

# **Intel® Atom™ Processor E3800 Product Family Customer Reference Board**

**Platform Guide**

---

***September 2014***



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm>

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: <http://www.intel.com/products/processor/%5Fnumber/>

Intel, the Intel logo, and Intel Atom are trademarks of Intel Corporation in the U.S. and/or other countries.

\*Other names and brands may be claimed as the property of others.

Copyright © 2014, Intel Corporation. All rights reserved.



## Contents

---

<b>1.0</b>	<b>Introduction</b>	5
1.1	Purpose	5
1.2	Intended Audience	5
1.3	Related Documents	5
1.4	Conventions	5
1.5	Acronyms and Terminology	6
<b>2.0</b>	<b>Intel® Atom™ Processor E3800 Product Family Hardware Platform</b>	7
2.1	Intel® Firmware Support Package	8
<b>3.0</b>	<b>Customer Reference Boards</b>	9
<b>4.0</b>	<b>Example Boot Loader</b>	11
4.1	Example Boot Loader Design	11
4.2	Boot Loader Development Environment	11
4.3	Preparing the Coreboot Build Environment	11
<b>5.0</b>	<b>Building the Example Boot Loader</b>	13
<b>6.0</b>	<b>Updating the Firmware</b>	15
6.1	Programming the Firmware	15
6.1.1	Connecting the SF100 to the Reference Platform	16
6.2	Creating a Firmware Backup	18
6.3	Programming a Complete Firmware Image	18
6.4	Updating Only the Boot Loader	19
6.5	Bootting the Example Boot Loader	20
<b>7.0</b>	<b>Creating Custom Images</b>	21
7.1	Edit Coreboot Image Specifications	21
7.2	Binary Configuration Tool (BCT)	21

## Figures

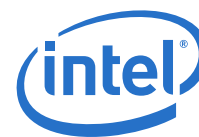
---

1	Atom™ Processor E3800 Product Family Platform Block Diagram	8
2	Customer Reference Platform Block Diagram	9
3	Micro-USB Jack Location	10
4	SF100 Programmer	15
5	DediProg Cable	16
6	Programming Header Locations	17
7	Programming Header Location Close-up	17
8	Location of Power and Reset Buttons	20

## Tables

---

1	Acronyms and Terminology	6
2	Sequence Details	19



## Revision History

---

Date	Revision	Description
September 2014	002	Technical collateral updates.
July 2014	001	Initial release.



## 1.0 Introduction

### 1.1 Purpose

The purpose of this document is to provide information about the Intel® Atom™ Processor E3800 Product Family Customer Reference Boards, code-named Bayley Bay and Bakersport (hereafter referred to as “the CRBs”), with guidance for building an example boot loader for these CRBs that is based on the Intel® Firmware Support Package (FSP). Relevant differences between the two CRBs is noted where appropriate in this document.

### 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. This group includes, but is not limited to, system BIOS developers, boot loader developers, and system integrators.

### 1.3 Related Documents

- *Intel® Firmware Support Package: Introduction Guide* – available at <http://www.intel.com/fsp>
- *Intel® Firmware Support Package for Atom Processor E3800 Product Family Integration Guide* – included in the corresponding FSP kit – available at <http://www.intel.com/fsp>
- *Binary Configuration Tool for Intel® Firmware Support Package* – available at <http://www.intel.com/fsp>

### 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 1. Acronyms and Terminology**

Acronym	Description
AMT	Advanced Management Technology
AVT	Advanced Vector Extensions
BCT	Binary Configuration Tool
BSP	Boot Strap Processor
BWG	BIOS Writer's Guide
CI	Current Image
CRB	Customer Reference Board
DMI	Direct Media Interface
FDI	Flexible Display Interface
FSP	Firmware Support Package
FSP API	Firmware Support Package Interface
FWG	Firmware Writer's Guide
KGI	Known Good Image
ME	Management Engine
PCD	Platform Configuration Database
PCH	Platform Controller Hub
SMI	System Management Interrupt
SMM	System Management Mode
SMRAM	System Management RAM
SPI	Serial Peripheral Interface
TSEG	Top Segment, a reserved segment of memory at the top of its address space to be used as SMRAM





## 2.0 Intel® Atom™ Processor E3800 Product Family Hardware Platform

The Intel® Atom™ Processor E3800 Customer Reference Boards are based on the Intel® Atom™ Processor E3800 Product Family hardware platform, code-named Bay Trail (hereinafter referred to as “the hardware platform”).

This hardware platform consists of a system on a chip (SoC), which is based on one to four third-generation Intel® Atom™ processor cores. The hardware platform includes the Gen7 Intel® graphics architecture and is built on 22-nanometer process technology.

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

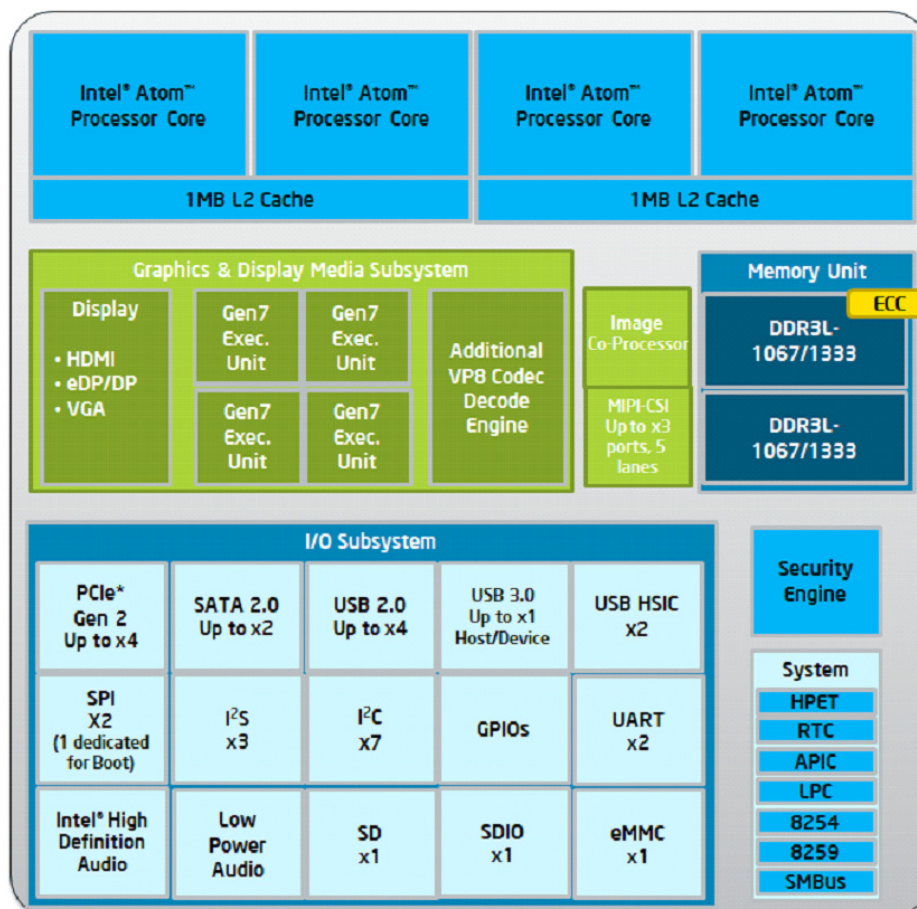
- Increased I/O integration
- Wide range of optional I/O flexibility
- Rugged and reliable
- Greater media competence
- Superior process technology
- Leading performance
- Enhanced graphics engine

*Note:* The hardware platform was previously code named Valleyview SoC, prior to being code-named Bay Trail.

Figure 1 illustrates the hardware platform’s major components.



**Figure 1. Atom™ Processor E3800 Product Family Platform Block Diagram**



## 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.

### §

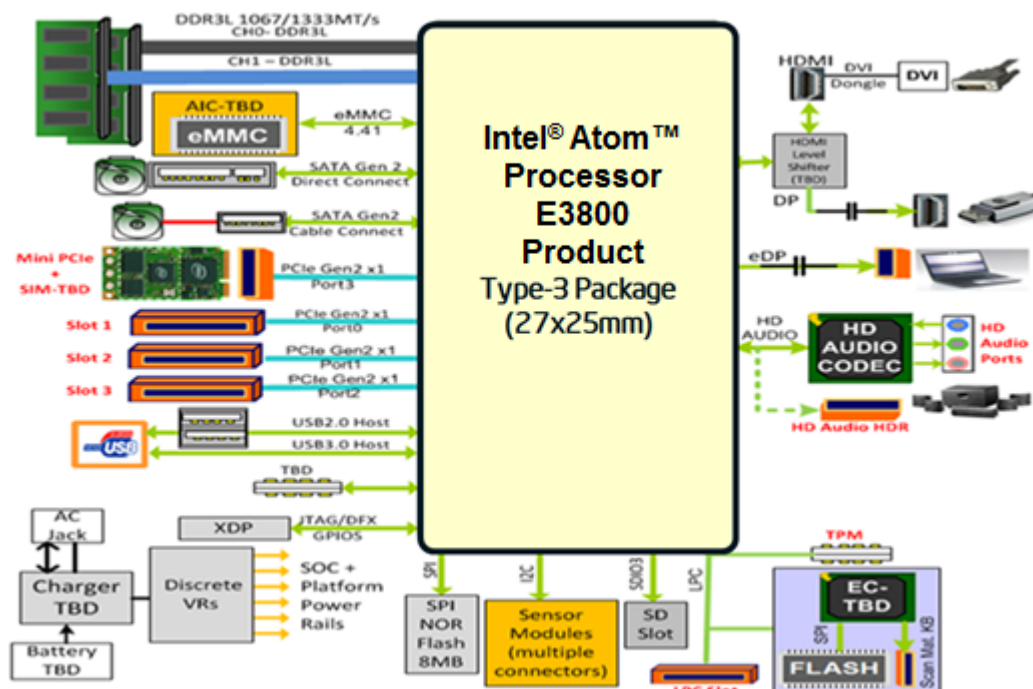




### 3.0 Customer Reference Boards

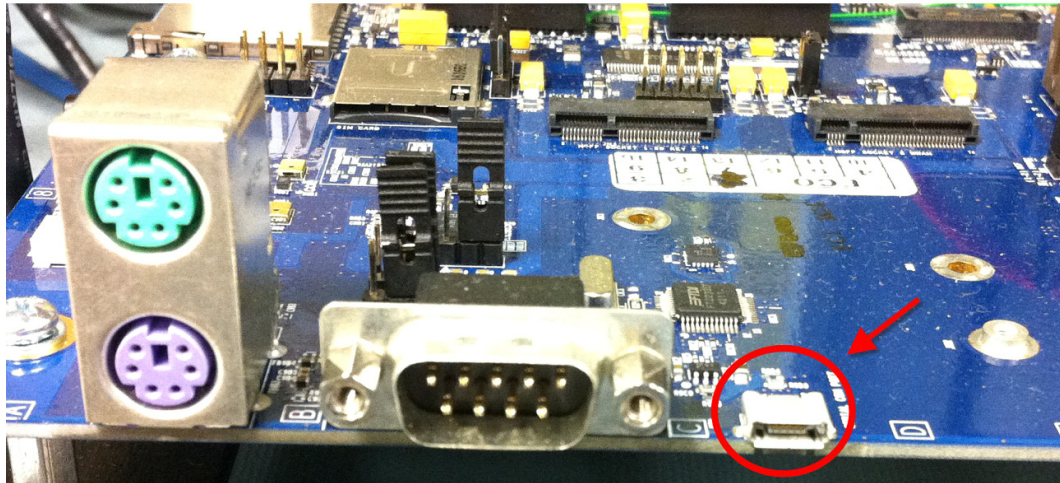
The CRBs are based on the hardware platform, as illustrated in Figure 2.

**Figure 2. Customer Reference Platform Block Diagram**



**Note:** The Bakersport CRB has a similar design and board layout to the Bayley Bay CRB except that it uses a single ECC DIMM instead of two non-ECC DIMMs.

The CRBs provide both serial console and graphics output capabilities. If you are developing a boot loader for which all user interaction is performed through the serial console, use the micro-USB jack located on the left edge of the board, just to the right of the DB-9 connector, as shown in Figure 3.

**Figure 3. Micro-USB Jack Location**

Connect a standard USB to micro-USB cable between a development host system and the board. Once connected, the host system should detect two USB-to-serial adapters on the reference platform. Compatible drivers should install automatically on host systems that are running Microsoft Windows\*. Other host operating systems may require downloading and installing appropriate drivers. The serial console is provided through the first of these two USB-to-serial adapters.

The default serial communication parameters for the serial console are:

- 115,200 baud
- 8-N-1 bit configuration
- No flow control

## §



## 4.0 Example Boot Loader

Intel® provides and supports the Intel® Atom™ Processor E3800 Product Family Firmware Support Package (hereafter referred to as the “FSP kit”) for the CRBs. However, Intel does not provide or support a complete boot loader solution for these CRBs.

The embedded firmware ecosystem has developed an example boot loader solution for the CRBs that uses the FSP. This solution is based on the open source Coreboot project at [coreboot.org](http://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 CRBs are 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 CRB.

### 4.2 Boot Loader Development Environment

Although the FSP itself 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 toolchain required by the Coreboot project. In order to build the Coreboot-specific toolchain, your development host must have its own distribution-provided GCC tool chain. Consult the documentation at [coreboot.org](http://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 CRBs, we will refer to this as the `BB` directory.
2. Navigate into your new `BB` 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` directory.
- 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](http://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 CRBs, 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 CRB and its build options.
- Run **make**.

Follow these steps:

1. Go to [www.intel.com/fsp](http://www.intel.com/fsp) and download the Intel® Atom™ Processor E3800 Product Family FSP kit, which is distributed both as a Microsoft Windows® executable file, `BAY_TRAIL_FSP_KIT_GOLD3.exe`, and as a Linux archive, `BAY_TRAIL_FSP_KIT_GOLD3.tgz`. You can use either version of the FSP kit to build the example boot loader.
2. Install the kit:
  - For Windows: Execute the `BAY_TRAIL_FSP_KIT_GOLD3.exe` file and follow the on-screen dialogs to install the Bay Trail kit. The default installation directory is `C:\BAY_TRAIL_FSP_KIT`.
  - For Linux: Extract the contents of the `BAY_TRAIL_FSP_KIT_GOLD3.tgz` file and follow the instructions in the `Readme_Extract.txt` file. The FSP kit extracts into a subdirectory named `BAY_TRAIL_FSP_KIT`.
3. Navigate to the `BB` directory created in the previous chapter. Create a new subdirectory named `intel` parallel to the `coreboot` directory.
4. Copy files from the `BAY_TRAIL_FSP_KIT` directory where the FSP kit was installed to the `BB/intel` directory on your development host as follows, creating the path to the specified target directories as required:
  - a. Copy `BAY_TRAIL_FSP_KIT/FSP/*.fd` to `BB/intel/fsp/baytrail` and rename the file to `BAYTRAIL_FSP.fd`.

**Note:** If you are using the Bakersport CRB, the FSP binary file must be modified with the Binary Configuration Tool (BCT) to enable ECC RAM support. Name the resulting modified FSB binary file `BAYTRAIL_FSP_ECC.fd`.

- b. Copy `BAY_TRAIL_FSP_KIT/FSP/include/*.h` to `BB/intel/fsp/baytrail/include`.
  - c. Copy `BAY_TRAIL_FSP_KIT/FSP/src/*.c` to `BB/intel/fsp/baytrail/src`.
  - d. Copy `BAY_TRAIL_FSP_KIT/Microcode/*.h` to `BB/intel/cpu/baytrail/microcode`.
  - e. Copy `BAY_TRAIL_FSP_KIT/Graphics/INTEL_EMGD.VBIOS_GOLD_VERSION_36_2_3_3698/vga.dat` to `BB/intel/cpu/baytrail/vbios`.
5. Navigate to the `BB/coreboot` directory and run:

```
make menuconfig
```

6. Open the **mainboard** menu. Select **Intel** as the Mainboard vendor. Then select either **Bayley Bay FSP-based CRB** or **Bakersport FSP-based CRB** as the Mainboard model, depending on which CRB you are using.



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, and select the **Payload** option. In the list of options, select **SeaBIOS version (1.7.4)**, and select **master**. This selects the latest version of the SeaBIOS payload that includes xHCI support.
9. 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**.
10. Select **Exit** twice to return to the command prompt. Save the configuration when prompted to do so.
11. Now, build the boot loader with a single command:

```
make
```

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

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

13. 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 CRBs are both equipped with a single 8 MB flash device that contains all of the system firmware, including the Firmware Descriptor, Security Firmware, and the BIOS or Boot Loader. By default, these CRBs come from Intel® with a standard UEFI BIOS installed.

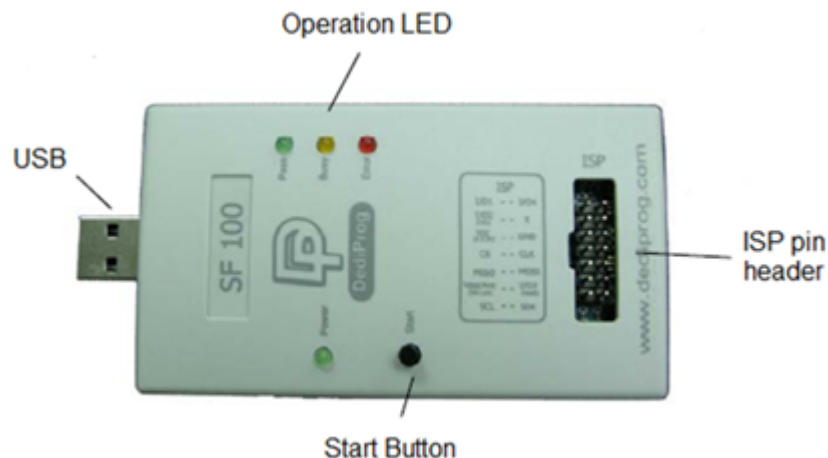
To replace the factory BIOS with boot loader firmware, you can generate a complete 8 MB firmware image and update the whole firmware flash device, or you can update only the BIOS region of the firmware image with the new boot loader. For preproduction or development purposes, updating only the boot loader 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.

**Note:** **Important!** The non-BIOS portions of the factory-installed firmware flash image may be configured in a way that is not compatible with the boot loader that you built in [Section 5.0](#). If you are not generating a full 8 MB firmware image, then prior to programming your boot loader image to the CRB, you must first program the 8 MB SPI.bin file (provided in the Bay Trail FSP kit) to your board, following the procedure in [Section 6.3](#). This ensures that you are using firmware components that are compatible with the example boot loader. The SPI.bin file is located in the BayleyBay subdirectory of the Bay Trail FSP kit installation. Note that you only need to do this one time.

### 6.1 Programming the Firmware

The flash devices on the CRBs can be programmed with new firmware using a DediProg\* SF100 programmer, which 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>





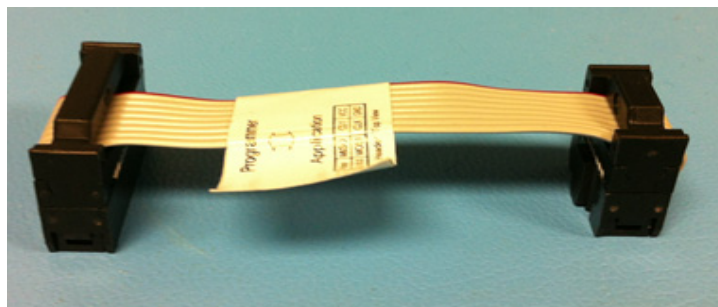
The SF100 programmer is also supported by the Linux\* *flashrom* utility. The flashrom utility is not supported by DediProg or by Intel. Additional technical information about the flashrom utility can be obtained from the flashrom website at: <http://www.flashrom.org/>

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 Customer Reference Board

The SF100 includes a cable (shown in Figure 5) that connects between the ISP pin header and the 8-pin header labeled as J7A2 on the Bayley Bay CRB or J3A2 on the Bakersport CRB (shown in Figure 6 and Figure 7).

**Figure 5. DediProg Cable**



**Note:** The header is located in the same location on both the Bayley Bay and Bakersport CRBs. Only the label for the header is different.

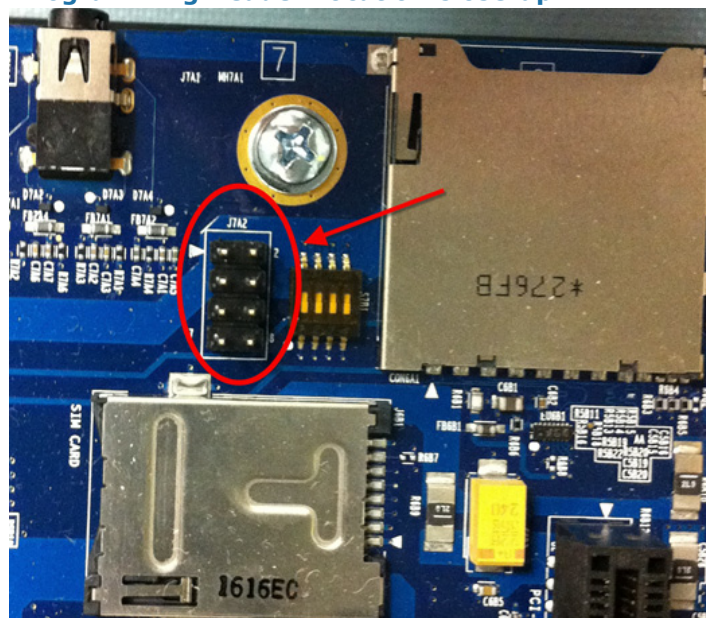
**Caution:** Do not connect the SF100 to the 8-pin header labeled J8E2 on the Bayley Bay CRB or J2E2 on the Bakersport CRB, which is noted as the PROGRAMMING PORT on the board and is shown crossed off in Figure 6. This is not the correct connector for programming the firmware.

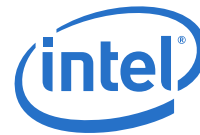


**Figure 6. Programming Header Locations**



**Figure 7. Programming Header Location Close-up**





## 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. Connect the cable from the SF100 programmer to header J7A2 on the Bayley Bay CRB or header J3A2 on the Bakersport CRB.
3. Open the DediProg Engineering application.
4. Next to the **Currently working on** section near the top of the window, make sure that **Application Memory Chip 1** is selected.
5. Click the **Edit** button at the top of the window.
6. Click **Read** at the top of the window. Wait for the completion of the read operation.
7. Click **Chip Buffer to File** to save the 8 MB firmware image to a file.
8. Remove the cable from header J7A2 on the Bayley Bay CRB or header J3A2 on the Bakersport 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.

You must also use this procedure to program the Bay Trail FSP kit's `SPI.bin` file to the CRB prior to updating the BIOS region of the firmware image with a new boot loader. Note that the `SPI.bin` file only needs to be programmed to the CRB one time.

1. Disconnect the power from the CRB. **Do not apply power to the CRB at any time during this procedure.**
2. Connect the cable from the SF100 programmer to header J7A2 on the Bayley Bay CRB or header J3A2 on the Bakersport CRB.
3. Open the DediProg Engineering application.
4. Next to the **Currently working on** section near the top of the window, make sure that **Application Memory Chip 1** is selected.
5. Click the **Config** button at the top of the window.
6. In the Advanced Settings dialog, click the **Prog** button on the left.
7. Select **Program a whole file starting from address 0 of a chip.**
8. Click the **Flash Options** button on the left.
9. Select the check box **Unprotect block automatically when block(s) protected.**
10. Click **OK**.
11. Click the **File** button at the top of the window. This opens a file-open dialog for selecting the complete 8 MB firmware file to be programmed to the flash device.
12. Click the **Erase** button at the top of the window to erase the entire flash memory device. Wait for completion of the erase operation.
13. 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.
14. 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.



15. Remove the cable from header J7A2 on the Bayley Bay CRB or header J3A2 on the Bakersport 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 the power from the CRB. **Do not apply power to the CRB at any time during this procedure.**
2. Connect the cable from the SF100 programmer to header J7A2 on the Bayley Bay CRB or header J3A2 on the Bakersport CRB.
3. Open the DediProg Engineering application.
4. Next to the **Currently working on** section near the top of the window, make sure that **Application Memory Chip 1** is selected.
5. 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 was built in section 5.0.
6. Click the **Config** button at the top of the window.
7. In the Advanced Settings dialog that appears, click the **Batch** button on the left.
8. Under **Batch Operation Options**, select **Update memory according to Region configuration**, select **Region 1**, and enter **600000** to **7FFFFF**.
9. Select the **Identify Chip** and **Require Verification after completion** check boxes. Make sure all other check boxes are cleared (unchecked).
10. Verify that the Sequences Details at the bottom of the window are as shown in Table 2.

**Table 2. Sequence Details**

Steps	Actions
1	Identify before operation starts.
2	Read from the chip.
3	Blank check.
4	Erase chip (if not blank).
5	Program chip.
6	Verify after operation completes.

11. Click the **Flash Options** button on the left.
12. Select the check box **Unprotect block automatically when block(s) protected**.
13. Click **OK** to close the Advanced Settings dialog.
14. Click the **Batch** button at the top of the window. Wait for completion of the batch operation.
15. Remove the cable from header J7A2 on the Bayley Bay CRB or header J3A2 on the Bakersport CRB before applying power to the CRB.

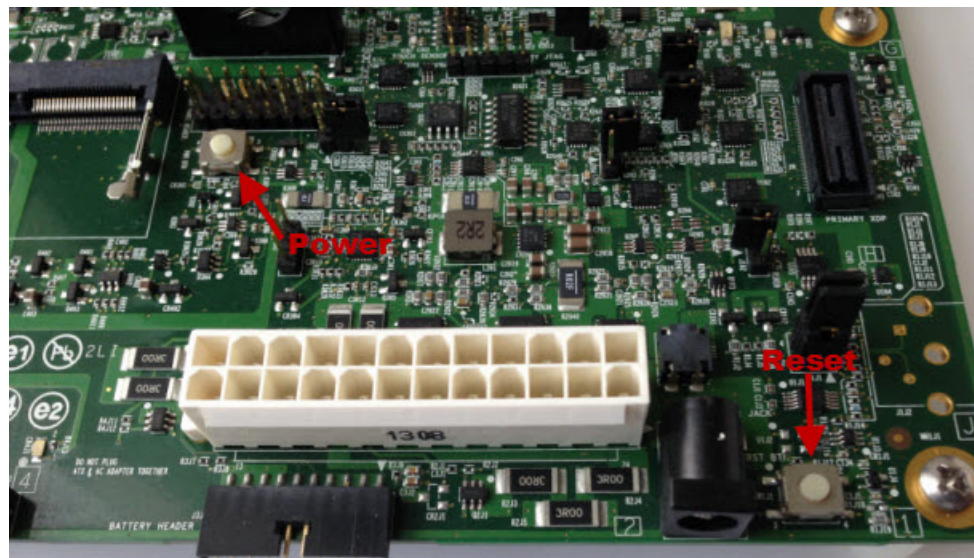


## 6.5 Booting the Example Boot Loader

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 development host to the micro-USB connector, as discussed in section 3.0. The CRBs provide **Power** and **Reset** buttons, as shown in Figure 8. To power up the board, turn on the power supply, and then press the board's **Power** button.

**Figure 8. Location of Power and Reset Buttons**



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 micro-USB port.

If a storage device with a bootable OS image is connected to one of the CRB'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 either the USB ports or the PS/2 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 Bash 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 [Section 6.4](#), except in step 8 specify a region of 400000 to 7FFFFFFF.

### 7.2 Binary Configuration Tool (BCT)

Intel provides the Binary Configuration Tool (BCT) that lets you edit the FSP binary file delivered with the Bay Trail 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 0xFFFC0000, 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.

