

# Using NXP Smart Card Composer

## Abstract

This document describes the usage of the NXP Smart Card Composer as target system. The minimum tested version of the NXP Smart Card Composer GdbServer is 15.0.7.14899.

**Important Note:** You need a functional NXP Smart Card Composer project which can successfully build a target binary and launch a debug session.

## Table of Contents

Abstract .....	1
1 Introduction.....	2
2 TESSY Configuration .....	2
3 TEE Configuration .....	3
4 NXP Smart Card Composer Configuration .....	4
4.1 Create a new configuration .....	4
4.2 Add TESSY Library .....	4
4.3 Add TESSY symbol.....	5
4.4 Adapt <code>appMain.c</code> .....	6
5 Switching between Configurations .....	6
5.1 Switch to preferred next configuration .....	6
5.2 Adjust TESSY_TEST and apply Library again .....	7
5.3 Rebuild UserModeAppB again .....	8
6 Interactive Debugging.....	8
7 Troubleshooting.....	10
7.1 Function „ <code>tslows_sync</code> “ not defined .....	10
7.2 Execution gets stuck .....	11
7.3 (ERROR) [main] Invalid segment descriptor for execute.....	11

## 1 Introduction

Basically TESSY generates a library which will be linked to your own NXP Smart Card Composer project. Therefore no startup code or linker file has to be set within TESSY. The communication runs via pipes to NXP Smart Card Composer's GdbServer. TESSY parses the GdbServer's output to find the right moment when to source the monitor commands file and when to startup the simulator automatically. However, using the NXP Smart Card Composer as TESSY's target system requires some pre-adjustments to be performed. You have to

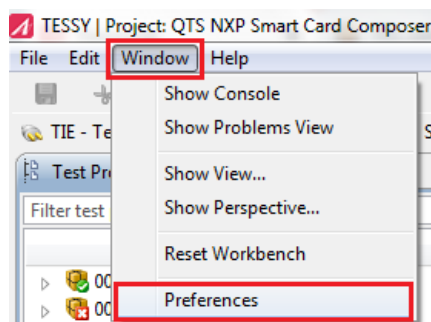
- adjust file `appMain.c` of your project and add a library.
- set the paths to the NXP installation directory and to your NXP workspace directory within TEE,
- set the SystemSDK Configuration and the monitor commands file,
- set the target device and the target driver, and
- set the path to the NXP command line simulator.

This application note describes how to find these values and where to set them.

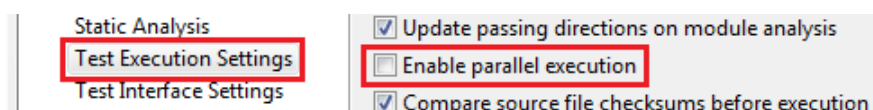
Since the GdbServer allows only one connection at a time, there is no direct interactive debugging supported for this adaption. However, there is an easy and comfortable way to debug the generated target binary having your TESSY test data built-in.

## 2 TESSY Configuration

Since the NXP built system allows only one built of the same target binary at once, you have to disable the parallel execution within the TESSY configuration. From TESSY's menu select *Window->Preferences*.



From the Preferences dialog select **Test Execution Settings** and disable **Enable parallel execution** as show below.

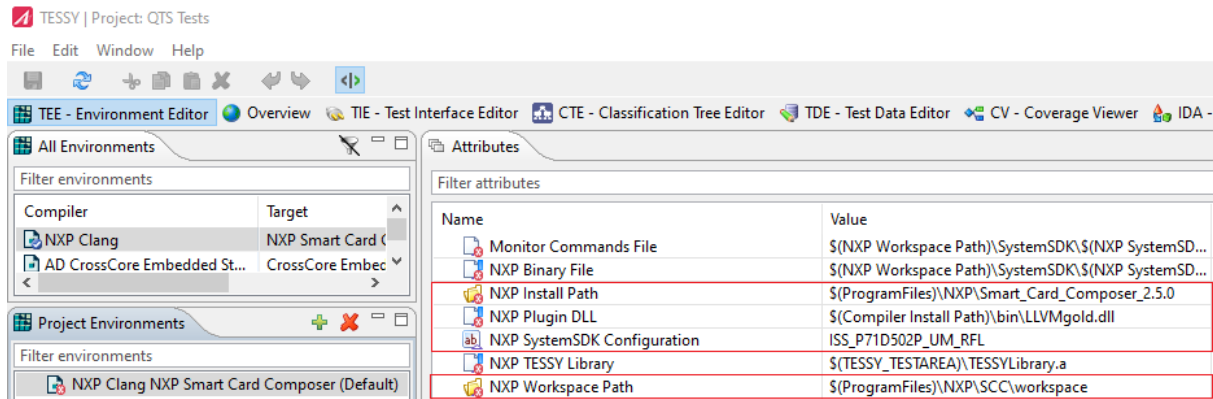


Klick **Apply** and **OK**.

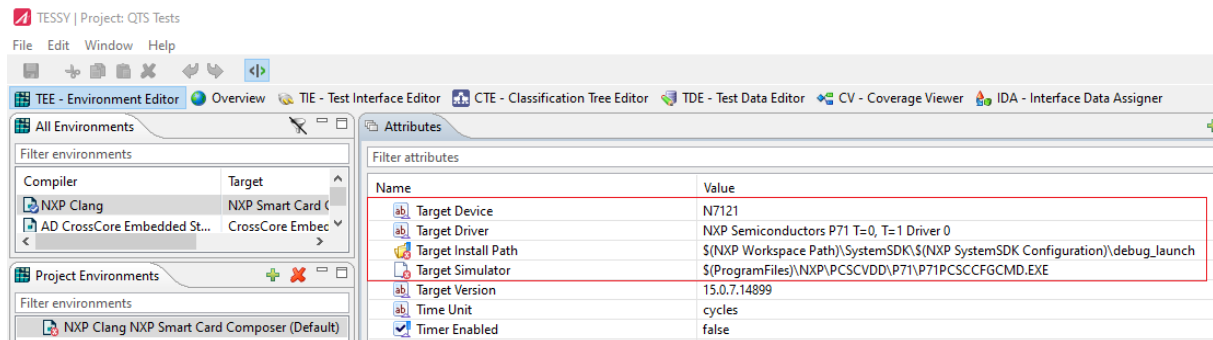
### 3 TEE Configuration

There are quite some TEE attributes bound to the adaption. The best way to start with is to set the following TEE attributes in the given order.

- NXP Install Path
- NXP Workspace Path
- NXP SystemSDK Configuration



- Target Simulator
- Target Device
- Target Driver



Furthermore, you may want to check if the attributes **NXP Plugin DLL**, **Target Install Path**, and **Compiler Install Path** fit. Please, also adjust the path to your NXP workspace monitor commands file, which is needed by the GdbSserver, within attribute **Monitor Command File**.

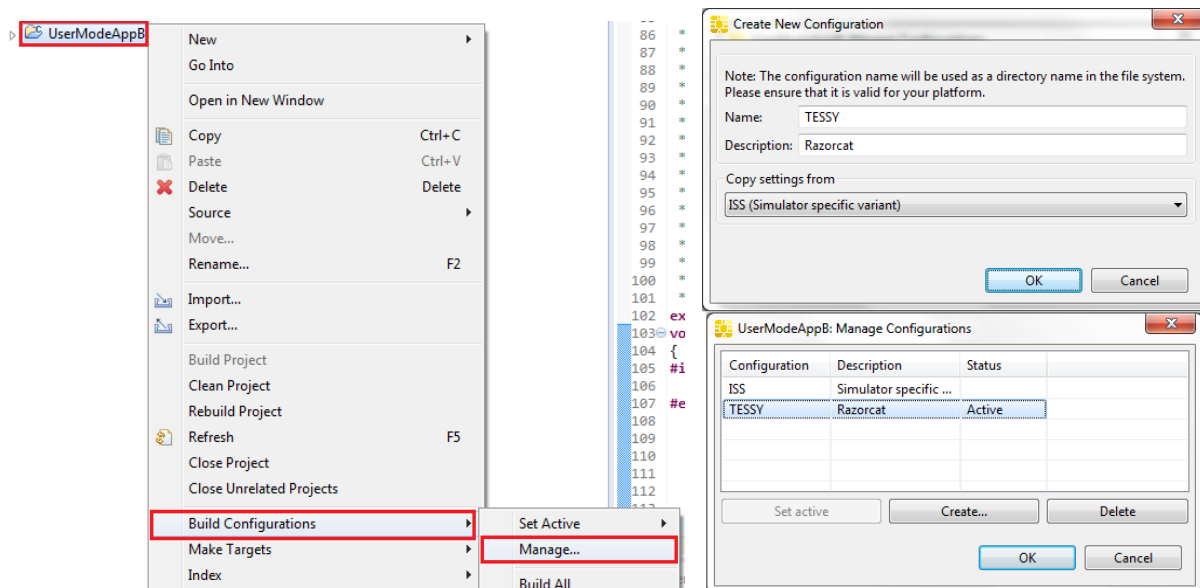
You should not alter attribute **NXP Build Command** unless you really know what you are doing. There is a special `build_script.bat` file within TESSY's installation directory which sets up the correct build command.

## 4 NXP Smart Card Composer Configuration

This chapter describes how to setup and adjust your NXP Smart Card Composer project to work with TESSY.

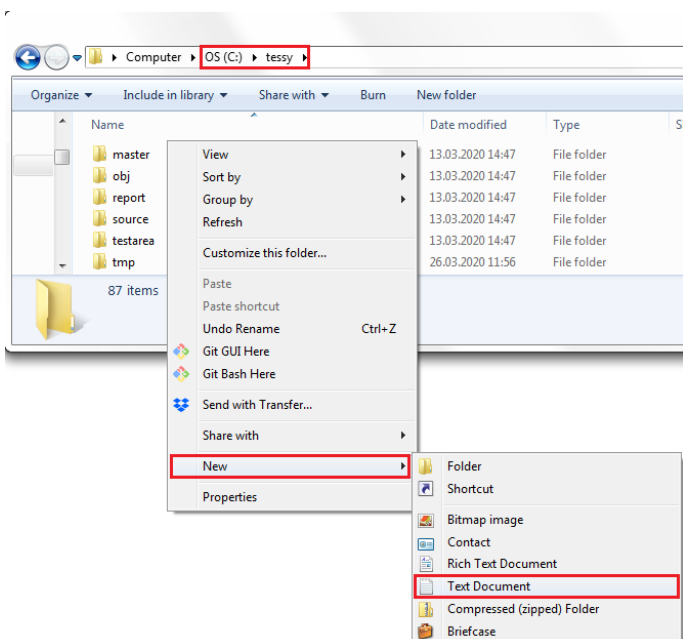
### 4.1 Create a new configuration

At first create a new configuration within your project so that you may switch between your own settings and the settings required by TESSY. Select **Build Configurations->Manage...** from the menu to open the **Manage Configurations** dialog. Select **Create...** to create a new configuration based on your original project configuration.

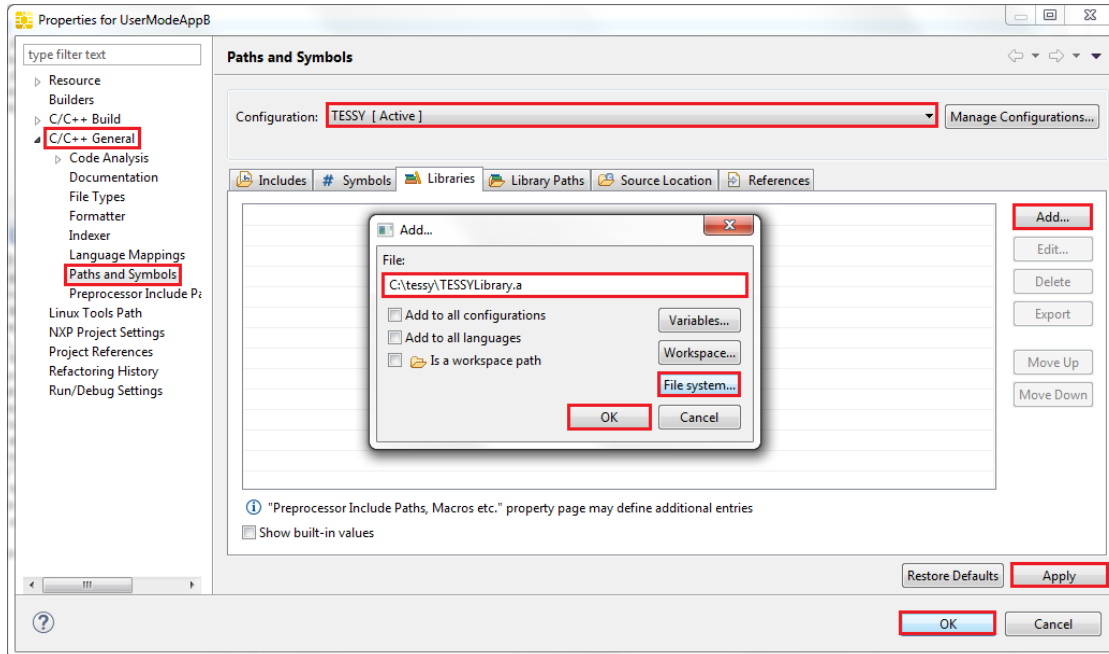


### 4.2 Add TESSY Library

Create a new file within the TESSY test area, which is by default in `c:\tessy`. If you have chosen a different path for the TESSY test area, you will have to adjust the path accordingly. Rename the new empty file to **TESSYLibrary.a**

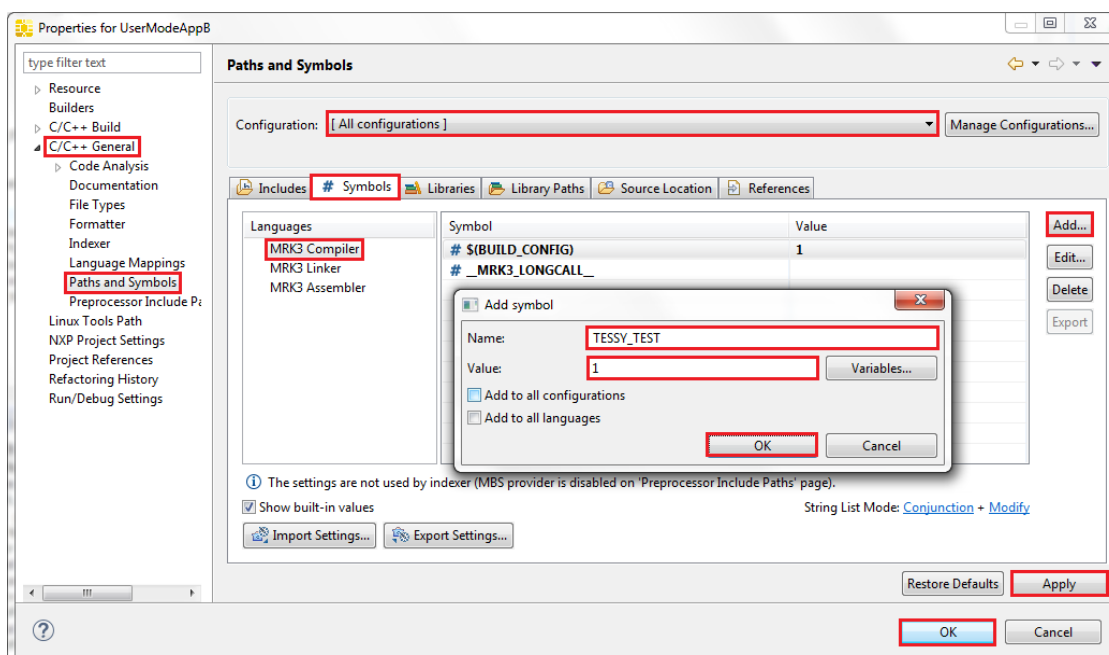


and add the file to your NXP Smart Card Composer project's TESSY configuration as follows. From `UserModeAppB`'s context menu select **Properties->C/C++ General->Paths and Symbols->Libraries**. Click **Add...** and add the path to the file via **File system...** Finally, click **OK**, click **Apply**, and click **OK**.



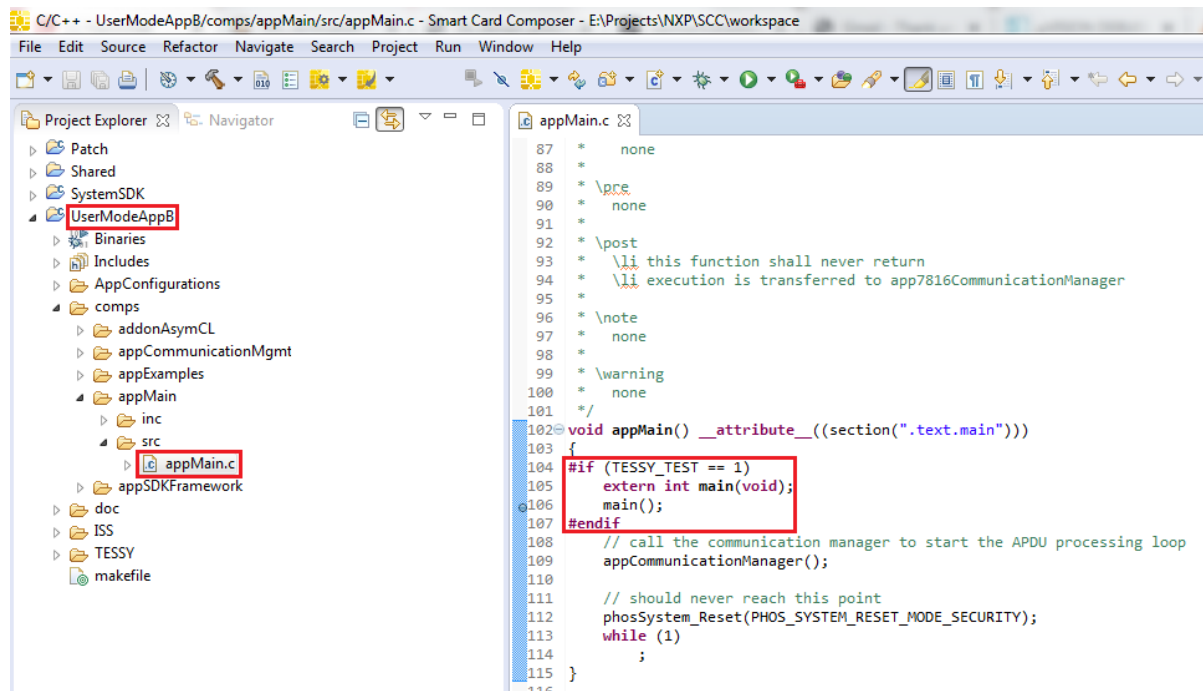
### 4.3 Add TESSY symbol

From `UserModeAppB`'s context menu select **Properties->C/C++ General->Paths and Symbols->Symbols->MRK3 Compiler**. Click **Add...** and add `TESSY_TEST` with value 1 as shown below. Finally, click **OK**, click **Apply**, and click **OK**. Since Smart Card Composer cannot handle defines by configuration, you will have to select **All configurations**.



## 4.4 Adapt appMain.c

From the Smart Card Composer **Project Explorer** open file `appMain.c` and insert the code as shown below and save the file.

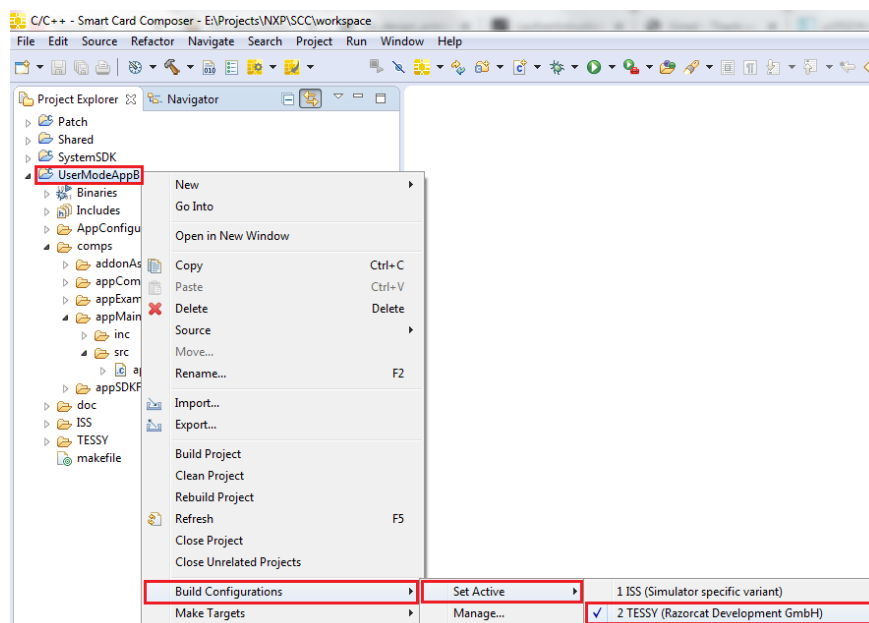


## 5 Switching between Configurations

It is a little bit tricky to switch between the ISS and the TESSY configuration to and fro. So it is important that you follow this procedure exactly.

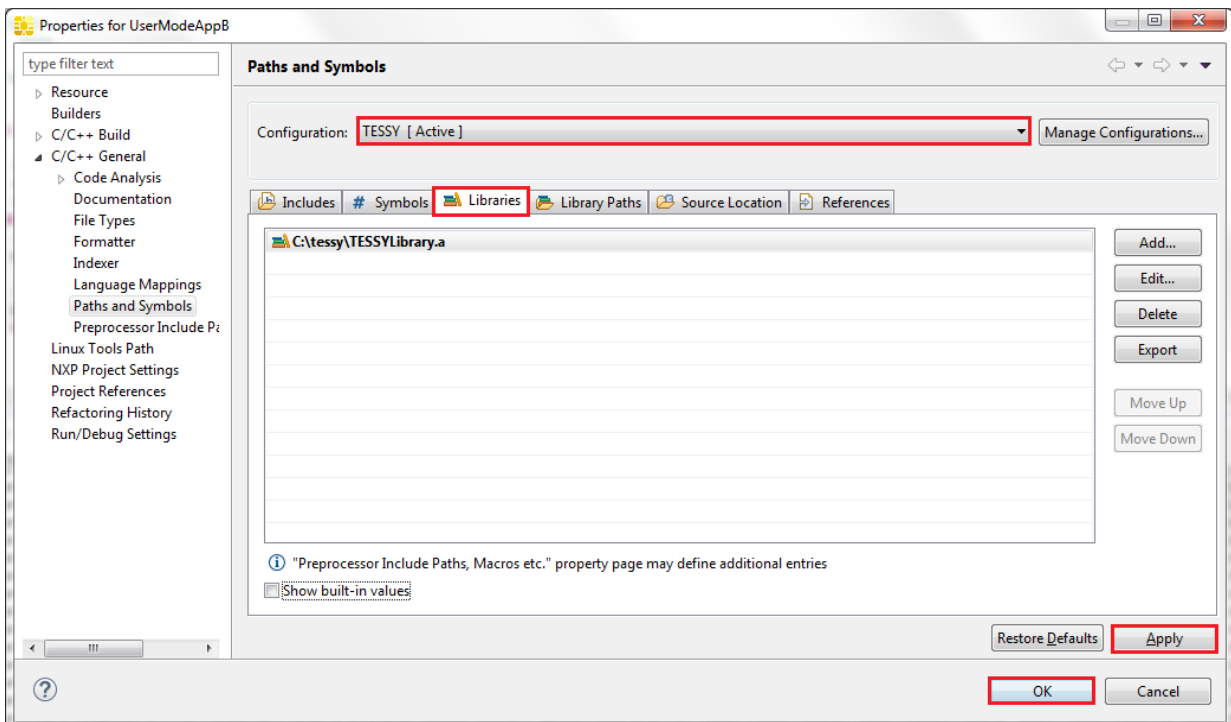
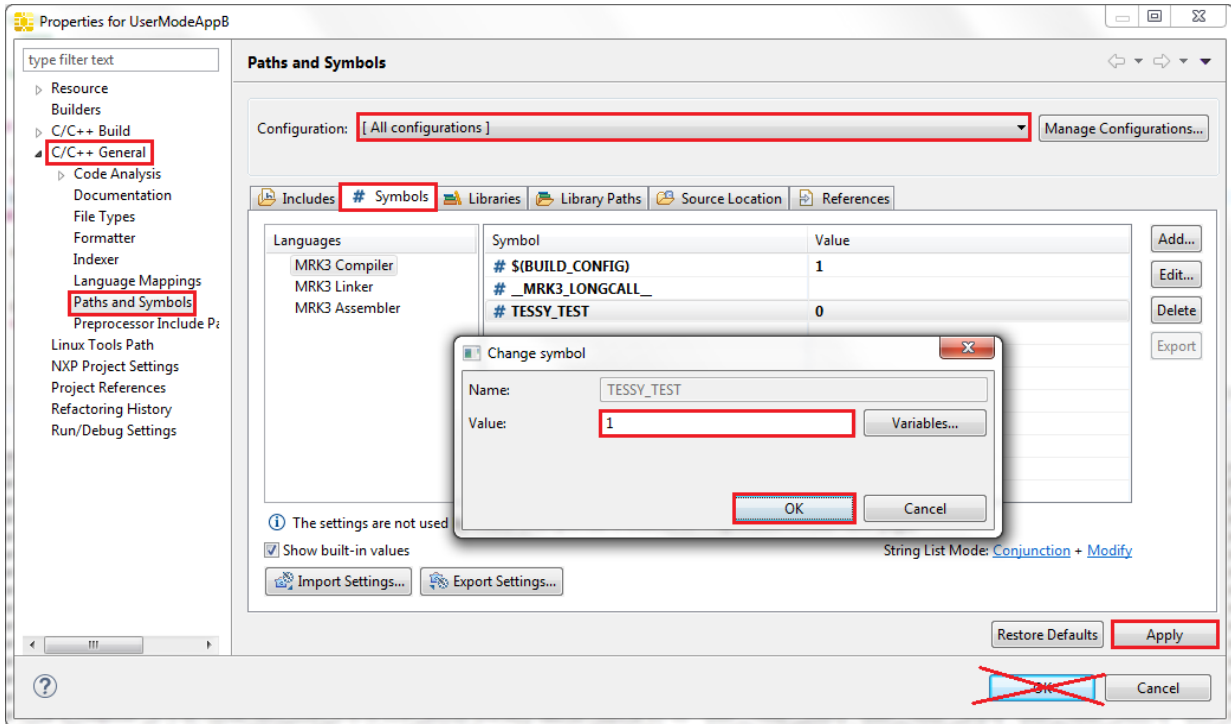
### 5.1 Switch to preferred next configuration

At first set the active configuration to the one you want to turn to. For example let us switch from ISS to TESSY.



## 5.2 Adjust TESSY\_TEST and apply Library again

Next open the **Properties** view of **UserModeAppB** select **Paths and Symbols**, select tab **Symbols**, select **[All configurations]** as configuration, and alter the **TESSY\_TEST** define to 1 which means that the TESSY code snippet in appMain.c is activated. Click **Apply** but do **not** click **OK**. Instead select tab **Libraries**, select the **TESSY** configuration, and click **Apply** as well.



If you need to switch from **TESSY** to **ISS**, set **TESSY\_TEST** to 0, select **Libraries**, and select **ISS [Active]** as Configuration instead of **TESSY [Active]**.

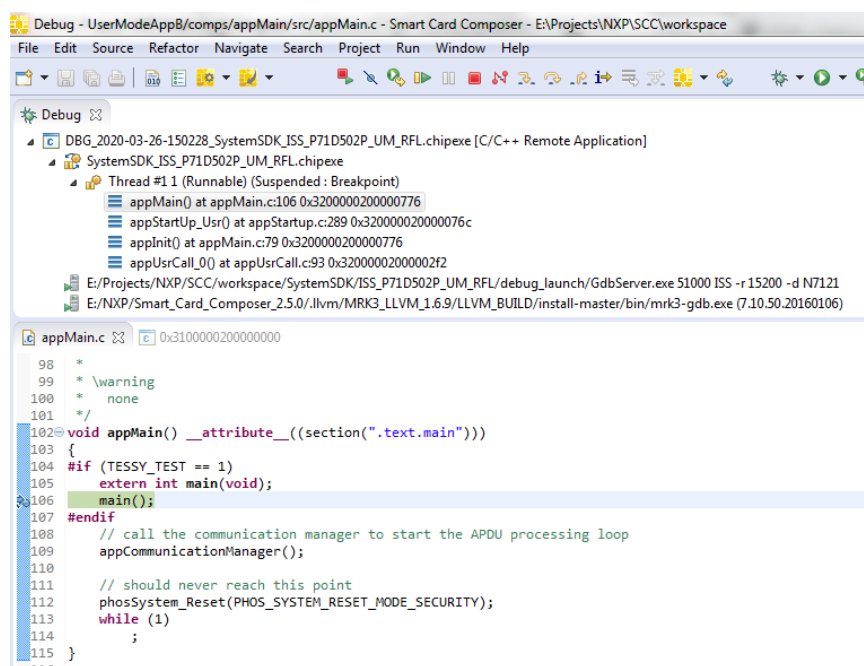
### 5.3 Rebuild UserModeAppB again

Finally rebuild the UserModeAppB project again.



## 6 Interactive Debugging

Interactive debugging using TESSY’s built-in feature is quite simple. Make sure your Smart Card Composer project is set to the **TESSY** configuration as described in chapter 5. Set attribute **Generate Builtin Data** from within TESSY’s **Properties** view to **true** and click **Execute Test** from TESSY’s toolbar. Now TESSY will generate a target library having the test data built-in and will link the library to the NXP target binary. So, all you have to do within NXP Smart Card Composer is to start the debugger, wait until the binary is load, and finally press the **Present** button of the NXP simulator. It is recommended to activate the **stop on startup at: appMain** toggle button from Smart Card Composer’s Debug Configurations dialog so that the debugger halts at **main ()**.



Step into `main()`, step over until `tessy_execute_task()`, and step into the latter.

```

212 #if !defined TS_OTHER_MAIN
213 int main(void)
214 {
215     tessy_init_task();
216
217     while (1) {
218         tstcomm_disable_send = 1;
219         tessy_start_task();
220         tstcomm_disable_send = 0;
221         tessy_execute_task();
222         tessy_end_task();
223     }
224     return 0;
225 }
226 #endif
    
```

Step into `TESSY_TestobjectCall()`.

```

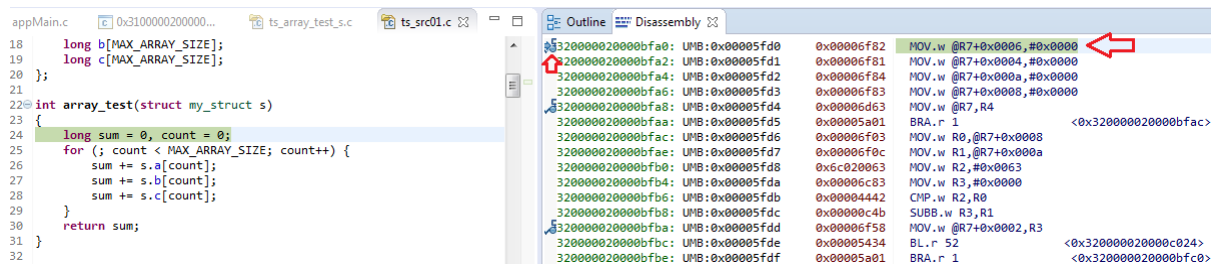
169 void tessy_execute_task(void)
170 {
171     if (ts_mode == TS_MODE_TESTSTEP) {
172         while (TS_REPEAT_COUNT) {
173             TESSY_TestobjectCall();
174             TS_REPEAT_COUNT--;
175         }
176     }
177 }
    
```

Step into your test object. In this example the function is called `array_test()`.

```

185 void TESSY_TestobjectCall(void) {
186     TS_TESTOBJECT_RETURN = array_test(TESSY_TESTOBJECT_PARM_01);
187 }
    
```

It is recommended to set a breakpoint at the very first statement of your test object within the **Disassembly** view, so that you may continue to the next test step.



The test object contains two global variables which hold the current test case and the current test step which you may find helpful to be added to the **Expressions** view.

Expression	Type	Value
(x)- TS_CURRENT_TESTCASE	TESSY_uint32	0x1
(x)- TS_CURRENT_TESTSTEP	TESSY_uint32	0x1
+ Add new expression		

**Please note:** At the end of debugging do not forget to reset the **Generate Builtin Data** attribute of the current test object to `false` to enable normal test execution without debugging again.

## 7 Troubleshooting

### 7.1 Function „tslow\_sync“ not defined

```

Console | Problems | Variants | Notes | Suspicious Elements
Messages
Building test driver succeeded.
-----+-----
| Execute Test [test/array_test]
-----+-----

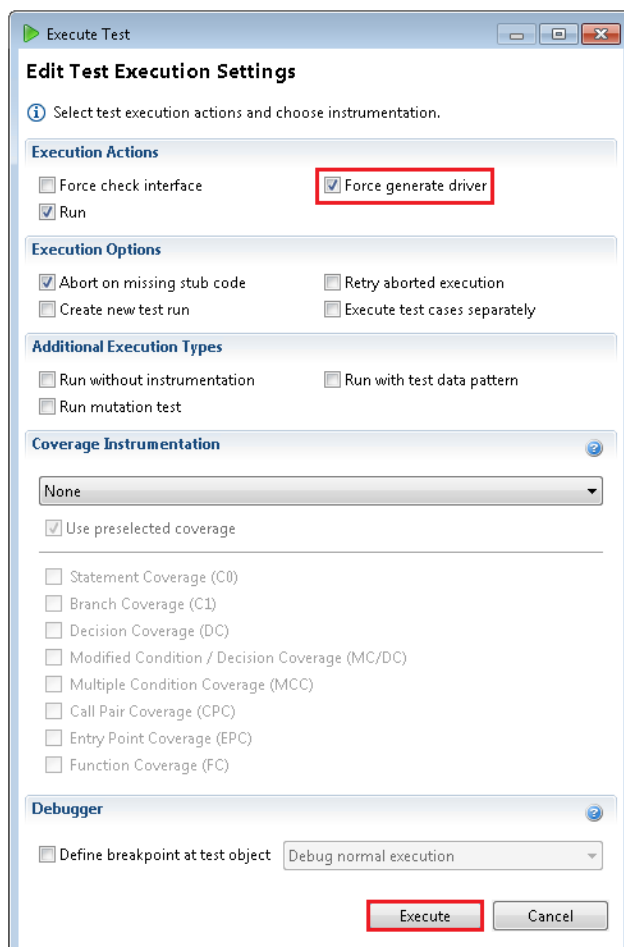
Device N7121
GdbServer (v.15.0.7.14899)
Listening for RSP on port 51000
Remote debugging from host
[tslow:setBreakpoints] Failed to set synchronization breakpoint - Function "tslow_sync" not defined.

Make breakpoint pending on future shared library load? (y or [n]) [answered N; input not from terminal]
-->
Target Communication: Master signaled error 0x1a: Could not connect to the slave
[tslow_error_abort] '26'
driver32: Error during test execution.
Error executing Master.
The specified disk or diskette cannot be accessed.

exp32.exe: Failed to execute 'driver32.exe "C:\tessy\tmp\tbs\3960.tbs"'
Failed to execute Test Object 'array_test'. Process with pid 3960 terminated with exit code 1.
Test execution finished with errors.

Execution job finished after 7710 milliseconds
  
```

Please, switch to the **TESSY** configuration for your Smart Card Composer project by following the procedure from chapter 5. Than start TESSY’s **Execute Test** dialog, enable **Force Generate Driver**, and click **Execute** again.



## 7.2 Execution gets stuck

Make sure that TEE attribute **NXP TESSY Library** path matches the library path within NXP Smart Card Composer (see 4.2).

## 7.3 (ERROR) [main] Invalid segment descriptor for execute

Verify TEE attribute **Monitor Commands File**. The monitor commands file give in your Smart Card Composer's project is required to setup the GdbServer correctly.