Activity: Unit Tests

Build and execute unit tests for Operating System Abstraction Layer (OSAL) and core Flight Executive (cFE). Create unit tests for a custom app and inspect the code coverage of the unit tests.

# Built the OSAL and cFE Unit Tests

|  |  |  |
| --- | --- | --- |
| 1 | Install lcov. This will provide a graphical front end to visualize code coverage of unit tests. | debian@beaglebone:~/cFS$ sudo apt-get install lcov  [sudo] password for debian:  Reading package lists... Done  Building dependency tree  Reading state information... Done  The following additional packages will be installed:  libgd-perl  The following NEW packages will be installed:  lcov libgd-perl  0 upgraded, 2 newly installed, 0 to remove and 18 not upgraded.  <snip>  Setting up libgd-perl (2.53-3) ...  Setting up lcov (1.13-1) ...  Processing triggers for man-db (2.7.6.1-2) ...  debian@beaglebone:~/cFS$ |
| 2 | Build the OSAL and cFE unit tests. Are there any errors? Address them. | debian@beaglebone:~/cFS$ make distclean  rm -rf "build"  debian@beaglebone:~/cFS$ make SIMULATION=native ENABLE\_UNIT\_TESTS=TRUE OSAL\_USER\_C\_FLAGS='-fprofile-arcs -ftest-coverage' prep  mkdir -p "build"  (cd "build/" && cmake -DCMAKE\_INSTALL\_PREFIX=/exe -DCMAKE\_BUILD\_TYPE=debug "/home/debian/cFS/cfe")  <snip>  echo " -DCMAKE\_INSTALL\_PREFIX=/exe -DCMAKE\_BUILD\_TYPE=debug" > "build/.prep"  debian@beaglebone:~/cFS$ make  make --no-print-directory -C "build" mission-all  Scanning dependencies of target elf2cfetbl  <snip>  debian@beaglebone:~/cFS$ |
| 3 | Run the cFE unit tests. | debian@beaglebone:~/cFS$ make test  lcov --capture --initial --directory build/native --output-file build/native/coverage\_base.info  Capturing coverage data from build/native  Found gcov version: 6.3.0  Scanning build/native for .gcno files ...  Found 236 graph files in build/native  Processing cpu1/CMakeFiles/core-cpu1.dir/src/target\_config.c.gcno  Processing cfe\_core\_default\_cpu1/unit-test/CMakeFiles/ut\_cfe\_sb\_object.dir/\_\_/src/sb/cfe\_sb\_priv.c.gcno  <snip>  Processing ut\_cfe\_core\_stubs/CMakeFiles/ut\_cfe-core\_stubs.dir/ut\_tbl\_stubs.c.gcno  Processing ut\_cfe\_core\_stubs/CMakeFiles/ut\_cfe-core\_stubs.dir/ut\_time\_stubs.c.gcno  Finished .info-file creation  (cd build/native && ctest -O ctest.log)  Test project /home/debian/cFS/build/native  Start 1: bin-sem-flush-test  1/23 Test #1: bin-sem-flush-test ............... Passed 6.04 sec  Start 2: bin-sem-test  2/23 Test #2: bin-sem-test ..................... Passed 10.05 sec  Start 3: bin-sem-timeout-test  3/23 Test #3: bin-sem-timeout-test ............. Passed 19.06 sec  Start 4: count-sem-test  4/23 Test #4: count-sem-test ................... Passed 12.56 sec  Start 5: file-api-test  5/23 Test #5: file-api-test .................... Passed 0.05 sec  Start 6: mutex-test  6/23 Test #6: mutex-test ....................... Passed 5.74 sec  Start 7: osal-core-test  7/23 Test #7: osal-core-test ................... Passed 0.81 sec  Start 8: queue-timeout-test  8/23 Test #8: queue-timeout-test ............... Passed 9.95 sec  Start 9: sem-speed-test  9/23 Test #9: sem-speed-test ................... Passed 5.05 sec  Start 10: symbol-api-test  10/23 Test #10: symbol-api-test .................. Passed 0.04 sec  Start 11: timer-test  11/23 Test #11: timer-test ....................... Passed 30.05 sec  Start 12: osal\_core\_UT  12/23 Test #12: osal\_core\_UT ..................... Passed 8.94 sec  Start 13: osal\_loader\_UT  13/23 Test #13: osal\_loader\_UT ................... Passed 0.13 sec  Start 14: osal\_filesys\_UT  14/23 Test #14: osal\_filesys\_UT .................. Passed 0.05 sec  Start 15: osal\_file\_UT  15/23 Test #15: osal\_file\_UT ..................... Passed 0.14 sec  Start 16: osal\_network\_UT  16/23 Test #16: osal\_network\_UT .................. Passed 0.04 sec  Start 17: osal\_timer\_UT  17/23 Test #17: osal\_timer\_UT .................... Passed 4.56 sec  Start 18: cfe\_core\_default\_cpu1\_es\_UT  18/23 Test #18: cfe\_core\_default\_cpu1\_es\_UT ...... Passed 0.46 sec  Start 19: cfe\_core\_default\_cpu1\_sb\_UT  19/23 Test #19: cfe\_core\_default\_cpu1\_sb\_UT ...... Passed 1.31 sec  Start 20: cfe\_core\_default\_cpu1\_evs\_UT  20/23 Test #20: cfe\_core\_default\_cpu1\_evs\_UT ..... Passed 0.11 sec  Start 21: cfe\_core\_default\_cpu1\_tbl\_UT  21/23 Test #21: cfe\_core\_default\_cpu1\_tbl\_UT ..... Passed 0.14 sec  Start 22: cfe\_core\_default\_cpu1\_time\_UT  22/23 Test #22: cfe\_core\_default\_cpu1\_time\_UT .... Passed 0.07 sec  Start 23: cfe\_core\_default\_cpu1\_fs\_UT  23/23 Test #23: cfe\_core\_default\_cpu1\_fs\_UT ...... Passed 0.08 sec  100% tests passed, 0 tests failed out of 23  Total Test time (real) = 115.64 sec  debian@beaglebone:~/cFS$ |
| 4 | Observe the detailed unit test results. | debian@beaglebone:~/cFS/build/native/Testing/Temporary$ cat LastTest.log  Start testing: Mar 09 16:40 UTC  <snip>  [BEGIN] 11 Test\_CFE\_FS\_GetUncompressedFile  [ PASS] 11.001 fs\_UT.c:546 - CFE\_FS\_GetUncompressedFile - Nominal Case  [ PASS] 11.002 fs\_UT.c:551 - CFE\_FS\_GetUncompressedFile - Output file name correct  [ PASS] 11.003 fs\_UT.c:560 - CFE\_FS\_GetUncompressedFile - Bad Input Argument  [ PASS] 11.004 fs\_UT.c:569 - CFE\_FS\_GetUncompressedFile - Name Too Long  [ PASS] 11.005 fs\_UT.c:579 - CFE\_FS\_GetUncompressedFile - Decompress failure  [ END] 11 Test\_CFE\_FS\_GetUncompressedFile TOTAL::5 PASS::5 FAIL::0 MIR::0 TSF::0 N/A::0  [ END] 11 SUMMARY TOTAL::41 PASS::41 FAIL::0 MIR::0 TSF::0 N/A::0  COMPLETE: 11 tests Segment(s) executed  <end of output>  Test time = 0.08 sec  ----------------------------------------------------------  Test Passed.  "cfe\_core\_default\_cpu1\_fs\_UT" end time: Sep 25 13:26 UTC  "cfe\_core\_default\_cpu1\_fs\_UT" time elapsed: 00:00:00  ----------------------------------------------------------  End testing: Sep 25 13:26 UTC  debian@beaglebone:~/cFS/build/native/Testing/Temporary$ |
| 5 | Generate the coverage report. | debian@beaglebone:~/cFS$ make lcov  lcov --capture --directory build/native --output-file build/native/coverage\_test.info  Capturing coverage data from build/native  Found gcov version: 6.3.0  Scanning build/native for .gcda files ...  Found 161 data files in build/native  Processing cfe\_core\_default\_cpu1/unit-test/CMakeFiles/ut\_cfe\_sb\_object.dir/\_\_/src/sb/ccsds.c.gcda  Processing cfe\_core\_default\_cpu1/unit-test/CMakeFiles/ut\_cfe\_sb\_object.dir/\_\_/src/sb/cfe\_sb\_buf.c.gcda  <snip>  Processing ut\_cfe\_core\_stubs/CMakeFiles/ut\_cfe-core\_stubs.dir/ut\_es\_stubs.c.gcda  Processing ut\_cfe\_core\_stubs/CMakeFiles/ut\_cfe-core\_stubs.dir/ut\_time\_stubs.c.gcda  Finished .info-file creation  lcov --add-tracefile build/native/coverage\_base.info --add-tracefile build/native/coverage\_test.info --output-file build/native/coverage\_total.info  Combining tracefiles.  Reading tracefile build/native/coverage\_base.info  Reading tracefile build/native/coverage\_test.info  Writing data to build/native/coverage\_total.info  Summary coverage rate:  lines......: 81.8% (27170 of 33201 lines)  functions..: 76.3% (1417 of 1858 functions)  branches...: no data found  lcov --remove build/native/coverage\_total.info 'unit-test/\*' --output-file build/native/coverage\_filtered.info  Reading tracefile build/native/coverage\_total.info  Deleted 0 files  Writing data to build/native/coverage\_filtered.info  Summary coverage rate:  lines......: 81.8% (27170 of 33201 lines)  functions..: 76.3% (1417 of 1858 functions)  branches...: no data found  genhtml build/native/coverage\_filtered.info --output-directory build/native/lcov  Reading data file build/native/coverage\_filtered.info  Found 169 entries.  Found common filename prefix "/home/debian/cFS"  Writing .css and .png files.  Generating output.  Processing file apps/bcamp\_io\_app/fsw/src/bcamp\_io\_app.c  Processing file apps/bcamp\_mn\_app/fsw/src/bcamp\_mn\_app.c  <snip>  Processing file psp/fsw/shared/cfe\_psp\_port.c  Processing file psp/fsw/ut-stubs/ut\_psp\_stubs.c  Writing directory view page.  Overall coverage rate:  lines......: 81.8% (27170 of 33201 lines)  functions..: 76.3% (1417 of 1858 functions)  Coverage Report Link: file:/home/debian/cFS/build/native/lcov/index.html  debian@beaglebone:~/cFS$ |
| 6 | Modify the nginx configuration file to access debian’s home directory. | debian@beaglebone:/etc/nginx/sites-available$ diff default default.0  57,62d56  < location /debian {  < # First attempt to serve request as file, then  < # as directory, then fall back to displaying a 404.  < try\_files $uri $uri/ =404;  < }  <  debian@beaglebone:/etc/nginx/sites-available$  debian@beaglebone:/etc/nginx/sites-available$ sudo systemctl restart nginx  debian@beaglebone:/etc/nginx/sites-available$ |
| 7 | View the coverage report in a web browser from the host machine. |  |

# Enhance the cFE Unit Test Stubs

|  |  |  |
| --- | --- | --- |
| 1 | As is, the cFE unit test stubs only store the event id of each sent event without the corresponding message. For robust unit testing, it is better to match not only the event id, but also the corresponding message. Let’s extend the EVS unit test stubs to store not only the event id but also the corresponding message. | debian@beaglebone:~/cFS/cfe$ git diff fsw/cfe-core/unit-test/ut\_support.h  diff --git a/fsw/cfe-core/unit-test/ut\_support.h b/fsw/cfe-core/unit-test/ut\_support.h  index 1c8d8b1..c027058 100644  --- a/fsw/cfe-core/unit-test/ut\_support.h  +++ b/fsw/cfe-core/unit-test/ut\_support.h  @@ -131,6 +131,16 @@ typedef struct  } UT\_SoftwareBusSnapshot\_Entry\_t;  /\*  + \* Information about an event including the event id and  + \* the message for the event  + \*/  +typedef struct  +{  + uint16 EventID;  + char Msg[CFE\_MISSION\_EVS\_MAX\_MESSAGE\_LENGTH];  +} EventInfo\_t;  +  +/\*  \* Information to identify a message in the "Task Pipe"  \* or message dispatch routines, to indicate which of  \* the message handlers the routine should invoke.  @@ -551,6 +561,28 @@ bool UT\_EventIsInHistory(uint16 EventIDToSearchFor);  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*  +\*\* \brief Search the event history for a specified event ID and message  +\*\*  +\*\* \par Description  +\*\* Search the event history for a specified event ID and message  +\*\*  +\*\* \par Assumptions, External Events, and Notes:  +\*\* None  +\*\*  +\*\* \param[in] EventIDToSearchFor Event ID to search for in the event history  +\*\* \param[in] MsgToSearchFor Message to search for  +\*\*  +\*\* \returns  +\*\* true if the event ID to search for is found in the event history  +\*\* with the designated message  +\*\* false if the event ID and message combination are not found.  +\*\*  +\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  +bool UT\_EventIsInHistoryWithMessage(uint16 EventIDToSearchFor,  + const char \*MsgToSearchFor);  +  +/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  +/\*\*  \*\* \brief Search the syslog history for a specified log message template  \*\*  \*\* \par Assumptions, External Events, and Notes:  debian@beaglebone:~/cFS/cfe$ git diff fsw/cfe-core/unit-test/ut\_support.c  diff --git a/fsw/cfe-core/unit-test/ut\_support.c b/fsw/cfe-core/unit-test/ut\_support.c  index 00ecfbf..116fb5b 100644  --- a/fsw/cfe-core/unit-test/ut\_support.c  +++ b/fsw/cfe-core/unit-test/ut\_support.c  @@ -64,9 +64,9 @@ static union  CFE\_ES\_ResetData\_t ResetData;  } UT\_CFE\_ES\_ResetData;  -static uint16 UT\_SendEventHistory[UT\_EVENT\_HISTORY\_SIZE];  -static uint16 UT\_SendTimedEventHistory[UT\_EVENT\_HISTORY\_SIZE];  -static uint16 UT\_SendEventAppIDHistory[UT\_EVENT\_HISTORY\_SIZE \* 10];  +static EventInfo\_t UT\_SendEventHistory[UT\_EVENT\_HISTORY\_SIZE];  +static EventInfo\_t UT\_SendTimedEventHistory[UT\_EVENT\_HISTORY\_SIZE];  +static EventInfo\_t UT\_SendEventAppIDHistory[UT\_EVENT\_HISTORY\_SIZE \* 10];  extern int32 dummy\_function(void);  @@ -387,22 +387,31 @@ void UT\_ClearEventHistory(void)  UT\_SetDataBuffer(UT\_KEY(CFE\_EVS\_SendEventWithAppID), UT\_SendEventAppIDHistory, sizeof(UT\_SendEventAppIDHistory), false);  }  -static bool UT\_CheckEventHistoryFromFunc(UT\_EntryKey\_t Func, uint16 EventIDToSearchFor)  +static bool UT\_CheckEventHistoryFromFunc(UT\_EntryKey\_t Func, uint16 EventIDToSearchFor,  + const char \*MsgToSearchFor)  {  bool Result = false;  uint32 Position;  uint32 MaxSize;  - uint16 \*EvBuf;  + EventInfo\_t \*EvBuf;  UT\_GetDataBuffer(Func, (void\*\*)&EvBuf, &MaxSize, &Position);  if (EvBuf != NULL && MaxSize > 0)  {  while (Position > 0)  {  - if (\*EvBuf == EventIDToSearchFor)  + if (EvBuf->EventID == EventIDToSearchFor)  {  - Result = true;  - break;  + if (MsgToSearchFor == NULL)  + {  + Result = true;  + break;  + }  + else if (strcmp(EvBuf->Msg, MsgToSearchFor) == 0)  + {  + Result = true;  + break;  + }  }  ++EvBuf;  --Position;  @@ -418,11 +427,20 @@ static bool UT\_CheckEventHistoryFromFunc(UT\_EntryKey\_t Func, uint16 EventIDToSea  \*/  bool UT\_EventIsInHistory(uint16 EventIDToSearchFor)  {  - return (UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendEvent),EventIDToSearchFor) ||  - UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendEventWithAppID),EventIDToSearchFor) ||  - UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendTimedEvent),EventIDToSearchFor));  + return (UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendEvent),EventIDToSearchFor, NULL) ||  + UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendEventWithAppID),EventIDToSearchFor, NULL) ||  + UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendTimedEvent),EventIDToSearchFor, NULL));  }  +bool UT\_EventIsInHistoryWithMessage(uint16 EventIDToSearchFor,  + const char \*MsgToSearchFor)  +{  + return (UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendEvent),EventIDToSearchFor, MsgToSearchFor) ||  + UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendEventWithAppID),EventIDToSearchFor, MsgToSearchFor) ||  + UT\_CheckEventHistoryFromFunc(UT\_KEY(CFE\_EVS\_SendTimedEvent),EventIDToSearchFor, MsgToSearchFor));  +}  +  +  /\*  \*\* Return number of events issued  \*/  @@ -434,11 +452,11 @@ uint16 UT\_GetNumEventsSent(void)  void \*EvBuf;  UT\_GetDataBuffer(UT\_KEY(CFE\_EVS\_SendEvent), &EvBuf, &MaxSize, &Position);  - Total += Position / sizeof(uint16);  + Total += Position / sizeof(EventInfo\_t);  UT\_GetDataBuffer(UT\_KEY(CFE\_EVS\_SendEventWithAppID), &EvBuf, &MaxSize, &Position);  - Total += Position / sizeof(uint16);  + Total += Position / sizeof(EventInfo\_t);  UT\_GetDataBuffer(UT\_KEY(CFE\_EVS\_SendTimedEvent), &EvBuf, &MaxSize, &Position);  - Total += Position / sizeof(uint16);  + Total += Position / sizeof(EventInfo\_t);  return Total;  }  debian@beaglebone:~/cFS/cfe$ git diff fsw/cfe-core/ut-stubs/ut\_evs\_stubs.c  diff --git a/fsw/cfe-core/ut-stubs/ut\_evs\_stubs.c b/fsw/cfe-core/ut-stubs/ut\_evs\_stubs.c  index 2d1432f..43ee7ae 100644  --- a/fsw/cfe-core/ut-stubs/ut\_evs\_stubs.c  +++ b/fsw/cfe-core/ut-stubs/ut\_evs\_stubs.c  @@ -36,6 +36,7 @@  #include "cfe.h"  #include "cfe\_platform\_cfg.h"  #include "utstubs.h"  +#include "ut\_support.h"  /\*  \*\* Functions  @@ -106,17 +107,22 @@ void CFE\_EVS\_TaskMain(void)  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int32 CFE\_EVS\_SendEvent(uint16 EventID,  uint16 EventType,  - const char \*Spec,  + const char \*Msg,  ...)  {  int32 status;  - UT\_Stub\_RegisterContext(UT\_KEY(CFE\_EVS\_SendEvent), &EventID);  + EventInfo\_t EventInfo;  + EventInfo.EventID = EventID;  + strncpy(EventInfo.Msg, Msg, CFE\_MISSION\_EVS\_MAX\_MESSAGE\_LENGTH);  + EventInfo.Msg[CFE\_MISSION\_EVS\_MAX\_MESSAGE\_LENGTH - 1] = '\0';  +  + UT\_Stub\_RegisterContext(UT\_KEY(CFE\_EVS\_SendEvent), &EventInfo);  status = UT\_DEFAULT\_IMPL(CFE\_EVS\_SendEvent);  if (status >= 0)  {  - UT\_Stub\_CopyFromLocal(UT\_KEY(CFE\_EVS\_SendEvent), (uint8\*)&EventID, sizeof(EventID));  + UT\_Stub\_CopyFromLocal(UT\_KEY(CFE\_EVS\_SendEvent), (uint8\*)&EventInfo, sizeof(EventInfo));  }  return status;  @@ -140,17 +146,22 @@ int32 CFE\_EVS\_SendEvent(uint16 EventID,  int32 CFE\_EVS\_SendTimedEvent(CFE\_TIME\_SysTime\_t Time,  uint16 EventID,  uint16 EventType,  - const char \*Spec,  + const char \*Msg,  ...)  {  int32 status;  - UT\_Stub\_RegisterContext(UT\_KEY(CFE\_EVS\_SendEvent), &EventID);  + EventInfo\_t EventInfo;  + EventInfo.EventID = EventID;  + strncpy(EventInfo.Msg, Msg, CFE\_MISSION\_EVS\_MAX\_MESSAGE\_LENGTH);  + EventInfo.Msg[CFE\_MISSION\_EVS\_MAX\_MESSAGE\_LENGTH - 1] = '\0';  +  + UT\_Stub\_RegisterContext(UT\_KEY(CFE\_EVS\_SendEvent), &EventInfo);  status = UT\_DEFAULT\_IMPL(CFE\_EVS\_SendEvent);  if (status >= 0)  {  - UT\_Stub\_CopyFromLocal(UT\_KEY(CFE\_EVS\_SendTimedEvent), (uint8\*)&EventID, sizeof(EventID));  + UT\_Stub\_CopyFromLocal(UT\_KEY(CFE\_EVS\_SendTimedEvent), (uint8\*)&EventInfo, sizeof(EventInfo));  }  return CFE\_SUCCESS;  @@ -213,17 +224,22 @@ int32 CFE\_EVS\_Register(void \*Filters,  int32 CFE\_EVS\_SendEventWithAppID(uint16 EventID,  uint16 EventType,  uint32 AppID,  - const char \*Spec,  + const char \*Msg,  ...)  {  int32 status;  - UT\_Stub\_RegisterContext(UT\_KEY(CFE\_EVS\_SendEvent), &EventID);  + EventInfo\_t EventInfo;  + EventInfo.EventID = EventID;  + strncpy(EventInfo.Msg, Msg, CFE\_MISSION\_EVS\_MAX\_MESSAGE\_LENGTH);  + EventInfo.Msg[CFE\_MISSION\_EVS\_MAX\_MESSAGE\_LENGTH - 1] = '\0';  +  + UT\_Stub\_RegisterContext(UT\_KEY(CFE\_EVS\_SendEvent), &EventInfo);  status = UT\_DEFAULT\_IMPL(CFE\_EVS\_SendEventWithAppID);  if (status >= 0)  {  - UT\_Stub\_CopyFromLocal(UT\_KEY(CFE\_EVS\_SendEventWithAppID), (uint8\*)&EventID, sizeof(EventID));  + UT\_Stub\_CopyFromLocal(UT\_KEY(CFE\_EVS\_SendEventWithAppID), (uint8\*)&EventInfo, sizeof(EventInfo));  }  debian@beaglebone:~/cFS/cfe$ git diff fsw/cfe-core/ut-stubs/CMakeLists.txt  diff --git a/fsw/cfe-core/ut-stubs/CMakeLists.txt b/fsw/cfe-core/ut-stubs/CMakeLists.txt  index f112090..e7be8bd 100644  --- a/fsw/cfe-core/ut-stubs/CMakeLists.txt  +++ b/fsw/cfe-core/ut-stubs/CMakeLists.txt  @@ -12,6 +12,12 @@  # Reference the UT assert include directory  include\_directories(${osal\_MISSION\_DIR}/ut\_assert/inc)  +# Reference the CFE Unit Test directory for support header file(s)  +include\_directories(../unit-test)  +include\_directories(../src/evs)  +include\_directories(../src/es)  +include\_directories(../src/time)  +  #  # Create the generic stubs library  # |
| 2 | Ensure the cFE unit tests still build and run. |  |

# Develop Unit Tests for the Bootcamp IO App

|  |  |  |
| --- | --- | --- |
| 1 | Create a unit test for the Bootcamp IO App. Specifically, exercise the BCAMP\_IO\_ProcessGroundCommand() function by sending a NOOP command. | debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ git diff ../src/bcamp\_io\_app.h  diff --git a/apps/bcamp\_io\_app/fsw/src/bcamp\_io\_app.h b/apps/bcamp\_io\_app/fsw/src/bcamp\_io\_app.h  index cf71262..a7272c3 100644  --- a/apps/bcamp\_io\_app/fsw/src/bcamp\_io\_app.h  +++ b/apps/bcamp\_io\_app/fsw/src/bcamp\_io\_app.h  @@ -10,6 +10,7 @@  #include "cfe\_sb.h"  #include "cfe\_es.h"  +#include "bcamp\_io\_app\_events.h"  #include "bcamp\_io\_app\_perfids.h"  #include "bcamp\_io\_app\_msgids.h"  #include "bcamp\_io\_app\_msg.h"  debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ cat bcamp\_io\_app\_testcase.h  #ifndef \_bcamp\_io\_testcases\_h\_  #define \_bcamp\_io\_testcases\_h\_  void TestBCAMP\_IO\_Setup(void);  void TestBCAMP\_IO\_Teardown(void);  void TestBCAMP\_IO\_ProcessGroundCommand(void);  #endif /\* #ifndef \_bcamp\_io\_testcases\_h\_ \*/  debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ cat bcamp\_io\_app\_testcase.c  #include "ut\_support.h"  #include "bcamp\_io\_app.h"  #include "bcamp\_io\_app\_version.h"  extern Bcamp\_IO\_AppData\_t Bcamp\_IO\_AppData;  void TestBCAMP\_IO\_Setup(void)  {  UT\_Init("bcamp\_io\_app");  }  void TestBCAMP\_IO\_Teardown(void)  {  CFE\_PSP\_MemSet((void\*)&Bcamp\_IO\_AppData, 0x00,  sizeof(Bcamp\_IO\_AppData));  UT\_ClearEventHistory();  }  void TestBCAMP\_IO\_ProcessGroundCommand(void)  {  BCAMP\_IO\_NoArgsCmd\_t cmdMsg = {};  CFE\_SB\_MsgPtr\_t pMsg = (CFE\_SB\_MsgPtr\_t)&cmdMsg;  CFE\_SB\_MsgId\_t MID;  uint16 CC;  /\* Nominal NOOP \*/  /\* Configure environment \*/  Bcamp\_IO\_AppData.CmdCounter = 0; /\* should already be initialized to 0, but just to make sure \*/  MID = BCAMP\_IO\_APP\_CMD\_MID;  CC = BCAMP\_IO\_APP\_NOOP\_CC;  CFE\_SB\_SetMsgId(pMsg, MID);  CFE\_SB\_SetCmdCode(pMsg, CC);  CFE\_SB\_SetTotalMsgLength(pMsg, sizeof(cmdMsg));  /\* Execute test \*/  BCAMP\_IO\_ProcessGroundCommand(pMsg);  /\* Confirm results \*/  UtAssert\_True(Bcamp\_IO\_AppData.CmdCounter == 1, "TestBCAMP\_IO\_ProcessGroundCommand - Nominal NOOP");  UtAssert\_True(UT\_GetNumEventsSent() == 1, "TestBCAMP\_IO\_ProcessGroundCommand - Nominal NOOP Eent Count");  UtAssert\_True(UT\_EventIsInHistoryWithMessage(BCAMP\_IO\_COMMANDNOP\_INF\_EID,  "BCAMP\_IO: NOOP command Version %d.%d.%d.%d"),  "TestBCAMP\_IO\_ProcessGroundCommand - Nominal NOOP Event");  }  debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ cat bcamp\_io\_app\_testrunner.c  /\*  \* Filename: bcamp\_io\_app\_testrunner.c  \*  \* Purpose: This file contains a unit test runner for the Bootcamp IO App.  \*  \*/  /\*  \* Includes  \*/  #include "ut\_support.h"  #include "bcamp\_io\_app\_testcase.h"  /\*  \* Function Definitions  \*/  void UtTest\_Setup(void)  {  UT\_Text("Bootcamp IO App Unit Test Output File\n\n");  UtTest\_Add(TestBCAMP\_IO\_ProcessGroundCommand, TestBCAMP\_IO\_Setup, TestBCAMP\_IO\_Teardown, "TestBCAMP\_IO\_ProcessGroundCommand");  }  debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ cat makefile  ##############################################################################  ## GNU Makefile for building UT unit tests  #  # Supported MAKEFILE targets:  # clean - deletes object files, executables, output files, and gcov files  # all - makes utf\_test\_runner.exe  # run - runs utf\_test\_runner.exe  # gcov - prints a GCOV coverage report (make all, make run, make gcov)  #  APP=bcamp\_io\_app  $(info $$APP is [${APP}])  mkfile\_path := $(dir $(abspath $(lastword $(MAKEFILE\_LIST))))  CFS\_MISSION := $(realpath $(mkfile\_path)/../../../../)  CFS\_APP\_SRC = $(CFS\_MISSION)/apps/$(APP)  UT\_ASSERT\_SRC := $(CFS\_MISSION)/osal/ut\_assert  CFE\_DIR = $(CFS\_MISSION)/cfe  OSAL\_DIR = $(CFS\_MISSION)/osal  BSP\_TYPE ?= pc-linux  BSP\_DIR := $(OSAL\_DIR)/src/bsp/$(BSP\_TYPE)/ut-src  PSP\_DIR = $(CFS\_MISSION)/psp  APP\_DIR = $(CFS\_MISSION)/apps  #  # VPATH specifies the search paths for source files outside of the current directory. Note that  # all object files will be created in the current directory even if the source file is not in the  # current directory.  #  VPATH := $(CFS\_APP\_SRC)/fsw/src  VPATH += $(UT\_ASSERT\_SRC)/src  VPATH += $(BSP\_DIR)  VPATH += $(PSP\_DIR)/fsw/ut-stubs  VPATH += $(CFE\_DIR)/fsw/cfe-core/ut-stubs  VPATH += $(CFE\_DIR)/fsw/cfe-core/unit-test  VPATH += $(OSAL\_DIR)/src/ut-stubs  #  # INCLUDES specifies the search paths for include files outside of the current directory.  # Note that the -I is required.  #  INCLUDES += -I../src  INCLUDES += -I../platform\_inc  INCLUDES += -I../mission\_inc  INCLUDES += -I$(UT\_ASSERT\_SRC)/inc  INCLUDES += -I$(CFE\_DIR)/fsw/cfe-core/src/inc  INCLUDES += -I$(CFE\_DIR)/fsw/cfe-core/src/time  INCLUDES += -I$(CFE\_DIR)/fsw/cfe-core/src/es  INCLUDES += -I$(CFE\_DIR)/fsw/cfe-core/src/evs  INCLUDES += -I$(CFE\_DIR)/fsw/cfe-core/unit-test  INCLUDES += -I$(CFE\_DIR)/fsw/mission\_inc  INCLUDES += -I$(CFE\_DIR)/fsw/platform\_inc/cpu1  INCLUDES += -I$(CFE\_DIR)/cmake/target/inc  INCLUDES += -I$(OSAL\_DIR)/src/os/inc  INCLUDES += -I$(OSAL\_DIR)/src/os/shared  INCLUDES += -I$(OSAL\_DIR)/src/bsp/pc-linux/config  INCLUDES += -I$(PSP\_DIR)/fsw/inc  #  # LIB\_OBJS specifies library object files.  #  LIB\_OBJS := $(APP).o  #  # GCOV\_OBJS specifies the object files targeted for code coverage  # They're not necessarily the same set as the LIB\_OBJS.  #  GCOV\_OBJS := $(APP).o  #  # UT\_OBJS specifies unit test object files.  #  UT\_SRCS := $(wildcard $(UT\_ASSERT\_SRC)/src/\*.c)  UT\_SRCS := $(foreach d,$(UT\_SRCS),$(notdir $(d)))  UT\_OBJS := $(patsubst %.c,%.o,$(UT\_SRCS))  BSP\_SRCS := $(wildcard $(BSP\_DIR)/\*.c)  BSP\_SRCS := $(foreach d,$(BSP\_SRCS),$(notdir $(d)))  BSP\_OBJS := $(patsubst %.c,%.o,$(BSP\_SRCS))  UT\_OBJS += $(BSP\_OBJS)  PSP\_SRCS := $(wildcard $(PSP\_DIR)/fsw/ut-stubs/\*.c)  PSP\_SRCS := $(foreach d,$(PSP\_SRCS),$(notdir $(d)))  PSP\_OBJS := $(patsubst %.c,%.o,$(PSP\_SRCS))  UT\_OBJS += $(PSP\_OBJS)  CFE\_SRCS := $(wildcard $(CFE\_DIR)/fsw/cfe-core/ut-stubs/\*.c)  CFE\_SRCS := $(foreach d,$(CFE\_SRCS),$(notdir $(d)))  CFE\_OBJS := $(patsubst %.c,%.o,$(CFE\_SRCS))  UT\_OBJS += $(CFE\_OBJS)  OSAL\_SRCS := $(wildcard $(OSAL\_DIR)/src/ut-stubs/\*.c)  OSAL\_SRCS := $(foreach d,$(OSAL\_SRCS),$(notdir $(d)))  OSAL\_OBJS := $(patsubst %.c,%.o,$(OSAL\_SRCS))  UT\_OBJS += $(OSAL\_OBJS)  UT\_OBJS += ut\_support.o  UT\_OBJS += $(APP)\_testcase.o  #UT\_OBJS += $(APP)\_stubs.o  ###############################################################################  COMPILER=gcc  LINKER=gcc  #  # Compiler and Linker Options  #  ENABLE\_GCOV = TRUE  ifeq ($(ENABLE\_GCOV), TRUE)  GCOV\_COPT = -fprofile-arcs -ftest-coverage -pg -p  GCOV\_LOPT = -pg -p -fprofile-arcs -ftest-coverage -lgcov  endif  #WARNINGS = -Wall -W -ansi -Werror -Wstrict-prototypes -Wundef  WARNINGS = -Wall -Wstrict-prototypes  DEBUGGER = -g  #COPT = $(WARNINGS) $(DEBUGGER) $(GCOV\_COPT) -DSOFTWARE\_LITTLE\_BIT\_ORDER -D\_EL -D\_\_x86\_64\_\_ -D\_LINUX\_OS\_  COPT = $(WARNINGS) $(DEBUGGER) $(GCOV\_COPT) -DSOFTWARE\_LITTLE\_BIT\_ORDER -D\_EL -D\_ix86\_ -D\_LINUX\_OS\_ -DUT\_VERBOSE -DOSAPI\_NO\_SPECIAL\_ATTRIBS  LOPT = $(GCOV\_LOPT)  ###############################################################################  ## Rule to make the specified TARGET  ##  %.exe: %.o  $(LINKER) $(LOPT) $^ -o $\*.exe  ###############################################################################  ## "C" COMPILER RULE  ##  %.o: %.c  $(COMPILER) -c $(COPT) $(INCLUDES) $<  ###############################################################################  ##  ##  all:$(APP)\_testrunner.exe  $(APP)\_testrunner.exe: $(APP)\_testrunner.o $(UT\_OBJS) $(LIB\_OBJS)  clean ::  rm -f \*.o \*.exe \*.gcda \*.gcno \*.gcov gmon.out  rm -rf ram0 ram1 ram2 ram3 ram4 ram5 eeprom1  run ::  ./$(APP)\_testrunner.exe  gcov ::  @gcov -f -b $(GCOV\_OBJS:.o=.gcda)  # end of file |
| 2 | Build the unit tests. | debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ make  $APP is [bcamp\_io\_app]  gcc -c -Wall -Wstrict-prototypes -g -fprofile-arcs -ftest-coverage -pg -p -DSOFTWARE\_LITTLE\_BIT\_ORDER -D\_EL -D\_ix86\_ -D\_LINUX\_OS\_ -DUT\_VERBOSE -DOSAPI\_NO\_SPECIAL\_ATTRIBS -I../src -I../platform\_inc -I../mission\_inc -I/home/debian/cFS/osal/ut\_assert/inc -I/home/debian/cFS/cfe/fsw/cfe-core/src/inc -I/home/debian/cFS/cfe/fsw/cfe-core/src/time -I/home/debian/cFS/cfe/fsw/cfe-core/src/es -I/home/debian/cFS/cfe/fsw/cfe-core/src/evs -I/home/debian/cFS/cfe/fsw/cfe-core/unit-test -I/home/debian/cFS/cfe/fsw/mission\_inc -I/home/debian/cFS/cfe/fsw/platform\_inc/cpu1 -I/home/debian/cFS/cfe/cmake/target/inc -I/home/debian/cFS/osal/src/os/inc -I/home/debian/cFS/osal/src/os/shared -I/home/debian/cFS/osal/src/bsp/pc-linux/config -I/home/debian/cFS/psp/fsw/inc bcamp\_io\_app\_testrunner.c  <snip>  gcc -pg -p -fprofile-arcs -ftest-coverage -lgcov bcamp\_io\_app\_testrunner.o utstubs.o uttools.o utassert.o uttest.o utlist.o bsp\_ut\_voltab.o bsp\_ut.o ut\_psp\_stubs.o ut\_evs\_stubs.o ut\_sb\_stubs.o ut\_tbl\_stubs.o ut\_time\_stubs.o ut\_es\_stubs.o ut\_fs\_stubs.o osapi-utstub-printf.o osapi-utstub-network.o osapi-utstub-idmap.o osapi-utstub-errors.o osapi-utstub-file.o osapi-utstub-task.o utstub-helpers.o osapi-utstub-filesys.o osapi-utstub-time.o osapi-utstub-heap.o osapi-utstub-countsem.o osapi-utstub-fpu.o osapi-utstub-select.o osapi-utstub-sockets.o osapi-utstub-interrupts.o osapi-utstub-timebase.o osapi-utstub-binsem.o osapi-utstub-mutex.o osapi-utstub-dir.o osapi-utstub-module.o osapi-utstub-common.o osapi-utstub-clock.o osapi-utstub-queue.o ut\_support.o bcamp\_io\_app\_testcase.o bcamp\_io\_app.o -o bcamp\_io\_app\_testrunner.exe |
| 3 | Run the unit tests. | debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ make run  $APP is [bcamp\_io\_app]  ./bcamp\_io\_app\_testrunner.exe  [BEGIN] PC-LINUX UNIT TEST  Creating mount point directory: ./ram0  Creating mount point directory: ./ram1  Creating mount point directory: ./ram2  Creating mount point directory: ./ram3  Creating mount point directory: ./ram4  Creating mount point directory: ./ram5  Creating mount point directory: ./eeprom1  [BEGIN] 01 SETUP  [ INFO] ut\_support.c:207:Bootcamp IO App Unit Test Output File  [ END] No test cases  [BEGIN] 01 TestBCAMP\_IO\_ProcessGroundCommand  [ PASS] 01.001 bcamp\_io\_app\_testcase.c:43 - TestBCAMP\_IO\_ProcessGroundCommand - Nominal NOOP  [ PASS] 01.002 bcamp\_io\_app\_testcase.c:44 - TestBCAMP\_IO\_ProcessGroundCommand - Nominal NOOP Eent Count  [ PASS] 01.003 bcamp\_io\_app\_testcase.c:47 - TestBCAMP\_IO\_ProcessGroundCommand - Nominal NOOP Event  [ END] 01 TestBCAMP\_IO\_ProcessGroundCommand TOTAL::3 PASS::3 FAIL::0 MIR::0 TSF::0 N/A::0  COMPLETE: 1 tests Segment(s) executed |
| 4 | Generate the coverage report. | debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ make gcov  $APP is [bcamp\_io\_app]  Function 'BCAMP\_IO\_VerifyCmdLength'  Lines executed:50.00% of 10  Branches executed:100.00% of 2  Taken at least once:50.00% of 2  Calls executed:25.00% of 4  Function 'BCAMP\_IO\_SetDeltaValueCC'  Lines executed:0.00% of 6  No branches  Calls executed:0.00% of 2  Function 'BCAMP\_IO\_SetCurrentTempCC'  Lines executed:0.00% of 6  No branches  Calls executed:0.00% of 2  Function 'BCAMP\_IO\_ProcessCC'  Lines executed:0.00% of 10  No branches  Calls executed:0.00% of 3  Function 'BCAMP\_IO\_ResetCounters'  Lines executed:0.00% of 5  No branches  Calls executed:0.00% of 1  Function 'BCAMP\_IO\_NoopCmd'  Lines executed:100.00% of 4  No branches  Calls executed:100.00% of 1  Function 'BCAMP\_IO\_ReportHousekeeping'  Lines executed:0.00% of 8  No branches  Calls executed:0.00% of 2  Function 'BCAMP\_IO\_ProcessGroundCommand'  Lines executed:33.33% of 21  Branches executed:50.00% of 16  Taken at least once:12.50% of 16  Calls executed:25.00% of 12  Function 'BCAMP\_IO\_ProcessCommandPacket'  Lines executed:0.00% of 10  Branches executed:0.00% of 3  Taken at least once:0.00% of 3  Calls executed:0.00% of 4  Function 'BCAMP\_IO\_AppInit'  Lines executed:0.00% of 48  Branches executed:0.00% of 8  Taken at least once:0.00% of 8  Calls executed:0.00% of 11  Function 'BCAMP\_IO\_AppMain'  Lines executed:0.00% of 17  Branches executed:0.00% of 6  Taken at least once:0.00% of 6  Calls executed:0.00% of 11  File '/home/debian/cFS/apps/bcamp\_io\_app/fsw/src/bcamp\_io\_app.c'  Lines executed:11.03% of 145  Branches executed:28.57% of 35  Taken at least once:8.57% of 35  Calls executed:9.43% of 53  Creating 'bcamp\_io\_app.c.gcov' |
| 5 | View the coverage report. | debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ ls \*.gcov  bcamp\_io\_app.c.gcov  debian@beaglebone:~/cFS/apps/bcamp\_io\_app/fsw/unit-test$ cat bcamp\_io\_app.c.gcov  -: 0:Source:/home/debian/cFS/apps/bcamp\_io\_app/fsw/src/bcamp\_io\_app.c  -: 0:Graph:bcamp\_io\_app.gcno  -: 0:Data:bcamp\_io\_app.gcda  -: 0:Runs:1  -: 0:Programs:1  -: 1:/\*  -: 2:\*\* Include Files:  -: 3:\*/  -: 4:#include "bcamp\_io\_app\_events.h"  -: 5:#include "bcamp\_io\_app\_version.h"  -: 6:#include "bcamp\_io\_app.h"  -: 7:  -: 8:#include <string.h>  -: 9:  -: 10:/\*  -: 11:\*\* global data  -: 12:\*/  -: 13:Bcamp\_IO\_AppData\_t Bcamp\_IO\_AppData;  <snip>  -: 371:/\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*\*/  -: 372:/\* \*/  -: 373:/\* BCAMP\_IO\_NoopCmd -- BCAMP\_IO NOOP commands \*/  -: 374:/\* \*/  -: 375:/\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*\*/  function BCAMP\_IO\_NoopCmd called 1 returned 100% blocks executed 100%  1: 376:void BCAMP\_IO\_NoopCmd( const BCAMP\_IO\_Noop\_t \*Msg )  -: 377:{  -: 378:  1: 379: Bcamp\_IO\_AppData.CmdCounter++;  -: 380:  1: 381: CFE\_EVS\_SendEvent(BCAMP\_IO\_COMMANDNOP\_INF\_EID,  call 0 returned 100%  -: 382: CFE\_EVS\_INFORMATION,  -: 383: "BCAMP\_IO: NOOP command Version %d.%d.%d.%d",  -: 384: BCAMP\_IO\_APP\_MAJOR\_VERSION,  -: 385: BCAMP\_IO\_APP\_MINOR\_VERSION,  -: 386: BCAMP\_IO\_APP\_REVISION,  -: 387: BCAMP\_IO\_APP\_MISSION\_REV);  -: 388:  1: 389: return;  -: 390:  -: 391:} /\* End of BCAMP\_IO\_NoopCmd \*/  -: 392:  -: 393:/\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*\*/  -: 394:/\* Name: BCAMP\_IO\_ResetCounters \*/  -: 395:/\* \*/  -: 396:/\* Purpose: \*/  -: 397:/\* This function resets all the global counter variables that are \*/  -: 398:/\* part of the task telemetry. \*/  -: 399:/\* \*/  -: 400:/\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*/  function BCAMP\_IO\_ResetCounters called 0 returned 0% blocks executed 0%  #####: 401:void BCAMP\_IO\_ResetCounters( const BCAMP\_IO\_ResetCounters\_t \*Msg )  -: 402:{  -: 403:  #####: 404: Bcamp\_IO\_AppData.CmdCounter = 0;  #####: 405: Bcamp\_IO\_AppData.ErrCounter = 0;  -: 406:  #####: 407: CFE\_EVS\_SendEvent(BCAMP\_IO\_COMMANDRST\_INF\_EID,  call 0 never executed  -: 408: CFE\_EVS\_INFORMATION,  -: 409: "BCAMP\_IO: RESET command");  -: 410:  #####: 411: return;  -: 412:  -: 413:} /\* End of BCAMP\_IO\_ResetCounters() \*/  -: 414:  <snip> |
| 6 | Can you improve the coverage by testing another ground command?  Can you improve coverage by testing all the functions? |  |