Controller Tester Troubleshooting Guides

3.4 — Last update: Feb 09, 2021

Suresofttech

Table of Contents

1. Controller Tester Troubleshooting Guide

This is a guide to solving common problems you may encounter using Controller Tester.

1.1. How to assign values to flexible array members

Here's how to assign values to flexible array members added in c99.

```
#include <stdio.h>

typedef struct _line {
        int size;
        int data[];
}line;

void funfun(line * abc){
        if(abc->data[0] == 1 ){
                printf("You can't touch this");

        }
        else{
                printf("MC Hammer");
        }

}
```

To design a test so that the if branch in the code above is true, it is not possible to put a value in the usual test data entry method.
This is because it is a flexible array member.

Therefore, the value should be entered as shown below using the code before the call in the test information tab.

```
size_t this_length = 5;
abc = (line *)malloc (sizeof (line) + this_length);
abc->data[0] = 1;
```

! You cannot use the above method in an environment without malloc.

1.2. How to increase heap memory in JAVA when memory is low

When the Controller Tester written in Java language based on Eclipse RCP and executed in jvm runs out of memory, the performance may deteriorate due to frequent GC (Garbage collection) or the program may not operate normally due to insufficient memory.
In this case, you can use the Controller Tester 64bit package (maximum heap memory default value of 8G), or manually adjust the heap memory settings.

Here's how to manually adjust the heap memory settings.

1. Open the *(product installation path)* \CodeScroll.ini file with an editor(requires modification with administrator privileges).
2. Run Controller Tester again, increasing the value of -Xmx. Controller Tester cannot be executed if too large a value is entered. Find and set the maximum value by decreasing or increasing the value little by little.

> The maximum value of Xmx depends on the free memory status of the PC. In general, when using 4GByte memory and executing only Controller Tester, it will have a value in the range of approximately 1000MByte ~ 2000MByte.

1.3. Project DB file (*.csp) is damaged due to abnormal termination

Controller Tester saves project information in SQLite Database file.

```
(Workspace path)\(project name)\.csdata\(project name).csp
```

DB may be damaged when Controller Tester is abnormally terminated during operation due to a power failure or a Windows error.
In this case, you can recover the damaged DB file in the following way(not always possible).

1. Run command prompt from below path
    - `(Product installation path)\plugins\com.codescroll.gp.core_1.0.2.201202152119\lib`
2. Run the below command at the command prompt
    - `$ sqlite3 corrupted.db ".dump" | sqlite3 new.db`

1.4. Testrun.exe crashes when testing QNX software

When testing C++ code, if the global variable is of class type, abnormal termination may occur in the constructor of the class.

In this case, you should check the following:

1. If the __get_errno_ptr() function is made of stubs, add the following to the stub body.
    - `int* x = (int *)calloc(sizeof(int),1); return x;`
2. If you are using a debug utility or a utility related to shm (shared memory), you need to make sure that the file path is not hardcoded. If it is hard coded, you should create system functions (fprintf, etc.) to access the file as stubs.
3. You need to make sure that you do exit code at initialization time. If you run the exit code, you must generate the exit code as a stub.
4. You need to check if the constructor stub has a throw. If there is a throw, it should be removed.

1.5. When a specific function is not displayed in the coverage view

If the function is not displayed in the coverage view and the solution is as follows:

Function not showing in coverage view	Resolution
If the source file to which the function belongs is excluded from analysis	Remove the corresponding source file or included path from [Inclusion and Exclusion] of project properties or [Exclusion] of environment setting
The function or the source file to which the function belongs is selected as an exclusion of coverage measurement	Remove the function or source file from [Exclusion of coverage] of project properties
If the function is not executed	Check whether the test was executed normally or check the design of the test case

1.6. If the error message is not displayed after the project analysis fails

When creating a unit test after project creation (analysis), if it fails without any special error message, please check the following:

Analysis fails without message	Resolution
Controller Tester is not run with administrator privileges	Redo as administrator
The path of the generated project or the path or name of the source file to be tested is very long.	Modify the path of the workspace or source file under test to the shortest path possible
If there is a path related to MYSQL in the PATH of the environment variable, and the path contains “&”	Restart Controller Tester after deleting MYSQL path if “&” is present

1.7. The screen is cut off when the Windows display magnification is not 100%

If you set the Windows display magnification to a value other than 100%, setting the following will solve the problem that the screen is cut off or the image is broken.
(Controller Tester automatically reflected from version 3.1)

1. Right-click the executable file (CodeScroll.exe) or shortcut file and click [Properties]
2. On the [Compatibility] tab, click Change High DPI Settings
3. In [Override High DPI Adjustment], select Override High DPI Adjustment Behavior.
4. Select [Coordinator] as the system and confirm

> If you set it as above, the Windows system will automatically adjust the scale, so the screen may appear slightly whitish.

1.8. Windows.h file not found

When installing the Windows SDK 6.1 and using the Visual Studio 2008 compiler, the following error message may be displayed in certain environments.

```
Cannot open include file: 'windows.h': No such file or directory
```

This error is caused by missing content in the batch file that sets up the Visual Studio build environment. The solution is as follows.

Search for and locate the windows.h header file in the C:\Program Files\Microsoft SDKs directory.

```
Example) C:\Program Files\Microsoft SDKs\Windows\v6.1\Include
```

Open the file C:\Program Files (x86)\Microsoft Visual Studio 9.0\VC\bin\vcvars32.bat with an editor and add the path SDKs\windows\v6.0A\include, SDKs\windows\v6.0A\lib to the INCLUDE, LIB, and LIBPATH variables.

```
Example) line 26  @set Path to add to INCLUDE: C:\Program Files\Microsoft SDKs\
Windows\v6.1\Include

@set INCLUDE=C:\Program Files\Microsoft SDKs\Windows\v6.1\Include;%VCINSTALLDI
R%\ATLMFC\INCLUDE;%VCINSTALLDIR%\INCLUDE;%INCLUDE%\
```

1.9. If INFO [ut.hio]: runTest:testrun exit code(105) is displayed in the log and the unit test is not executed

The message INFO [ut.hio]: runTest:testrun exit code(105) is displayed in (project name)_full.log, and the unit test may not be executed.

In this case,
`(Worksapce)\(project name)\.csdata\build\testrun.log`
Check the file for the following error message.
bc. error:add virtual address : memory alloc error [E000000-E1FFFFFF]
error:fail to init iohandle(105)

This error occurs when the address range set in [Virtual Address] in the preferences settings is invalid.

In the [Virtual Address] of the environment setting, modify the memory address range that can be used in the current system.

1.10. When “Extract Toolchain Info” fails

When the [Extract Toolchain Info] button is clicked in the Add a Toolchain Wizard, it may fail.
In this case, you can solve the problem by taking the following measures.

- In [Toolchain Information] of the Add a Toolchain Wizard, enter the environment information file (*.bat) to run the compiler in [Env Script].
    - Example) For the Visual Studio cl compiler, enter the vcvars32.bat file in the same path as the cl.exe file.
- Check if the security program (such as ALYac) has blocked the tce.exe executable and add it to the scan exceptions list.
- Check if any other value besides the default value is added to ComSpec variable among Windows environment variables, and if so, modify it to the default value.
    - Default value for ComSpec variable: ComSpec=C:\Windows\system32\cmd.exe

1.11. Messages display abnormally in the error view

Depending on the environment in which Controller Tester is used, the encoding of the error message that is output may be different, and the message may be displayed abnormally.
In this case, you can solve the problem in the following way.

`(Product install path)\plugins\com.codescroll.gp.rcp.helios_1.0.0.20190924035 1\plugin_customization.ini`
Open the file in an editor.

in line 85,

```
#log file encoding (log plugin)
com.codescroll.gp.log/log.file.encoding=euc-kr
```

After changing euc-kr to the encoding suitable for the current environment, restart Controller Tester.

1.12. When testing with source code containing C++20 items

The analyzer currently used by Controller Tester does not support C++20 items and some new headers, so if you want to test with source code that includes those items, you need to do something extra.

1. Edit *(Controller Tester global path)* \1.1\parserConfig\ *vs2019 toolchain name*.conf
    - Add “ms_c++20” to the last line
2. Edit *(Controller Tester install path)* \plugins\com.codescroll.ut_3.3.2\config\cl.flag.txt file
    - Add “&/std:c++latest” to the last line of
    - This value must be cleared again when analyzing/performing sources that do not contain new items.

1.13. 테스트 실행시 발생할 수 있는 에러

1. 기본 생성자를 사용할 수 없는 에러

예제코드

```
class A{
public:
      A(int a, int b){}
      testFunction();
}
```

- Controller Tester에서는 클래스 멤버 함수에 대한 테스트 생성시 기본 생성자로 인스턴스를 생성한 후에 함수를 호출하여 커버리지를 측정합니다. 하지만 위의 예제 코드를 보면 class A의 기본 생성자가 없어 인스턴스 생성시 에러가 발생하게 됩니다. 이와 같은 에러가 발생하면 테스트를 수정하여 적절한 생성자로 인스턴스를 생성하도록 수정해야합니다.

2. instance를 생성하지 못하는 에러

- 추상 클래스의 함수를 테스트로 생성하는 경우 추상 클래스로 인스턴스를 생성하고 생성된 인스턴스에서 함수를 호출하려고 하여 에러가 발생할 수 있습니다. 이런 경우 '테스트 정보'에서 해당 클래스를 찾은 후 생성자에서 '사용자 코드'로 바꾸어줍니다. 이 후 사용자 코드에서 추상 클래스를 상속받고 있는 클래스로 인스턴스를 생성하도록 수정해주시면 됩니다.

1.14. 통합 테스트 가져오기 후 전역 변수를 찾지 못하는 경우

기존에 잘 수행되던 통합 테스트 데이터를 가져오기 후 테스트 실행을 했을 때 특정 전역 변수에 대해 undefined reference 에러가 발생할 수 있습니다.
에러가 발생한 테스트의 ‘테스트 정보’에서 ‘전역 변수’ 클릭시 Not Found element 메세지가 출력이 된다면 코드 수정으로 인해 전역 변수의 TU가 달라졌을 수 있습니다.
테스트의 ‘설정’ 탭에서 연관 파일을 전역 변수의 TU가 바뀐 소스 파일로 변경해주시면 테스트가 정상 동작하는 것을 확인할 수 있습니다.

1.15. 테스트 실행시 “C2118 : 첨자가 음수입니다 “ 에러 발생시

visual studio 툴체인 혹은 변환 툴체인 사용시 발생할 수 있는 에러 메세지입니다.
모듈 특성의 ‘컴파일 플레그’에 추가 되어 있는 `/Zp` 옵션을 제거해주시면 됩니다.

1.16. Visual Studio 2015 툴체인 사용시 SDK 버전 문제

Visual Studio 2015 툴체인 사용시 SDK 버전이 8.1과 10.0.10240.0 이상이 같이 설치되어 있는 경우 테스트 실행이 정상적으로 수행되지 않을 수 있습니다.
Visual Studio 2015의 vcvars32.bat 환경 설정 스크립트가 최신 버전 sdk로 설정이 되기 때문에 발생하는 문제입니다.
모듈 특성의 '컴파일 플레그'에 /I 옵션으로 최신 sdk 버전의 시스템 헤더를 추가해주어야합니다.

ex) sdk 최신 버전이 10.0.18362.0인 경우
/I"C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\include"
/I"C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\atlmfc\include"
/I"C:\Program Files (x86)\Windows Kits\10\Include\10.0.18362.0\ucrt"
/I"C:\Program Files (x86)\Windows Kits\10\Include\10.0.18362.0\um"
/I"C:\Program Files (x86)\Windows Kits\10\Include\10.0.18362.0\shared"
/I"C:\Program Files (x86)\Windows Kits\10\Include\10.0.18362.0\winrt"
/I"C:\Program Files (x86)\Windows Kits\10\Include\10.0.18362.0\cppwinrt"

1.17. Controller Tester가 설치되지 않는 경우

설치 파일이 일반적인 디스크가 아닌 드라이브에 있어서 인스톨러가 접근할 수 없거나 보안 프로그램의 영향 등으로 설치가 실패하는 경우

1. 설치 파일(.msi 혹은 .exe)을 시스템 드라이브로 옮깁니다.
2. 파일을 우클릭한 후, 관리자 권한으로 실행합니다.

설치 파일이 전송 중 손상된 경우

설치 파일을 다시 받아서 설치합니다.

그 외

위 두 가지 방법을 시도해도 동일한 오류가 발생하는 경우에 다음과 같이 해보시기 바랍니다.

1. 명령 프롬프트(cmd.exe)를 열고 실행 파일이 있는 경로로 이동합니다.
2. 다음 명령어를 수행하여 설치 마법사를 동일하게 진행합니다.
   - .msi 파일로 설치하는 경우
     `msiexec /I "****설치 파일 이름****.msi" /L*V "install.log"`
   - .exe 파일로 설치하는 경우
     `"***설치 파일 이름***.exe" /L*v "install.log" /I`

3. 설치 마법사를 진행하면 해당 경로에 install.log 파일이 생성됩니다. install.log 파일에 다음과 같은 로그가 있는 경우 Windows Installer 실행 관련 오류입니다.
   ```
   MainEngineThread is returning 1603
   ```

4. 이 경우 우선 컴퓨터를 재부팅합니다.
5. 작업 관리자를 연 뒤 msiexec.exe 프로세스가 실행중이라면 종료시킨 뒤 Controller Tester를 다시 설치하시기 바랍니다.
6. 그래도 문제가 해결되지 않는다면 아래 Microsoft의 문제 해결 가이드대로 수행하시기 바랍니다.
   https://support.microsoft.com/ko-kr/help/17588/windows-fix-problems-that-block-programs-being-installed-or-removed
   https://support.microsoft.com/ko-kr/help/834484/you-receive-an-error-1603-a-fatal-error-occurred-during-installation

7. Microsoft의 문제 해결 가이드대로 수행하여도 문제가 해결되지 않는 경우 아래 내용을 기술지원 연락처로 보내주시기 바랍니다.
   - 설치를 시도한 패키지의 상세 정보
     - ex) Controller Tester (Host) 3.4.2 ×64
   - 명령 프롬프트로 설치하여 생성된 install.log 파일
   - 특정 PC에서만 발생하는지 여부
   - 문제가 발생한 PC의 환경
   - (Controller Tester 외에 사용하는 자사 제품이 있는 경우) 다른 제품을 설치할 때도 동일한 현상이 발행했는 지

기술 지원 연락처

- support@sursofttech.com

2. Controller Tester Target Plug-in Troubleshooting Guide

2.1. Build issues after exporting target test code

2.1.1. When the entry point function name is not main (for versions before CT 3.2)

The Controller Tester Target Plugin replaces the original main function with the cs_renamed_main function so that when the software is run on the target, the main function defined in Controller Tester is executed.
If the original entry point function name is not main, you need to modify the cs_tfx.c and cs_build_macro.h files.
The cs_tfx.c file is included in the Controller Tester installation path, but cs_build_macro.h is generated every time you export the target test code, so you need to apply the patch to fix the problem.

If you suspect an entry point function name problem, you can check the code below to modify it.

cs_tfx.c

```
#if defined main /* normal */
#undef main // replace with entry point function name
#endif
```

cs_build_macro.h

```
#if !defined CS_START_FROM_IUT
#define main        __cs_renamed_main // replace with entry point function nam
e
```

2.1.2. Multiple definition error of the function used to save the target test result (for versions before CT 3.2)

Target test results are output in the form of ‘target log’ when the test is run on the target. Because the target log uses the memset function to store data, it is declared with the extern keyword in the cs_tfx.h file to use the memset function.
When you export the target test code, the cs_tfx.h file is included in the original source code. At this time, multiple definition error may occur depending on whether memset or strncpy function is included in the source code to be tested.

```
ex) multiple definition of `memset' error
```

If you encounter this error, comment out the multiple error line in the cstfx.h file in the Controller Tester installation path and then ‘Export Target Test Code’ to fix the problem.

```
Example cstfx.h path: C:\Program Files\CodeScroll Controller Tester 3.1\plugin
s\com.codescroll.ut_3.1.2\target\lib\controller_nc\cs_tfx.h
```

2.1.3. ‘sprintf’ has not been declared or CS_FLT_OUTPUT error (for versions before CT 3.2)

Sometimes the sprintf function is not available in the source code under test.
In this case, you can fix the problem by adding the following code to the cstfx.h file in the Controller Tester installation path.

```
Example cstfx.h path: C:\Program Files\CodeScroll Controller Tester 3.1\plugin
s\com.codescroll.ut_3.1.2\target\lib\controller_nc\cs_tfx.h
```

Add

```
int sprintf(char* str, const char* format, ...);
```

to line 210 of the cs_tfx.h file

2.1.4. Target log interface settings

The target log interface file name is cs_io_implementation.c. Performing ‘Export Target Test Code’ will include the target log interface file for each source file.
Using the target log interface’s cs_io_putbyte function, the log is output in char units. Depending on the implementation of the target log interface, the target log can be output in various ways such as Ethernet, UART (Serial), and JTAG.
Depending on the communication method used, appropriate initialization logic must be implemented in the cs_io_initialize function. In particular, if you do not set the baudRate or dataBits settings when using UART, the target log may be displayed incorrectly.
Accurate setup for target testing requires an understanding of the target and software, so you should contact your target development staff.

2.1.5. Check for signal errors

You can check the output of the target log to see if an error occurred while the test was running in the target environment.
You can check the target log to see if an error occurred during test case execution or if an error occurred before the test case was run.
When the test run is completed normally, “CSET” is printed at the end of the target log. If the log is truncated after “CSTC” is printed, you should find the test case in the project DB.

```
Project DB file: { ProjectName }.csp file in .csdata directory under project p
ath in [Project Properties]-[Info]-[Location]
Testcase output format: CSTC <TEST_ID, TEST_CASE_NO>
```

Target log output example

```
CSTR<374069146,130789013,1,0,1>CSTR#
CSST<test,1574932569>CSST#
CSTC<133143986176,1>CSTC#
CSOS<1returnVar,0>CSOS#
CSES<1>CSES#
CSTB<1>CSTB#
CSET< test,1574932569>CSET#
```

2.1.6. Problems When Testing Functions Related to UART Communication in Target Software

If a test case is performed that changes the UART communication settings, subsequent target logs may not be output correctly.
(Example: test case to change baudrate setting)
In this case, you must switch from the settings in the test case editor to unmanaged code, and then modify (implement the code) the test case to back up the settings before the test case is executed and restore them after the test case is run.

2.1.7. When building with CodeWarrior, test related files such as cs_tfx.c are not found

In CodeWarrior's project settings, make sure the system path contains the following path, and add it if it doesn't exist.

```
- {CT_Workspace}/.metatdata/.plugins/com.codescroll.ut.embedded/{ ProjectName
}/TestFixture/cs
- {CT_Workspace}/.metatdata/.plugins/com.codescroll.ut.embedded/{ ProjectName
}/TestFixture
```

2.1.8. Target output value does not appear or is output as 0 for float type in CodeWarrior

When the variable value of the real type is output in the target test, the result value is retrieved through the sprintf function.
However, there are cases where the library provided by CodeWarrior cannot use the sprintf function for real types.
In this case, you need to replace or add the library linked from the CodeWarrior project.
Libc99_E200z650.a is a library that can use the sprintf function for the real type currently identified.
After replacing with the library, edit the contents of the cs_tfx.h file as follows.

Before modification

```
#define TFX_ftoa_writeBytes(value) \
do {\
        sprintf(buf, "%g", value);\
        TFX_writeBytes(buf);\
} while(0)
```

After modification

```
 #define TFX_ftoa_writeBytes(value) \
do {\
        sprintf(buf, "%f", value);\
        TFX_writeBytes(buf);\
} while(0)
```

2.1.9. undefined error of type codescroll_int32 and codescroll_uint32

Check if there are codescroll_int32 and codescroll_uint32 in the info file of the toolchain set in the project. If not, add the type. After editing the info file, you will need to ‘Export Target Test Code’ again.

```
Toolchain info file path: { ToolchainName }.info in [Preferences]-[ToolChain]-
[Open configuration Folder]
```

2.1.10. When cannot use ‘long’ or undefined type to ‘long’ error occurs in codescroll_int, codescroll_uint type

This error can occur if the target software does not support the long type. Change the codescroll_int and codescroll_uint type to int in the info file of the toolchain set in the project.

```
Toolchain info file path: { ToolchainName }.info in [Preferences]-[ToolChain]-
[Open configuration Folder]
```

```
ex) unsigned long long,unsigned,0,9223372036854775807,8,codescroll_uint
        -> unsigned int,unsigned,0,9223372036854775807,8,codescroll_uint
```

2.1.11. When the address of cs_io_putbyte is not found in Code Composer Studio

```
SEVERE: Unable to get address for symbol: cs_io_putbyte
org.mozilla.javascript.WrappedException: Wrapped com.ti.ccstudio.scripting.env
ironment.ScriptingException: Unable to get address for symbol: cs_io_putbyte
```

If the target_binary_path in the execution tab of the target environment setting and the actually built binary are different, the error may occur.

In this case, it works normally if you change the target_binary_path to the path of the actual built binary.

The way to find out in the console log that this is the cause of the error is as follows.

```
Check if the .out file of the finished building target and the .out file of lo
adProgram: ENTRY sFileName: are the same
```

2.1.12. GreenHills AdaMulti에서 declaration is incompatible with “void cs_io_putbyte” 에러 발생시

AdaMulti에서 빌드시 **declaration is incompatible with “void cs_io_putbyte(codescroll_byte)”** 에러가 출력된다면 codescroll_byte 타입이 char 가 아닌 signed char, unsigned char 로 설정되어 발생한 경우입니다. 해당 프로젝트 분석시 사용했던 툴체인.info 파일에 아래와 같은 형태에서 codescroll_byte 타입을 수정해야합니다.**(툴체인.info 파일은 ‘환경 설정’ > ‘툴체인’ > ‘설정 디렉터리 열기’ 에서 확인할 수 있습니다.)**

수정 전

```
#typeName,valueKind,min,max,size,csType
signed char,signed,-128,127,1,codescroll_byte
```

수정 후

```
#typeName,valueKind,min,max,size,csType
char,signed,-128,127,1,codescroll_byte
```

수정 후 Controller Tester 의 툴체인 편집창에서 툴체인 선택 후 ‘편집’ > ‘완료’ > ‘적용 후 닫기’ 를 해야 수정된 내용이 반영됩니다.

2.2. TRACE32 related issues

- symbol not found error “ct_target_log”
- target reset failed
- ‘&binary_path’ symbol not found or an error occurred at Data.Load.Elf “{file_path}” /LPATH while running the cmm script

2.2.1. symbol not found error “ct_target_log”

The results of the tests on the target are stored in ct_target_log.
If you get an error that can’t find the ct_target_log, there are two things to check:

1. Make sure your target test code is built correctly
    - If the test code was not applied correctly or if the binaries built with the test code were not deployed to the target (build the original software only), you might get an error that a variable cannot be found to store the test results.

2. Position of the GO.HLL instruction in the cmm script
    - You must write a cmm script to specify a breakpoint with the BREAK.SET command and begin debugging.

ex)

```
BREAK.SET cs_io_initialize /Program /cmd " … " /RESUME
BREAK.SET cs_write_log /Program /cmd "…"  /RESUME
BREAK.SET cs_io_finalize /Program /cmd "…" /RESUME /cmd "…" /RESUME

GO.HLL <- Enter after setting BREAK
```

2.2.2. target reset failed

There are two things to check.

1. Make sure the CPU_NAME in System.CPU {CPU_NAME} is set correctly in the cmm script
2. Make sure that the target binary size is too big to load on the target

2.2.3. ‘&binary_path’ symbol not found or an error occurred at Data.Load.Elf “{file_path}” /LPATH while running the cmm script

When testing targets using Trace32, the Controller Tester calls the target.cmm script from the start.cmm script.
When calling the target.cmm script from the start.cmm script, pass “&binary_path” as a parameter. This error can occur if the build is not successful or if the path passed as a parameter is incorrect.
In this case, you should check that the build is successful and check the path in the start.cmm script.

2.3. Importing target log (test results) issues

2.3.1. Failed or Error When Performing ‘Import Target Test Log’

You should open and check the target log file. If the log is output normally, the log ‘CSET <…> # CSET’ is output last. If this log is not printed, a signal error likely occurred while running tests on the target.
To fix the problem, run the test cases one by one and find and fix the test case that caused the signal error.

2.3.2. When the target log buffer size is exceeded

If you store the char value of the cs_io_putbyte function in a fixed-size buffer, such as char * buff[BUFF_SIZE], large test results can cause you to exceed the buffer size.
To fix the problem, modify the BUFF_SIZE in the “Target Log Interface” and export the target test code again.

2.3.3. When an exception occurs in the script when running after test building in Code Composer Studio

When an error occurs:

*CCS Caused by: com.ti.ccstudio.scripting.environment.ScriptingException: Could not open session. Found devices matching: .**

When running the target test using the Code Composer Studio debugger in Controller Tester, Controller Tester uses the execution script written in Javascript.
The script opens a debugging session from the available debugging probes and runs the target binary. Basically, there is one debugging probe and one debugging session. Depending on the target environment setting, there may be two or more debugging probes, and the above error occurs.
You can fix the error by specifying the debugging probe you want to use.
The debugging probe can be set in Project Properties > Target Test > Target environment settings > [Run] tab > **debug_probe** option.

> When you do not know which debugging probe to choose, run the target project in debugging mode in the Code Composer Studio IDE and set the debugging probe value displayed in the debug view to the debug_probe option.

When an error occurs: invalid target memory page

When you build the Code Composer Studio project, a .map file is generated. The .map file contains information about the address and page location to which variables are assigned.
The Code Composer Studio execution script provided by Controller Tester is implemented to store the target test result value in the data area and get this value.
However, if it is stored in the program area, not in the data area, an error message may appear when the script is executed, indicating that the memory area cannot be accessed.
Check the **char_format** option in Project Properties> Target Test > Target environment settings > [Run] tab.

2.3.4. 타깃 디버그 정보 로그를 가져오기시 실패하는 경우

타깃 테스트시 디버거가 없는 환경에서 ‘디버그 정보 확인’ 기능을 사용하면 디버깅 정보를 로그를 통해 확인할 수 있습니다.
타깃 테스트 환경에서 ‘디버그 정보 확인’을 통해 테스트 코드를 내보내어 빌드 및 타깃 실행을 하게 되면 커버리지 측정값이 아닌 디버깅 정보가 출력되게 됩니다.
이 때, 기본 로그 출력 사이즈가 1000으로 지정되어 있는데 로그 정보가 1000글자를 넘어서면 그대로 실행이 끝나게 되어 디버깅 로그가 잘리는 현상이 발생할 수 있습니다.
프로젝트별로 생성되는 ut.ini 파일에서 **TARGET_DEBUG_PROBE_INDEX** 옵션을 통해 로그 출력 사이즈를 조절할 수 있습니다.

ut.ini 파일 위치 : Controller Tester_프로젝트_경로\.csdata\ut.ini
옵션 : TARGET_DEBUG_PROBE_INDEX=1000

2.3.5. '타깃 로그 가져오기' 실패시 타깃 로그로 확인하는 방법

테스트 실행 결과는 아래와 같은 패턴으로 출력됩니다.

```
CSTR<212668380,3136998082,2,0>CSTR#
CSST<ct_test_log,425747>CSST#
CSTC<4294967296,2>CSTC#
CSOS<2s_aDoJobCnt[0],0>CSOS#
CSES<1>CSES#
CSTB<110011000000100101100000>CSTB#
CSET<ct_test_log,425747>CSET#
```

- 2번 테스트 케이스 실행시 시그널 에러가 발생한 경우

```
CSTR<212668380,3136998082,2,0>CSTR#
CSST<ct_test_log,425747>CSST#
CSTC<4294967296,2>CSTC#// CSTC 태그 사이에 2번째에 있는 번호가 테스트 케이스 번호를 나타냄
```

```
CSTR<212668380,3136998082,2,0>CSTR#
CSST<ct_test_log,425747>CSST#
CSTC<4294967296,2>CSTC#
CSOS<2s_aDoJobCnt[0],0>CSOS#
```

- 테스트 케이스의 특정 변수가 null인데 출력값을 체크한 경우

```
CSTR<212668380,3136998082,2,0>CSTR#
CSST<ct_test_log,425747>CSST#
CSTC<4294967296,2>CSTC#
CSOS<2s_aDoJobCnt[0], //s_aDoJobCnt[0] 변수값이 null이라 값 출력 중 에러 발생
```

- 테스트 실행중 무한 루프로 인해 커버리지 결과가 출력되지 않는 경우

```
CSTR<212668380,3136998082,2,0>CSTR#
CSST<ct_test_log,425747>CSST#
CSTC<4294967296,2>CSTC#
CSOS<2s_aDoJobCnt[0],0CSOS#
CSES<1>CSES#
CSTB<11001100000010010110  //커버리지 비트맵이 출력중에 무한 루프로 인해 더 이상 출력되지 않음
```

2.3.6. CodeComposerStudio에서 프로젝트의 최적화 옵션이 2단계 이상인 경우 실행 에러 발생 시

Code Composer Studio에서 최적화 2단계 이상에서 break point의 제약이 있어 현재 도구에 반영되어 있는 스크립트로 실행시 정상적인 타깃 테스트 실행이 불가할 수 있습니다.(3.4 버전 이하)
최적화 옵션 2단계 이상에서 테스트 실행시 문제가 발생하면 Controller Tester 패키지에 포함되어 있는 default_ccs_script.js 파일을 아래 링크에 있는 default_ccs_script.js 파일로 교체해주시면 됩니다.

default_ccs_script.js 경로 : Constroller Tester 설치 경로\plugins\com.codescroll.ut.embedded_1.0.0\script\targets\ccs\default_ccs_script.js

default_ccs_script.js

default_ccs_script.js 파일을 수정한 후, 타깃 환경 설정을 다시 적용 버튼을 클릭해야만 프로젝트에 해당 스크립트 내용이 반영됩니다.

그 후, 프로젝트 경로에 있는 ide.c를 교체해주어야 합니다.

ide.c 경로 : Controller Tester Workspace 경로\Project 이름\.csdata\target\io\ide.c

ide.c

2.4. Other tips

- .map file
- TRACE32 debugging

2.4.1. .map file

After building the target software, a .map file may also be created in the path where the binaries are created.
The file contains symbol information for the target binary. You can check whether the main function has been replaced normally, or check the function symbols contained in the binary file.

2.4.2. TRACE32 debugging

If you use TRACE32, you can debug your test binaries.

1. Run TRACE32 and open the cmm script file (start.cmm).
2. Run “Debug”, set a breakpoint after Go.HLL in the script, and run “Continue”.
3. When the break occurs, select [view]-[Var] to see the variable and function information currently running on the target.