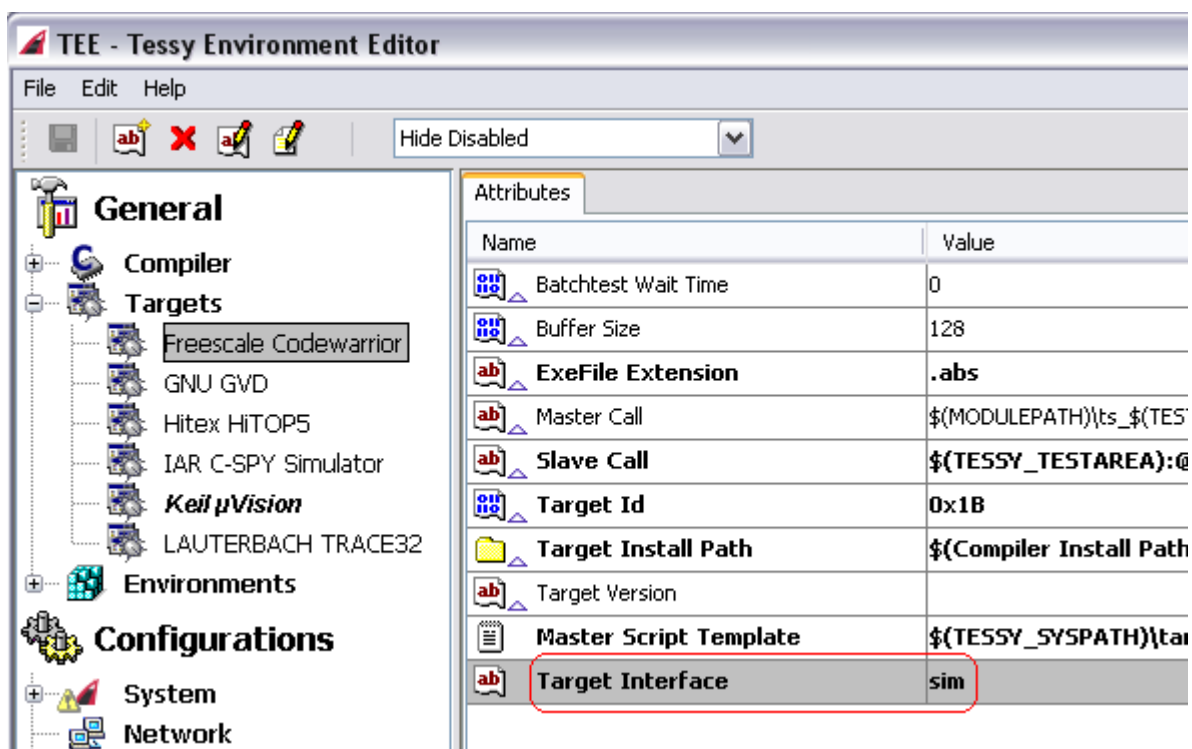
**Please note:** The default value of the **Slave Call** attribute contains the `hiwave.exe` with the `start.cmd` file as only argument. The default value will work with the standard CodeWarrior simulator only!

### 3 Simulator execution for HC(S)08/HC(S)12/S12X

This setup description is for the default simulator target of CodeWarrior. If you want to use the full chip simulation or any hardware target, please refer to chapter 2.

#### 3.1 Setting the target interface

The CW debugger target may be selected within TEE (the default target is the simulator). If you want to execute tests using debugger hardware, you should follow the steps described within chapter 2. Special handling of target hardware like flashing or hardware communication settings may not be supported with the default target setting.



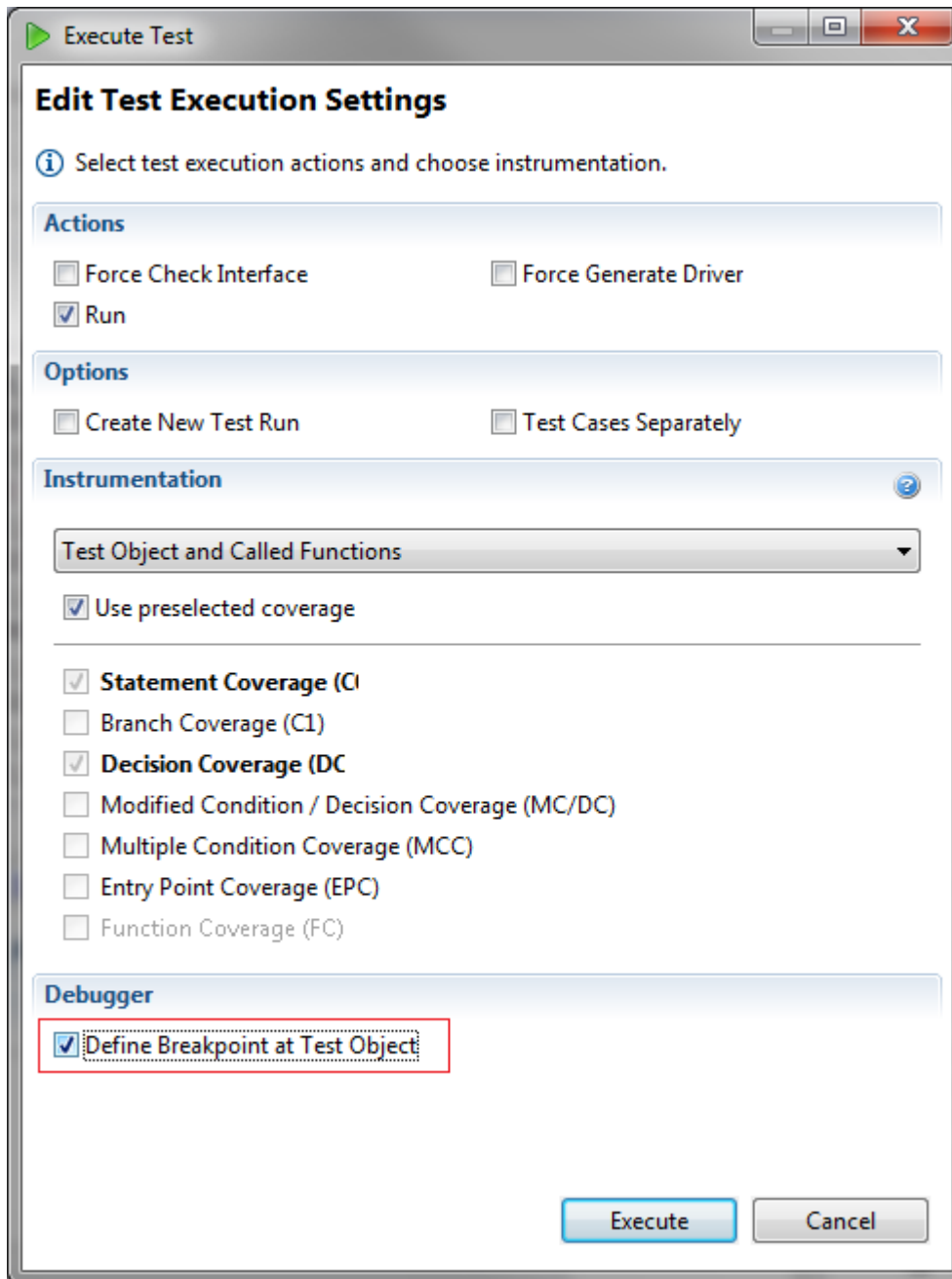
You may choose all supported targets of the Freescale Hiwave debugger. A complete list of targets may be found within the Freescale installation subdirectory:

```
C:\Program Files\Freescale\CW for HCS12X V4.6\prog
```

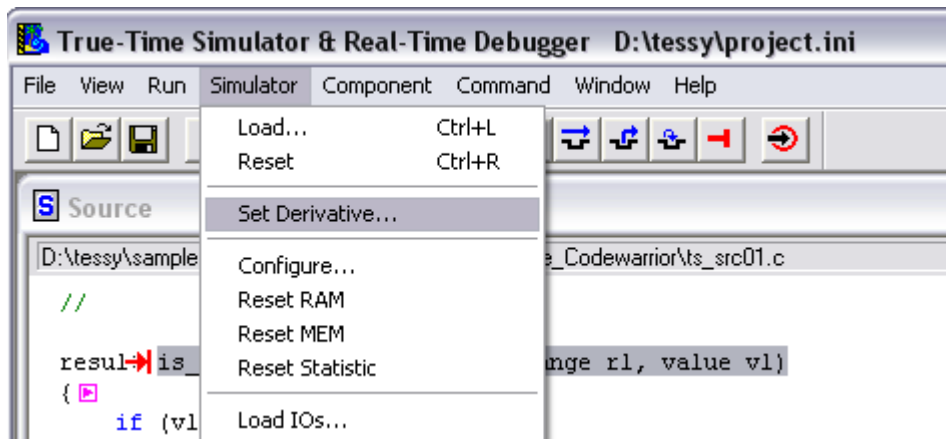
All files with the extension `.tgt` represent a possible target. You need to specify the name of the file without extension within the **Target Interface** attribute.

#### 3.2 Setting the CPU derivative

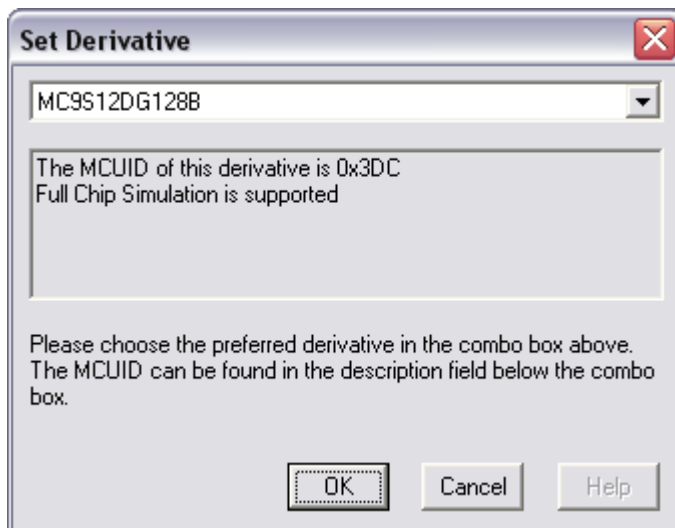
If you want to set the CPU derivative, start the test execution with the test object breakpoint set as shown below:



The CW debugger will start execution and break at the test object. Select **Set Derivative ...** from the **Simulator** menu (within the CW debugger) to choose the derivative:



Within the **Set Derivative** dialog, you may select your CPU derivative (e.g. MC9S12DG128B like shown below):



After pressing **OK**, this setting will then be active for all future test and debug sessions (in fact, the derivative setting will be stored within a `project.ini` file located within the `TESSY_TESTAREA` directory, normally `c:\tessy`. This location is the current working directory where the CW debugger `hiwave.exe` will be started).

### 3.3 Troubleshooting

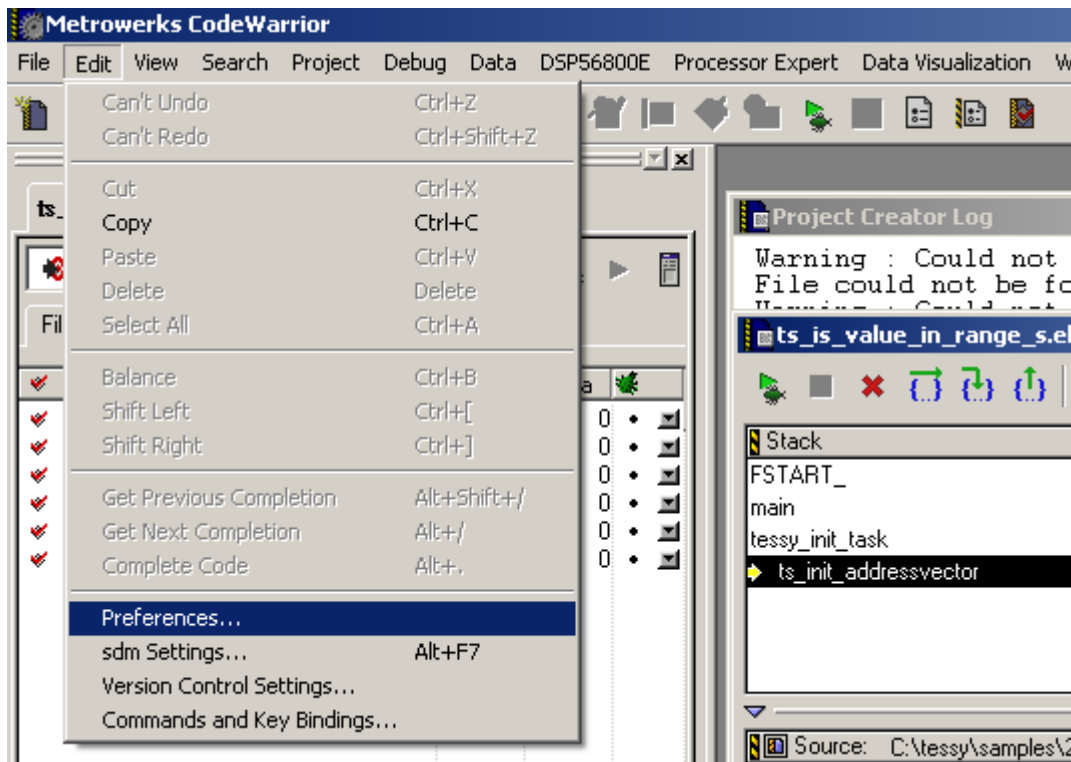
If the test fails to execute properly, please inspect the CPU derivative like described above to ensure that the correct CPU is selected.

## 4 DSP5xx

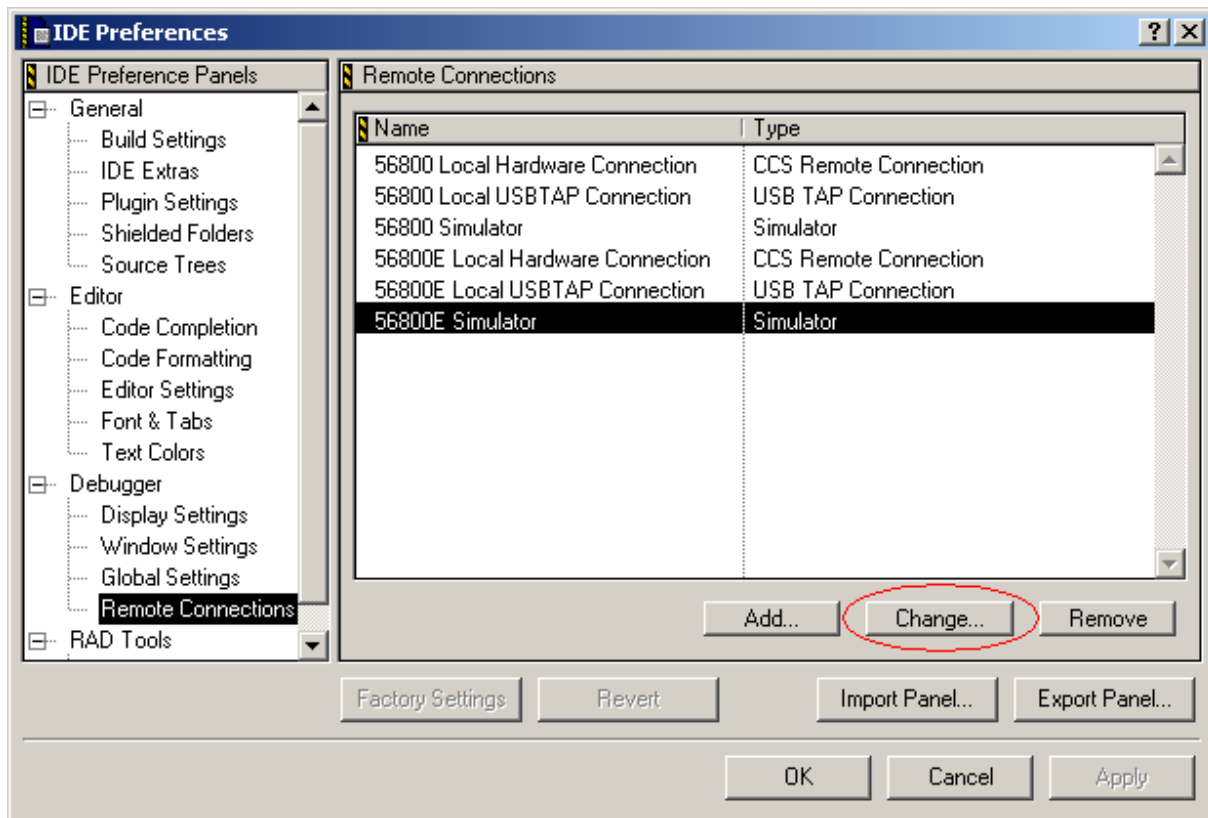
There is only one special setting required in order to execute tests within CodeWarrior IDE debugger for DSP5xx. If this setting is missing, there will be a dialog on startup of CodeWarrior asking for the desired remote connection.

## 4.1 Setting the remote connection

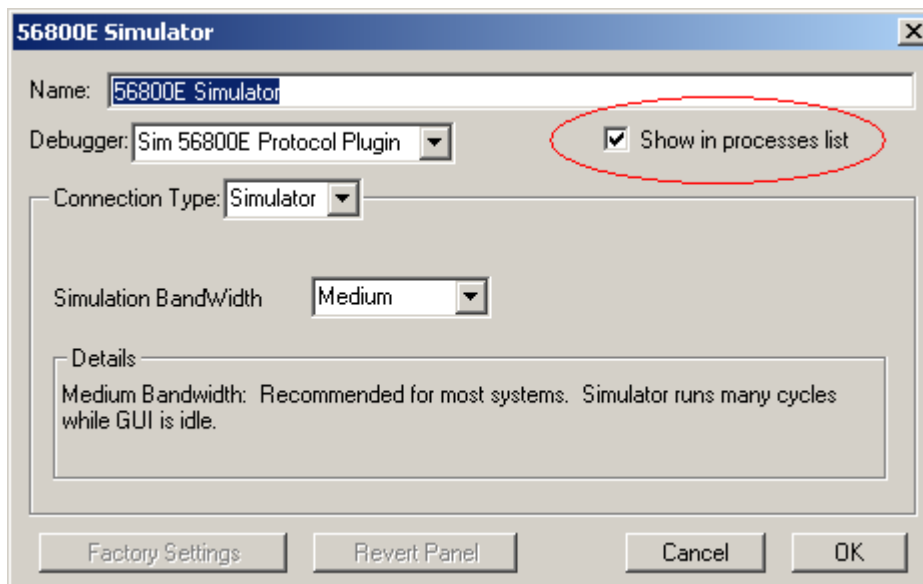
Select the **Preferences** from the **Edit** menu to open the **IDE Preferences** dialog like shown below:



Within the **IDE Preferences** dialog, select the **Remote Connections** entry within the **IDE Preferences Panel**, select the appropriate entry from the list of remote connections and click on the **Change** button.



The respective dialog for the remote connection will appear. Please make sure, that the **Show in processes list** flag is set like shown below:



## 4.2 Restrictions of the DSP568E controller

Because of the special nature of the DSP568E hybrid controller, some restrictions concerning pointer variables apply: The processor can only handle pointers to char or unsigned char correctly when assigning addresses of variables.

If the test object contains such pointers, the passing direction needs to be set to EXTERN and the pointer value need to be assigned using the usercode views within the TDE perspective.