3rd Generation Intel® Core™ Processor with Mobile Intel® HM76/QM77 Express Chipset Customer Reference Board

Platform Guide

September 2014

Document Number: 331208-001

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, Intel Core, Intel Celeron, and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

Contents

Figures

Revision History

Date	Revision	Description
September 2014	001	Initial release.

§

1 Introduction

1.1 Purpose

The purpose of this document is to provide information about the 3rd Generation Intel® Core™ Processor with Mobile Intel® HM76/QM77 Express Chipset Customer Reference Board (CRB) code named Cougar Canyon 2 (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. 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
- 3rd Generation Intel® Core™ Processor with Mobile Intel® HM76/QM77 Express Chipset 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

AMT	Advanced Management Technology
AVT	Advanced Vector Extensions
BCT	Binary Configuration Tool
BSP	Boot Strap Processor
BWG	BIOS Writer's Guide
CRB	Customer Reference Board
DMI	Direct Media Interface
FDI	Flexible Display Interface
FSP	Firmware Support Package
FSP API	Firmware Support Package Application Programming Interface
ME	Management Engine
FWG	Firmware Writer's Guide
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 Intel® 3rd Generation Intel® Core™ Processor with Mobile Intel® HM76/QM77 Express Chipset Hardware Platform

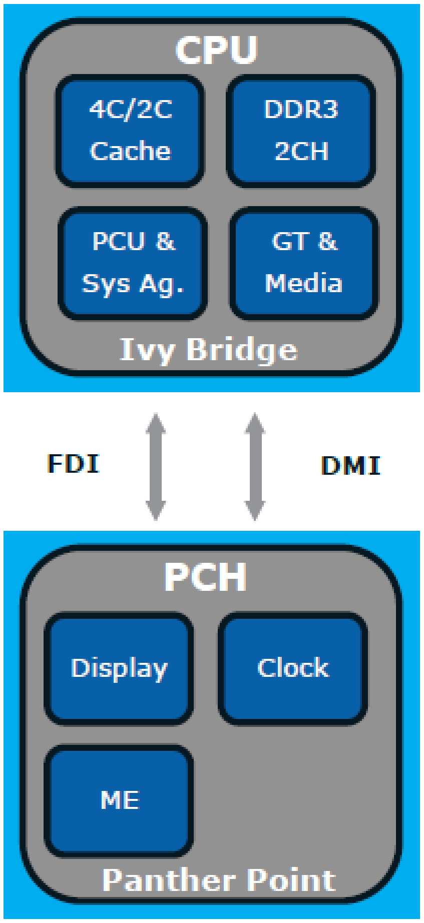
The CRB is based on the Intel® 3rd Generation Intel® Core™ Processor with Mobile Intel® HM76/QM77 Express Chipset Hardware platform, code named Chief River (hereafter referred to as "the hardware platform"). The hardware platform is built on 22 nm process technology.

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

- Integrated graphics
- Low power consumption
- PCIe* Gen 3.0 Graphics (PEG) port
- Flexible display interface
- Full integration of CPU, GPU, and memory controller
- Proactive security and manageability with Intel® Advanced Management Technology (Intel® AMT) 8.0
- Intel® Advanced Vector Extensions (Intel® AVX v1.0)

Figure 1 illustrates the hardware platform’s major components.

Figure 1. Hardware Platform



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.

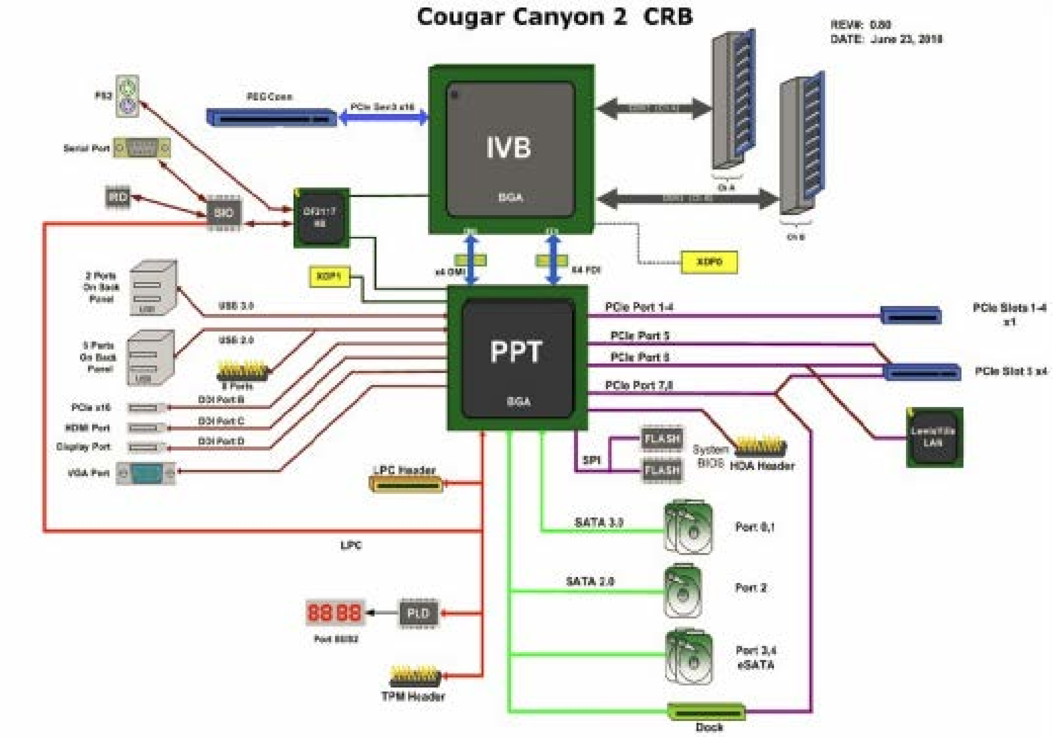
The FSP for the hardware platform handles the initialization of the processor, memory, and the Platform Controller Hub (PCH) for hardware designs based on this hardware platform.

§

3 Customer Reference Board

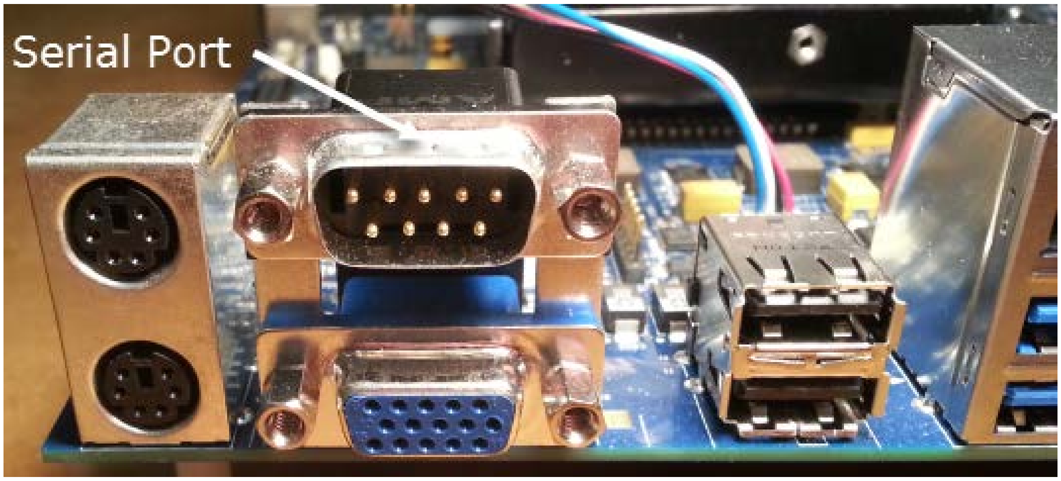
The CRB is based on the hardware platform as illustrated in Figure 2.

Figure 2. Customer Reference Board Block Diagram



Note: The CRB provides both a 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 DB-9 serial port located on the back edge of the board, directly above the VGA port, as illustrated in Figure 3.

Figure 3. Serial Port Location



Connect a null modem cable between a development host system and the board.

The default serial communication parameters for the serial console are as follows:

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

§

4 Example Boot Loader

Intel provides and supports the 3rd Generation Intel® Core™ Processors with Mobile Intel® HM76/QM77 Express Chipsets 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 CRB.

The embedded firmware ecosystem has developed an example boot loader solution for the CRB that uses the Intel FSP. 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 the way to integrate the Intel FSP into a complete boot loader solution.

Note: The steps to generate the example boot loader for the CRB 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 PCH. 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* 14 and Fedora* 16 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 environment, 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 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 CC2 directory.
2. Navigate into your new CC2 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, to 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 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 Building the Example Boot Loader

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

- Download, install, and unzip 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 select the CRB and its build options.
- Run **make**.

Follow these steps:

1. Go to www.intel.com/fsp and download the FSP kit, which is distributed both as a Microsoft Windows® executable file, `CHIEF_RIVER_FSP_KIT_GOLD2.exe`, and as a Linux archive, `CHIEF_RIVER_FSP_KIT_GOLD2.tgz`. You can use either version of the FSP kit to build the example boot loader.
2. Install the kit:
   - For Windows: Execute the `CHIEF_RIVER_FSP_KIT_GOLD2.exe` file and follow the on-screen dialogs to install the Chief River FSP kit. The default installation directory is `C:\CHIEF_RIVER_FSP_KIT`.
   - For Linux: Extract the contents of the `CHIEF_RIVER_FSP_KIT_GOLD2.tgz` file and follow the instructions in the `Readme_Extract.txt` file.

     The FSP kit will extract into a `CHIEF_RIVER_FSP_KIT` subfolder.
3. Navigate to the CC2 directory on your development host that you created in the previous chapter. Create a new subdirectory named `intel` parallel to the `coreboot` directory.
4. Copy files from the `CHIEF_RIVER_FSP_KIT` subdirectory where the FSP kit was installed to the `CC2/intel` directory on your development host as follows, creating the path to the specified target directories as required:

   a. Copy `CHIEF_RIVER_FSP_KIT/FSP/*.fd` to `CC2/intel/fsp/ivybridge_bd82x6x` and rename the file to `FvFsp.bin`.

   b. Copy `CHIEF_RIVER_FSP_KIT/FSP/include/*.h` to `CC2/intel/fsp/ivybridge_bd82x6x/include`.

   c. Copy `CHIEF_RIVER_FSP_KIT/FSP/src/fsphob.c` to `CC2/intel/fsp/ivybridge_bd82x6x/src`.

   d. Copy `CHIEF_RIVER_FSP_KIT/Microcode/*.h` to `CC2/intel/cpu/ivybridge/microcode`.

   e. Copy `CHIEF_RIVER_FSP_KIT/CougarCanyon2/snm_2170.dat` to `CC2/intel/mainboard/intel/cougar_canyon2/vbios`.

5. Navigate to the `CC2/coreboot` directory and run

   make menuconfig

6. Open the mainboard menu. Select **Intel** as the Mainboard vendor and **Cougar Canyon 2 CRB** as the Mainboard model.
7. Use **Exit** at the bottom of the screen to back out to the main menu, then select the **Chipset** menu.
8. Move to the **Use Intel Firmware Support Package** option and press the space bar to select it. Leave all other settings in their default state.
9. Back out to the main menu, then select the **Console** menu. 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 one 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

    The default `coreboot.rom` file is 8 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 Updating the Firmware

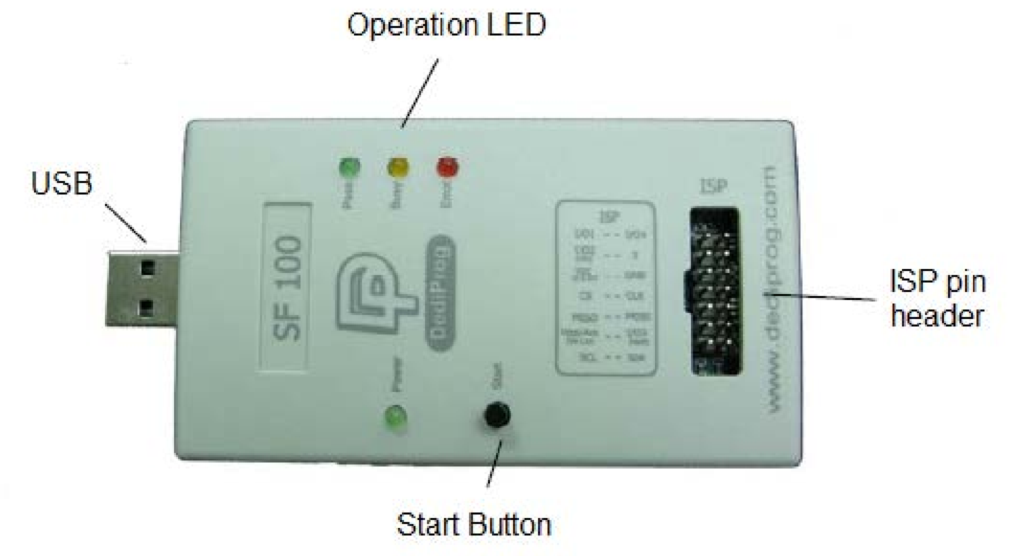
By default, the CRB comes from Intel with a standard BIOS installed.

The CRB is equipped with two 8 MB Serial Peripheral Interface (SPI) flash memory devices designated SPI-0 and SPI-1 that together contain the system firmware. The firmware descriptor, management engine (ME) firmware, and Gigabit Ethernet firmware are all contained in SPI-0, while the BIOS or boot loader is contained in SPI-1.

6.1 Programming the Firmware

The flash devices on the CRB 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; it connects to the flash device to be programmed through 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. Note, however, that 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 use of the SF100 programmer assume that you have the DediProg* SF100 drivers and software installed on a PC running Microsoft Windows.

6.1.1 Connecting the SF100 to the Reference Platform

The SF100 includes a cable (shown in Figure 5) that connects between the ISP pin header and the 8-pin header labeled J2E1 on the CRB (shown in Figure 6).

Figure 5. DediProg* Cable

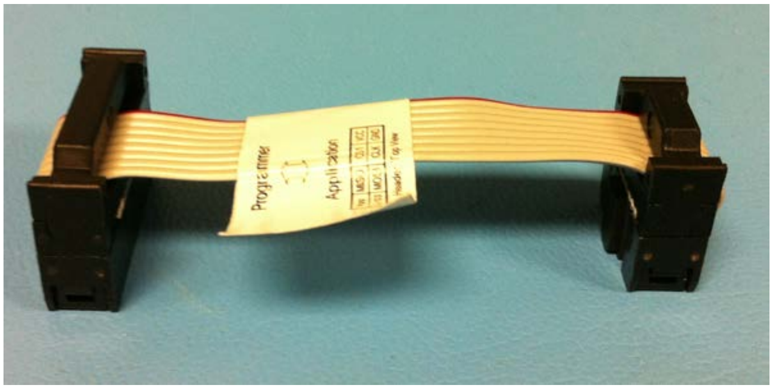


Figure 6. CRB Headers and Jumpers

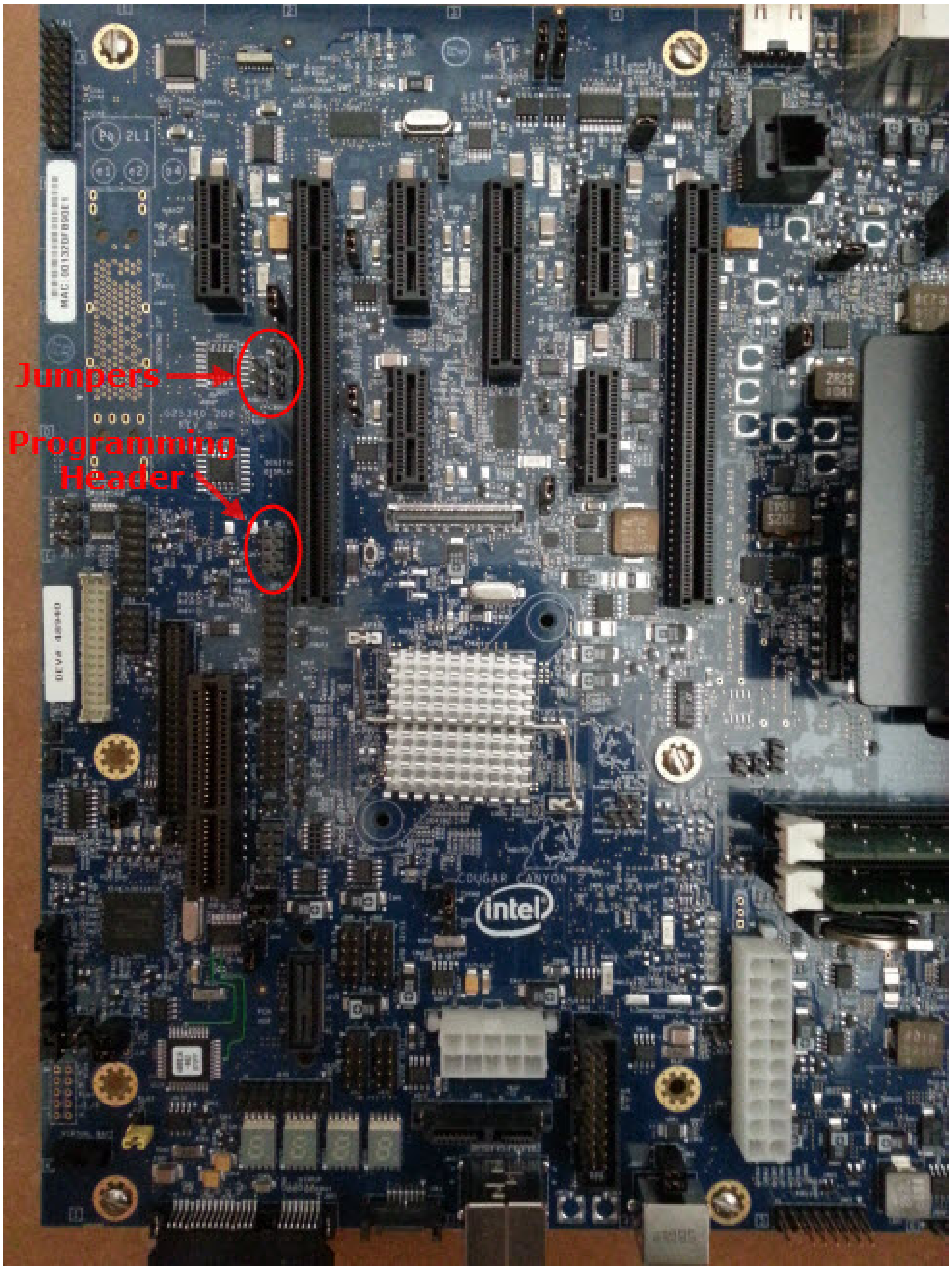


Figure 7 shows a close-up of the J2E1 header location.

Figure 7. J2E1 Header Location

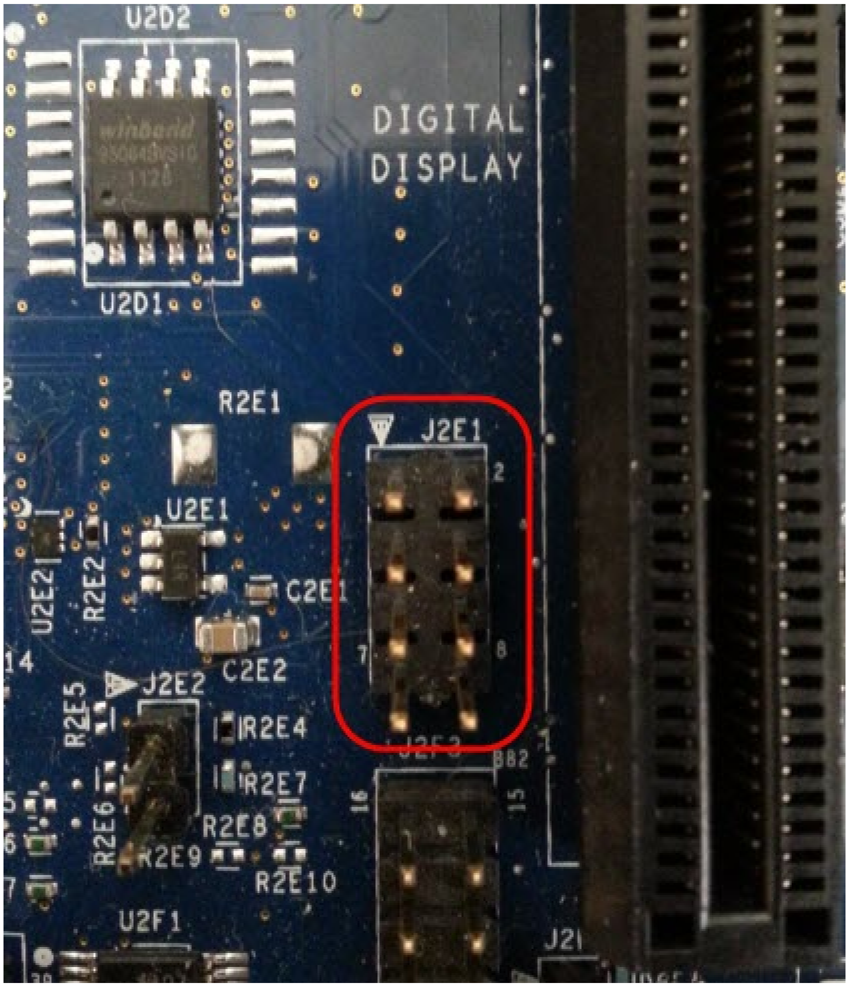
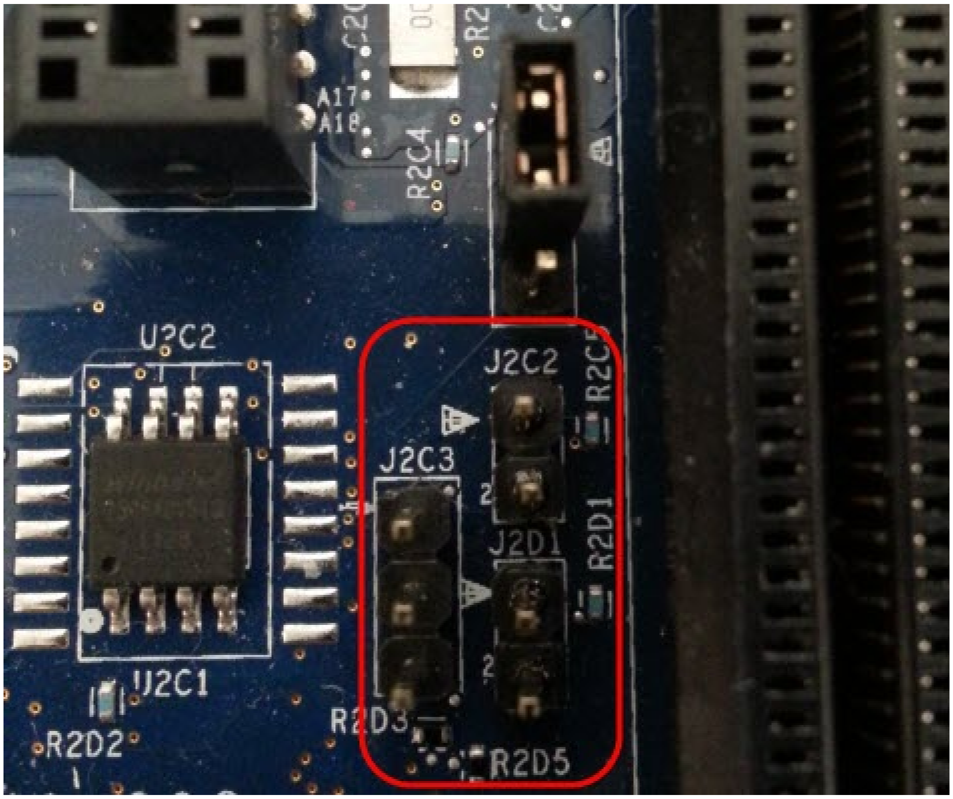


Figure 8 shows a close-up of the jumpers you must change while backing up and programming the SPI-0 and SPI-1 chips.

Figure 8. Jumper Locations



6.2 Creating a Firmware Backup

It is important to back up any existing working firmware so that you can always restore the system back to a known working condition. The CRB ships with a BIOS installed.

Use the following procedure to back up the as-shipped 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 J2E1 on the CRB.
3. Make sure that jumpers are installed at locations J2C2 and J2D1.
4. Set the jumper at J2C3 to connect pins 1-2. This step selects SPI-0.
5. Open the DediProg Engineering application.
6. Select **W25Q64BV** when prompted for the memory device type.
7. In the DediProg* Engineering application, next to the **Currently working on** section near the top of the window, be sure to select **Application Memory Chip 1**.
8. Click the **Edit** button at the top of the window.

9. Click **Read** at the top of the window. Wait for the completion of the read operation.
10. Click **Chip Buffer to File** to save the 8 MB firmware image to a file.
11. Close the **View Contents in the Memory Chip** window.
12. Set the jumper at J2C3 to connect pins 2-3. This step selects SPI-1.
13. Repeat steps 8–11 to save the SPI-1 firmware image to a file.
14. Remove the cable from header J2E1 before applying power to the CRB.

Note: Contrary to other documentation, you do not need to remove the jumpers from locations J2C2, J2D1, or J2C3 before powering on the CRB.

6.3 Replacing the SPI-0 Contents

Boot loaders that are based on the FSP are programmed to the SPI-1 chip. However, for the boot loader to function on the CRB, the SPI-0 chip must contain a firmware descriptor, the management engine (ME) firmware, and the Gigabit Ethernet firmware. An 8 MB file containing the contents for SPI-0 is included in the FSP kit, with filename `Rom00_8M_MB_PPT.bin`. The firmware descriptor in this file is configured so that all 8 MB of SPI-1 on the CRB can be used for the boot loader firmware image.

You must load this `Rom00_8M_MB_PPT.bin` file into SPI-0 one time, after which SPI-0 does not need to be changed. The following steps must be performed only once:

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 J2E1 on the CRB.
3. Make sure that jumpers are installed at locations J2C2 and J2D1.
4. Set the jumper at J2C3 to connect pins 1-2. This step selects SPI-0.
5. Open the DediProg Engineering application.
6. Select **W25Q64BV** when prompted for the memory device type.
7. Next to the **Currently working on** section near the top of the window, make sure that **Application Memory Chip 1** is selected.
8. Click the **Config** button at the top of the window.
9. In the Advanced Settings dialog that appears, click the **Prog** button on the left.
10. Select **Program a whole file starting from address 0 of a chip**.
11. Click the **Flash Options** button on the left.
12. Select the check box **Unprotect block automatically when block(s) protected.**
13. Click **OK**.
14. Click the **File** button at the top of the window. This step opens a File Open dialog for selecting the firmware image to be programmed to the board. Select the 8 MB image file named `Rom00_8M_MB_PPT.bin` delivered with the FSP kit in the `CHIEF_RIVER_FSP_KIT/CougarCanyon2` directory. Click **OK**.
15. Click the **Erase** button at the top of the window to erase the entire flash memory device. Wait for completion of the erase operation.
16. Click the **Prog** button at the top of the window to program the Intel-provided firmware image to the flash memory device on the target platform. Wait for completion of the programming operation.

17. 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.
18. Remove the cable from header J2E2 before applying power to the CRB.
19. Go on to program the boot loader into chip SPI-1 before applying power to the CRB.

6.4 Programming a Boot Loader Firmware Image in SPI-1

Use the following procedure to program an 8-MB boot loader firmware image to the CRB. If your boot loader is less than 8 MB in size, see Section 6.4.1 instead.

1. Disconnect the power from the CRB. **Do not apply power to the CRB at any time during this procedure.**
2. Transfer the `coreboot.rom` file you built in in section 5 from your Linux development system to the Windows* system running the DediProg* software.
3. Connect the cable from the SF100 programmer to the header J2E1 on the CRB.
4. Make sure that jumpers are installed at locations J2C2 and J2D1.
5. Set the jumper at J2C3 to connect pins 2-3. This step selects SPI-1.
6. Open the DediProg* Engineering application.
7. Select **W25Q64BV** when prompted for the memory device type.
8. Next to the **Currently working on** section near the top of the window, make sure that **Application Memory Chip 1** is selected.
9. Click the **Config** button at the top of the window.
10. In the Advanced Settings dialog that appears, 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 step opens a File Open dialog box for selecting the file containing the firmware image to be programmed to the target platform. Select the `coreboot.rom` file that you transferred to this PC in step 2, then click **OK**.
16. Click the **Erase** button at the top of the window to erase the entire 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. Remove the cable from header J2E2 before applying power to the CRB.

Note: Contrary to other documentation, you do not need to remove the jumpers from locations J2C2, J2D1, or J2C3 before powering on the CRB.

6.4.1 If Boot Loader Image Is Less Than 8 MB

If your boot loader image is less than 8 MB (such as 4 MB or 1 MB), the boot loader image be programmed to the end of the SPI-1 chip. For example, a 4 MB image must be programmed to the last 4 MB of the 8 MB chip, and a 1 MB image must be programmed to the last MB of the 8 MB chip.

Follow the exact procedure in Section 6.4 above, except the following:

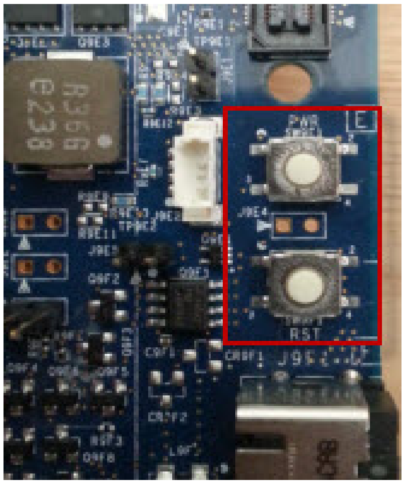
Instead of step 11, select **Program from specific address of a chip**, and enter the appropriate starting address to match your firmware image. For example, if your firmware image is 4 MB in size, specify a starting address of **0X0400000**. If your firmware image is 1 MB in size, specify a starting address of **0x0700000**.

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 terminal emulator to the DB-9 serial port. The CRB provides **Power** and **Reset** buttons, as shown in Figure 9. To power up the board, turn on the power supply, and then press the board's **Power** button.

Figure 9. Power (Top) 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 serial 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 VGA monitor, as determined by the configuration of the OS image. If the OS image provides a graphical user interface on the VGA 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 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 an 8-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, to rebuild the `coreboot.rom` image file, type:

   make

7.2 Binary Configuration Tool

Intel provides the Binary Configuration Tool (BCT) that lets you edit the FSP binary file delivered with the 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 kit 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.

§