# Building and Running Test Applications

## 1: I2C Test Application

This test source code is located at the following location: “\I2CAPP”.

Add I2CAPP as a subproject:

1. In Visual Studio click “Project” -> “Add Existing Subproject”
2. Navigate to “\I2CAPP” and select the sources file.
3. I2CAPP will be added to subprojects and will be built with and included in the BSP.

Build the subproject:

1. To Build just the subproject:
   1. Right click the I2CAPP subproject and select build.
   2. Compiled binary should be present in “\I2CAPP\obj\ARMV7\retail” (or debug depending on the build environment).
2. To build with your OS Image:
   1. Click “Build” -> “Advanced Build Commands” -> “Build Current BSP and Subprojects”.
   2. I2CAPP will be included in OS image in the “windows” directory.

Run the application:

1. Double click I2CAPP.exe in the windows directory.
   1. Application icon may be hidden. To reveal, unhide hidden files and folders in the display options.
2. Or run from the command line “I2CAPP.exe”

Interpreting the results:

* Application will write to the I2C1, 2, and 3 registers. It will then read the same registers.
* If I2C fails to write, it is considered a failure. This will be printed out on the debug port.
* If I2C fails to read, it is considered a failure. This will be printed out on the debug port.
* If I2C writes a value, and reads a different value than written, this is not necessarily a failure. This could be due to read-only or write-only registers, in which case, this is expected behavior. If this happens, check the registers being written to on specified device (i.e. – SGTL5000, HDMI, or LVDS) to determine if registers are read only.

## 7.2: SPI Test Application

This test source code is located at the following location: “\ECSPI”.

Add ECSPI as a subproject:

1. In Visual Studio click “Project” -> “Add Existing Subproject”
2. Navigate to “\ECSPI” and select the sources file.
3. ECSPI will be added to subprojects and will be built with and included in the BSP.

Build the subproject:

1. To Build just the subproject:
   1. Right click the ECSPI subproject and select build.
   2. Compiled binary should be present in “\ECSPI\obj\ARMV7\retail” (or debug depending on the build environment).
2. To build with your OS Image:
   1. Click “Build” -> “Advanced Build Commands” -> “Build Current BSP and Subprojects”.
   2. ECSPI will be included in OS image in the “windows” directory.

Run the application:

1. Double click ECSPIapp.exe in the windows directory.
   1. Application icon may be hidden. To reveal, unhide hidden files and folders in the display options.
2. Or run from the command line “ECSPIapp.exe”

Interpreting the results:

* Application will write to the SPI1: port.
* Application will output onto the debug port whether the write and echo back on the SPI port succeeded or failed.

## 7.3: UART Test Application

This test source code is located at the following location: “\SerialWriteTest”.

Add SerialWriteTest as a subproject:

1. In Visual Studio click “Project” -> “Add Existing Subproject”
2. Navigate to “\ SerialWriteTest” and select the sources file.
3. SerialWriteTest will be added to subprojects and will be built with and included in the BSP.

Build the subproject:

1. To Build just the subproject:
   1. Right click the SerialWriteTest subproject and select build.
   2. Compiled binary should be present in “\ SerialWriteTest \obj\ARMV7\retail” (or debug depending on the build environment).
2. To build with your OS Image:
   1. Click “Build” -> “Advanced Build Commands” -> “Build Current BSP and Subprojects”.
   2. SerialWriteTest will be included in OS image in the “windows” directory.

Run the application:

1. Connect serial connector (DE-9) pins 2 and 3 together. This will create a loopback between TX and RX lines.
2. Double click SerialWriteTest.exe in the windows directory.
   1. Application icon may be hidden. To reveal, unhide hidden files and folders in the display options.
   2. Double clicking the exe will run the test with default settings. (This will test UART1/COM1)
3. Or run from the command line “SerialWriteTest.exe”
   1. An additional parameter of 1 or 2 will run the test on UART1 or UART2 respectfully. (i.e. – “serialwritetest.exe 1” will run the test on UART1)
   2. Running SerialWriteTest.exe without parameters will default to testing UART1.

Interpreting the results:

* On debug output, application will print out data written, and data received.
* For test to pass, the number of bytes read need to be equal to that written.
* For test to pass, the characters read need to be equal to that written.

## 7.4: USB Camera Test Application

This test source code is located at the following location: “\DShowCamTest”.

Add DShowCamTest as a subproject:

1. In Visual Studio click “Project” -> “Add Existing Subproject”
2. Navigate to “\ DShowCamTest” and select the sources file.
3. DShowCamTest will be added to subprojects and will be built with and included in the BSP.

Build the subproject:

1. To Build just the subproject:
   1. Right click the DShowCamTest subproject and select build.
   2. Compiled binary should be present in “\DShowCamTest\obj\ARMV7\retail” (or debug depending on the build environment).
2. To build with your OS Image:
   1. Click “Build” -> “Advanced Build Commands” -> “Build Current BSP and Subprojects”.
   2. DShowCamTest will be included in OS image in the “windows” directory.

Run the application:

1. Connect a USB Camera to a USB port on the device.
2. Double click DShowCamTest.exe in the windows directory.
   1. Application icon may be hidden. To reveal, unhide hidden files and folders in the display options.

Interpreting the results:

* Video test application should pop up and display video stream from the camera.