Converting Library Build to Split Image

All BLE-Stack 3.01.00.05 projects will support the Library build configuration. Most Application and Stack projects will not have a split-image build configuration.

A split Application and Stack configuration is required for on-chip OAD applications where only the stack or the application image is planned to be upgraded. Library configuration only supports full application + stack image off-chip OAD.

Convert the Stack Project in an executable image (CCS)

1. **Locate the StackLibrary .projectspec file**
   - An example file path for the **.projectspec** file for the **Multi Role project** is the following. Replace **<ver>** with the version of your SDK :
     `C:\ti\simplelink_cc2640r2_sdk_<ver>\examples\rtos\CC2640R2_LAUNCHXL\blestack\multi_role\tirtos\ccs\multi_role_cc2640r2lp_stack_library.projectspec`
   - Open it in a file editor and remove the line **outputType="staticLibrary"**
     This will change the output type of the library to "executable"
     Note: This change will make any pre-existing **FlashROM_StackLibrary** configurations unusable in your project
2. **Import the Application and Stack Project**
   - Changes to the .projectspec file will only reflect in your project when it is reimported
   - Go to **Project->Import CCS Projects…** and locate your application and stack project and import it
3. **Select the Stack Project as the Active Project**
   - It's recommended to create a new project build configuration: **Project** –> **Build Configurations** –> **Manage…**
   - Select **New** and supply a new name for the build configuration. We recommend using the name **FlashROM**.
   - Select **FlashROM_StackLibrary** for "Existing configuration".
4. **Set Created Build Configuration as Active**
   - Go to **Project** –> **Build Configurations** –> **Set Active** and select the build configuration created in Step 3.
5. **Open Project Properties**
   - Go to **Project** –> **Properties**
6. **Go to Build -> ARM Compiler -> Predefined Symbols**
   - If listed, remove the following predefined symbols in the "Pre-define NAME" list
     1. `STACK_LIBRARY`
     2. `GATT_NO_CLIENT`
7. **Go to Build -> ARM Linker**
   Click on **Edit Flags** and replace the text with:

```
-m"${ProjName}_${ConfigName}.map" --heap_size=0 --stack_size=256 -i"${XDC_LIBRARY_PATH}" -i"${CG_TOOL_ROOT}/lib" -i"${CG_TOOL_ROOT}/include" --reread_libs --define=CC26X0ROM=2 --
define=OSAL_SNV=1 --diag_suppress=16002-D --diag_suppress=10247-D --diag_suppress=10325-D --diag_suppress=10229-D --diag_suppress=16032-D --diag_wrap=off --display_error_number --
warn_sections --xml_link_info="${ProjName}_${ConfigName}_linkInfo.xml" --entry_point=startup_entry --rom_model
```

8. **Go to Build and select the Steps tab**
   - Within the **Pre-build steps** window, edit the file path to the **lib_linker.cmd** file to direct to the correct folder**.** This should only require replacing **<BuildConfigName>** in the file path below with the name of your new Build Configuration:
     `${PROJECT_LOC}/<BuildConfigName>/lib_linker.cmd`
     For example if your Build Configuration name is **FlashROM** the file path should now be:
     `${PROJECT_LOC}/FlashROM/lib_linker.cmd`
   - Add the *Frontier* boundary tool in **Post-build steps**:

```
${CG_TOOL_HEX} -order MS --memwidth=8 --romwidth=8 --intel -o ${ProjName}_${ConfigName}.hex ${ProjName}.out
```

```
${TOOLS_BLE_DIR}/frontier/frontier ccs ${PROJECT_LOC}/${ConfigName}/${ProjName}_${ConfigName}_linkInfo.xml ${PROJECT_LOC}/TOOLS/ccs_compiler_defines.bcfg
${PROJECT_LOC}/TOOLS/ccs_linker_defines.cmd
```

   Note: This tool hands information to the Application regarding Stack Entry Point.

9. **Go to Build->ARM Linker->File Search Path**
   - Add the following to the **Include library file or command file as input** window. Replace **<BuildConfigName>** with your Build Configuration name (ie. FlashROM):
     1. `${XDC_LIBRARIES}`
     2. `"${XDC_LIBRARIES}"`
     3. `"${SRC_BLE_DIR}/../../../source/ti/devices/cc26x0r2/driverlib/bin/ccs/driverlib.lib"`
     4. `"${PROJECT_LOC}/<BuildConfigName>/lib_linker.cmd"`
     5. `"${PROJECT_LOC}/<BuildConfigName>/ble_r2.symbols"`
     6. `"${SRC_BLE_DIR}/common/cc26xx/ccs/cc26xx_stack.cmd"`
     7. `libc.a`

10. **Go to Build->ARM Linker->Advanced Options->Command File Preprocessing**
    - Add the following to the **Pre-define preprocessor macro name_to _value_** window:
        1. `CC26X0ROM=2`
        2. `OSAL_SNV=1`
11. **Go to Build->ARM Linker->Advanced Options->Diagnostics**
    - Add the following to the **Suppress diagnostic <id>** window:
        1. `16002-D`
        2. `10247-D`
        3. `10325-D`
        4. `10229-D`
        5. `16032-D`
12. **Edit the cc26xx_stack.cmd file**
    - If your project is a **BLEStack Project** then go to the following file path in the SDK. Replace **<ver>** with the version of your SDK**:**
      `C:\ti\simplelink_cc2640r2_sdk_<ver>\source\ti\blestack\common\cc26xx\ccs\cc26xx_stack.cmd`
    - If your project is a **BLE5Stack Project** then go to the following file path in the SDK. Replace **<ver>** with the version of your SDK:
      `C:\ti\simplelink_cc2640r2_sdk_<ver>\source\ti\ble5stack\common\cc26xx\ccs\cc26xx_stack.cmd`
    - Open the **cc26xx_stack.cmd** in a text editor and add the following directly after the line `.emb_text` **:**

      ```
      .snvSectors     LOAD_START(SNVStart)
      ```

      Note: The file being edited is a part of the SDK. Any projects dependent on this file will also reflect this change.

Update the Application Project to use the executable Stack Project image (CCS)

1. **Select the Application Project as the Active Project**
    - It's recommended to create a new project build configuration: **Project** –> **Build Configurations** –> **Manage…**
    - Select **New** and supply a new name for the build configuration. We recommend using the name **FlashROM**.
    - Select **FlashROM_StackLibrary** for "Existing configuration".
2. **Set Created Build Configuration as Active**
    - Go to **Project** –> **Build Configurations** –> **Set Active** and select the build configuration created in Step 1.
3. **Open Project Properties**
    - Go to **Project** –> **Properties**
4. **Go to Build -> ARM Compiler -> Predefined Symbols**
    - If listed, remove the following predefined symbols in the "Pre-define NAME" list
        1. `STACK_LIBRARY`
        2. `ICALL_STACK0_ADDR`
5. **Go to Build->ARM Compiler->Advanced Options->Command Files**
    - **Add** the following to the **Read options from specified files** window. Replace **<StackProjectName>** with your stack project's name:

      ```
      "${WORKSPACE_LOC}/<StackProjectName>/TOOLS/ccs_compiler_defines.bcfg"
      ```

    - For example, if your stack project name is **multi_role_cc2640r2lp_stack_library**, the following should be added:

      ```
      "${WORKSPACE_LOC}/multi_role_cc2640r2lp_stack_library/TOOLS/ccs_compiler_defines.bcfg
      ```

6. **Go to Build->ARM Linker->File Search Path**
    - Remove the following from the **Include library file or command file as input** window.

      Note: **<StackProjectName>** is your stack project's name

        1. `"${WORKSPACE_LOC}/<StackProjectName>/FlashROM_Library/<StackProjectName>.lib"`
        2. `"${WORKSPACE_LOC}/<StackProjectName>/FlashROM_Library/ble_r2.symbols"`
        3. `"${WORKSPACE_LOC}/<StackProjectName>/FlashROM_Library/lib_linker.cmd"`
    - Add the following from the **Include library file or command file as input** window. Replace **<StackProjectName>** with your stack project's name:

        1. `"${WORKSPACE_LOC}/<StackProjectName>/TOOLS/ccs_linker_defines.cmd"`
        2. `"${SRC_BLE_DIR}/rom/ble_rom_releases/cc26xx_r2/Final_Release/common_r2.symbols"`

Now build and download the stack onto the device.
Then build, download and run the application on the device.