

QCVS BOOTROM Tool User Guide



Contents

- Chapter 1 BOOTROM Configuration..... 3**
- 1.1 Introduction..... 3
 - 1.1.1 Power-on reset configuration.....3
 - 1.1.2 Boot ROM data configuration.....4
 - 1.1.3 BootSequencer/BootROM vs PBL.....5
- 1.2 Using BOOTROM configuration tool..... 5
 - 1.2.1 Create a new QorIQ configuration project..... 5
 - 1.2.2 Configure POR configuration settings..... 7
 - 1.2.3 Configure boot ROM data..... 10
- 1.3 Generating code from BOOTROM component..... 12
- 1.4 BOOTROM component output file types..... 16

Chapter 1

BOOTROM Configuration

This document introduces the BOOTROM configuration tool, represented by a BOOTROM embedded component of QorIQ Configuration and Validation Suite (QCVS).

The document also describes how to create a new QorIQ configuration project and configure the BOOTROM component.

NOTE

The BOOTROM component of QCVS is only available for projects created with B, P, or T family of processors.

This chapter contains the following sections:

- [Introduction](#) on page 3
- [Using BOOTROM configuration tool](#) on page 5
- [Generating code from BOOTROM component](#) on page 12
- [BOOTROM component output file types](#) on page 16

1.1 Introduction

The BOOTROM configuration tool helps you in power-on reset (POR) configuration, clocking, and pre-initialization of P1/P2 and G1 devices, using the signal values from a POR configuration.

The tool also helps you initialize memory content to enable booting from various memory devices. This tool generates an overview of the current configuration in a readable form as well as the boot ROM configuration data or EEPROM memory content that can be used for external memory boot configuration.

The BOOTROM component has two parts:

- Power-on reset configuration
- Boot ROM data configuration

The power-on reset configuration setting allows choosing required configuration values of the power-on reset signals for selected device using a set of properties. Each power-on reset configuration signal has its own property (or set of properties) named respectively for setting its desired value(s).

The boot ROM data configuration setting allows you to create either configuration data structure for external memory boot configuration or EEPROM data to be used for an I2C boot sequencer mode. The configuration of data structure or EEPROM data definition consists of a set of properties under the **Control Words** and the **Configuration Words** setting or the **I2C EEPROM Data** setting according to the selected output configuration. All these properties are provided in the Component Inspector view.

This section contains the following subsections:

- [Power-on reset configuration](#) on page 3
- [Boot ROM data configuration](#) on page 4
- [BootSequencer/BootROM vs PBL](#) on page 5

1.1.1 Power-on reset configuration

The power-on reset configuration user interface allows you to specify how the POR signals of the processor should be set.

When you initiate generating Processor Expert code, the tool outputs a set of reports describing how each POR signal should be set, based on the properties set in the user interface. The reports are in various formats: HTML, XML, and text.

These reports are only useful if you have a board with dual inline package (DIP) switches that control the POR signals, or if you are in the process of designing a custom board with a hard-wired POR configuration. In the former case, you can use the reports to guide you in toggling the DIP switches. In the latter case, you can use the reports to guide you in adding pull-up resistors to the board.

Setting the power-on reset configuration properties in the BOOTROM component is not very useful if you have a board without POR DIP switches and you are not looking to alter the board or design a new one.

1.1.2 Boot ROM data configuration

The boot ROM data configuration user interface allows you to configure how the processor will boot.

The tool provides the following two benefits:

- The boot ROM data configuration helps you configure the processor's Boot Sequencer. The QorIQ I2C-based Boot Sequencer allows you to have configuration registers set by the hardware after a power-on reset. The configuration registers you set are often ones in the processor's memory mapped register block (CCSR). For example, through CCSR, you can define LAWs, which connect portions of the physical memory space to specific interfaces, such as DDR or PCIe. While such a configuration is often done via boot code running on a core, the I2C Boot Sequencer allows you to configure registers without writing code and for the configuration to happen before the cores even run. This is done by storing configuration data on an EEPROM connected to the I2C bus and communicating to the processor via POR signals to initiate the I2C Boot Sequencer on reset.
- The boot ROM data configuration also helps you in planning on booting the processor from code residing on an SD, SDHC or MMC card, or an EEPROM or flash memory with an eSPI interface. To support these use cases, the processor's on-chip ROM memory has boot code with drivers for these external memories. This boot code will:
 1. Look for the boot configuration data and code on these specific external memories.
 2. Set system configuration registers (usually CCSR) based on your configuration data.
 3. Copy your boot code from the external memories into RAM (DDR SDRAM or L2/SRAM).
 4. Resume execution from within the boot code.

Using the POR configuration, you can instruct the processor to run its on-chip ROM boot code out of reset and to identify which external memory interface (eSDHC or eSPI) you have the boot configuration data and code in.

The boot configuration data mentioned above is, at a minimum, a set of address/data pairs set into a specific data structure. If you are booting from eSDHC or eSPI, the configuration data is followed by your boot code and related parameters. The parameters guide the processor's on-chip ROM boot logic in copying your boot code to RAM and running it. These parameters are your boot code's source address (on eSDHC or eSPI), destination address (in DDRAM or L2/SRAM), size, and entry point address.

The format of the boot configuration data is particular to the boot approach. The data structure for the I2C Boot Sequencer is different than the data structure for an SD(HC), MMC and eSPI boot. Also, the structures are not easy to assemble manually. By using the BOOTROM component in QCVS, you can enter the boot configuration data semantically through a user friendly interface without being concerned about the underlying data format. When you initiate Generate Processor Expert Code, the tool generates a file with the assembled data structure. You then flash or write that data structure to the I2C EEPROM, SD(HC) card, MMC, or eSPI memory.

The details of booting from external memory and using the I2C Boot Sequencer are lengthy and outside the scope of this guide. Users will require a deep understanding of these processor features to benefit from the BOOTROM Configuration tool. Such detail can be found in the reference manual of the SoC. There is also an Application Note (AN3659) that provides much insight into making use of the on-chip ROM boot code, with examples for running U-Boot and Linux from external memory. AN3659 is available on [Freescale website](#).

1.1.3 BootSequencer/BootROM vs PBL

In a QorIQ configuration project, you can use either the BOOTROM component or PBL component.

The pre-boot loader (PBL) hardware block is available in most mid-tier and high-tier QorIQ and Qonverge SoCs as a replacement and improvement to the I2C BootSequencer and BootROM, for configuring and booting the device after a reset. Depending on the processor you choose when creating a QorIQ configuration project, either the BOOTROM component or the PBL component will be available.

1.2 Using BOOTROM configuration tool

You can use the BOOTROM configuration tool to configure the power-on reset signals and create the external memory boot configuration file or I2C boot sequencer EEPROM data file content, by setting the component properties.

The component properties are available for configuration in the **Components** items displayed inside the **Component Inspector** view after adding the BOOTROM component to the QorIQ configuration project.

To configure BOOTROM, you first need to create a QorIQ configuration project with the BOOTROM configuration tool.

This section contains the following subsections:

- [Create a new QorIQ configuration project](#) on page 5
- [Configure POR configuration settings](#) on page 7
- [Configure boot ROM data](#) on page 10

1.2.1 Create a new QorIQ configuration project

This section explains how to create a new QorIQ configuration project for BOOTROM configuration.

To create a new QorIQ configuration project, follow these steps:

1. Run the Eclipse environment.
2. Choose **File > New > QorIQ Configuration Project** from the IDE menu bar. The **New QorIQ Configuration Project** wizard starts, showing the **Create a QorIQ Configuration Project** page.
3. Specify a name for your project and click **Next**. The **Devices** page appears.
4. Select the required target SoC including its silicon revision, for example, P1021, Rev 1.1. Click **Next**. The **Toolset selection** page appears.
5. Select the **BOOTROM Configuration** checkbox, as shown in the figure below.

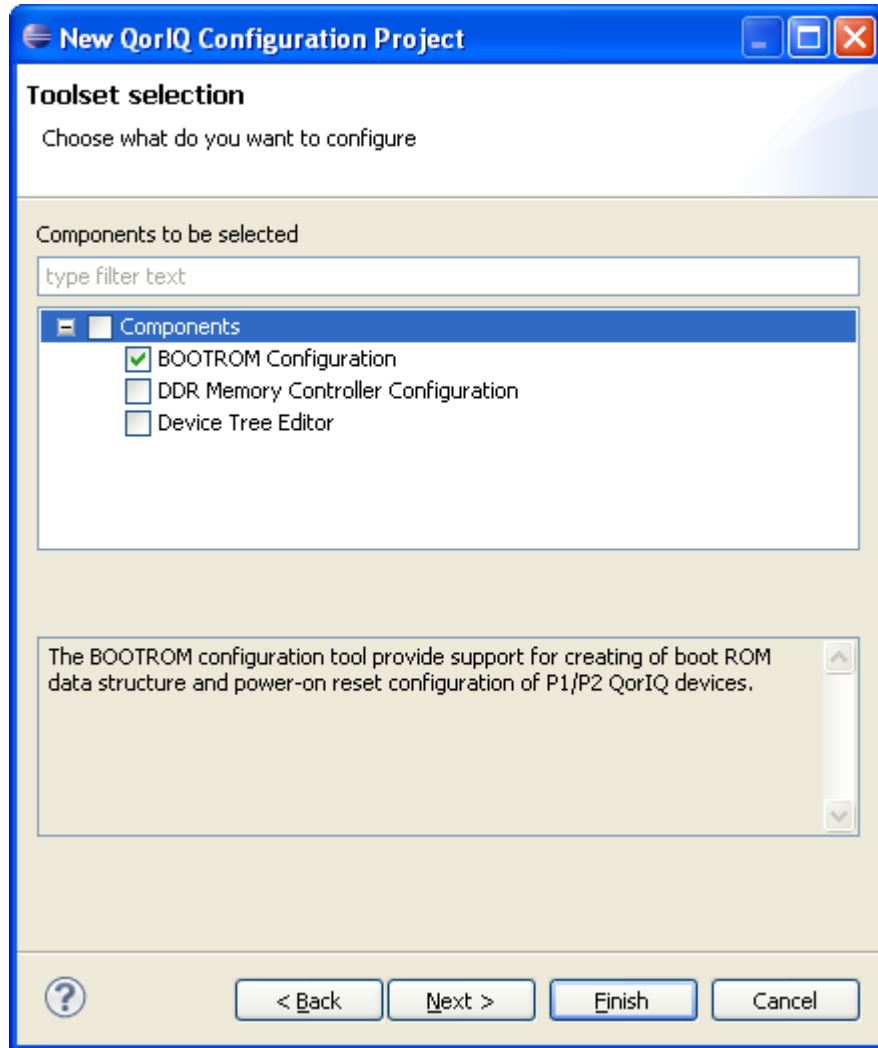


Figure 1. Toolset selection page

6. Click **Next**. The **BOOTROM Configuration** page appears (shown in the figure below).

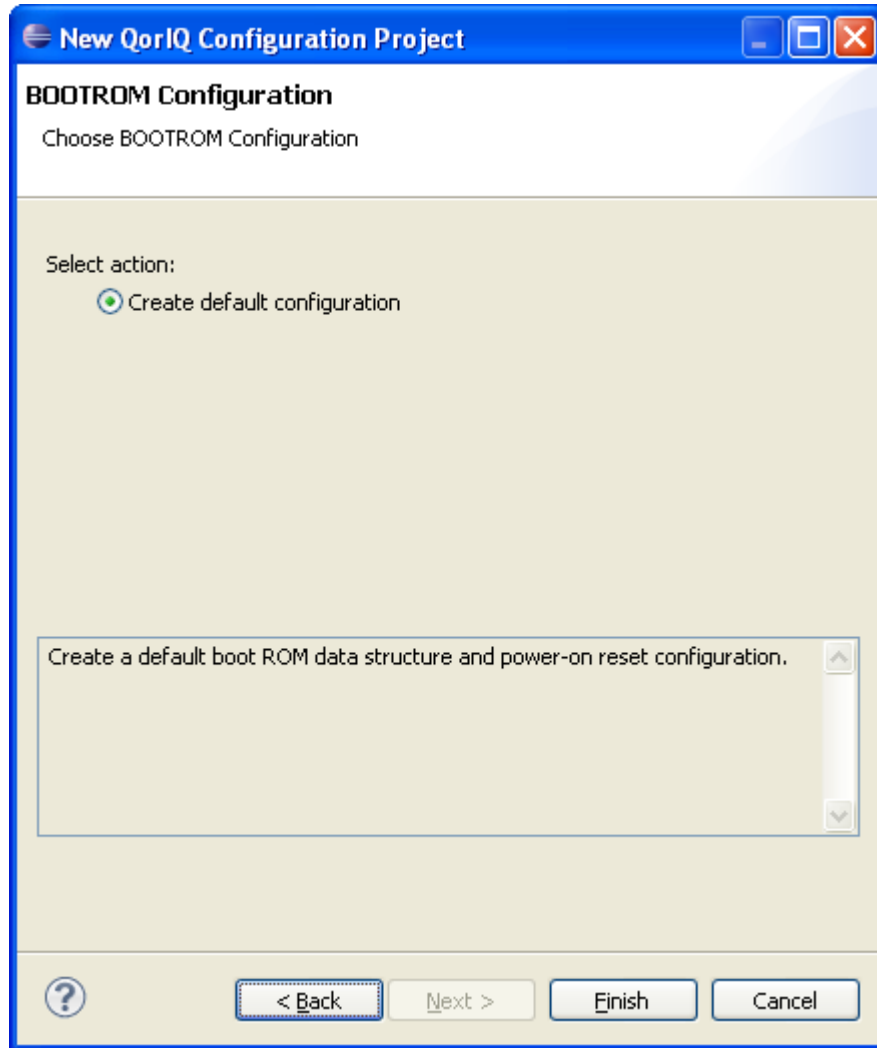


Figure 2. BOOTROM Configuration page

7. Click **Finish**. The **New QorIQ Configuration Project** wizard ends and your project is created.

1.2.2 Configure POR configuration settings

This section explains how to configure the BOOTROM component and its power-on reset configuration options in QCVS.

To configure the BOOTROM component and its power-on reset configuration options, perform these steps:

1. After creating the QorIQ configuration project with the BOOTROM Configuration tool, select the BOOTROM component under the **Components** folder in the **Components** view, as shown in the following figure. The properties of the BOOTROM component opens in the **Component Inspector** view.

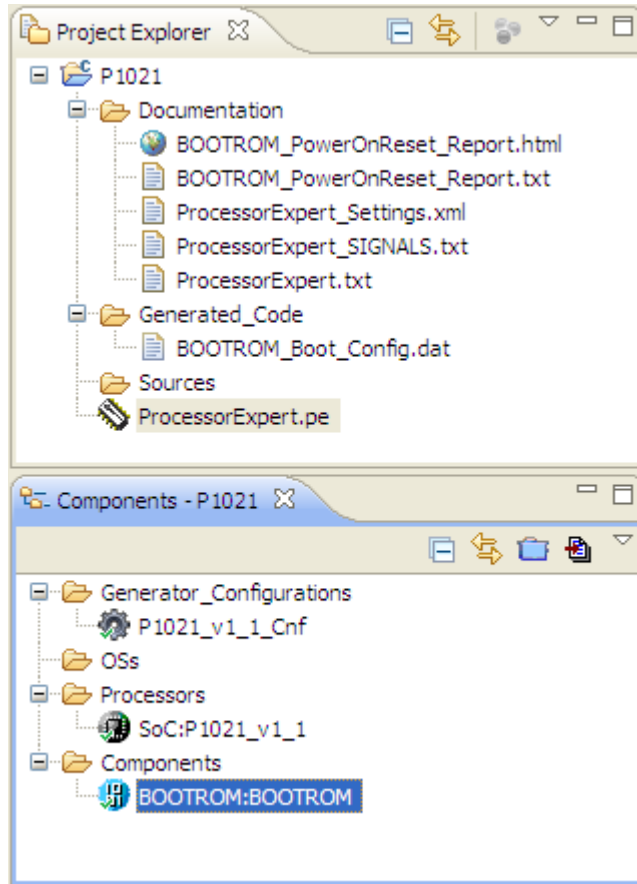


Figure 3. BOOTROM component selected in Components view

NOTE

If the **Component Inspector** view is closed, you can open it by right-clicking on the BOOTROM component in the **Components** view and then selecting the **Inspector** option from the pop-up menu.

2. Adjust power-on reset configuration options and/or edit external memory boot configuration file content according to your requirements. All the settings and properties of the BOOTROM component are grouped in categories.

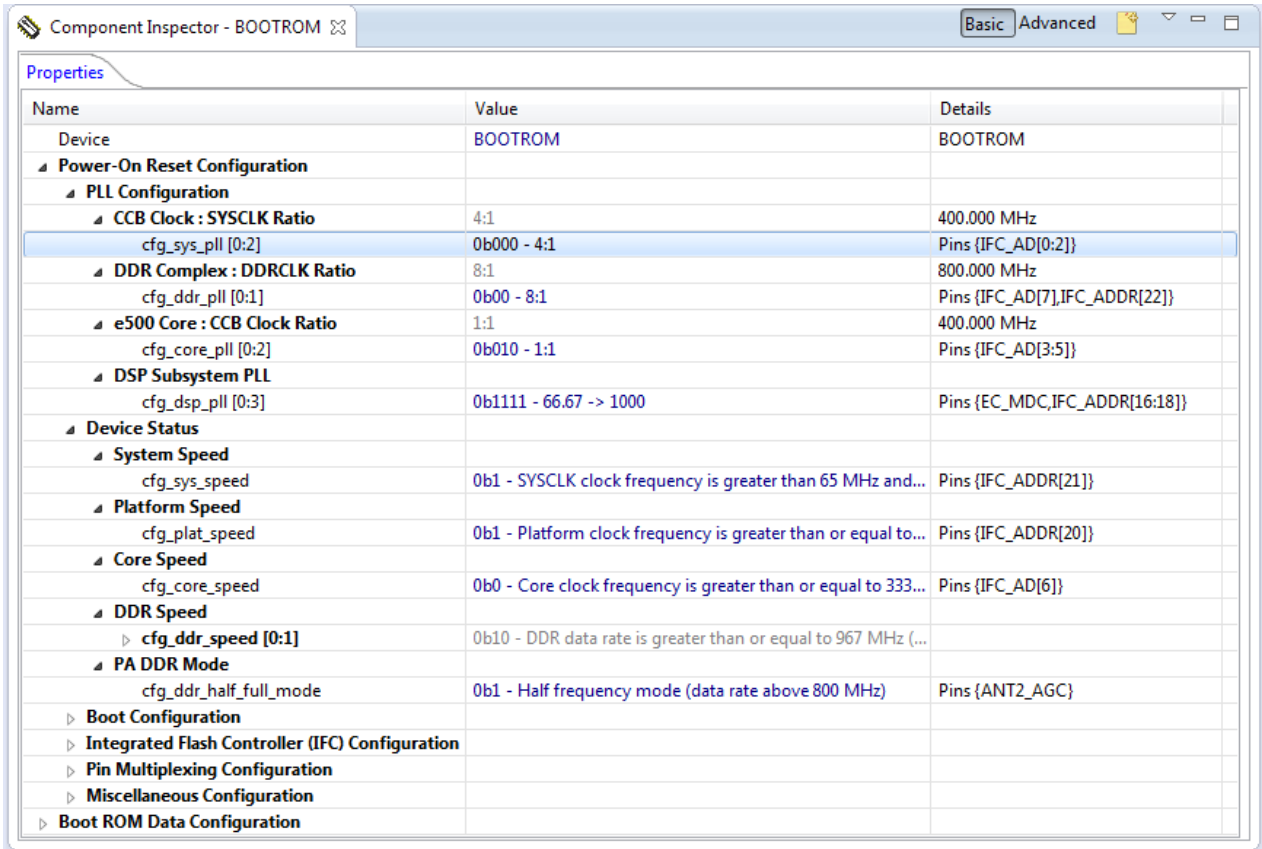


Figure 4. Setting POR configuration properties

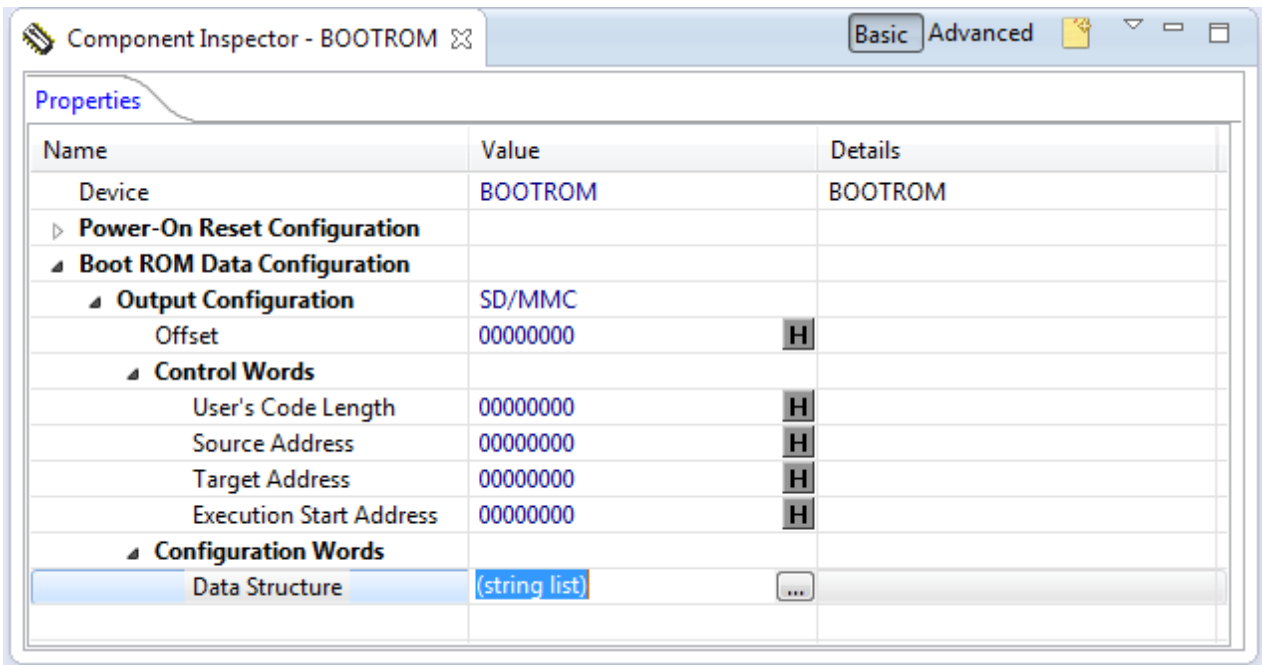


Figure 5. Setting boot ROM data configuration properties

1.2.3 Configure boot ROM data

This section explains how to configure the BOOTROM component external memory boot configuration or I2C boot sequencer EEPROM data content in QCVS.

To configure the BOOTROM component external memory boot configuration or the I2C boot sequencer EEPROM data content, follow these steps:

1. Expand **Boot ROM Data Configuration** to open its properties, and select the desired output configuration.

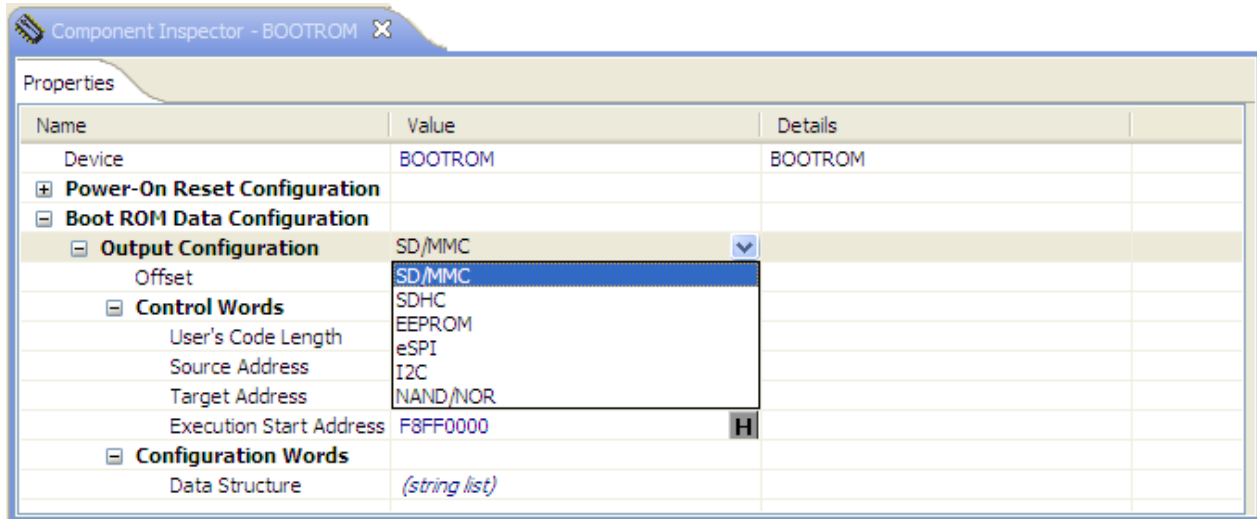


Figure 6. Output configuration options

NOTE

You can define proper content of the data structure depending on the flash or EEPROM memory type.

To configure external memory boot configuration for flash and SPI interfaces, you can edit **Configuration Words** settings independently using the *Data Structure* property. The *Data Structure* property helps you define specific address space initialization and data to be configured.

1. Switch to the Graphical Mode of the **Component Inspector** view to display the properties of the BOOTROM component.
2. Select the *Data Structure* property, as shown in the following figure, to display the **Data Structure** dialog on the right.
3. Select the configuration command type from the drop-down box.
4. In the **Address 0x** text box, specify the address where you want to add the data.
5. In the **Data 0x** text box, specify the data that needs to be added at the specified address.
6. Click **Add**.

The data structure definition gets added in the text area.

7. Click **Restore** to restore the last saved settings.
8. Click **Apply** to save the settings.

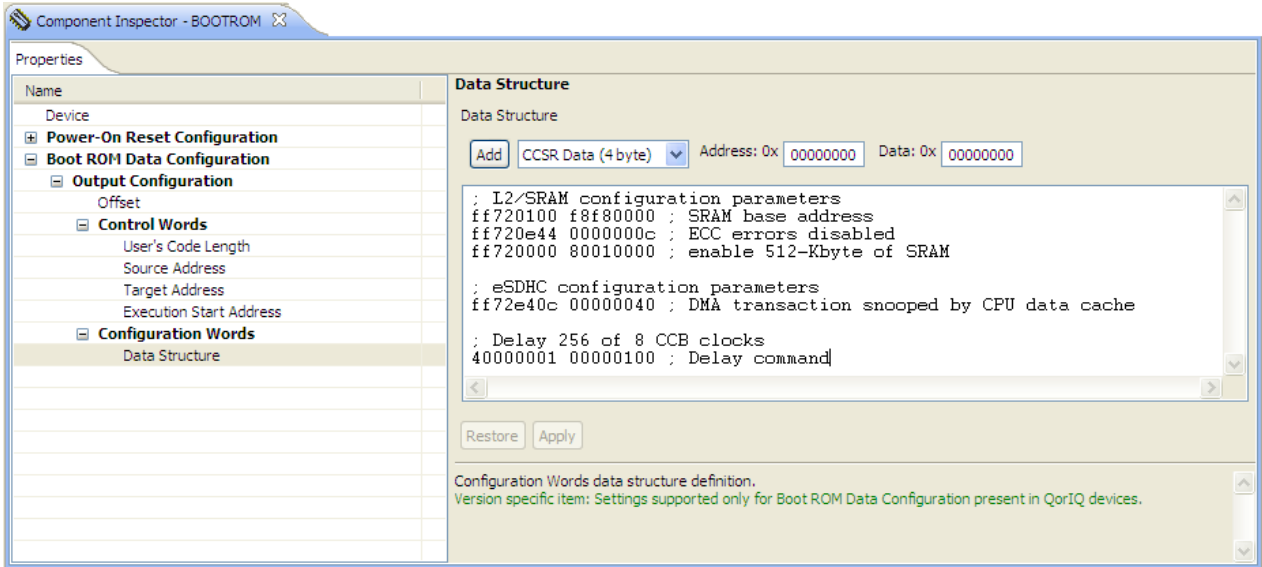




Figure 7. Editing Data Structure property

NOTE

You can also open the **Data Structure** dialog by clicking the  button in the *Data Structure* property as shown in the following figure. The  button is available only in Basic/Advanced/Expert modes.

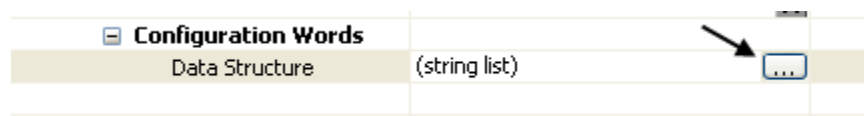


Figure 8. Opening Data Structure dialog

To configure EEPROM data for I2C boot sequencer mode:

1. Select the **I2C** option from the **Output Configuration** drop-down box.
2. Select the output format for the boot sequencer data from the **Output Format** property.

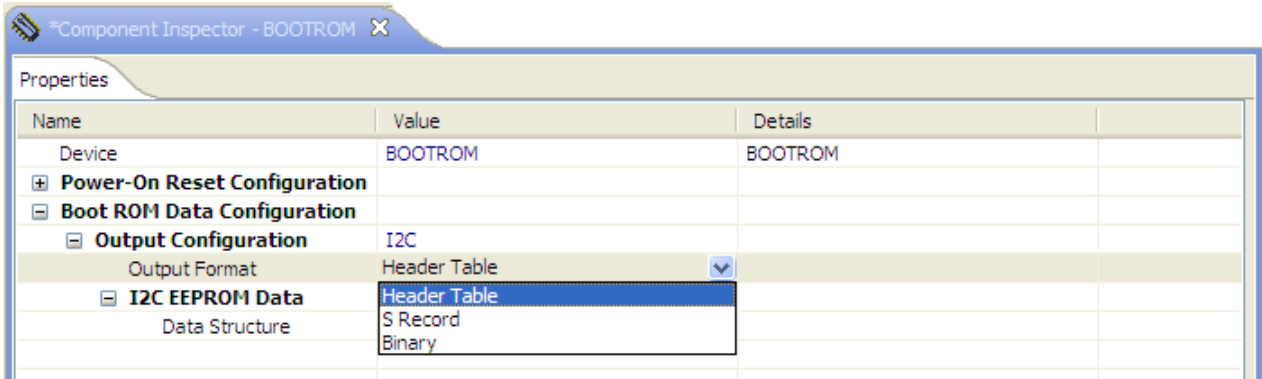


Figure 9. Selecting output format of I2C EEPROM data

You can edit **I2C EEPROM Data** settings independently using the **Data Structure** property. This property helps you define specific CCSR or ACS space address initialization and data to be configured. It also allows entering ACS data from binary file or user-defined data in a simple text value CSV format.

3. Switch to the Graphical Mode of the Component Inspector view to display the UI elements of the Data Structure.
4. Select the **Data Structure** property, as shown in the following figure, to display the **Data Structure** dialog.

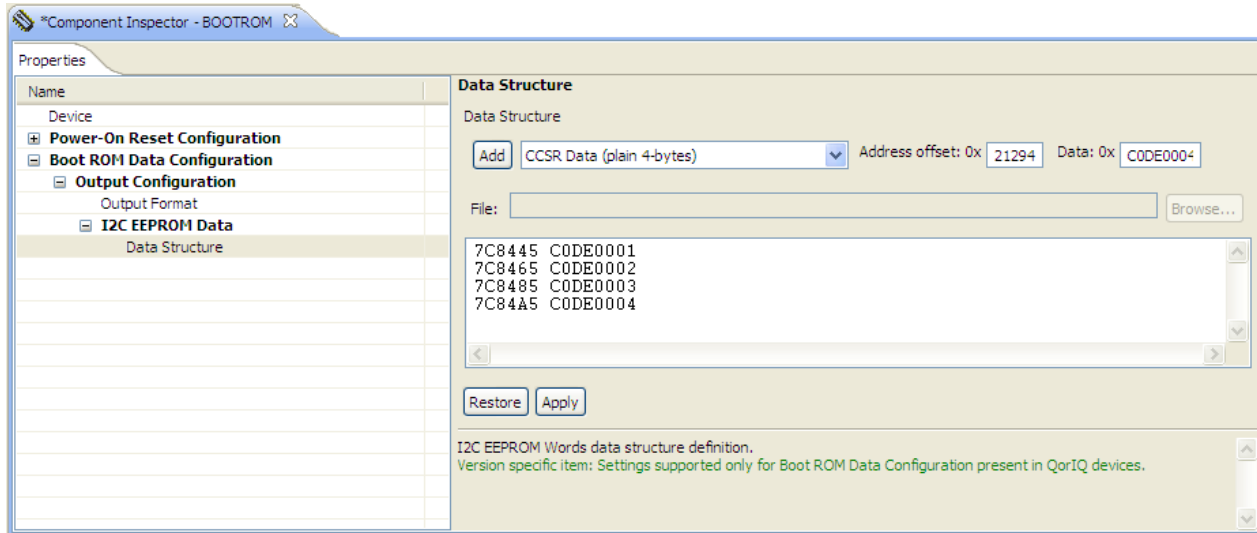


Figure 10. Editing Data Structure property - I2C EEPROM data

5. Select the configuration option from the drop-down box. You can use the UI elements according to the selected option.
6. In the **Address offset 0x** text box, specify the address offset where you want to add the data.
7. In the **Data 0x** text box, specify the data that needs to be added at the specified address.
8. In the **File** text box, specify the file from which you want to read either ACS data or any user-defined data.
9. Click **Add**.
The data structure definition gets added in the text area.
10. Click **Restore** to restore the last saved settings.
11. Click **Apply** to save the settings.

1.3 Generating code from BOOTROM component

During or after modifying configuration settings of the BOOTROM component, you can generate code to get output of the component.

The code generation from the BOOTROM component is based on the properties set in the **Component Inspector**. The output from the BOOTROM component is generated on user request.

The following output files can be generated by the BOOTROM tool depending on the tool configuration:

- Power-on reset configuration overview report
- External Memory Boot Configuration Data Structure file
- I2C Boot Sequencer EEPROM Data file

See [BOOTROM component output file types](#) on page 16 for more details on these file types.

Perform any of the following steps to generate code:

1. Click the **Generate Processor Expert Code** icon  in the **Components** view, as shown in the following figure.

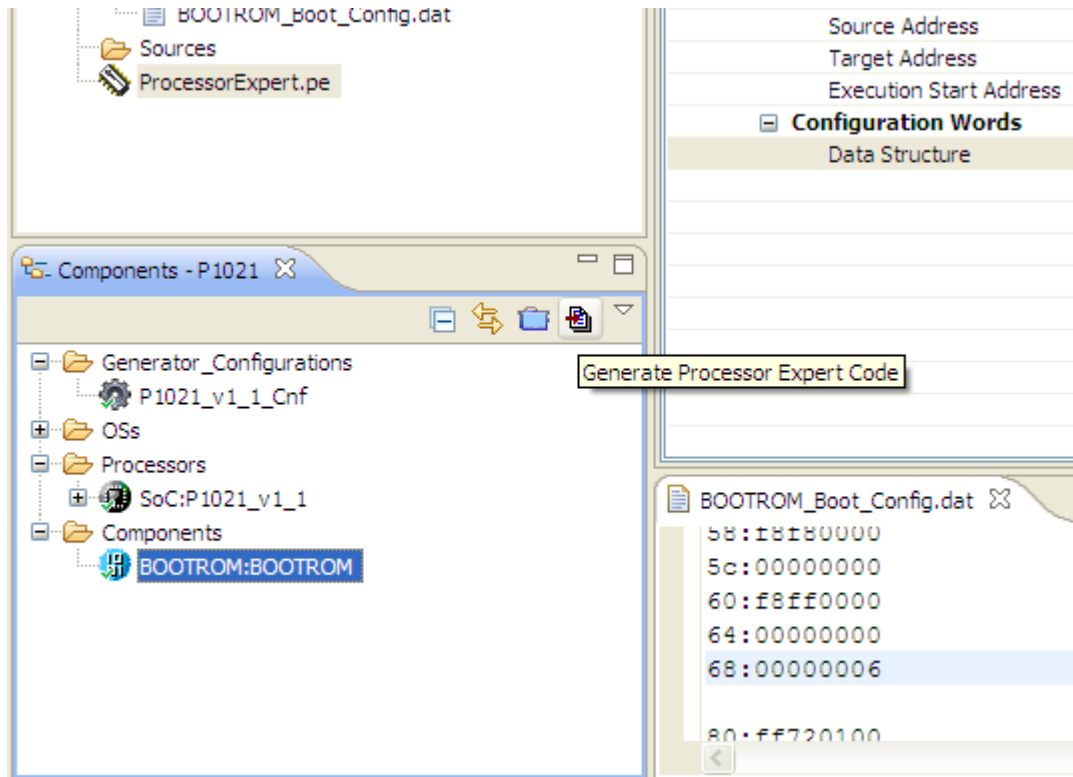


Figure 11. Generate Processor Expert Code icon

2. Right-click the `ProcessorExpert.pe` node in the **Project Explorer** view, and select the **Generate Processor Expert Code** option from the context menu.

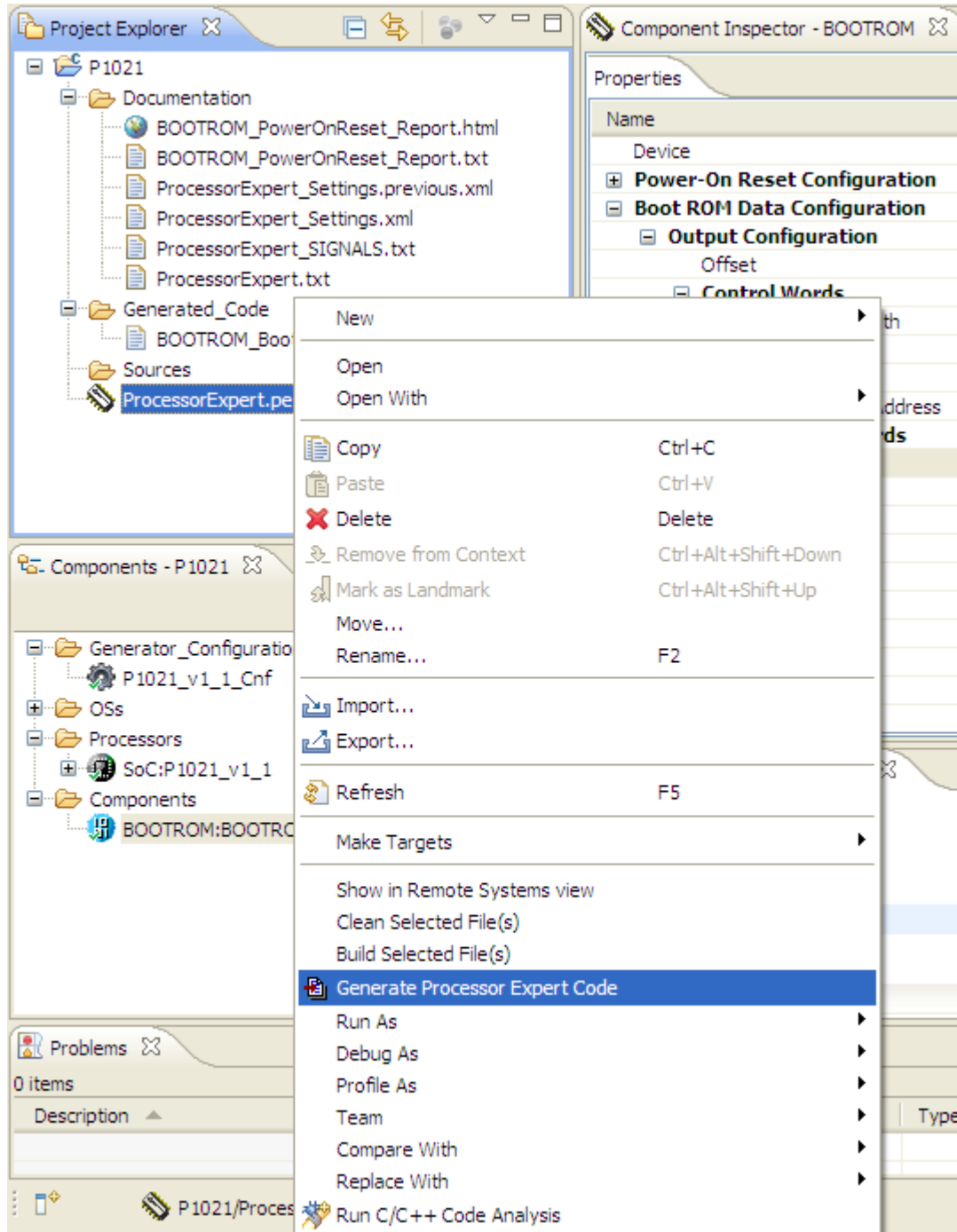


Figure 12. Generate Processor Expert code

3. Select the **Project > Generate Processor Expert Code** option from the Eclipse IDE menu bar.
The **Generating Code** dialog appears.

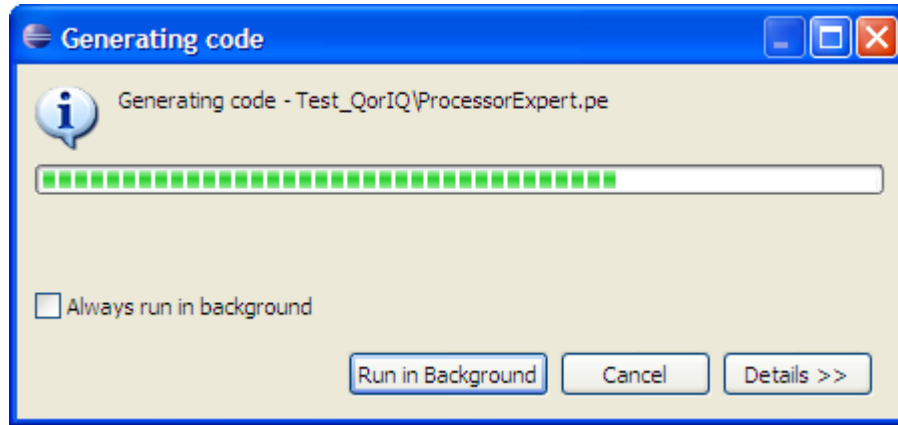


Figure 13. Code generation

The output file is generated and added to the **Project Explorer** view in the **Generated_Code** folder. The names of the output files use the name of the BOOTROM component included in the QorIQ configuration project.

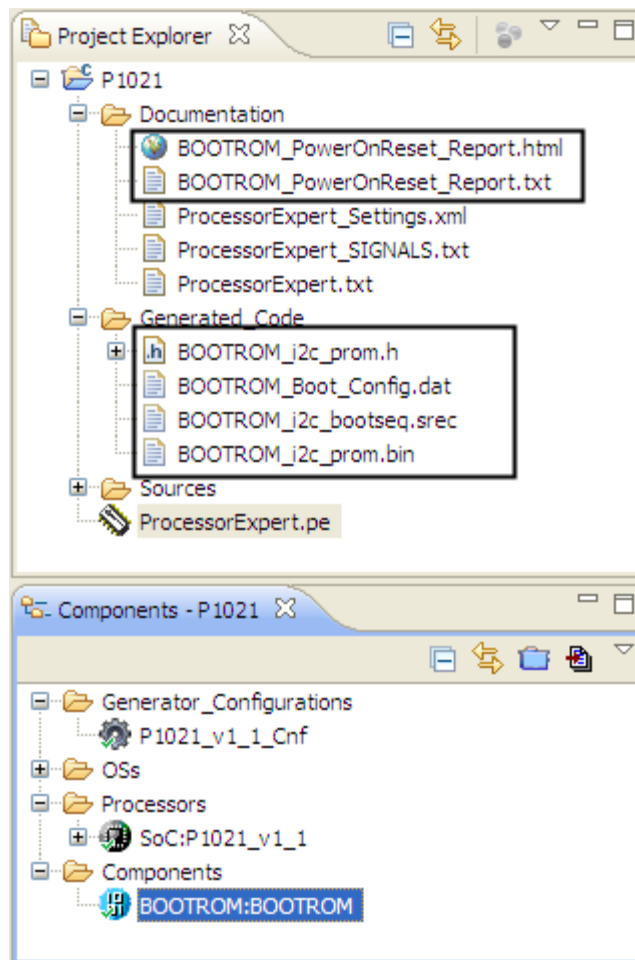


Figure 14. BOOTROM-generated code files in Project Explorer

4. Double-click the .html and .txt files (in the **Documentation** folder of the **Project Explorer**) to view the Power-on Reset Configuration Overview report in the html and text formats.

5. Double-click the .dat file (in the **Generated_Code** folder) to view the External Memory Boot Configuration Data Structure file.
6. Double-click the .h, .bin or .srec file (in the **Generated_Code** folder) to view the I2C EEPROM data file content.

1.4 BOOTROM component output file types

This section describes the output files generated by the BOOTROM tool.

Power-on reset (POR) target configuration overview report

This overview report is always generated, and it includes all Power-On Reset configuration details as configured within the respective section of the UI properties. It contains description of the power-on reset configuration signals and their actual values configured in the tool as well as the pin locations where the respective values shall be exposed during power-on reset time.

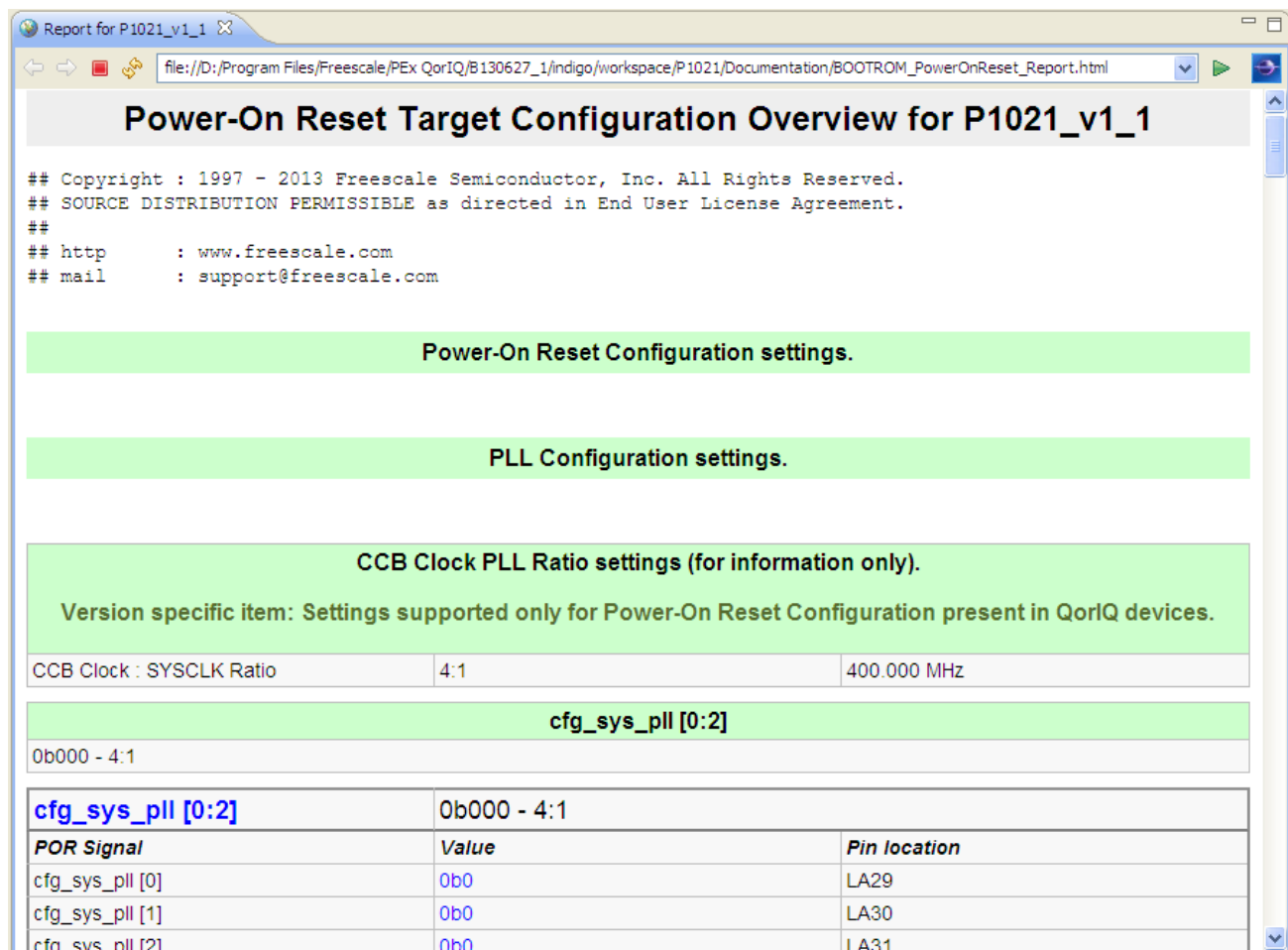


Figure 15. POR target configuration overview report

External memory boot configuration data structure file

This data structure file is generated for output configuration of SD/MMC, SDHC, EEPROM, eSPI on NAND/NOR flash option selected in the respective UI property.

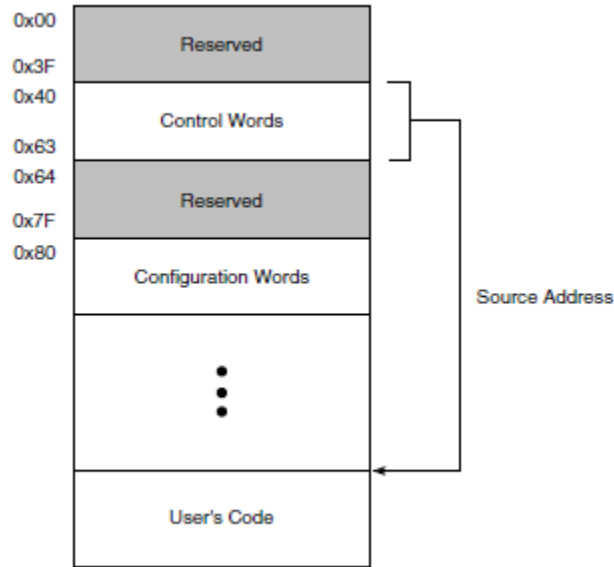


Figure 16. External memory data structure

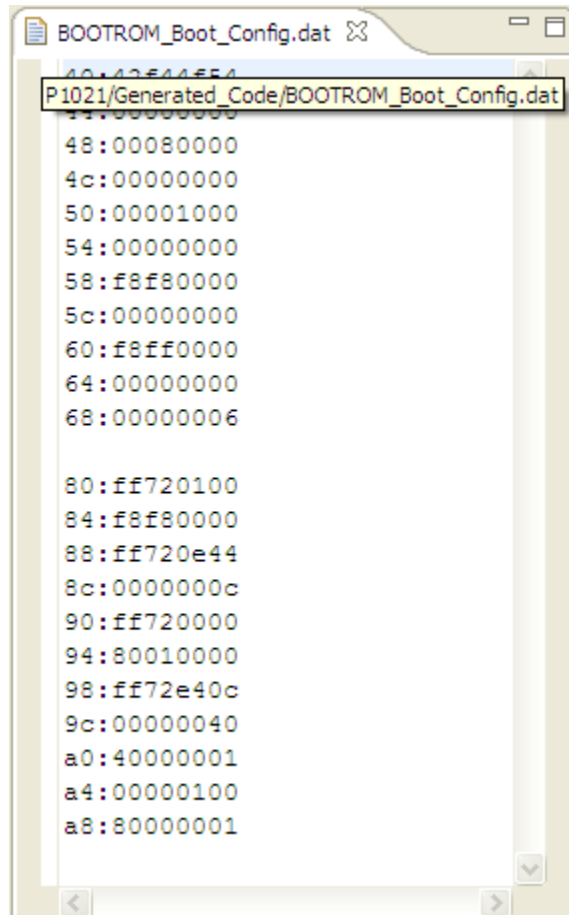


Figure 17. Boot configuration data structure file example

I2C boot sequencer EEPROM data file

BOOTROM Configuration

BOOTROM component output file types

This EEPROM data file, which can be used in boot sequencer mode of the I2C module, is generated for output configuration of the I2C option selected in the respective UI property ([Figure 9. Selecting output format of I2C EEPROM data](#) on page 11).

0	1	4	5	6	7
ACS	BYTE_EN		CONT	ADDR[0-1]	
ADDR[2-9]					
ADDR[10-17]					
DATA[0-7]					
DATA[8-15]					
DATA[16-23]					
DATA[24-31]					

Figure 18. EEPROM data format for one register preload

0	1	2	3	4	5	6	7	
1	0	1	0	1	0	1	0	Preamble
0	1	0	1	0	1	0	1	
1	0	1	0	1	0	1	0	
ACS	BYTE_EN			1	ADDR[0-1]			
ADDR[2-9]								
ADDR[10-17]								
DATA[0-7]								
DATA[8-15]								
DATA[16-23]								
DATA[24-31]								
ACS	BYTE_EN			1	ADDR[0-1]			
ADDR[2-9]								
ADDR[10-17]								
DATA[0-7]								
DATA[8-15]								
DATA[16-23]								
DATA[24-31]								
.								
.								
.								
ACS	BYTE_EN			1	ADDR[0-1]			
ADDR[2-9]								
ADDR[10-17]								
DATA[0-7]								
DATA[8-15]								
DATA[16-23]								
DATA[24-31]								
0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	

Figure 19. EEPROM contents

The following three EEPROM data output file formats are supported by BOOTROM:

- Header Table file with an EEPROM data structure defined as an array of properly formatted data including all user-defined configuration data as shown in [Figure 10. Editing Data Structure property - I2C EEPROM data](#) on page 12.

```

P1021/Generated_Code/BOOTROM_i2c_prom.h *****
** THIS COMPONENT MODULE IS GENERATED BY THE TOOL. DO NOT MODIFY IT.
**   Filename: BOOTROM_i2c_prom.h
**
** Copyright : 1997 - 2013 Freescale Semiconductor, Inc. All Rights Reserved.
** SOURCE DISTRIBUTION PERMISSIBLE as directed in End User License Agreement.
**
** http      : www.freescale.com
** mail     : support@freescale.com
** ***** */

/* Use a flash programmer to flash processing of this file. */

unsigned char eeprom_data [] = {
    0xAA, 0x55, 0xAA,
    0x7C, 0x84, 0x45, 0xC0, 0xDE, 0x00, 0x01,
    0x7C, 0x84, 0x65, 0xC0, 0xDE, 0x00, 0x02,
    0x7C, 0x84, 0x85, 0xC0, 0xDE, 0x00, 0x03,
    0x7C, 0x84, 0xA5, 0xC0, 0xDE, 0x00, 0x04,
    0x00, 0x00, 0x00, 0xD5, 0xED, 0xB4, 0x1E,
    0x00, 0x00, 0x00,
};
  
```

Figure 20. Header table output file example

- Binary file which is the binary form of the header file with the EEPROM data definition that can be used by external EEPROM programmer tool.

```

P1021/Generated_Code/BOOTROM_i2c_prom.bin
*U* | ,EÀP00 | ,eÀP00 | ,ÀP00 | ,YÀP0000000i '000
  
```

Figure 21. Binary output file example

- S Record, which is another form of the properly formatted EEPROM data that can be used by external EEPROM programmer tool.

```

P1021/Generated_Code/BOOTROM_i2c_bootseq.srec
S00600004844521B
S30800000000AA55AA4E
S30C000000037C8445C0DE00010C
S30C0000000A7C8465C0DE0002E4
S30C000000117C8485C0DE0003BC
S30C000000187C84A5C0DE000494
S30C0000001F000000D5EDB41E40
  
```

Figure 22. S Record output file example

How To Reach Us**Home Page:**

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

NXP, the NXP logo, Freescale, the Freescale logo, CodeWarrior, QorIQ, and Processor Expert are trademarks of NXP B.V. All other product or service names are the property of their respective owners. All rights reserved.

© 2017 NXP B.V.

QCVS_BOOTROM_User_Guide
Rev. 4.x
02/2017

