Using a Dummy Function for Verification

CoverageMaster winAMS V3.3 or later

1. The Dummy Function

Normally when using CoverageMaster winAMS it is unnecessary to make changes to the application (C program) to be unit tested. However, in some cases the debug information that the cross compiler creates does not include the function's argument (parameter) information (e.g. argument order, argument bit width) required by CoverageMaster. When this occurs, the input values set in the test CSV file cannot be applied to the function's argument through the winAMS process.

As a workaround, a dummy function can be created for use as a "test driver" that calls the function to be tested. Through the dummy function, the desired input values can be set to the function's arguments.

Creating the Dummy Function

The dummy function can be created directly within the program source files. However, in order to avoid modifications to the program source code, we recommend creating the dummy function in a separate source file.

In the following example, the function func1() within the program source file will be tested using a dummy function created in a separate source file, dummy.c.

Program source file of function to be tested

```
struct ST_PARAM
 {
     int data:
     int ret_code;
 } gb_result;
▶void func1( int enable, int mode, int input ) /* target function to test */
{
     if( enable )
     {
              /* code for enable true */
     }
     else
     {
              gb_result.data = 0;
              gb_result.ret_code = FALSE;
     }
```

```
Dummy function source code (dummy.c)
```

```
struct DUMMY_ST_PARAM /* dummy function output variables */
{
 int data;
 int ret_code;
} DUMMY_gb_result;
int DUMMY_enable;
                       /* dummy function input variables */
int DUMMY_mode;
                         /* dummy function input variables */
int DUMMY_input;
                        /* dummy function input variables */
/* prototype declaration of target function to test */
extern void func1(int, int, int);
void DUMMY_func1_New(void) /* dummy function */
{
 - func1(DUMMY_enable, DUMMY_mode, DUMMY_input);
  DUMMY_gb_result.data = gb_result.data;
  DUMMY_gb_result.ret_code = gb_result.ret_code;
}
```

2. Create the Test CSV File and Test Data

Create a Unit Test CSV.

Select CSV File Type	—
Unit Test CSV	ОК
Start/End Symbol Test CSV	Capital
Start/End Address Test CSV	Cancer
Sequence Test CSV	Help
Automated Test CSV	
Class Test CSV	

Unit Test CSV Settings

- Set the dummy function "DUMMY_func1_New" as the "Function"
- Enable the "Using test driver" setting
- Set the function to be tested, "func1", as the "Measure coverage for function"
- Set the Input variables as the dummy function's input variables
- Set the output variables as the dummy function's output variables

Unit Test CSV Settings					
Filename: dummy_func1.csv	Test Description:				
Function: DUMMY_func1_New	Pass arguments through the stack				
Using test driver Measure coverage for function	on: func1				
Simulation time limit 1 b: Base Time					
Startup Command File:	Browse) Create				
Register Maps Defined Symbols All Variables Detected Variables Variable: Find Input Variables Find DUMMY_enable DUMMY_input DUMMY_input DUMMY_mode DUMMY_mode DUMMY_gb_result Output Variables Output Variables DUMMY_gb_result DUMMY_gb_result.data DUMMY_gb_result.ret_code Static Local Variables Function Parameter Function Retum List Options Show subfunction variables Hide unused struct members Show "const" data	Add ()-> Input Delete Add ()-> Edit Add I/O UMMY_mode Add I/O Up Down Output Down Output DUMMY_gb_result.data Delete DUMMY_gb_result.ret_code Edit Add I/O Up Down Check for Missing Clear Output DUMMY_gb_result.ret_code Edit Down Check for Missing Clear Down Check for Missing Clear Clear Check for Missing Clear Clear				
OK Initial Values Stub Set	ttings Enter Data Cancel <u>H</u> elp				

Test Data

_							
	🔁 ATDEditor	- [Edit CSV Dat	a]				
	Eile Edit	<u>V</u> iew <u>T</u> oo	ls <u>W</u> indow <u>H</u> elp				_ 8
	🔟 🔛 X	• • • •	' 🏢 🎇 🔛 😽 🗐	률 ☆ ☆ ♠	📪 🧇		
		Value	:			•	
		COMMENT	1	2	3	4	5
	COMMENT						
	NAME	Comment	DUMMY_enable	DUMMY_input	DUMMY_mode	DUMMY_gb_result.data	DUMMY_gb_result.ret_code
	1		1	101	3	10000	1
	2		0	101	3	0	0
	3		1	100	3	10000	1
	4		1	101	0	101	1
	5		1	101	1	1010	1
	6		1	101	2	10100	1
	7		1	101	4	-1	1

The input variables: "DUMMY_enable", "DUMMY_input", "DUMMY_mode" are the dummy variables used to assign test data values to the test function's (func1) arguments. Enter input data values into these dummy input variables.

The output variables: "DUMMY_gb_result.data" and "DUMMY_gb_result.ret_code" are the dummy variables to capture the test function's (func1) output. Enter the expected output values into these dummy output variables.

Note: the example image above uses the built-in ATDEditor. MS Excel can also be used to set the test data values.

3. Coverage Results and the Test Report

Even though the dummy function is executed, coverage will be measured for the function set in the "Measure coverage for function" setting as shown below.

Date Tested: 2015/07/23 14:07:52 Total Coverage C0: 100% C1: 100%								
Function	C0	C1		Other Functions	CO	C1		
func1	100%	100%		DUMMY_func1_New	100%	100%		

Test Report

CSV File name	Туре	Test Title	Function	Tested Function	Used Stub Function/Origin al Function	Test V ector	Tested Date	Exec. on Sim ulator	Exec. on Boa rd	Check
func1_dummy.csv	mod		func 1	DUMMY_func1_New		7	2015/07/23 14:07:53	2003 ms	0.0053 9 ms	No Ch eck

Coverage Report

Function	C0 Coverage	C1 Coverage	Coverage Log
func 1	100%	100%	TestCoverLog¥main.c¥func1&0001.txt
DUMMY_func1_New	100%	100%	TestCoverLog¥dummy.c¥DUMMY_func1_New&0006.txt



User Support Division Tennozu First Tower 25th floor 2-2-4 Higashi-Shinagawa, Shinagawa-ku, Tokyo 140-0002 Japan TEL. +81-3-4455-4767 URL: <u>https://www.en.gaio.co.jp/</u>