Introduction

The core requirement of every SSP project is the Board Support Package (BSP). This document shows how to use the Custom BSP Creator to create a custom BSP. After creating the BSP, you can use this BSP exactly as a BSP provided with a Renesas Synergy kit.

For more information about what is provided in a BSP please refer to the SSP User’s Manual.

Note on SSP Compatibility

Custom BSPs are only verified to work with the SSP version that they were created from. For example, a custom BSP created from SSP v1.0.0 is not verified to work with SSP v1.1.0. A custom BSP may be forward compatible, but in the event that it is not compatible, the user will be responsible for manually migrating their custom BSP to the new SSP version. This would typically be done by repeating the steps in this document, but using a newer SSP pack.

Contents

1. BSP Custom Creator Tool ........................................................................................................... 2
   1.1 Why create a custom BSP? ........................................................................................................ 2
   1.2 What changes with a custom BSP? .......................................................................................... 2
   1.3 How does the tool work? .......................................................................................................... 2

2. Example walkthrough .............................................................................................................. 4
   2.1 Step 1: Create a base pack .................................................................................................... 4
   2.2 Step 2: Create a template pack ............................................................................................. 5
   2.3 Step 3: Create a new project for the BSP and modify the BSP .............................................. 6
   2.4 Step 4: Create the final pack ................................................................................................. 9
1. BSP Custom Creator Tool

The Custom BSP Creator is a command-line tool for creating a custom BSP that can be used in e² studio.

1.1 Why create a custom BSP?

The BSP is a requirement of every SSP project and Renesas provides BSPs for all Renesas boards. The BSP is responsible for getting the MCU from reset to the main application (the `main()` function). The BSP also provides information to the SSP modules, so that the modules can be automatically configured for your hardware. Typically, users develop an application on a Renesas Synergy board and eventually replace the Renesas-provided BSP with a custom BSP matching their custom board.

1.2 What changes with a custom BSP?

Not everything in the BSP changes when you create a custom BSP. The BSP is made up of three folders in its source tree (synergy/ssp/src/bsp):

- `board`
- `cmsis`
- `mcu`

The `cmsis` and `mcu` folders do not need to change for a custom BSP. The `cmsis` folder contains the CMSIS-CORE subset of the BSP. The `mcu` folder contains the source that is common to MCU groups. For example, if there are two boards that both use a S7G2 MCU then they share the `mcu/s7g2` source folder.

You will be making changes for the custom BSP in the `board` folder. Each folder in the `board` folder is specific to a particular board. When you make changes for a custom BSP, the tool creates a new folder in the `board` directory that is specific to your custom board.

There are three required files in any board folder:

- `bsp.h`
- `bsp_init.c`
- `bsp_init.h`

The `bsp.h` header file contains all of the necessary includes and function prototypes needed for this custom BSP. The `bsp_init.c` source file contains code that must be executed before `main()` and is specific to this board. The code in `bsp_init.c` is called when the core BSP executes the `bsp_init()` function which resides in `bsp_init.c`. The code in the `bsp_init()` function can then call other functions as needed.

Examples from boards provided by Renesas Synergy include initializing external memory, such as SDRAM and QSPI, and defining the LEDs available. The code for initializing SDRAM and QSPI is contained in `bsp_sdram.c` and `bsp_qspi.c` respectively.

There is also an XML file associated with each BSP. This file is used by e² studio to enable configuration of the BSP through the Synergy Configurator inside e² studio. The `BSP`, `Clocks`, and `ICU` tabs are data derived from the information contained in this XML file.

1.3 How does the tool work?

The figure below provides an overview of the steps involved in creating a custom BSP with the Custom BSP Creator. The procedure includes using two executables and following four basic steps.
The first step is to create a base BSP to work from using the Custom BSP Creator executable, `custom_bsp_creator.exe`. Using this tool you can pick an existing BSP (for example the DK-S7G2) as a base for the new custom BSP.

Running this tool creates a base BSP. This base BSP exists as a set of files and folders on your PC. To be able to use and configure the base BSP in e2 studio, you need to create a new CMSIS Pack in step 2. CMSIS Packs are the mechanism that is used to deliver code inside e2 studio.

To create a pack, use the Custom Pack Creator, `custom_pack_creator.exe`. This tool takes the base BSP created in the previous step and packages it, so it can be used in e2 studio. One of the options of the tool is to create a template pack. Creating a template pack is required to make changes to the source code later in e2 studio.

Normal, non-template packs, do not allow the source code they provide to be modified. If you modify the code, then those modifications are overwritten the next time the project is built. To disable the overwriting feature, you must create a pack that can tell e2 studio to allow modifications.

After creating the template pack, you can modify the base BSP in e2 studio. At this point the base BSP is still a copy of an existing BSP. The existing BSP must be modified to meet the requirements of your new custom BSP. To do so, you can create a new project in e2 studio. Once the project is created, you can modify the BSP through the System Configurator in the same way as editing any other BSP. In addition, you can also add, modify, and remove source code from the BSP as needed. Typical updates to the BSP are:

- Pin configuration
- Stack and heap sizes
- Clock tree configuration
- Additional source in `board` directory to be run before `main()`

After making the changes, you can build and debug the project in e2 studio.

Once the custom BSP has been tested and is ready to be finalized, you can create the final pack in step 4. This step is essentially the same as step 2 except this time you are creating a standard pack instead of a template pack with the Custom Pack Creator. The BSP in this pack acts like any other BSP provided with a Renesas Synergy kit. Other users of the BSP can modify the BSP settings in the Synergy Configurator but cannot add, remove, or modify source files inside the BSP folder. The pack created from the last step can then be distributed to other team members to be used inside e2 studio.
2. Example walkthrough

This section shows how to create an example custom BSP based on the DK-S7G2. For illustration purpose, it is assumed that custom_bsp_creator.exe and custom_pack_creator.exe are located under the `<e2 studio_install_directory>\internal\projectgen\arm\Packs` directory. Otherwise, the absolute path of the SSP pack needs to be provided with the –p option when using these executables.

2.1 Step 1: Create a base pack

To create a base pack, follow these steps (the first step and second step are optional and serves as an introduction to the executable):

1. From the Windows command prompt, enter the directory where the custom_bsp_creator.exe is located. Next, run `custom_bsp_creator.exe -h` to print the command line options for the tool.

   ```
   custom_bsp_creator.exe -h
   Usage: custom_bsp_creator.exe [options]
   Custom BSP Creator - This script is used to provide a customized BSP based on the user's base board and board name. The custom BSP is generated from a PACK file (eg. To create a board named 'wolfpack' based off the S7G2-DK, run the command: custom_bsp_creator.exe -p pack_location -b s7g2_dk -n wolfpack
   -# part_number -d output
   Options:
   -h, --help            show this help message and exit
   -p PACK_PATH, --pack=PACK_PATH
   Path to pack which custom BSP will be based upon
   -l, --list_boards     Prints list of available base boards
   -b BOARD, --base_board=BOARD
   Board which custom BSP will be based upon (eg s3a7_dk)
   -n USER_BOARD_NAME, --name=USER_BOARD_NAME
   Name of custom board (eg wolfpack)
   -# PART_NUMBER, --part=PART_NUMBER
   Part Number for MCU (eg R7FS7G27H2A01CBD, R7FS3A77C3A01CFB, R7FS124773A01CFM)
   -d GENERATED_BSP_DESTINATION, --dest=GENERATED_BSP_DESTINATION
   Destination path (eg C:\Users\John\Documents). Default is 'tmp'.
   --verbose             Enable verbose output
   ```

2. Find the available base BSPs using the -l option. A valid pack must be provided using -p option. Packs can be found under the `<e2 studio_install_directory>\internal\projectgen\arm\Packs` directory.

   ```
   custom_bsp_creator.exe -l -p Renesas.Synergy.1.0.0.pack
   s3a7_dk
   s3a7_user
   s7g2_dk
   s7g2_pe_hmi1
   s7g2_sk
   s7g2_user
   ```

   Note: This operation will take more than several seconds to complete. In addition, this example uses SSP1.0.0 as example. User can choose the SSP pack they are using in this command.

3. Create a custom BSP with the following options:

   a. As shown previously, use the v1.0.0 pack (`Renesas.Synergy.1.0.0.pack`) for the -p option.

   b. Since we are using the DK-S7G2 for the base BSP, use `s7g2_dk` as the -b option.

   c. The part number of our MCU is the same as the DK-S7G2. Use `R7FS7G27H2A01CBD` for the -# option.
**2.2 Step 2: Create a template pack**

Follow these steps to create a template pack, one that will allow modifications to the BSP source code. Step 1 is optional and serves an introduction to the executable. Creating the template pack involves using the new BSP in e² studio:

1. Use the `custom_pack_creator.exe` with the `-h` option to see the command-line options.

   ```shell
custom_pack_creator.exe -h
Usage: custom_pack_creator.exe [options]
Pack Generator for Custom BSP - This is used to create a custom pack based on
a modified bsp which originated from the custom_bsp_creator script. eg)
custom_pack_creator.exe -p CUSTOM_BSP_PATH -n PACK_NAME -v 1.0.0
```

Options:

- `-h`, `--help` show this help message and exit
- `-p CUSTOM_BSP_PATH`, `--path=CUSTOM_BSP_PATH`
  Path to Custom BSP (eg C:\CustomBSP”). Folder must contain 'synergy' folder. Default is current directory.
- `-n USER_BOARD_NAME`, `--name=USER_BOARD_NAME`
  Name of custom board (eg wolfpack)
- `-v PACK_VERSION`, `--version=PACK_VERSION`
  Version Number for Generated Pack (eg 1.0.0). Default is 1.0.0.
- `-t`, `--template`
  Creates a pack with modifiable BSP that can be used for creating final BSP
- `-e E2STUDIO_PATH`, `--e2studio=E2STUDIO_PATH`
  Path to e2studio (eg C:\Renesas\e2_studio). Default is C:\Renesas\e2_studio.
- `--verbose` Enable verbose output

2. Before continuing, make sure that e² studio is not open. If e² studio is open during a new pack generation, the tool may fail since e² studio can have the files open.

3. Create a template pack using the following options:
   
a. The base BSP is located in the `output` folder that was just created. We will provide this path for the `-p` option.
   
b. The board name is `wolfpack` so we will provide that as the `-n` option
   
c. For this example we will set the version using the `-v` option to 1.0.0
   
d. This is a template pack so we will use the `-t` option
e. The path to e² studio will vary based on where you installed it. In this example we will use the e² studio installation located at `C:\Renesas\synergy-e2_studio`. This will be provided as the `-e` option.

f. Once again if `--verbose` is not provided then the tool will not output any status. We will provide the `--verbose` option.

```bash
custom_pack_creator.exe -p output -n wolfpack -v 1.0.0 -t -e C:\Renesas\synergy-e2_studio --verbose
```

Removing old release
Copying files
Generating .psdc

The custom pack has now been created.

### 2.3 Step 3: Create a new project for the BSP and modify the BSP

To configure the custom BSP follow these steps:

1. Start e² studio and create a new project based on this BSP. The ISDE project configurator shows the BSP template you created as selectable Board option in the Project Configuration and Project Template Selection windows.
If you expand the `synergy/ssp/src/bsp/board` directory in the Project Explorer window you will see your custom board folder.

![Project Explorer window with custom board folder]

After the project has been created, e² studio must be configured to treat the BSP module as a template module. Only perform this step when working with template pack BSPs.

2. Close e² studio.

3. On your PC, navigate to the folder of the project you just created. In this example the folder was `C:\Workspace\wolfpack_template`.

4. Open the `configuration.xml` file in a text editor.

5. Find the `<component>` element that relates to the BSP. It is the only element with its `class` attribute set to "BSP". Below is the `<component>` for this example.

   ```xml
   <component apiversion="" class="BSP" condition="" group="USER" subgroup="wolfpack" variant=""
   vendor="Renesas" version="1.0.0">  
   <description>[TEMPLATE] Board Support Package for wolfpack</description>  
   <originalPack>user.wolfpack_template.1.0.0.pack</originalPack>
   </component>
   ```

6. Under this `<component>` element add the following element: `<template>true</template>`. Using the example in the previous step the `<component>` now looks like this:

   ```xml
   <component apiversion="" class="BSP" condition="" group="USER" subgroup="wolfpack" variant=""
   vendor="Renesas" version="1.0.0">  
   <description>[TEMPLATE] Board Support Package for wolfpack</description>  
   <originalPack>user.wolfpack_template.1.0.0.pack</originalPack>  
   <template>true</template>
   </component>
   ```

7. Save the file and close it.

8. Restart e² studio and reopen the project you created previously.

   **WARNING:** If you accidentally create a second component instead of modifying the first one, e² studio will crash.

In the next step, make modifications to the BSP to simulate the customization process. As previously mentioned, in most cases, the modification included pin configuration, stack and heap size, clock tree configuration and any additional source in board directory that needs to run before main.
9. Add two empty files to the synergy/ssp/src/bsp/board/wolfpack/ directory: `bsp_extra.c` and `bsp_extra.h`.

These two files are used to test the creation of the final non-template pack.

10. Open the Synergy Configurator and make the following changes:
    - In Properties window of the BSP tab, change the Main Stack size to 0x2000. In your application, you may need a different Main Stack size.
• Change the system clock source (Clock Src) to the HOCO. In your application, you should choose whichever clock suitable for you.

• Enable the CAN0 pins in the Pin Configurator using the Peripherals view. This is an example of configuring a peripheral in the customer BSP package.

11. Save the changes by clicking the Generate Project button and build the project.

The final, customized BSP has been created. Now we need to put it in a final pack.

2.4 Step 4: Create the final pack

You can now create a final, non-template, pack with the Custom Pack Creator. Remember to close e² studio before performing this step. Go through the previous steps with the following differences:

• The path to the custom BSP is now below the root of the project you created in earlier steps. For this example, we will assume the project was created under C:\Workspace\wolfpack_template and there is a synergy directory under this folder. Notice that there must be a synergy directory in the path that is given to the tool.
• Since this pack will not be a template we will remove the `-t` option that we used previously.

  ```
  custom_pack_creator.exe -p C:\Workspace\wolfpack_template -n wolfpack -v 1.0.0
  -e C:\Renesas\synergy-e2_studio --verbose
  ```

  Removing old release
  Copying files
  Generating `.psdc`

  Before using the final template, remove the template pack from e² studio:

  Navigate to `<e² studio_install_directory>\internal\projectgen\arm\Packs` and delete the template pack or move it outside of the e² studio directory. For this example the name of the template pack is `user.wolfpack_template.1.0.0.pack`.

  Open up e² studio and create a new project based on the new custom board.

![Image](image.png)

Once the project has been created, verify that the changes made to the template are still in the new project.

To verify that this is the final pack, delete the `bsp_extra.c` and `bsp_extra.h` files in the `synergy/ssp/src/bsp/board/wolfpack/` directory. After deleting the files, build the project and verify that the files are brought back into the project.

You have now created a custom BSP. To share this BSP, you can distribute the newly created pack in the `<e² studio_install_directory>\internal\projectgen\arm\Packs` directory. For this example, the name of the final pack is `user.wolfpack.1.0.0.pack`. 
Website and Support

Support: https://synergygallery.renesas.com/support

Technical Contact Details

- America: https://renesas.zendesk.com/anonymous_requests/new
- Europe: http://www.renesas.eu/support/index.jsp
- Japan: http://japan.renesas.com/contact/index.jsp
## Revision History

<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Page</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Oct 12, 2015</td>
<td>-</td>
<td>Initial Version</td>
</tr>
<tr>
<td>1.01</td>
<td>Oct 16, 2015</td>
<td>All</td>
<td>Minor editorial updates</td>
</tr>
<tr>
<td>1.02</td>
<td>Jan 12, 2016</td>
<td>12</td>
<td>Support URL updated</td>
</tr>
<tr>
<td>1.03</td>
<td>Mar 17, 2016</td>
<td>-</td>
<td>Minor typos fixed. Updates based on latest pack creator.</td>
</tr>
<tr>
<td>1.04</td>
<td>Nov 18, 2016</td>
<td>1</td>
<td>Title format changed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12</td>
<td>Support URLs updated</td>
</tr>
</tbody>
</table>