## Contents

1.0 Introduction ......................................................................................................................... 6  
   1.1 Purpose ................................................................................................................................. 6  
   1.2 Intended Audience ................................................................................................................. 6  
   1.3 Related Documents .............................................................................................................. 6  
   1.4 Conventions ........................................................................................................................... 6  
   1.5 Acronyms and Terminology ................................................................................................. 7  
2.0 Intel® Atom™ Processor C2000 Product Family Hardware Platform ................................. 8  
   2.1 Intel® Firmware Support Package ......................................................................................... 9  
3.0 Customer Reference Board .................................................................................................... 10  
4.0 Example Boot Loader ............................................................................................................. 12  
   4.1 Example Boot Loader Design ............................................................................................... 12  
   4.2 Boot Loader Development Environment .............................................................................. 12  
   4.3 Preparing the Coreboot Build Environment ........................................................................ 13  
5.0 Building the Example Boot Loader ........................................................................................ 14  
6.0 Updating the Firmware .......................................................................................................... 16  
   6.1 Programming the Firmware ................................................................................................. 16  
   6.1.1 Connecting the SF100 to the CRB ................................................................................. 17  
   6.2 Creating a Firmware Backup .............................................................................................. 19  
   6.3 Programming a Complete Firmware Image ....................................................................... 20  
   6.4 Updating Only the Boot Loader ......................................................................................... 21  
   6.5 Booting the Example Boot Loader ..................................................................................... 22  
7.0 Creating Custom Images ....................................................................................................... 24  
   7.1 Edit Coreboot Image Specifications ...................................................................................... 24  
   7.2 Binary Configuration Tool .................................................................................................. 24  
   7.2.1 How to Enable Memory Down ...................................................................................... 25  
   7.2.2 How to Enable Fast Boot .............................................................................................. 25  
   7.2.3 Debug and Validation .................................................................................................... 25
Figures

Figure 1. Intel® Atom™ Processor C2000 Product Family Platform Block Diagram........9
Figure 2. Intel® Atom™ Processor C2000 CRB Block Diagram.................................10
Figure 3. Mini-USB Port Location..............................................................................11
Figure 4. SF100 Programmer.....................................................................................16
Figure 5. DediProg SF100 Universal Adapter.........................................................17
Figure 6. DediProg 1.27 mm 8-pin Header Cable Shown Connected to Adapter Board..18
Figure 7. SPI_EMUL Header Location on the CRB....................................................19
Figure 8. Pinout for the CRB's F_PANEL header....................................................23

Tables

Table 1. Related Documents......................................................................................6
Table 2. Acronyms and Terminology.........................................................................7
Table 3. Sequence Details.........................................................................................22
## Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 2015</td>
<td>002</td>
<td>Modified for POSTGOLD4 release: • Added procedures for enabling Rank Margining Tool, Fast Boot, Memory Down, Serial Debug Messages</td>
</tr>
<tr>
<td>December 2014</td>
<td>001</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>
1.0 Introduction

1.1 Purpose

The purpose of this document is to provide information about the Intel® Atom™ Processor C2000 Product Family Customer Reference Board (CRB), code-named Mohon Peak (hereafter referred to as “the CRB”), with guidance for building an example boot loader for the CRB that is based on the Intel® Firmware Support Package (FSP).

1.2 Intended Audience

This document is targeted at all platform and system developers who intend to use an FSP-based boot loader for the firmware solution for their overall design based on the Intel® Atom™ Processor C2000 Product Family. This group includes, but is not limited to, system BIOS developers, boot loader developers, and system integrators.

1.3 Related Documents

Table 1. Related Documents

<table>
<thead>
<tr>
<th>Document</th>
<th>Document No./Location</th>
</tr>
</thead>
</table>

1.4 Conventions

To better illustrate some of its points, this document may provide code snippets. Such code snippets follow the GNU C Compiler and GNU Assembler syntax.
1.5 Acronyms and Terminology

Table 2. Acronyms and Terminology

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCT</td>
<td>Binary Configuration Tool</td>
</tr>
<tr>
<td>CRB</td>
<td>Customer Reference Board</td>
</tr>
<tr>
<td>FSP</td>
<td>Firmware Support Package</td>
</tr>
<tr>
<td>PCD</td>
<td>Platform Configuration Database</td>
</tr>
<tr>
<td>SoC</td>
<td>System on a Chip</td>
</tr>
<tr>
<td>SPI</td>
<td>Serial Peripheral Interface</td>
</tr>
<tr>
<td>TSEG</td>
<td>Top Segment, a reserved segment of memory at the top of its address space to be used as System Management RAM (SMRAM)</td>
</tr>
</tbody>
</table>

§
2.0  Intel® Atom™ Processor C2000 Product Family Hardware Platform

The Customer Reference Board (CRB) is based on the Intel® Atom™ Processor C2000 Product Family hardware platform, code-named Rangeley (hereinafter referred to as "the hardware platform").

This hardware platform consists of a system on a chip (SoC), which is based on two to eight fourth-generation Intel® Atom™ processor cores. The hardware platform is built on 22-nanometer process technology.

Some of the features of the hardware platform are the following:

- Support for optional Error Correcting Code (ECC) RAM
- Support for optimizing data plane solutions with the Intel® Data Plane Development Kit
- Intel® Virtualization Technology (Intel® VT)
- Intel® Advance Encryption Standard New Instructions (Intel® AES-NI)
- Intel® QuickAssist Technology on select SKUs
- Four integrated Intel® 10/100/1000/2500 Gigabit Ethernet controllers
- Up to four SATA 2 ports, and two SATA 3 ports
- Up to 16 PCI Express Gen 2.0 lanes with four controllers

Figure 1 illustrates the hardware platform’s major components.
2.1 Intel® Firmware Support Package

An Intel® Firmware Support Package (FSP) is a firmware component provided in binary form that contains initialization code for a specific Intel platform. Engineers building systems that are based on a particular platform can integrate the corresponding FSP with the boot loader of their choice.

The FSP for the hardware platform handles the initialization of the processor, memory, and I/O subsystems for hardware designs based on this hardware platform.
3.0 Customer Reference Board

The CRB is based on the Intel® Atom™ C2000 Processor hardware platform. Figure 2 shows a block diagram of the CRB’s components.

Figure 2. Intel® Atom™ Processor C2000 CRB Block Diagram

The board provides an on-board USB to UART bridge for the rear panel mini-USB connector, which is shown circled in red in Figure 3. If you are developing a boot loader for which all user interaction is performed through the serial console, connect a terminal to this mini-USB port.
Once connected, the host system should detect the USB-to-serial adapter. Compatible drivers should install automatically on host systems running Microsoft Windows®. Other host operating systems may require downloading and installing appropriate drivers from the bridge chip manufacturer’s site at:

The default serial communication parameters for the serial console are:

- 115,200 baud
- no parity
- 8 data bit, 1 stop bit configuration
- No flow control

If you are developing a boot loader for which user interaction is performed through a monitor, connect a monitor to the included video card from ASPEED Technology, Inc. or to another video card you provide.
**4.0 Example Boot Loader**

Intel® provides and supports the Intel® Atom™ Processor C2000 Product Family Firmware Support Package (hereafter referred to as the “FSP kit”) for the CRB. However, Intel does not provide or support a complete boot loader solution for this board.

The embedded firmware ecosystem has developed an example boot loader solution for the CRB that uses the FSP kit. This solution is based on the open source Coreboot project at coreboot.org. While Intel does not endorse or support boot loader solutions based on the Coreboot project, the example Coreboot-based boot loader provides a good teaching model for how to integrate the Intel FSP into a complete boot loader solution.

*Note:* The steps to generate the example boot loader for the CRB is provided to Intel customers as-is, with no warranty or support. Please contact a firmware ecosystem vendor to help you develop a production-worthy firmware solution based on the Intel® FSP for your hardware designs.

**4.1 Example Boot Loader Design**

The FSP’s role is to initialize the processor, memory, and I/O subsystem. The example Coreboot-based boot loader calls into the FSP for these initialization steps, then goes on to prepare and load a primary target that the Coreboot project calls a payload. See the Coreboot project’s documentation for further details.

The default payload for the example Coreboot-based boot loader is the SeaBIOS, which is provided by a related open source firmware project. The SeaBIOS attempts to boot an OS image from a storage device attached to the board.

**4.2 Boot Loader Development Environment**

Although the FSP can be used with any software development environment, the example Coreboot-based boot loader described herein was developed using Fedora® 18 Linux® and the standard GNU development tools.

*Note:* Intel does not endorse or support any specific development environment for developing boot loader firmware that integrates with the FSP.
4.3 Preparing the Coreboot Build Environment

To download the Coreboot.org development system, your development host must have the Git version control system installed.

Once Coreboot has been downloaded, you must run a command to download and build the exact GCC tool chain required by the Coreboot project. In order to build the Coreboot-specific tool chain, your development host must have its own distribution-provided GCC tool chain. Consult the documentation at coreboot.org for the tool chain components required.

On your development host, follow these steps to download the Coreboot.org development environment and then build its required GCC tool chain:

1. Create a directory to contain your project. This directory is to be the parent of the coreboot directory. For the purpose of these example steps for the CRB, we will refer to this as the MP directory.
2. Navigate into your new MP project directory.
3. Use the git clone command as follows:

   ```
   git clone http://review.coreboot.org/p/coreboot
   ```

   This step downloads a directory named coreboot.
4. Navigate into the coreboot directory.
5. Run the following command to build the project-specific GCC tool chain required by the Coreboot project:

   ```
   make crossgcc-i386
   ```

   This command can take from a few minutes up to an hour or more to complete, depending on the power of your development host.

   If make crossgcc-i386 fails to build (which can happen on recent Ubuntu Linux** systems), try these alternate steps:
   a. In the coreboot directory, navigate to the util/grossgcc subdirectory.
   b. Run ./buildgcc. This command takes the same amount of time to complete as the make version.
   c. When complete, return to the root of the coreboot directory with this command:

   ```
   cd ../..
   ```

   If you continue to have difficulty getting the tool chain to build, make sure that you have all the required development tools installed on your Linux* system, as listed on the coreboot.org site. If you are still unable to get the tool chain to build, then consider using an alternate Linux* distribution such as Fedora 18.
5.0  Building the Example Boot Loader

To integrate the Intel® FSP with the Coreboot.org project to build the example boot loader for the Mohon Peak CRB, you must do the following:

- Download and install the FSP kit.
- Copy files from the FSP kit to a new directory parallel to the coreboot directory that was created in the previous chapter.
- Run the make menuconfig command to specify the Mohon Peak CRB and its build options.
- Run make.

Follow these steps:

1. Go to www.intel.com/fsp and download the Intel® Atom™ Processor C2000 Product Family FSP kit, which is distributed both as a Microsoft Windows® executable file, RANGELEY_FSP_KIT_POSTGOLD*.exe, and as a Linux* archive, RANGELEY_FSP_KIT_POSTGOLD*.tgz. You can use either version of the FSP kit to build the example boot loader.

2. Install the kit:
   - For Windows: Execute the RANGELEY_FSP_KIT_POSTGOLD*.exe file and follow the on-screen dialogs to install the FSP kit. The default installation directory is C:\RANGELEY_FSP_KIT.
   - For Linux*: Extract the contents of the RANGELEY_FSP_KIT_POSTGOLD*.tgz file and follow the instructions in the Readme_Extract.txt file. The FSP kit extracts into a subdirectory named RANGELEY_FSP_KIT.

3. Navigate to the MP directory created in the previous chapter. Create a new subdirectory named intel parallel to the coreboot directory.

4. Copy files from the RANGELEY_FSP_KIT directory where the FSP kit was installed to the MP/intel directory on your development host as follows, creating the path to the specified target directories as required:
   a. Copy RANGELEY_FSP_KIT/Microcode/*.h to MP/intel/cpu/rangeley/microcode
   b. Copy RANGELEY_FSP_KIT/FSP/*.fd to MP/intel/fsp/Rangeley and rename the file to FvFsp.bin
   c. Copy RANGELEY_FSP_KIT/FSP/include/*.h to MP/intel/fsp/rangeley/include
   d. Copy RANGELEY_FSP_KIT/FSP/srx/*.c to MP/intel/fsp/rangeley/srx
5. Navigate to the MP/coreboot directory and run:

```
make menuconfig
```

6. Open the Mainboard menu. Select Intel as the Mainboard vendor; then select Mohon Peak CRB as the Mainboard model.

7. While still in the Mainboard menu, move to the Configure defaults for the Intel FSP package option and press the space bar to select it. Leave all other settings in their default state.

8. Back out to the main menu; then select the Console option. Navigate to the Default console log level option and set it to 4: WARNING.

9. Select Exit twice to return to the command prompt. Save the configuration when prompted to do so.

10. Now, build the boot loader with a single command:

```
make
```

11. If the build completes without errors, the newly created firmware image is generated into the following directory and file:

```
coreboot/build/coreboot.rom
```

12. The generated `coreboot.rom` file is 2 MB in size. This file can be programmed into the firmware flash memory device on the CRB by following the procedures in the next section.
6.0 Updating the Firmware

The CRB is equipped with two 8 MB flash devices that each contain all of the system firmware, including the Firmware Descriptor and the BIOS or Boot Loader. The CRB allows you to select which device to use via a jumper on the board. By default, the CRB comes from its manufacturer with a standard BIOS installed identically in both flash chips.

To replace the factory BIOS with boot loader firmware, you can generate a complete 8 MB firmware image and update the whole firmware flash device, or you can update only the BIOS region of the firmware image with the new boot loader. For preproduction or development purposes, updating only the boot loader is an acceptable method. However, for production purposes, use a complete 8 MB firmware image. Note that information about how to generate a complete 8 MB firmware image is outside the scope of this document.

6.1 Programming the Firmware

The flash device on the CRB can be programmed with new firmware using a DediProg SF100 programmer, is shown in Figure 4.

Figure 4. SF100 Programmer

The SF100 connects to a host development system through its USB plug for communication with the controller software and to obtain power, and it connects to the flash device to be programmed via the ISP pin header.
Additional technical information about the SF100 programmer, including drivers and software for Microsoft Windows* environments, can be obtained from DediProg’s website at:
http://www.dediprog.com/product/SPI%20Flash%20Solution/89

The SF100 programmer is also supported by the Linux** flashrom utility. The flashrom utility is not supported by DediProg or by Intel. Additional technical information about the flashrom utility can be obtained from the flashrom website at:
http://www.flashrom.org/

All of the following instructions regarding the use of the SF100 programmer assume that the DediProg SF100 drivers and software are installed on a PC running Microsoft Windows.

6.1.1 Connecting the SF100 to the CRB

The Dediprog SF100 includes the SF100 Universal Adapter, which is illustrated in Figure 5 and again in Figure 6, shown with a 1.27 mm 8-pin header cable attached.

Figure 5. DediProg SF100 Universal Adapter
Connect the DediProg 1.27 mm 8-pin header cable to the SF100 Universal adapter as shown in Figure 6. Connect the other end to the SPI_EMUL header on the CRB. Locate the SPI_EMUL header as shown circled in red in Figure 7.
6.2 Creating a Firmware Backup

It is important to back up any existing working firmware so that you can always restore the system to a known working condition. Use the following procedure to back up the existing firmware.

1. Disconnect the power from the CRB. Do not apply power to the CRB at any time during this procedure.
2. Select which of the CRB’s two BIOS chips you want to back up, using jumper JP4. Close pins 1-2 to select BIOS chip 1, or close pins 2-3 to select BIOS chip 2. The BIOS chips are in a redundant configuration, so you can use either one.
3. Move jumper J22 from closing pins 1-2 to pins 2-3. This allows the selected BIOS chip to be backed up.
4. Attach the SF100 Universal Adapter board to the ISP pin header of the SF100.
5. Connect the cable from the SF100 Universal Adapter to the CRB, as described in the previous section.
6. Open the DediProg Engineering application.
7. If a dialog asks you to select a chip part number, select M25P64.
8. Next to the currently working on section near the top of the window, make sure that Application Memory Chip 1 is selected. Set this the same way regardless of whether you are backing up BIOS chip 1 or 2.

9. Click the Edit button at the top of the window.

10. Click Read at the top of the window. Wait for the completion of the read operation.

11. Click Chip Buffer to File to save the 8 MB firmware image to a file.

12. To restore normal operation of the CRB, move the jumper at JP22 back to close pins 1-2.

13. Remove the cable from the SPI_EMUL header on the CRB before applying power to the CRB.

### 6.3 Programming a Complete Firmware Image

Use the following procedure to program a complete firmware image to the CRB, such as when restoring the firmware image that was backed up in the previous section, or when programming a complete 8 MB firmware image in a production scenario. If you just want to update the BIOS region of the firmware image with a new boot loader, skip this section and go to section 6.4, Updating Only the Boot Loader.

1. Disconnect power from the CRB. Do not apply power to the CRB at any time during this procedure.

2. Select which of the CRB’s two BIOS chips you want to program with the complete firmware image, using jumper JP4. Close pins 1-2 to select BIOS chip 1, or close pins 2-3 to select BIOS chip 2. The BIOS chips are in a redundant configuration, so you can use either one.

3. Move jumper J22 from closing pins 1-2 to pins 2-3. This allows the selected BIOS chip to be programmed.

4. Attach the SF100 Universal Adapter board to the ISP pin header of the SF100.

5. Connect the cable from the SF100 Universal Adapter to the CRB, as described in section 6.1.1, Connecting the SF100 to the CRB.

6. Open the DediProg Engineering application.

7. If a dialog asks you to select a chip part number, select M25P64.

8. Next to the currently working on section near the top of the window, make sure that Application Memory Chip 1 is selected. Set this the same way regardless of whether you are programming BIOS chip 1 or 2.

9. Click the Config button at the top of the window.

10. In the Advanced Settings dialog, click the Prog button on the left.

11. Select Program a whole file starting from address 0 of a chip.

12. Click the Flash Options button on the left.

13. Select the check box Unprotect block automatically when block(s) protected.

14. Click OK.
15. Click the File button at the top of the window. This opens a file selection dialog for selecting the complete 8 MB firmware file to be programmed to the flash device, such as a backup file made as described in the previous section.

16. Click the Erase button at the top of the window to erase the entire selected flash memory device. Wait for completion of the erase operation.

17. Click the Prog button at the top of the window to program the selected firmware image to the flash memory device on the target platform. Wait for completion of the programming operation.

18. Click the Verify button at the top of the window to verify that the firmware image was successfully programmed to the flash memory device. Wait for completion of the verify operation.

19. To restore normal operation of the CRB, move the jumper at JP22 back to close pins 1-2.

20. Remove the cable from the SPI_EMUL header on the CRB before applying power to the CRB.

6.4 Updating Only the Boot Loader

You can update only the boot loader portion of the firmware without having to program the complete 8 MB firmware image. For example, the 2 MB boot loader built in section 5.0 can be programmed to the last 2 MB of the 8 MB firmware image.

The DediProg software for the SF100 allows updating just the boot loader portion of the firmware image using a batch programming process. Use the following procedure to program the boot loader code to the BIOS region.

1. Disconnect power from the CRB. Do not apply power to the CRB at any time during this procedure.

2. Select which of the CRB's two BIOS chips you want to program with the boot loader image, using jumper JP4. Close pins 1-2 to select BIOS chip 1, or close pins 2-3 to select BIOS chip 2. The BIOS chips are in a redundant configuration, so you can use either one.

3. Move jumper J22 from closing pins 1-2 to pins 2-3. This allows the selected BIOS chip to be programmed.

4. Attach the SF100 Universal Adapter board to the ISP pin header of the SF100.

5. Connect the cable from the SF100 Universal Adapter to the CRB, as described in section 6.1.1, Connecting the SF100 to the CRB.

6. Open the DediProg Engineering application.

7. If a dialog asks you to select a chip part number, select M25P64.

8. Next to the Currently working on section near the top of the window, make sure that Application Memory Chip 1 is selected. Set this the same way regardless of whether you are programming BIOS chip 1 or 2.
9. Click the **File** button at the top of the window. This opens a file-open dialog for selecting the file containing the boot loader file to be programmed to the CRB. Select the `coreboot.rom` file that you built in section 5.0.

10. Click the **Config** button at the top of the window.

11. In the Advanced Settings dialog that appears, click the Batch button on the left.

12. Under Batch Operation Options, select Update memory according to Region configuration, select Region 1, and enter 600000 to 7FFFFF.

13. Select the Identify Chip and Require Verification after completion check boxes. Make sure all other check boxes are cleared (unchecked).

14. Verify that the Sequences Details at the bottom of the window are as shown in Table 2.

<table>
<thead>
<tr>
<th>Steps</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Identify before operation starts.</td>
</tr>
<tr>
<td>2</td>
<td>Read from the chip.</td>
</tr>
<tr>
<td>3</td>
<td>Blank check.</td>
</tr>
<tr>
<td>4</td>
<td>Erase chip (if not blank).</td>
</tr>
<tr>
<td>5</td>
<td>Program chip.</td>
</tr>
<tr>
<td>6</td>
<td>Verify after operation completes.</td>
</tr>
</tbody>
</table>

15. Click the Flash Options button on the left.

16. Select the check box Unprotect block automatically when block(s) protected.

17. Click OK to close the Advanced Settings dialog.

18. Click the Batch button at the top of the window. Wait for completion of the batch operation.

19. To restore normal operation of the CRB, move the jumper at JP22 back to close pins 1-2.

20. Remove the cable from the SPI_EMUL header on the CRB before applying power to the CRB.

6.5 **Booting the Example Boot Loader**

By default, the Coreboot-based example boot loader boots into the Coreboot-provided SeaBIOS. The default behavior of SeaBIOS is to seek a bootable OS image on an attached storage device, such as a SATA or USB disk.

To interact with the boot loader, connect a terminal or development host to the mini-USB port, as discussed in section 3.0, Customer Reference Board.
The CRB provides a header labeled `F_PANEL` for connecting power and reset buttons. If the CRB is installed in a case, then connect the case's power and reset buttons to this header. If the CRB is not in a case, connect power and reset buttons directly to this header. The pinout for the `F_PANEL` header is shown in Figure 8.

Figure 8. Pinout for the CRB's F_PANEL header

To power up the board, turn on the power supply, and then press the power button.

When power is applied to the CRB and the boot loader initializes the board and boots the SeaBIOS payload, various messages appear on the terminal connected to the mini-USB port.

If a storage device with a bootable OS image is connected to one of the board's SATA or USB ports, the booting OS displays messages on the serial port terminal, or on an attached monitor, as determined by the configuration of the OS image.

If the OS image provides a graphical user interface on the monitor, you may need to attach a keyboard and mouse to the USB ports in order to fully interact with the booted operating system.
7.0 Creating Custom Images

There are two ways to edit the components of a boot loader image to specify custom settings:

- Edit the Coreboot image specifications, then rebuild the coreboot.rom image file.
- Edit the binary FSP file to include or exclude support for hardware features, or to rebase the firmware volume image.

7.1 Edit Coreboot Image Specifications

Use the menuconfig utility provided with the Coreboot distribution to edit the specifications of the coreboot.rom image file to be generated.

The following example shows the steps to build a 4 MB image file instead of a 2 MB image file.

1. At the shell prompt on your development host, navigate to the coreboot directory.
2. Start the menuconfig utility with the following command:
   
   make menuconfig

3. Use the arrow keys to select the Mainboard option.
4. Select the ROM chip size option.
5. Use the arrow keys to select the 4096 KB option.
6. Select Exit twice.
7. Select Yes to save your new configuration and exit the menuconfig utility.
8. Back at the Coreboot directory, type make to rebuild the coreboot.rom image file.

To update the boot loader with a 4 MB coreboot.rom file, follow the steps in 6.4, Updating Only the Boot Loader, except in step 12 specify a region of 400000 to 7FFFFFF.

7.2 Binary Configuration Tool

Intel provides the Binary Configuration Tool (BCT) that lets you edit the FSP binary file delivered with the Rangeley FSP kit. Use the BCT for two purposes:

- To change the values in the Platform Configuration Database (PCD) within the FSP binary. For example, you might need to change the SMBus addresses for the DIMM slots on your production board if they are different from the addresses used for the DIMM slots on the CRB.
To rebase the firmware binary. The code within the FSP binary is not relocatable code. Therefore, to locate it at an address in the boot loader other than its default address of 0xFFF80000, you must rebase the FSP binary to the desired address.

Each Intel® FSP release is packaged with a platform-specific binary settings file (.bsf), which is a text file that represents the default PCD settings in the FSP binary file as it is provided by Intel. Using the BCT, you can change the values of the settings listed in the .bsf file. The modified settings are saved in an as-built settings file (.absf). After modifying the settings, the BCT lets you patch those changes back into the binary image.

The BCT package is a standalone tool with its own user guide, and is not dependent on a particular CPU, chipset, or platform. Please refer to the BCT release package for further information on using this tool.

7.2.1 How to Enable Memory Down

Memory down is disabled by default. The following steps show how to enable Memory Down using BCT.
1. Start the BCT and open the RangeleyFsp.bsf file that was included with the FSP kit.
2. In the North Complex section, set Memory Down to Enabled.
4. Apply the settings to the FSP binary file.

7.2.2 How to Enable Fast Boot

Fast Boot is disabled by default. The following steps show how to enable Fast Boot using BCT.
1. Start the BCT and open the RangeleyFsp.bsf file that was included with the FSP kit.
2. Set Fast Boot to Enabled.
4. Apply the settings to the FSP binary file.

7.2.3 Debug and Validation

The BCT tool can be used to enable debug logging thru serial port that can assist with debug. BCT can enable the Rank Margining Tool for Memory testing.

7.2.3.1 How to Enable Serial Debug Messages

Serial Debug Messages is disabled by default. The following steps show how to enable Serial Debug Messages using BCT.
Creating Custom Images

1. Start the BCT and open the RangeleyFsp.bsf file that was included with the FSP kit.
2. In the Platform Settings section:
   a. Set Serial Debug Messages to Enabled.
4. Apply the settings to the FSP binary file.

Note: Enabling the Serial Debug Messages does not automatically provide debug messages during FSP API execution. This setting only allows the debug messages to be routed to the serial port. The Debug version of the FSP binary is needed for detailed debug information. Enabling debug messages in Coreboot requires using `make menuconfig` and selecting appropriate debug messages under Debugging.

7.2.3.2 How to Enable the Rank Margining Tool (RMT)

The BCT tool can be used to modify the FSP binary to enable the Rank Margining Tool for Memory Testing.

1. Start the BCT and open the RangeleyFsp.bsf file that was included with the FSP kit.
2. In the North Complex section:
   a. Set Enable Rank Margin Tool to Enabled.
   b. Set RMT CPGC exp_loop_cnt to 13.
   c. Set RMT CPGC num_bursts to 13.
3. In the Platform Settings section:
   a. Set Enable Serial Debug Messages to Enabled.
5. Apply the settings to the FSP binary file.