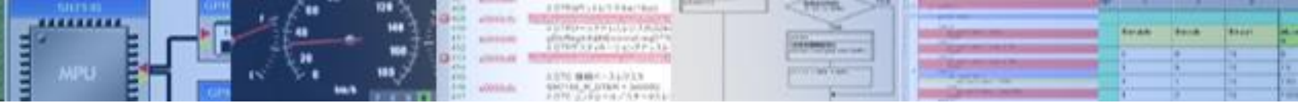


2016.01.07_V3.5

Known Issues: Coverage Measurement with Hook Code

This document is a list of know issues (and their solutions) when measuring Code Coverage with the Object with Hook Code that has been successfully built.

For information on how to set up and use the Object with Hook Code, please refer to the "[Application] Measuring Coverage by Hook Code" section of WinAMS Tutorial:

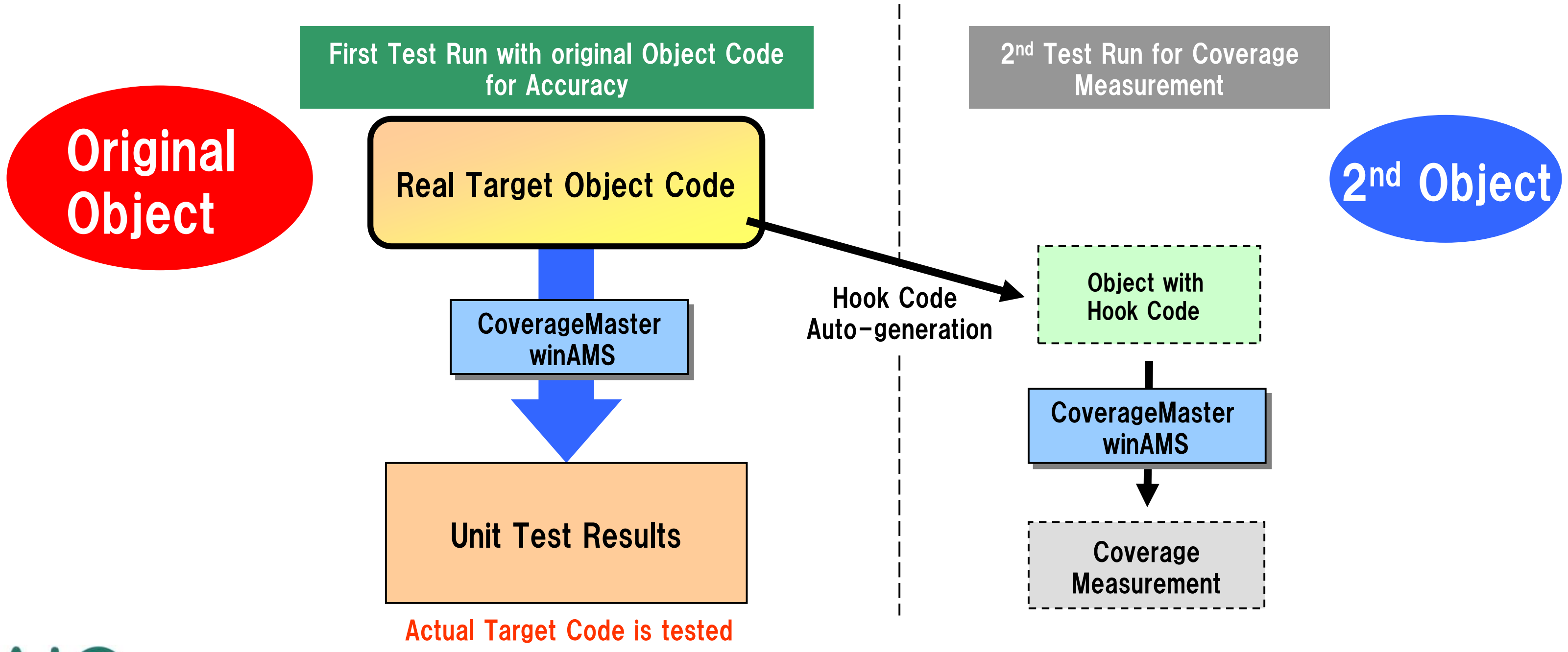
http://www.gaio.com/users/CM1_5EB42ZKXSZDX/CoverageMaster_Tutorial_Eng.pdf

Outline①

About Coverage Measurement with Hook Code

Two different objects

- **A second Object with inserted Hook Code is built for Coverage Measurement**
  - C1 is measured automatically; C0 can be measured with Hook Code as well
  - MC/DC and Function Call Coverage as additional options
- **Independent from the Compiler's settings and optimizations**

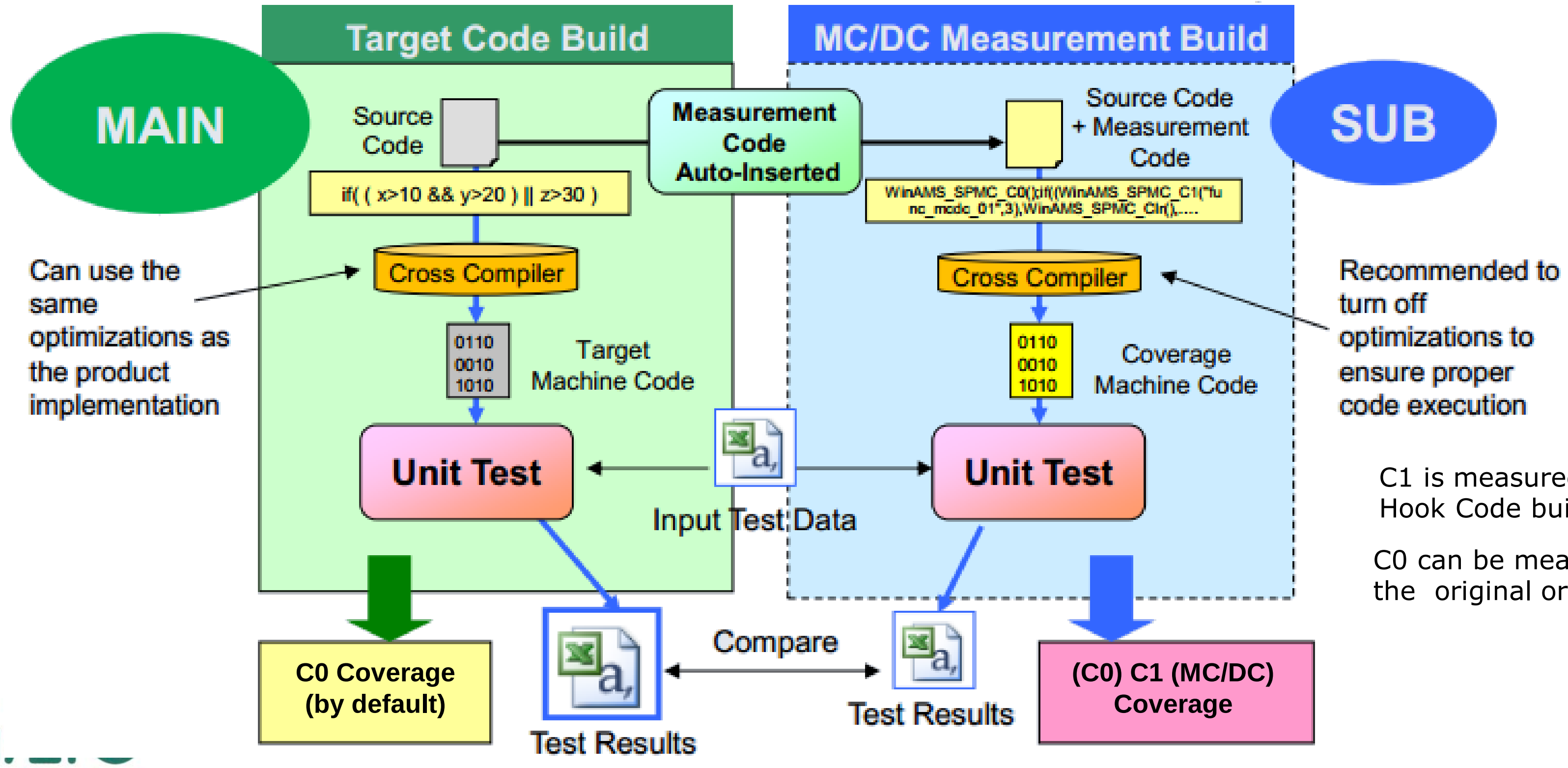


GAIO TECHNOLOGY

How Coverage with Hook Code is measured

- **The test is run twice: original Target Object and Object with Hook Code**

  -Main run: the unit test results are acquired from the unmodified target source code and compared with the expected results
  -Sub run: coverage measurement code inserted into the source code is used for measuring coverage
  -The results of the two executions are compared for accuracy


Target Code Build
MC/DC Measurement Build
MAIN
SUB
Source Code
if((x>10 && y>20) || z>30)
Measurement Code Auto-Inserted
Source Code + Measurement Code
WinAMS_SPMC_C0();if((WinAMS_SPMC_C1("func_mcdc_01",3),WinAMS_SPMC_Clr(),....
Can use the same optimizations as the product implementation
Cross Compiler
Cross Compiler
Recommended to turn off optimizations to ensure proper code execution
0110 0010 1010
Target Machine Code
0110 0010 1010
Coverage Machine Code
Unit Test
Input Test Data
Unit Test
C0 Coverage (by default)
Test Results
Compare
Test Results
(C0) C1 (MC/DC) Coverage

C1 is measured from the Hook Code build when it exists

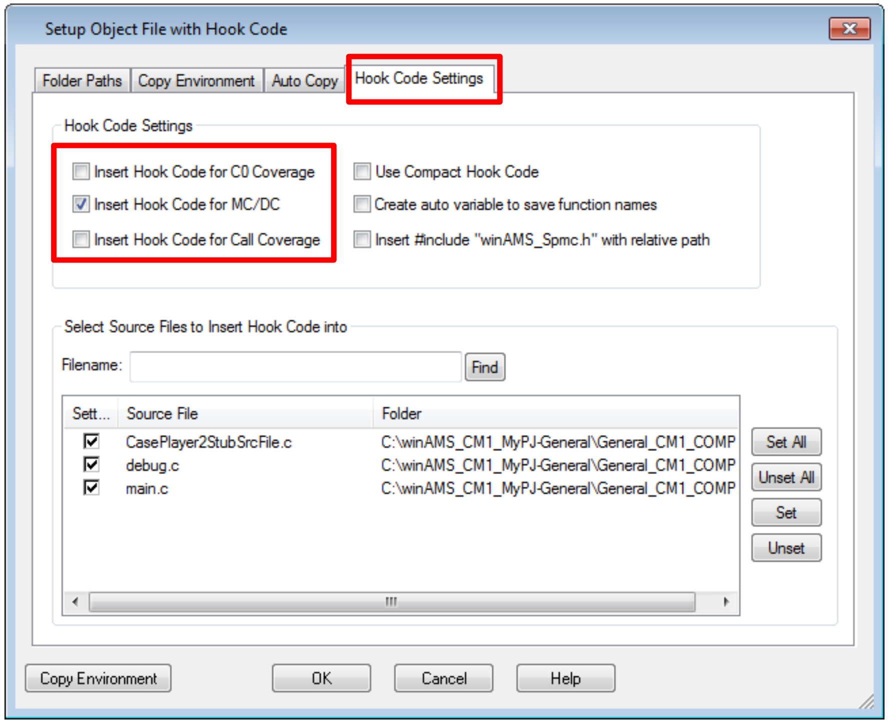
C0 can be measured from either the original or Hook Code build

The various Coverage measures

The various Coverage measures

- **The following Coverage measurements can be performed with Hook Code:**
  - C0, MC/DC, Function Call Coverage
    (Optional License required for MC/DC and Function Call Coverage)
  - C1 is automatically measured with Hook Code
  - If C0 is not selected, it is measured from the original Target Object

CasePlayer2's Project menu->
Setup Object File with Hook Code:



Hook Code Settings

- [ ] Insert Hook Code for C0 Coverage
- [✓] Insert Hook Code for MC/DC
- [ ] Insert Hook Code for Call Coverage
- [ ] Use Compact Hook Code
- [ ] Create auto variable to save function names
- [ ] Insert #include "winAMS_Spmc.h" with relative path

Select Source Files to Insert Hook Code into

Filename:

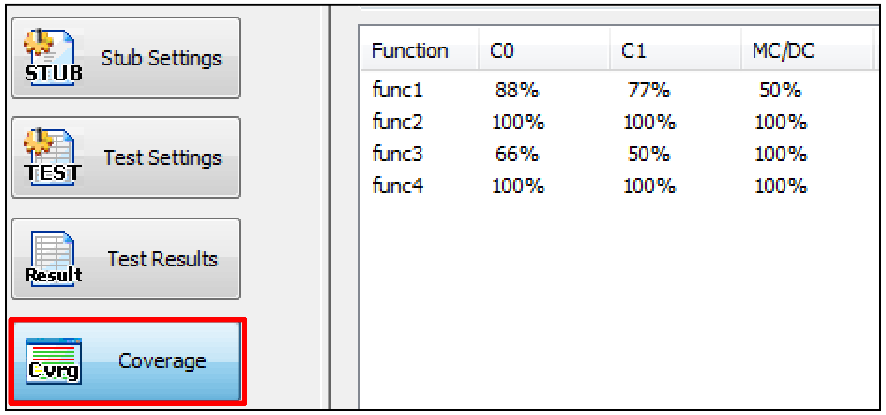
Sett...	Source File	Folder
✓	CasePlayer2StubSrcFile.c	C:\winAMS_CM1_MyPJ-General\General_CM1_COMP
✓	debug.c	C:\winAMS_CM1_MyPJ-General\General_CM1_COMP
✓	main.c	C:\winAMS_CM1_MyPJ-General\General_CM1_COMP

GAIO TECHNOLOGY

Common Issues with Coverage Measurement

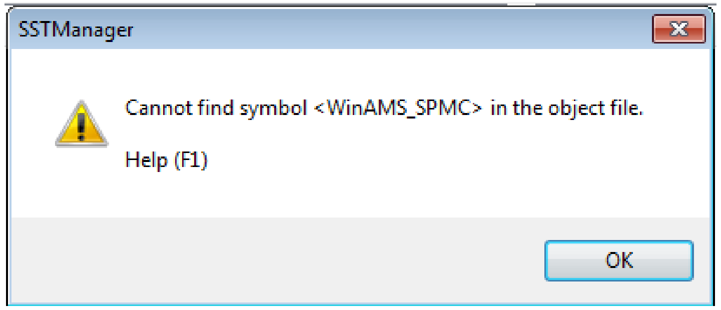
Common Issues with Coverage Measurement

- **Correct Measurement:**



Function	C0	C1	MC/DC
func1	88%	77%	50%
func2	100%	100%	100%
func3	66%	50%	100%
func4	100%	100%	100%

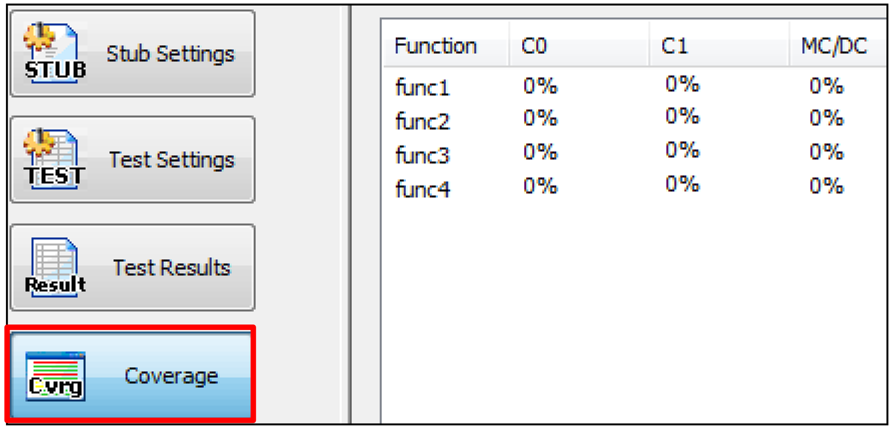
- **Incorrect Measurements:**


SSTManager

Cannot find symbol <WinAMS_SPMC> in the object file.

Help (F1)

Usual causes: Original Object is set instead of the Object with Hook Code, or Object with Hook Code wasn't built properly, or the Build Environment Copy wasn't done properly



Function	C0	C1	MC/DC
func1	0%	0%	0%
func2	0%	0%	0%
func3	0%	0%	0%
func4	0%	0%	0%

Usual cause: Architecture-specific. The header files for the Coverage measurement functions must be edited according to the target MPU.

Outline of Common causes

Outline of Common causes①

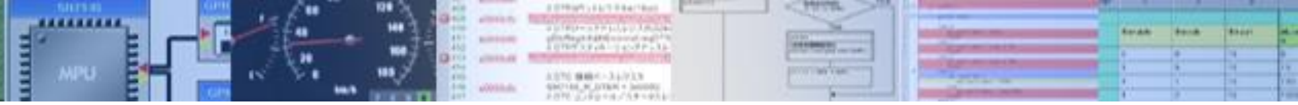
■ **3 usual reasons (① below is the most common)**

① **The strings containing the function's name or a file path used as parameters of the Coverage measurement functions are not passed properly**
->In this case C1 and MC/DC are 0%

A) Because some architecture-specific feature (ex: Memory Mirroring), the string isn't passed correctly to the Coverage measurement function.
Affected MPUs:
FMC16LX, FMC16FX, RL78(Core S2,S3), C2000(C28x), M16C, TX03(ARM Cortex-M3)

B) Due to incomplete initialization of the MPU, the Base Address for constant data is incorrect and the string isn't passed correctly
Affected MPUs:
e200z710/e200z410/e200z425, V850E2M, RH850

② **The variables used for Coverage Measurement are re-initialized during execution**
->In this case MC/DC is 0% but C1 is correct



Outline of Common causes②

■ **3 usual reasons (① is the most common)**

③ MPU/Compiler specific issues

A) A wrong LSB/MSB setting causing incorrect execution path
->In this case MC/DC is 0% but C1 is correct

- ➢ Affected MPU/Compiler:
  V850/GHS, V850E2M/GHS

B) Compile optimization causing incorrect Coverage measurement on the original Target Object
->In this case C0 is 0%

- ➢ Affected MPU(Compiler):
  R32C(Hew)

【①A】 Const string not passing

String not passing: Preliminary Info

- **For details on the *const* string types and Hook Code, refer to the CoverageMaster winAMS Manual:**

Help->Technical Information->Object with Hook Code Coverage Test->Object with Hook Code Setup

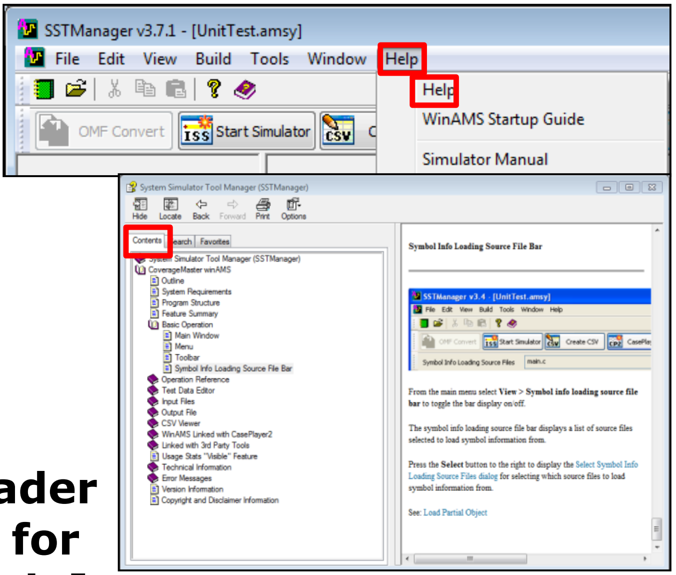
-Without Compact Hook Code:

[Coverage measurement Header Files]

[Changing the character string type of the code for coverage measurement]

-With Compact Hook Code:

[Changing the Coverage measurement Header Files and the character string type of the code for coverage measurement with Compact Hook Code]


SSTManager v3.7.1 - [UnitTest.amsy]
File Edit View Build Tools Window Help
Help
WinAMS Startup Guide
Simulator Manual
OMF Convert
Start Simulator
System Simulator Tool Manager (SSTManager)
Hide Locate Back Forward Print Options
Contents Search Favorites
System Simulator Tool Manager (SSTManager)
CoverageMaster winAMS
Outline
System Requirements
Program Structure
Feature Summary
Basic Operation
Main Window
Menu
Toolbar
Symbol Info Loading Source File Bar
Operation Reference
Test Data Editor
Input Files
Output File
CSV Viewer
WinAMS Linked with CasePlayer2
Linked with 3rd Party Tools
Usage Stats "Visible" Feature
Technical Information
Error Messages
Version Information
Copyright and Disclaimer Information
SSTManager v3.4 - [UnitTest.amsy]
Create CSV
Symbol Info Loading Source Files
main.c

Symbol Info Loading Source File Bar

From the main menu select **View > Symbol info loading source file bar** to toggle the bar display on/off.

The symbol info loading source file bar displays a list of source files selected to load symbol information from.

Press the **Select** button to the right to display the Select Symbol Info Loading Source Files dialog for selecting which source files to load symbol information from.

See: Load Partial Object

String not passing: Preliminary Info 2

■ **Coverage Measurement functions: (string types in red)**

① Without Compact Hook Code(WinAMS_SPMC.c)
  ① C0: BOOL WinAMS_SPMC_C0(WinAMS_SPMC_TFUNCNAME funcname,WinAMS_SPMC_U4 line)
  ② C1: BOOL WinAMS_SPMC_C1(WinAMS_SPMC_TFUNCNAME funcname,U4 blkID)
  ③ MC/DC: BOOL WinAMS_SPMC_Res(WinAMS_SPMC_TFUNCNAME funcname,U4 resID,BOOL res,U2 expcnt,U4 blkID)
  ④ Function Call: void WinAMS_SPMC_CALL(WinAMS_SPMC_TFUNCNAME funcname,WinAMS_SPMC_U4 callID)

② With Compact Hook Code (WinAMS_SPMC_Com.c)
  ① C0: BOOL WinAMS_SPMC_Com(WinAMS_SPMC_TFILENAME_PTR file,U4 index)
  ② C1: BOOL WinAMS_SPMC_Com(WinAMS_SPMC_TFILENAME_PTR file,U4 index)
  ③ MC/DC: BOOL WinAMS_SPMC_Res_Com(WinAMS_SPMC_TFILENAME_PTR file,U4 index,BOOL res)
  ④ Function call:
  BOOL WinAMS_SPMC_Com(WinAMS_SPMC_TFILENAME_PTR file,U4 index)

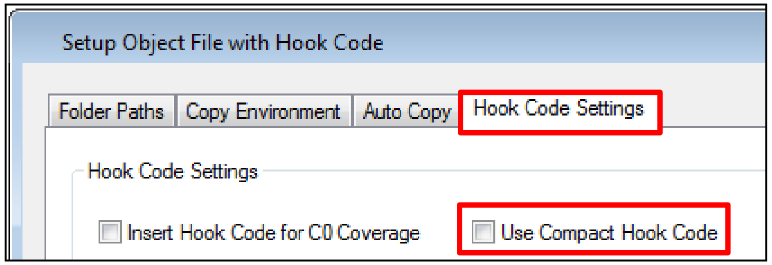
Const string not passing ①

Solution Outline:

1. **Confirm that the string's address is not passing properly**
2. **In this case, the header files for Code coverage Measurement functions must be edited**
   - ➢ With Compact Hook Code ON, the headers (and functions) differ

     With Compact Hook Code OFF : winAMS_Spmc.c, winAMS_Spmc.h, winAMS_SpmcDefine.h

     With Compact Hook Code ON: winAMS_Spmc_Com.c, winAMS_Spmc_Com.h, winAMS_SpmcDefine_Com.h



Setup Object File with Hook Code

Folder Paths | Copy Environment | Auto Copy | Hook Code Settings

Hook Code Settings

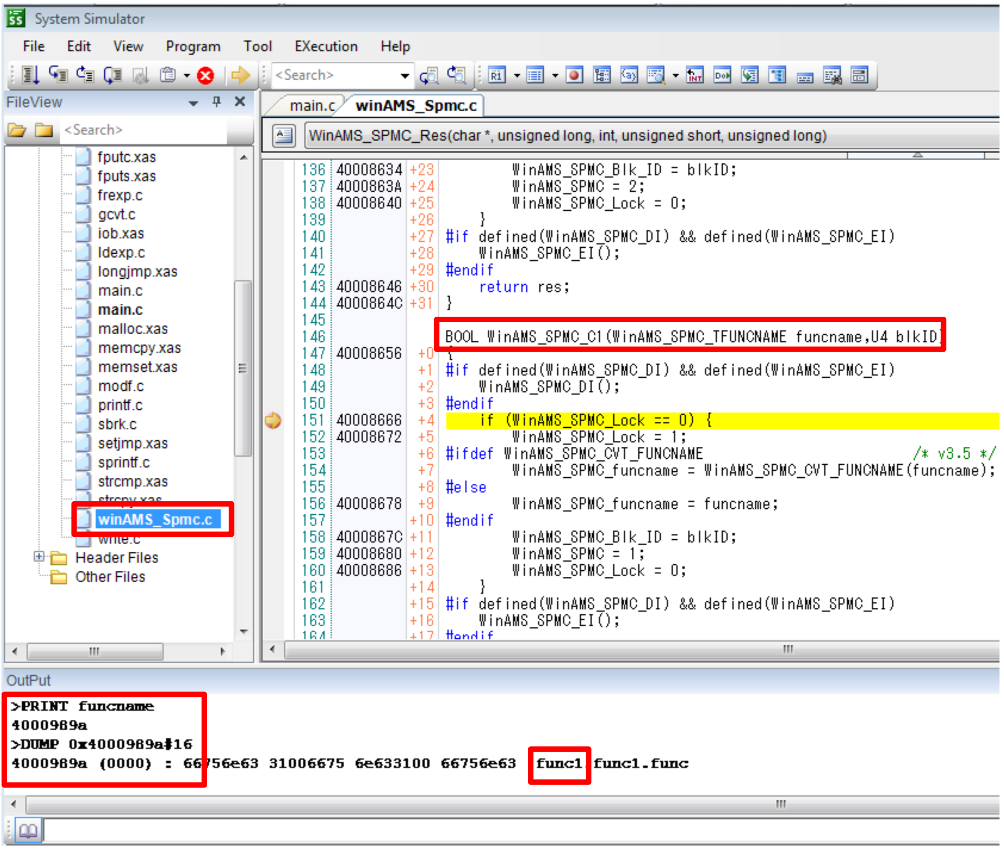
☐ Insert Hook Code for C0 Coverage

☐ Use Compact Hook Code

   - ➢ The original header files are in CasePlayer2's Install folder (under Template) and get copied to your Build Environment for Object with Hook Code. By modifying these original files, the changes will be automatically applied to all future projects.
     Ex: C:\Program Files\gaio\CasePlayer2\template\winAMS_Spmc.c
     C:\Program Files\gaio\CasePlayer2\template\winAMS_Spmc.h
     C:\Program Files\gaio\CasePlayer2\template\winAMS_SpmcDefine.h
3. **Rebuild the Object with Hook Code with the edited header files**

Const string not passing ②

■ **How to check if the string is passed correctly, with Compact Hook Code OFF (Default setting)**



```
136 40008634 +23         WinAMS_SPMC_Blk_ID = blkID;
137 4000863A +24         WinAMS_SPMC = 2;
138 40008640 +25         WinAMS_SPMC_Lock = 0;
139          +26     }
140          +27 #if defined(WinAMS_SPMC_DI) && defined(WinAMS_SPMC_EI)
141          +28     WinAMS_SPMC_EI();
142          +29 #endif
143 40008646 +30     return res;
144 4000864C +31 }
145
146              BOOL WinAMS_SPMC_C1(WinAMS_SPMC_TFUNCNAME funcname,U4 blkID)
147 40008656  +0 {
148           +1 #if defined(WinAMS_SPMC_DI) && defined(WinAMS_SPMC_EI)
149           +2     WinAMS_SPMC_DI();
150           +3 #endif
151 40008666  +4     if (WinAMS_SPMC_Lock == 0) {
152 40008672  +5         WinAMS_SPMC_Lock = 1;
153           +6 #ifdef WinAMS_SPMC_CVT_FUNCNAME                          /* v3.5 */
154           +7         WinAMS_SPMC_funcname = WinAMS_SPMC_CVT_FUNCNAME(funcname);
155           +8 #else
156 40008678  +9         WinAMS_SPMC_funcname = funcname;
157          +10 #endif
158 4000867C +11         WinAMS_SPMC_Blk_ID = blkID;
159 40008680 +12         WinAMS_SPMC = 1;
160 40008686 +13         WinAMS_SPMC_Lock = 0;
161          +14     }
162          +15 #if defined(WinAMS_SPMC_DI) && defined(WinAMS_SPMC_EI)
163          +16     WinAMS_SPMC_EI();
164          +17 #endif
```

```
>PRINT funcname
4000989a
>DUMP 0x4000989a#16
4000989a (0000) : 66756e63 31006675 6e633100 66756e63  func1 func1.func
```

Check that the function's name is passed correctly to the Coverage Measurement function:

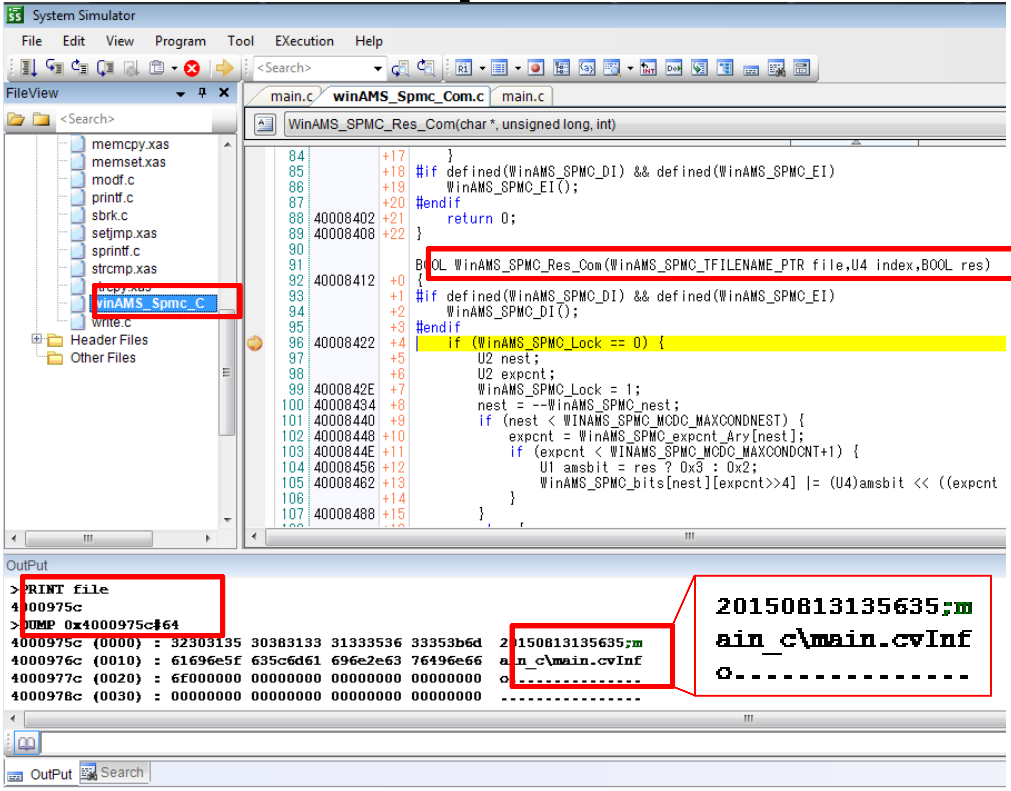
1. Run a test for your function (ex:func1) with the Object with Hook Code.
2. Set a breakpoint in the C1 Function WinAMS_SPMC_C1 in WinAMS_Spmc.c (ex: L151) and run the test to this point.
3. Run the [print funcname] command in the Console to see the string's address.
4. Then run the [dump] command followed by [0x] and the address found with "print funcname" in 3. and [#16] or [#32].
5. The content of the Memory at that address is displayed, and it should contain your function's name. If not, this is the cause of your issue.

Ex:

```
>print funcname
00050850
>dump 0x00050850#32(or 64, or 128)
00050850(0000):xxx xxx func1.func1.func1
```

Const string not passing ③

■ How to check if the string is passed correctly, with Compact Hook Code ON



```
         +17      }
         +18  #if defined(WinAMS_SPMC_DI) && defined(WinAMS_SPMC_EI)
         +19      WinAMS_SPMC_EI();
         +20  #endif
40008402 +21      return 0;
40008408 +22  }

              BOOL WinAMS_SPMC_Res_Com(WinAMS_SPMC_TFILENAME_PTR file,U4 index,BOOL res)
40008412  +0  {
          +1  #if defined(WinAMS_SPMC_DI) && defined(WinAMS_SPMC_EI)
          +2      WinAMS_SPMC_DI();
          +3  #endif
40008422  +4      if (WinAMS_SPMC_Lock == 0) {
          +5          U2 nest;
          +6          U2 expcnt;
4000842E  +7          WinAMS_SPMC_Lock = 1;
40008434  +8          nest = --WinAMS_SPMC_nest;
40008440  +9          if (nest < WINAMS_SPMC_MCDC_MAXCONDNEST) {
40008448 +10              expcnt = WinAMS_SPMC_expcnt_Ary[nest];
4000844E +11              if (expcnt < WINAMS_SPMC_MCDC_MAXCONDCNT+1) {
40008456 +12                  U1 amsbit = res ? 0x3 : 0x2;
40008462 +13                  WinAMS_SPMC_bits[nest][expcnt>>4] |= (U4)amsbit << ((expcnt
         +14              }
40008488 +15          }
```

```
>PRINT file
4000975c
>DUMP 0x4000975c#64
4000975c (0000) : 32303135 30383133 31333536 33353b6d  20150813135635;m
4000976c (0010) : 61696e5f 635c6d61 696e2e63 76496e66  ain_c\main.cvInf
4000977c (0020) : 6f000000 00000000 00000000 00000000  o...............
4000978c (0030) : 00000000 00000000 00000000 00000000  ................
```

```
20150813135635;m
ain_c\main.cvInf
o...............
```

Check that the function's name is passed correctly to the Coverage Measurement function:

1. Run a test for your function (ex:func1) with the Object with Hook Code.
2. Set a breakpoint in the WinAMS_SPMC_Com function in WinAMS_Spmc_Com.c (ex: L135) and run the test to this point.
3. Run the [print file] command in the Console to see the string's address.
4. Then run the [dump] command followed by [0x] and the address found with "print file" in 3. and [#16] or [#32].
5. The content of the Memory at that address is displayed, and it should contain the following: YYYYMMDDHHMM;filename_c\filename.cvInfo
 If not, this is the cause of your issue.

Ex:

```
>print file
00050850
>dump 0x00050850#64(or 128)
00050850(0000):xxx xxx 20150324155727;m
00050860(0010):xxx xxx ain_c\main.cvInf
00050870(0020):xxx xxx o
```

【①A】
Const string not passing
MPU: FMC16LX

MPU: FMC16LX ①

■ Cause

① In Small and Medium models, the address for the first parameter of the Coverage measurement function (the string) is in the ROM Mirror Area (0x8000-0xFFFF),

② but the actual string is stored in the non-mirror ROM area (0xFF8000-0xFFFFFF).

③ In the default configuration the Coverage Measurement function dereferences the pointer to the Mirror Area and doesn't access the correct string for the function name or File path.

■ Solution

① Modify the definition of the Coverage Measurement function

② Recompile the Object with Hook Code with the edited files

MPU: FMC16LX ②

■ Solution: With Compact Hook Code OFF

Modify [winAMS_SpmcDefine.h] as below (modified lines in red)

```
/*【Reference】Add :This declares the function parameters as const(for winAMS_Spmc.h) */
/*【Reference】Add :If this does not solve the issue, try to not add const(SW_WinAMS_SPMC_const_funcname 0) */
#define SW_WinAMS_SPMC_const_funcname 1
#if SW_WinAMS_SPMC_const_funcname
#define WinAMS_SPMC_const_funcname 1
#endif

/* A user defines the type of the string of the function name */
/*【Reference】Mod : Change WINAMS_SPMC_USR_DEF_TFUNCNAME from 0 to 1 */
#define WINAMS_SPMC_USR_DEF_TFUNCNAME 1                  /* 0:not define, 1:define */
#if WINAMS_SPMC_USR_DEF_TFUNCNAME

/* example */
/*【Reference】Mod :Change WinAMS_SPMC_BASE_TFUNCNAME from char to char __far */
//#define WinAMS_SPMC_BASE_TFUNCNAME char                /* base type=char */
/* #define WinAMS_SPMC_BASE_TFUNCNAME signed char */     /* base type=signed char */
/* #define WinAMS_SPMC_BASE_TFUNCNAME unsigned char */   /* base type=unsigned char */
#define WinAMS_SPMC_BASE_TFUNCNAME char __far            /* base type=char __far */

/*【Reference】Mod : Change fname to match with the ROM Mirroring feature (add 0xFF) */
//#define WinAMS_SPMC_CVT_FUNCNAME(fname)  (fname)       /* funcname pointer convert */
#define WinAMS_SPMC_CVT_FUNCNAME(fname) ((WinAMS_SPMC_TFUNCNAME)((unsigned long)(fname) | 0xff0000))
```

sample1

MPU: FMC16LX ③

■ Solution: With Compact Hook Code ON

Modify [winAMS_SpmcDefine_Com.h] as below (modified lines in red)

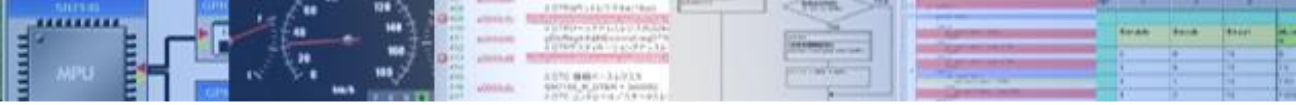
```
/* A user defines the type of the string of the file name */
/*【Reference】Mod : Change WinAMS_SPMC_BASE_TFILENAME from char to char __far */
//#define WinAMS_SPMC_BASE_TFILENAME char                /* base type=char */
/* #define WinAMS_SPMC_BASE_TFILENAME signed char */      /* base type=signed char */
/* #define WinAMS_SPMC_BASE_TFILENAME unsigned char */    /* base type=unsigned char */
/*【Reference】Add  :Change WinAMS_SPMC_BASE_TFUNCNAME from char to char __far*/
#define WinAMS_SPMC_BASE_TFILENAME char __far            /* base type=char __far */

#define WinAMS_SPMC_TABLE_PTR_TATTR1                     /* type tatt1 *name */
#define WinAMS_SPMC_TABLE_PTR_TATTR2                     /* type *tatt2 name */
/* #define WinAMS_SPMC_TABLE_PTR_TATTR1 far */           /* example */
#define WinAMS_SPMC_TABLE_TATTR                          /* type tatt name[] */
/* #define WinAMS_SPMC_TABLE_TATTR far */                /* example */

/*【Reference】Mod : Change fname to match with the ROM Mirroring feature (add 0xFF)  */
//#define WinAMS_SPMC_CVT_TABLE_PTR(fname) (fname)       /* table pointer convert */
#define WinAMS_SPMC_CVT_TABLE_PTR(fname) ((WinAMS_SPMC_TFILENAME_PTR)((unsigned long)(fname) |
0xff0000))          /* example */

#define WinAMS_SPMC_CONST const /* const table */
/* #define WinAMS_SPMC_CONST */ /* not const table */
```

sample2



【①A】
Const string not passing
MPU: FMC16FX

MPU: FMC16FX ①

Cause

① In Small and Medium models, the address for the first parameter of the Coverage measurement function (the string) is in the ROM Mirror Area (0x8000-0xFFFF),

② but the actual string is stored in the non-mirror ROM area (0xFn8000-0xFnFFFF), with n between 0x0 and 0xF depending on the Mirror Bank (selected with bits 4 to 7 of the ROM Mirror Control Register ROMM)

[Note]: When ROM Mirroring is off (bit0 of the ROMM is 0), the string might be stored in a different area. Verify in your Map File the first byte of the Addresses in the CONST Data area.
Ex: In the example on the right,
it is 0xFE.

```
MAP File Example (*.mp1)
00000180-0000018F  00000010  DATA  N RW-- 01 ABS  Register Bank No. 00
                                        …
00DF2002-........  00000000  CONST P R--I 02 REL  DCLEAR
00FE0000-00FE0403  00000404  CONST P R--I 02 REL  CONST
00FF0000-00FF0FE8  00000FE9  CODE  P R-XI 01 REL  CODE
```

③ In the default configuration the Coverage Measurement function dereferences the pointer to the Mirror Area and doesn't access the correct string for the function name or File path.

Solution

① Modify the definition of the Coverage Measurement function

② Recompile the Object with Hook Code with the edited files

MPU: FMC16FX ②

■ **Solution: With Compact Hook Code OFF**

Modify [winAMS_SpmcDefine.h] as below (modified lines in red)

```
/*【Reference】Add :This declares the function parameters as const(for winAMS_Spmc.h) */
/*【Reference】Add :If this does not solve the issue, try to not add const(SW_WinAMS_SPMC_const_funcname 0) */
#define SW_WinAMS_SPMC_const_funcname 1
#if SW_WinAMS_SPMC_const_funcname
#define WinAMS_SPMC_const_funcname 1
#endif

/* A user defines the type of the string of the function name */
/*【Reference】Mod : Change WINAMS_SPMC_USR_DEF_TFUNCNAME from 0 to 1 */
#define WINAMS_SPMC_USR_DEF_TFUNCNAME 1                  /* 0:not define, 1:define */
#if WINAMS_SPMC_USR_DEF_TFUNCNAME

/* example */
/*【Reference】Mod :Change WinAMS_SPMC_BASE_TFUNCNAME from char to char __far */
//#define WinAMS_SPMC_BASE_TFUNCNAME char                 /* base type=char */
/* #define WinAMS_SPMC_BASE_TFUNCNAME signed char */      /* base type=signed char */
/* #define WinAMS_SPMC_BASE_TFUNCNAME unsigned char */    /* base type=unsigned char */
#define WinAMS_SPMC_BASE_TFUNCNAME char __far             /* base type=char __far */

/*【Reference】Mod : Change fname to match with the ROM Mirroring feature */
//#define WinAMS_SPMC_CVT_FUNCNAME(fname)  (fname)        /* funcname pointer convert */
/*【Reference】Mod :Depending on the Mirror Bank (selected with bits 4 to 7 of the ROM Mirror Control Register ROMM,*/
/*the 2MSB of 0xff0000 can vary from 0xf0 to 0xff. Modify them according to your setup. In this example, 0xFF(ROMM bits=1111) is used. */
#define WinAMS_SPMC_CVT_FUNCNAME(fname) ((WinAMS_SPMC_TFUNCNAME)((unsigned long)(fname) | 0xff0000))
```

sample3

MPU: FMC16FX ③

■ Solution: With Compact Hook Code ON

Modify [winAMS_SpmcDefine_Com.h] as below (modified lines in red)

```
/* A user defines the type of the string of the file name */
/*【Reference】Mod : Change WinAMS_SPMC_BASE_TFILENAME from char to char __far */
//#define WinAMS_SPMC_BASE_TFILENAME char                  /* base type=char */
/* #define WinAMS_SPMC_BASE_TFILENAME signed char */       /* base type=signed char */
/* #define WinAMS_SPMC_BASE_TFILENAME unsigned char */     /* base type=unsigned char */
/*【Reference】Add : Change WinAMS_SPMC_BASE_TFUNCNAME from char to char __far */
#define WinAMS_SPMC_BASE_TFILENAME char __far              /* base type=char __far */

#define WinAMS_SPMC_TABLE_PTR_TATTR1                       /* type tatt1 *name */
#define WinAMS_SPMC_TABLE_PTR_TATTR2                       /* type *tatt2 name */
/* #define WinAMS_SPMC_TABLE_PTR_TATTR1 far */             /* example */
#define WinAMS_SPMC_TABLE_TATTR                            /* type tatt name[] */
/* #define WinAMS_SPMC_TABLE_TATTR far */                  /* example */

/*【Reference】Mod : Change fname to match with the ROM Mirroring feature */
//#define WinAMS_SPMC_CVT_TABLE_PTR(fname) (fname)         /* table pointer convert */
/*【Reference】Mod :Depending on the Mirror Bank (selected with bits 4 to 7 of the ROM Mirror Control Register ROMM,*/
/*the 2MSB of 0xff0000 can vary from 0xf0 to 0xff. Modify it according to your setup. in this example, 0xFF(ROMM bits=1111) is
used. */
#define WinAMS_SPMC_CVT_TABLE_PTR(fname) ((WinAMS_SPMC_TFILENAME_PTR)((unsigned long)(fname) |
0xff0000))        /* example */

#define WinAMS_SPMC_CONST const /* const table */
/* #define WinAMS_SPMC_CONST */ /* not const table */
```

sample4

【①A】 Const string not passing MPU: RL78 (Core S2,S3)

MPU: RL78 (Core S2,S3) ①

Cause

① In Small and Medium models, the first parameter of the Coverage measurement function (the string) only contains a 16bits long address(0x0000-0xFFFF),

② but the actual string is stored in the non-mirror ROM area (0xn0000-0xnFFFF), with n being 0 or 1 according to the MAA bit(=Bit0) of the Processor Mode Control (PMC) Register (Address=0xFFFFE).

[Note]: For S1 cores, MAA is necessarily 0 and the Mirror area is fixed (0x00000-0x05EFF is mirrored to 0xF8000-0xFDEFF)

③ In the default configuration the Coverage Measurement function dereferences the 16bits pointer to the 0x0000-0xFFFF area, and if MAA=1 then it doesn't access the correct string for the function name or File path. (If MAA=0 thee is no problem)

Solution

① Modify the definition of the Coverage Measurement function

② Recompile the Object with Hook Code with the edited files

MPU: RL78 (Core S2,S3) ②

■ Solution: With Compact Hook Code OFF

Modify [winAMS_Spmc.c] as below (modified lines in red)

```
#ifdef WinAMS_SPMC_const_funcname
WinAMS_SPMC_BASE_TFUNCNAME const *volatile WinAMS_SPMC_funcname;
#else
/*【Reference】Mod : Change WinAMS_SPMC_funcname's type to long so that it gets a 3bytes pointer*/
//WinAMS_SPMC_BASE_TFUNCNAME *volatile WinAMS_SPMC_funcname;
unsigned long volatile WinAMS_SPMC_funcname;
#endif /* WinAMS_SPMC_const_funcname */
```

sample5

Modify [winAMS_SpmcDefine.h] as below (modified lines in red)

```
/*【Reference】Mod : Change WINAMS_SPMC_USR_DEF_TFUNCNAME from 0 to 1*/
#define WINAMS_SPMC_USR_DEF_TFUNCNAME 1                 /* 0:not define, 1:define */
#if WINAMS_SPMC_USR_DEF_TFUNCNAME
/* example */
#define WinAMS_SPMC_BASE_TFUNCNAME char                 /* base type=char */
/* #define WinAMS_SPMC_BASE_TFUNCNAME signed char */    /* base type=signed char */
/* #define WinAMS_SPMC_BASE_TFUNCNAME unsigned char */  /* base type=unsigned char */
/* #define WinAMS_SPMC_BASE_TFUNCNAME char __far */     /* base type=char __far */

/*【Reference】Mod : Change fname to match with the Mirroring feature if PMC=1 (MAA=1)*/
//#define WinAMS_SPMC_CVT_FUNCNAME(fname) (fname)       /* funcname pointer convert */
#define WinAMS_SPMC_CVT_FUNCNAME(fname) ((unsigned long)(fname) & 0x00ffff | 0x010000)
```

sample6

MPU: RL78 (Core S2,S3) ③

■ Solution: With Compact Hook Code ON

Modify [winAMS_Spmc_Com.c] as below (modified lines in red)

sample7

```
#if __COMPILER_FCC907__
volatile WinAMS_SPMC_CONST WinAMS_SPMC_BASE_TFILENAME
WinAMS_SPMC_TABLE_PTR_TATTR1*WinAMS_SPMC_TABLE_PTR_TATTR2 WinAMS_SPMC_filename;
#else
/*【Reference】Mod : Change WinAMS_SPMC_funcname's type to long so that it gets a 3bytes pointer */
/* WinAMS_SPMC_CONST WinAMS_SPMC_BASE_TFILENAME
WinAMS_SPMC_TABLE_PTR_TATTR1*WinAMS_SPMC_TABLE_PTR_TATTR2 volatile WinAMS_SPMC_filename; */
unsigned long     volatile WinAMS_SPMC_funcname;
//volatile long WinAMS_SPMC_filename; ↑上記で問題が解決しない場合は、こちらを利用する
#endif /* __COMPILER_FCC907__ */
```

Modify [winAMS_SpmcDefine_Com.h] as below (modified lines in red)

sample8

```
/*【Reference】Mod : Change fname to match with the Mirroring feature if PMC=1 (MAA=1) */
//#define WinAMS_SPMC_CVT_TABLE_PTR(fname) (fname)                    /* table pointer convert */
#define WinAMS_SPMC_CVT_TABLE_PTR(fname) ((unsigned long)(fname) & 0x00ffff | 0x010000)
```

【①A】 Const string not passing MPU: M16C

MPU: M16C ①

Cause

① With Compact Hook Code OFF, the first parameter of the Coverage measurement function (the string) is a char* type and is treated as a near pointer,

[Note]: With Compact Hook Code ON, it becomes a const char* type which is treated as a far pointer, and no issues occur.

② but the actual string is stored in a far area (0xCXXXX-0xCXXXX)

③ In the default configuration the Coverage Measurement function dereferences the near pointer to the 0x0000-0xFFFF area, and doesn't access the correct string for the function name or File path.

Solution

① Modify the definition of the Coverage Measurement function
② Recompile the Object with Hook Code with the edited files

MPU: M16C ②

■ Solution: With Compact Hook Code OFF

Modify [winAMS_Spmc.h] as below (modified lines in red)

```
/*【Reference】Add : Define WinAMS_SPMC_const_funcname to change near pointers into far pointers */
#define WinAMS_SPMC_const_funcname

#if WINAMS_SPMC_USR_DEF_TFUNCNAME
#ifdef WinAMS_SPMC_const_funcname
typedef const WinAMS_SPMC_BASE_TFUNCNAME * WinAMS_SPMC_TFUNCNAME;
#else
typedef WinAMS_SPMC_BASE_TFUNCNAME * WinAMS_SPMC_TFUNCNAME;
#endif /* WinAMS_SPMC_const_funcname */
#else
#ifdef WinAMS_SPMC_const_funcname
typedef const char * WinAMS_SPMC_TFUNCNAME;    /*【Reference】const changes the near pointer into a far pointer */
#else
typedef char * WinAMS_SPMC_TFUNCNAME;
#endif /* WinAMS_SPMC_const_funcname */
#endif /* WINAMS_SPMC_USR_DEF_TFUNCNAME */
```

sample10

【①A】 Const string not passing MPU: TX03 (ARM Cortex-M3)

MPU: TX03（ARM Cortex-M3）①

■ Cause

① The first parameter of the Coverage measurement function (the string) is located on the RAM Area (0xFFXXXX-0xFFXXXX),

② but the actual string is stored in the ROM area (0xXXXX-0xXXXX)

③ In the default configuration the Coverage Measurement function accesses the RAM area and doesn't access the correct string for the function name or File path.

■ Solution

① In the Startup Command file, copy the ROM content into the RAM area.
[Note]: The Mirror feature is not implemented in the Simulation Engine.
Copying the ROM is also needed when standard (non Coverage-related) *const* data is accessed.

■Startup Command File example:

```
;Copy the content of 0x1000-0x10ff to 0xFF2000-0xFF20ff
COPY MEMORY 0x1000#0x100 0xFF2000
```

【①B】
Const string not passing
MPU: e200zxxx Series

MPU: e200zxxx Series ①

Possible Cause①

① The Base Register R13 (Base pointer for the Read-only Small Data Area (SDA)) has not been initialized.

② The first parameter of the Coverage measurement function (the string) is located in the SDA but is not accessed correctly

Possible Cause②

① The Base Register R2 (Base pointer for the Read-write Small Data Area 2(SDA2)) has not been initialized.

② The first parameter of the Coverage measurement function (the string) is located in the SDA but is not accessed correctly

Solution

① Run the Startup Routine of your application so that R2 and R13 are initialized properly

■For more details, see the following FAQ:

http://www.gaio.com/support/user/faq/winams/faq_011_03.html

GAIO TECHNOLOGY

【①B】
Const string not passing
MPU: V850E2M/RH850

MPU: V850E2M/RH850 ①

■ Cause

① The Base Register GP (Global Pointer) has not been initialized.

② The first parameter of the Coverage measurement function (the string) is located in the Global Data Area but is not accessed correctly.

■ Solution

① Run the Startup Routine of your test target application and set the base register R4[GP: Global pointer] / R5[TP: Text Pointer / R30 [EP: Element Pointer] ）

■For more details, see the following FAQ:
http://www.gaio.com/support/user/faq/winams/faq_011_03.html

【②】 Coverage Measurement variables are over-written

Coverage Measurement variables are over-written

■ **Possible Causes**
(In this case MC/DC is 0% but C1 is correct)

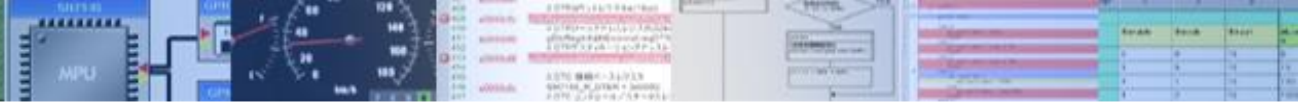
① When the Function under Test is called through a Test Driver, the variables for MC/DC Coverage measurement (WinAMS_SPMC_maxCondCnt, WinAMS_SPMC_maxCondNest) are re-initialized in the Test Driver.

② If there are a lot of other local variables initialized in the Test Driver, they fill up the Stack Area and the MC/DC variables are over-written

■ **Solution**

① 1. Save the MC/DC variables before the initialization of other local variables and restore their value just before calling the function under test.
Or:
2. Modify the Link Parameters and recompile the object so that the MC/DC variables are not in the area for local variables

② Modify the Stack Pointer SP so that there is sufficient Stack space



【③A】 MPU/Compiler specific Issues V850/GHS

MPU/Compiler specific Issues (V850/GHS)

■ Cause (In this case MC/DC is incorrect but C1 is correct)

① If the condition for an IF uses a Bit-field variable which is set in the CSV file, the bit order expected by the code is the opposite of the one set by CoverageMaster (based on the Debug Info created during the OMF Conversion and controlled by the [-LSB/-MSB] options, and the value is not correctly passed

■ Solution

① Depending on the Compiler's version, the default bit order changes. Set the OMF Converter’s option to [-LSB] or [-MSB] according to your Compiler's version:

[Note]: For OMF Converters for V850E2M and RH850, only [-MSB] is available

1. GHS[R3(LSB) / R7(LSB)]   OMF Option: None(=-LSB)
    1. Compile Option :-cpu v850f   MPU:v850(GHS) / V850E/GP1 Series
2. GHS[R8.1.3 and later: MSB] OMF Option: -MSB
    1. Compile Option :-cpu v850e1f   MPU:v850(GHS)   /V850E/GP1 Series
    2. Compile Option :-cpu v850e   MPU:v850E2M(GHS) / V850E2M Series
    3. Compile Option:-cpu v850e2   MPU:v850E2M(GHS) / V850E2M Series
    4. Compile Option :-cpu v850e2r   MPU:v850E2M(GHS) / V850E2M Series
    5. Compile Option :-cpu v850e2v3   MPU:v850E2M(GHS) / V850E2M Series

GAIO TECHNOLOGY

【③B】
MPU/Compiler specific Issues
R32C / NC100

MPU/Compiler specific Issues (R32C/NC100)

- **Cause**
  (In this case C0 is incorrect but C1 and MC/DC are correct)
  - ① Because of the Compiler's optimizations, Debug Info for several functions (ex: empty functions) is output in one section, and C0 coverage is measured on Code for several functions at the same time
- **Solution**
  - ① 1. Enable [Insert Hook Code for C0 Coverage]

    2. Turn the Compiler’s Optimizations OFF and recompile the Object with hook Code

END

For more information about our products: http://www.gaio.com/

‣ 中文 ‣ 日本語

Solution House GAIO TECHNOLOGY

The comprehensive embedded solution provider

SITE MAP HOME

Products | Seminars | News | User Support | GAIO CLUB | Company | Contact Us

Top >

Embedded Software and Model-based Development Solution Provider

Products and Services

Embedded Software Tools

GAIO develops original embedded development and test tools based on GAIO's own code analysis and microprocessor simulation technologies

GiS Global tool Interlink Solution

- Unit/Integration testing
- Model-based software testing
- Code analysis
- Functional safety compliance
- Cross compiler

Engineering Services

GAIO provides the following engineering services primarily for Japanese automotive companies.

MCD Model Centered Development

SAQT Solutions for Advancing Quality Technology

- Model-based development
- Software quality improvement
- Development process creation
- Unit test service
- Functional safety support
- Secure coding support

Functional Safety / Model-based Development Solutions

ISO26262 / IEC61508

ISO26262 Compliance Solutions

GAIO provides embedded software verification tools compliant with ISO 26262/IEC 61508. And we provide high quality embedded software engineering services to satisfy automotive functional safety standards ISO 26262.

MBD / MDD

Model-based Development Tool Solutions

GAIO provides comprehensive MBD/MDD verification solutions using the GIS concept to link with other company's tools. And we offers support services for establishing quality development processes and efficient tool introduction to meet the needs of the industry.

Latest News

2014.12.15 : GAIO will attend Embedded World 2015.

NEW GAIO will attend the exhibition show 'Embedded World 2015' held at Fachmesse in Nuremberg, Germany, Feb. 24th 2015.

More Info

embeddedworld Exhibition&Conference ... it's a smarter world

2014.9.1 : GAIO's offices have been relocated to new offices.

In order to support our rapid growth and business expansion, we are pleased to inform you that our offices

Shortcuts

MPU Support List

Product & Service Catalog

Information Links

embeddedworld Exhibition&Conference ... it's a smarter world

24 - 26 /Feb/2015
Exhibition Centre Nuremberg, Germany
GAIO's Booth : 4-603, Hall 4

Visit Us at GAIO's Booth!

EMBEDDED Tools BERATUNG VERTRIEB SUPPORT

Authorized Distributor in Germany, Austria and Switzerland

ISO 26262 Certified Unit Test Tool CoverageMaster

Model/Code Back-to-back Test Tool MC-Checker

Tweets from @GaioEurope

GAIO TECHNOLOGY CO.,LTD.

Sales Division

Tennouzu First Tower 25F
2-2-4 Higashi-Shinagawa, Shinagawa-ku,
Tokyo 140-0002 Japan

TEL: +81-3-4455-4767 Email: info@gaio.co.jp

* Company names and product names that appear in this presentation are trademarks of their respective company.
* Unauthorized distribution or duplication of this presentation material is prohibited.