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 ............................................................................................ 6

2.0 Intel® Celeron® Processor N3000 Product Family .......................................................... 8
  2.1 Hardware Platform .......................................................................................................... 8
  2.2 Intel® Firmware Support Package ................................................................................... 9

3.0 Customer Reference Boards .......................................................................................... 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 ................................................................... 12

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 Customer Reference Board ...................................... 17
     6.1.2 How to Flash the Firmware .................................................................................... 18
  6.2 Creating a Firmware Backup .......................................................................................... 19
  6.3 Programming a Complete Firmware Image .................................................................... 19
  6.4 Updating Only the Boot Loader ...................................................................................... 20
  6.5 Booting the Example Boot Loader ................................................................................ 21

7.0 Creating Custom Images .................................................................................................... 23
  7.1 Intel® Binary Configuration Tool (BCT) ...................................................................... 23

Figures

Figure 1. Intel® Celeron® Processor N3000 Product Family Platform Block Diagram .......... 9
Figure 2. Customer Reference Platform Block Diagram ....................................................... 10
Figure 3. Micro-USB Jack Location ....................................................................................... 11
Figure 4. SF100 Programmer ............................................................................................... 17
Tables

Table 1.  Acronyms and Terminology .................................................................6
Table 2.  Sequence Details ..............................................................................21
# Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>April 2016</td>
<td>1.0</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>

§
1.0 Introduction

1.1 Purpose

The purpose of this document is to provide information about Intel® Celeron® Processor N3000 Product Family Customer Reference Board, code-named Cherry Hill CRB (hereafter referred to as “the CRB”), and guidance for building an example boot loader for the CRB with Intel® Firmware Support Package.

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


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

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMT</td>
<td>Advanced Management Technology</td>
</tr>
<tr>
<td>AVT</td>
<td>Advanced Vector Extensions</td>
</tr>
<tr>
<td>BCT</td>
<td>Binary Configuration Tool</td>
</tr>
<tr>
<td>BSP</td>
<td>Boot Strap Processor</td>
</tr>
<tr>
<td>BWG</td>
<td>BIOS Writer’s Guide</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>CI</td>
<td>Current Image</td>
</tr>
<tr>
<td>CRB</td>
<td>Customer Reference Board</td>
</tr>
<tr>
<td>DMI</td>
<td>Direct Media Interface</td>
</tr>
<tr>
<td>FDI</td>
<td>Flexible Display Interface</td>
</tr>
<tr>
<td>FSP</td>
<td>Firmware Support Package</td>
</tr>
<tr>
<td>FSP API</td>
<td>Firmware Support Package Interface</td>
</tr>
<tr>
<td>FWG</td>
<td>Firmware Writer’s Guide</td>
</tr>
<tr>
<td>KGI</td>
<td>Known Good Image</td>
</tr>
<tr>
<td>ME</td>
<td>Management Engine</td>
</tr>
<tr>
<td>PCD</td>
<td>Platform Configuration Database</td>
</tr>
<tr>
<td>PCH</td>
<td>Platform Controller Hub</td>
</tr>
<tr>
<td>SMI</td>
<td>System Management Interrupt</td>
</tr>
<tr>
<td>SMM</td>
<td>System Management Mode</td>
</tr>
<tr>
<td>SMRAM</td>
<td>System Management RAM</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 SMRAM</td>
</tr>
</tbody>
</table>
2.0  Intel® Celeron® Processor N3000 Product Family

2.1  Hardware Platform

The Intel® Celeron® Processor N3000 Customer Reference Board is based on the Intel® Celeron® Processor N3000 Product Family hardware platform, code-named Braswell (hereinafter referred to as “the hardware platform”).

This hardware platform consists of a system on a chip (SoC), which is based on Intel® Airmont processor cores. The hardware platform includes the Gen8LP Intel® graphics architecture and is built on 14-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

Figure 1 illustrates the hardware platform's major components.
2.2 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 CRB is based on the hardware platform, as illustrated in Table 2.

Figure 2. Customer Reference Platform Block Diagram

![Customer Reference Platform Block Diagram](image)

The CRB provides 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 J3F2 connector, as shown in Figure 3.
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® Celeron® Processor N3000 Product Family Firmware Support Package (hereafter referred to as the “FSP kit”) for the CRB.

This boot loader 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 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 UEFI payload, which is provided by a related open source firmware project. The UEFI payload 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® Core 19 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 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 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 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:

1. In the coreboot directory, navigate to the util/grossgcc directory.

```
  RUN ./buildgcc
```

This command takes the same amount of time to complete as the make version.

2. 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 CRB, you must do the following:

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

Follow these steps:

1. Go to [www.intel.com/fsp](http://www.intel.com/fsp) and download the Intel® Celeron® Processor N3000 Product Family FSP kit, which is distributed both as a Microsoft Windows® executable file, <.exe> and as a Linux archive, <.tgz>. You can use either version of the FSP kit to build the example boot loader.

2. Install the kit:
   - For Windows: Execute the <.exe> file and follow the on-screen dialogs to install the Braswell kit.
   - For Linux: Extract the contents of the <.tgz> file and follow the instructions in the Readme_Extract.txt file. The FSP kit extracts into a subdirectory.

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

4. Copy files from the <Platform 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 <Platform FSP_KIT>/FSP/*.fd to BB/intel/fsp/braswell and rename the file to BSWFSP.fd
   b. Copy <Platform FSP_KIT>/FSP/FspSampleCode/*.h to BB/intel/fsp/braswell/include.
   c. Copy <Platform FSP_KIT>/VBT/*.* to BB/intel/gfx/braswell

5. Copy BB/intel/configs/config.bsw to BB/coreboot then run the following cmd:

   mv config.bsw .config

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

   make
7. If the build completes without errors, the newly created firmware image is generated into the following directory and file:
   `BB/coreboot/build/coreboot.rom`

8. 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 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, the CRB comes 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. To create the full (8MB) SPI image, download and install Intel® Trusted Execution Engine (Intel® TXE) Firmware. Refer to the Intel® TXE firmware kit in VIP and follow the instructions to build the appropriate full image as described in the “Braswell Platform, Intel® TXE Firmware Start-up Software Guide”.

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 Braswell 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 Braswell subdirectory of the Braswell FSP kit installation. Note that you only need to do this one time.

6.1 Programming the Firmware

The flash device on the CRB can be programmed with new firmware using a DediProg* SF100 programmer, which is shown in Figure 4.
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.
6.1.2 How to Flash the Firmware

6.1.2.1 Flash Instructions

Changing Dediprog* voltage to 1.8V is for the firmware flash update. **Do not apply power to the CRB at any time during the flashing process!

*Note:* Enable the QUAD MODE on the SPI part installed on your CRB board by setting the QE bit.

- For MX25U6435F it can be done by writing 0x40 to the status register by Dediprog*.
- In Config > Modify Status Register > Write Status Register(s) > Register1 Value(Hex).

*Note:* This is a one-time change. Once set, it persists in SPI flash part regardless of the FW image burned.
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 J2E1 on the Cherry Hill Fab B CRB, or header J3D1 on the Fab D CRB.
3. Open the DediProg Engineering application. Select MX25U6435F.
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 J2E1 on the Cherry Hill Fab B CRB, or header J3D1 on the Fab D 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 Braswell 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 J2E1 on the Cherry Hill Fab B CRB, or header J3D1 on the Fab D CRB.
3. Open the DediProg Engineering application, Select MX25U6435F.
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 and select the 8MB file that you want to flash. 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 J2E1 on the Cherry Hill Fab B CRB, or header J3D1 on the Fab D 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 J2E1 on the Cherry Hill Fab B CRB, or header J3D1 on the Fab D CRB.
3. Open the DediProg Engineering application, Select MX25U6435F.
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 – select 2MB coreboot.rom (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

<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>

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 J2E1 on the Cherry Hill Fab B CRB, or header J3D1 on the Fab D 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 UEFI Shell by default.

To interact with the boot loader, connect a development host to the micro-USB connector, as discussed in Section 3.0. The CRB provides Power and Reset buttons, as shown in Figure 7. To power up the board, turn on the power supply and then press the board's Power button.
When power is applied to the CRB and the boot loader initializes the board and boots the UEFI 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 a USB port in order to fully interact with the booted operating system.
7.0 Creating Custom Images

7.1 Intel® Binary Configuration Tool (BCT)

FSP allows a number of platform or board-specific settings to be changed during build time or boot loader run time. Each Intel® FSP release provides a platform-specific setting file (.bsf) – a text file that represents the default settings in the FSP binary file.

BCT is used to modify the FSP binary file provided in the FSP kit. For obtaining and further information on BCT, visit http://www.intel.com/fsp.

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

§