NXP Semiconductors
User's Guide

Document Number: QCVS_PBL_User_Guide
Rev. 4.x, 06/2017

QCVS PBL Tool User Guide

Contents

Chapter 1 PBL Configuration and Validation

This document introduces the Pre-Boot Loader (PBL) configuration and validation tool, which is an embedded component of QorIQ Configuration and Validation Suite (QCVS).

The PBL tool has two components: PBL configuration tool and PBL validation tool. The PBL configuration tool allows you to create a configuration for the PBL component and the PBL validation tool allows you to validate the PBL configuration.

PBL is an internal hardware IP block that loads initialization data from a specified memory device interface and performs pre-boot initialization of QorIQ device registers and local memory space.

NOTE

For QorIQ LS2 processor family, the PBL block has been replaced with software, specifically BootROM code running on the Service Processor. As such, the term PBL is no longer used in the LS2 processor reference manuals. However, for simplicity, QCVS will continue to refer to this SoC feature as PBL.

The QCVS PBL tool helps you create and modify pre-boot initialization data used for configuring QorIQ devices. The pre-boot initialization data has two parts:

- A reset configuration word (RCW), which is 512 or 1024 bits of information (depending on the processor)
- An optional pre-boot initialization (PBI) command sequence

For details on the pre-boot loader, see the reference manual of the SoC being used.

The PBL component represents the RCW as an organized set of modifiable properties in the **Component Inspector** view. Each RCW field is represented by a property. In addition, the PBI command data is accessible using a special property. The editor for this property has the user interface to view, add, and modify the PBI commands.

Supporting a wide range of workflows and use-cases, the PBL tool allows you to import or generate pre-boot initialization data in various file formats. These file formats are described in PBL component import file types on page 19 and PBL component output file types on page 24.

You can also use the PBL component to modify a PBL configuration stored in a file, such as a PBL binary image retrieved from the non-volatile memory of the board. For more details, see PBL component import file types on page 19.

For additional information on how to configure PBL for a QorIQ device using the QCVS PBL tool, see PBL Configuration using QCVS Application Note (AN5260).

This chapter is divided into the following sections:

- PBL configuration on page 3
- PBL validation on page 28

1.1 PBL configuration

This section describes how to work with the PBL configuration tool, for example, how to create a new QorIQ configuration project, how to configure the PBL component, and how to perform basic operations using the PBL configuration tool.

This chapter contains the following sections:

- Using PBL configuration tool on page 4
- Generating output from PBL component on page 12
- Importing PBL configuration on page 15
- PBL component import file types on page 19

- PBL component output file types on page 24
- PBI commands on page 26

1.1.1 Using PBL configuration tool

The PBL configuration tool allows you to create and modify pre-boot initialization data used to configure the QorIQ device out of reset.

To configure PBL, you need to first create a QorIQ configuration project.

This section describes the following topics:

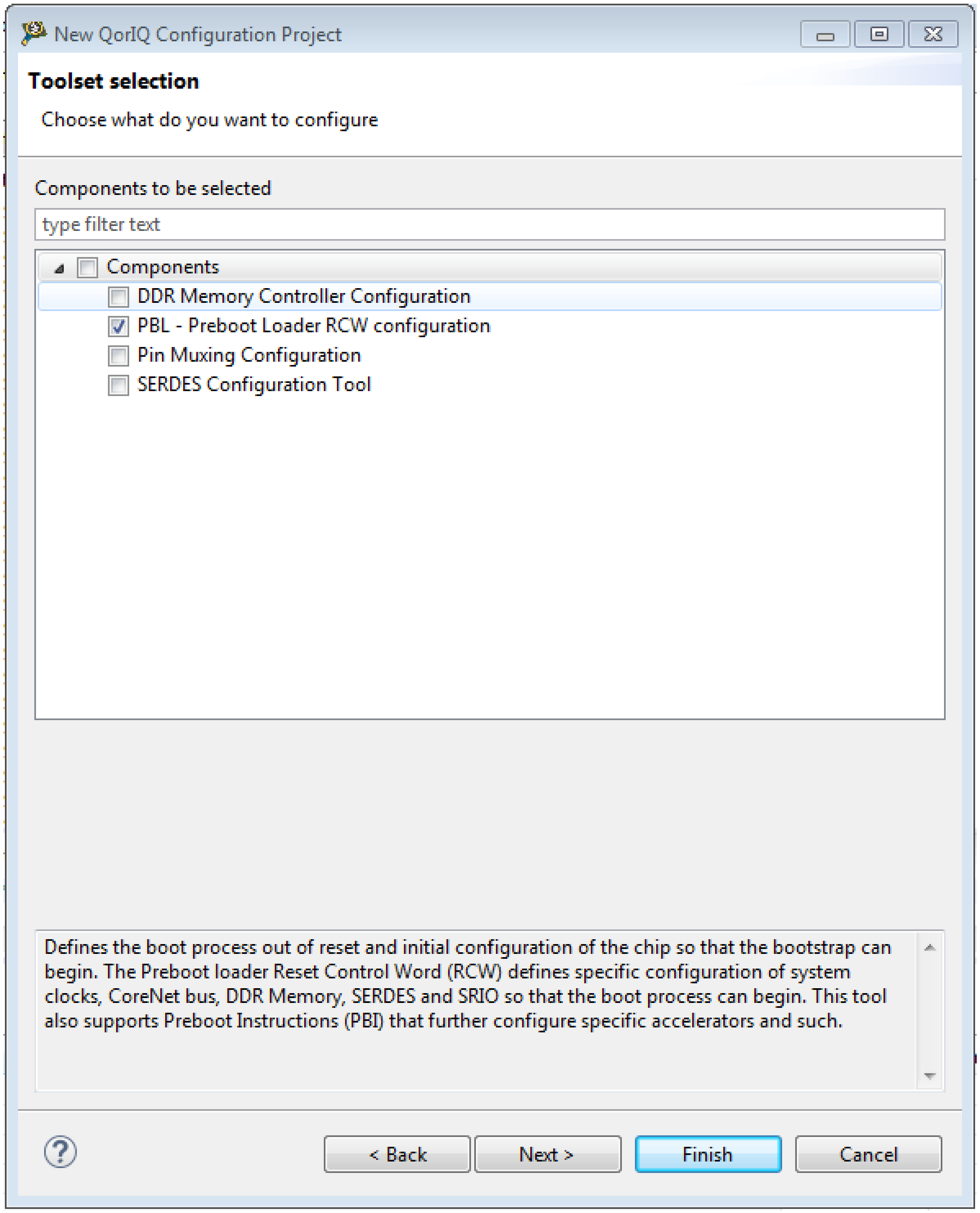
- Create a new QorIQ configuration project on page 4
- Configure PBL on page 7

1.1.1.1 Create a new QorIQ configuration project

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

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

1. Select **File > New > QorIQ Configuration Project**, and follow the steps in the **New QorIQ Configuration Project** Wizard.
2. Enter the project name.
3. Select the required target SoC including the silicon revision in the **Devices** page.
4. On the **Toolset selection** page, select the **PBL - Preboot Loader RCW configuration** checkbox.



Toolset selection

Choose what do you want to configure

Components to be selected

type filter text

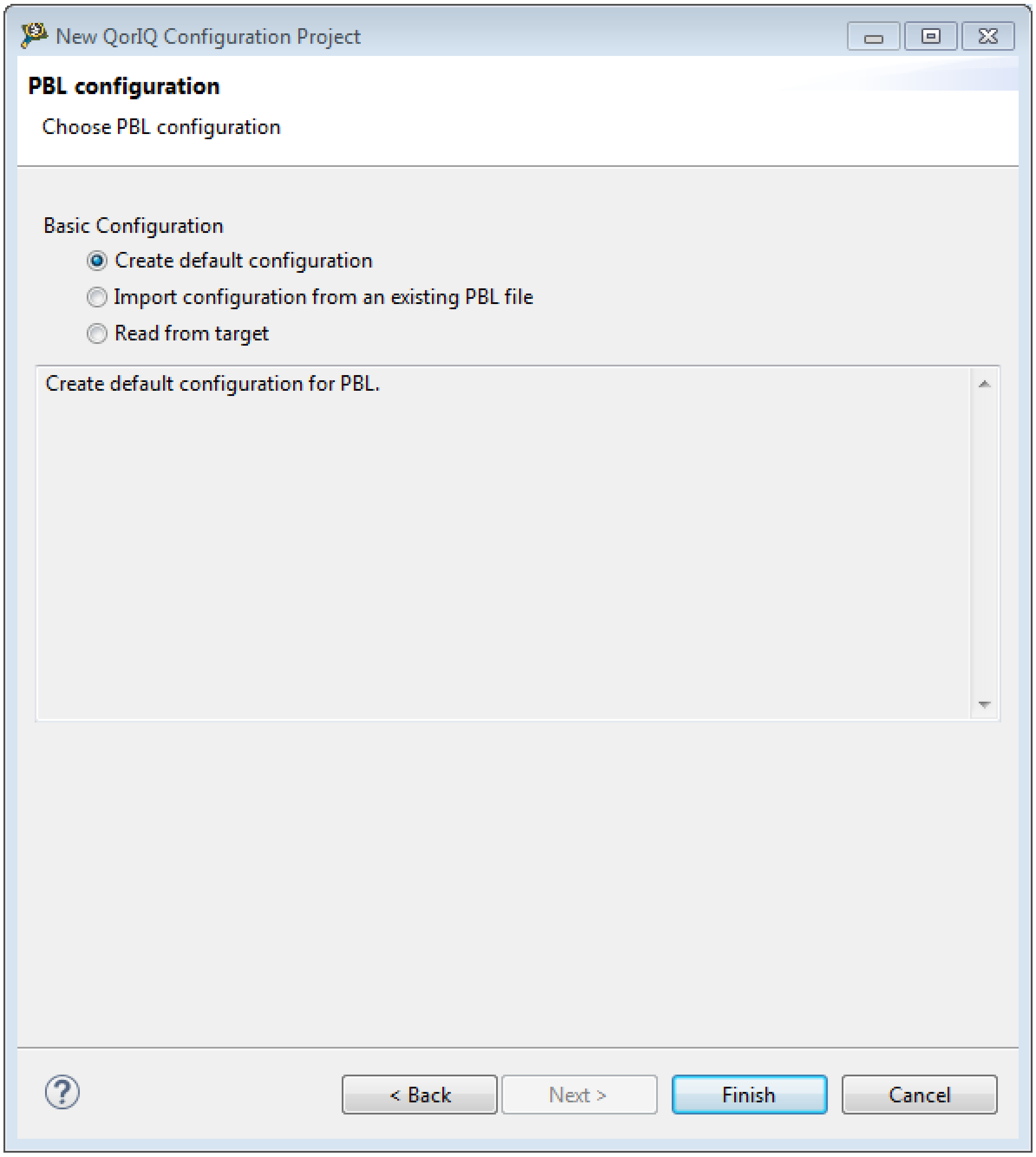
- [ ] Components
  - [ ] DDR Memory Controller Configuration
  - [x] PBL - Preboot Loader RCW configuration
  - [ ] Pin Muxing Configuration
  - [ ] SERDES Configuration Tool

Defines the boot process out of reset and initial configuration of the chip so that the bootstrap can begin. The Preboot loader Reset Control Word (RCW) defines specific configuration of system clocks, CoreNet bus, DDR Memory, SERDES and SRIO so that the boot process can begin. This tool also supports Preboot Instructions (PBI) that further configure specific accelerators and such.

Figure 1. Toolset selection page

5. Click **Next**.

   The **PBL configuration** page appears.



Basic Configuration

- [✓ Create default configuration]
- Import configuration from an existing PBL file
- Read from target

Create default configuration for PBL.

Figure 2. PBL configuration page

6. Specify the settings according to requirements.

NOTE

You can configure the PBL component while creating QorIQ Configuration project. On the **PBL Configuration** page (Figure 2. PBL configuration page on page 6), select the **Import configuration from an existing PBL file** or **Read from target** option, and provide the necessary details corresponding to the selected option. For details, see Importing PBL configuration section.

7. Click **Finish** to complete the project creation.

This creates a QorIQ configuration project for PBL configuration.

1.1.1.2 Configure PBL

This section explains how to configure the PBL component inside QCVS to generate pre-boot initialization data.

To configure PBL component:

1. After creating the QorIQ project with the PBL tool, select the PBL component under the **Components** folder in the **Components** view, as shown in figure below. The properties of the PBL component is displayed in the **Component Inspector** view.

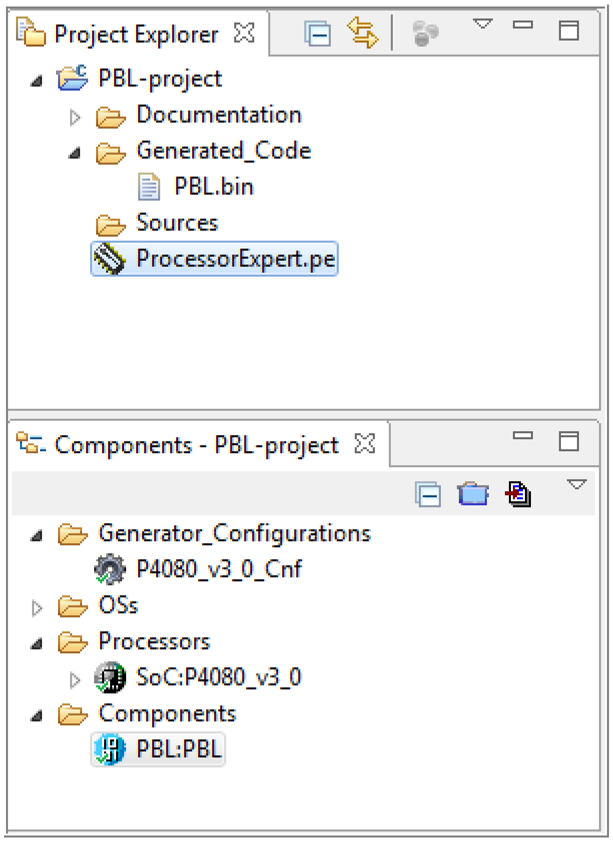
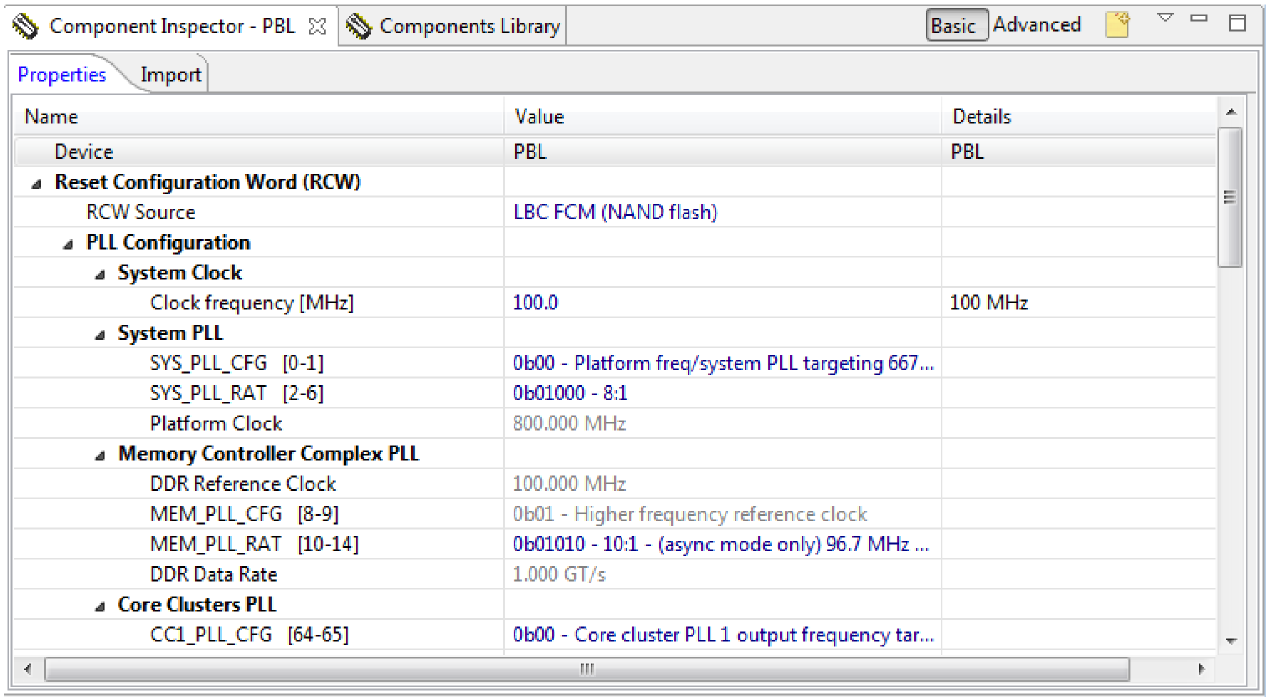


Figure 3. PBL component selected in Components view

NOTE

If the **Component Inspector** view is closed, then open it by double-clicking the component in the **Components** view.

2. Adjust RCW bit field values or edit PBI commands according to your requirements. All the settings and properties of the PBL component are sorted by RCW position.



Component Inspector - PBL | Components Library | Basic | Advanced

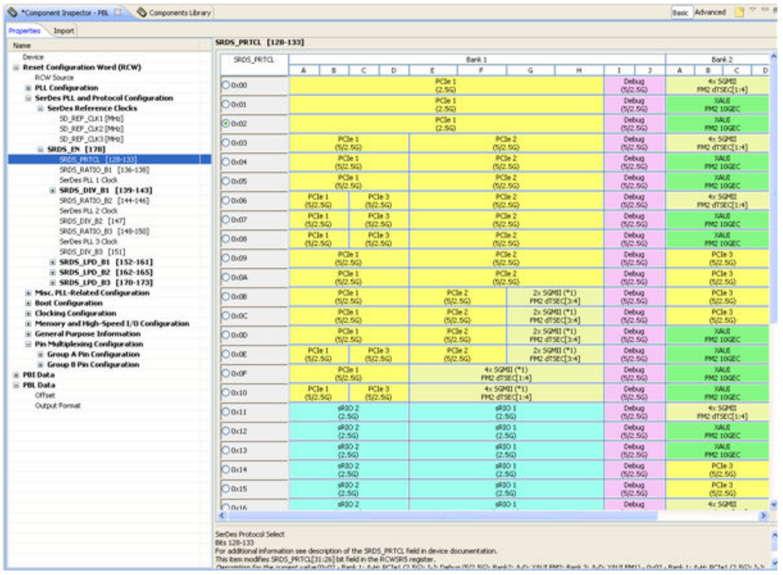
Properties | Import

Name	Value	Details
Device	PBL	PBL
Reset Configuration Word (RCW)		
RCW Source	LBC FCM (NAND flash)	
PLL Configuration		
System Clock		
Clock frequency [MHz]	100.0	100 MHz
System PLL		
SYS_PLL_CFG [0-1]	0b00 - Platform freq/system PLL targeting 667...	
SYS_PLL_RAT [2-6]	0b01000 - 8:1	
Platform Clock	800.000 MHz	
Memory Controller Complex PLL		
DDR Reference Clock	100.000 MHz	
MEM_PLL_CFG [8-9]	0b01 - Higher frequency reference clock	
MEM_PLL_RAT [10-14]	0b01010 - 10:1 - (async mode only) 96.7 MHz ...	
DDR Data Rate	1.000 GT/s	
Core Clusters PLL		
CC1_PLL_CFG [64-65]	0b00 - Core cluster PLL 1 output frequency tar...	

Figure 4. Component Inspector - PBL properties

NOTE

Most of the items of the PBL component are available at Basic level. Most RCW fields can be modified by typing in a value or by selecting a choice from a drop-down list. However, some fields may expose a more sophisticated GUI for modifying the field, as in the case of `SRDS_PRTCL`.



*Component Inspector - PBL | Components Library | Basic | Advanced

Properties | Import

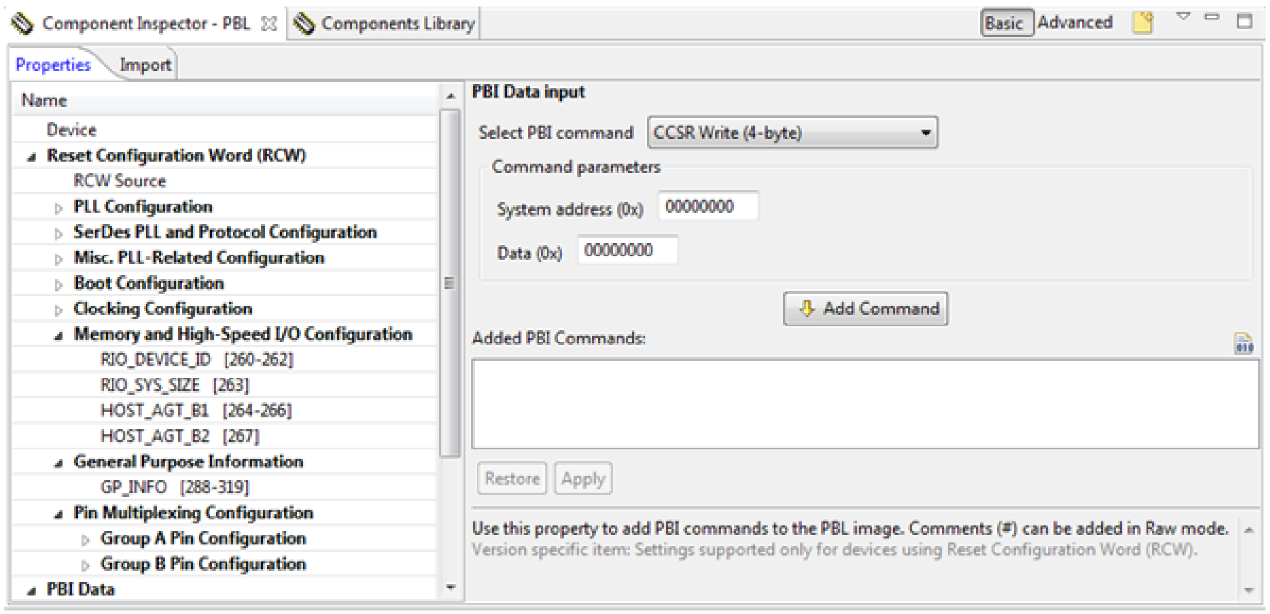
SRDS_PRTCL [128-133]

- Device
- Reset Configuration Word (RCW)
  - RCW Source
  - PLL Configuration
  - SerDes PLL and Protocol Configuration
    - SerDes Reference Clocks
      - SD_REF_CLK1 [MHz]
      - SD_REF_CLK2 [MHz]
      - SD_REF_CLK3 [MHz]
    - SRDS_EN [178]
      - SRDS_PRTCL [128-133]
      - SRDS_RATIO_B1 [136-138]
      - SerDes PLL 1 Clock
      - SRDS_DIV_B1 [139-143]
      - SRDS_RATIO_B2 [144-146]
      - SerDes PLL 2 Clock
      - SRDS_DIV_B2 [147]
      - SRDS_RATIO_B3 [148-150]
      - SerDes PLL 3 Clock
      - SRDS_DIV_B3 [151]
      - SRDS_LPD_B1 [152-161]
      - SRDS_LPD_B2 [162-165]
      - SRDS_LPD_B3 [170-173]
  - Misc. PLL-Related Configuration
  - Boot Configuration
  - Clocking Configuration
  - Memory and High-Speed I/O Configuration
  - General Purpose Information
  - Pin Multiplexing Configuration
    - Group A Pin Configuration
    - Group B Pin Configuration
- PBI Data
- PBL Data
  - Offset
  - Output Format

SerDes Protocol Select
Bits 128-133
For additional information see description of the SRDS_PRTCL field in device documentation.
This item modifies SRDS_PRTCL[31:26] bit field in the RCWSRS register.

Figure 5. SerDes protocol selection table

3. You can edit PBI commands using the **PBI Data input** property. To edit PBI commands, select the **PBI Data > PBI Data input** property in the **Name** column on the **Properties** page of **Component Inspector**.
4. Click the **Value** column for the property and click the ... button. The **PBI Data input** page appears, as shown in the figure below.



Select PBI command: CCSR Write (4-byte)

Command parameters

System address (0x): 00000000

Data (0x): 00000000

Use this property to add PBI commands to the PBL image. Comments (#) can be added in Raw mode.
Version specific item: Settings supported only for devices using Reset Configuration Word (RCW).

Figure 6. Editing PBI Data input property

5. Choose the PBI command you want to edit from the **Select PBI command** menu.
6. Edit command parameters in the **Command parameters** group and use the **Add Command** and **Apply** buttons to use the edited command.

NOTE

You can find more details about the supported data types of the PBI commands for PBI data input in PBI commands on page 26.

This section contains the following subsections:

- Specifying custom values on page 9
- Suppression of non-critical errors in PBL settings on page 11

1.1.1.2.1 Specifying custom values

The PBL tool will restrict field values to the ranges specified in the SoC reference manual.

It will also check entered values against known constraints and generate errors in the **Component Inspector** and **Problems** view when constraints are violated. Both of these features can be turned off. Doing so gives you the ability to set any RCW field to any value, even if such a configuration is likely or certain to cause the SoC to not boot out of reset or to function improperly.

To specify a custom value for an RCW field:

1. From the **View** menu in the **Component Inspector**, select the **Ignore Constraints and non-Critical Errors** option.

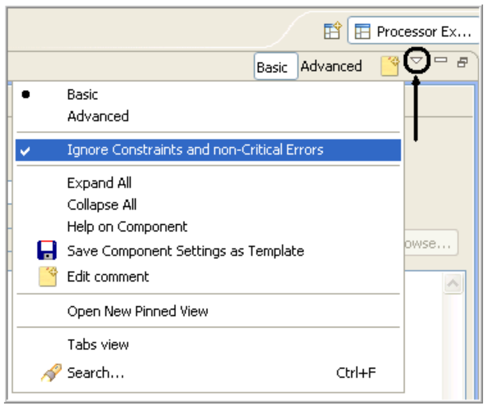
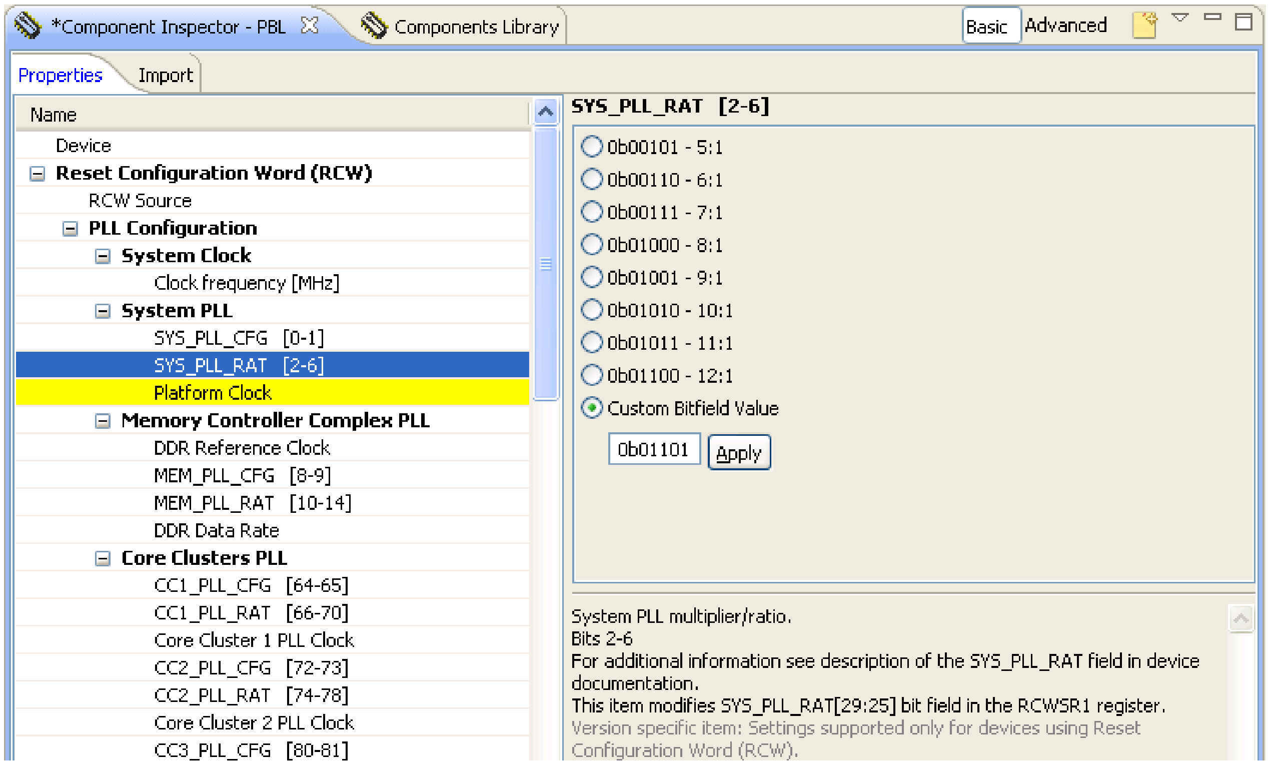


Figure 7. Enabling Ignore Constraints and non-Critical Errors option

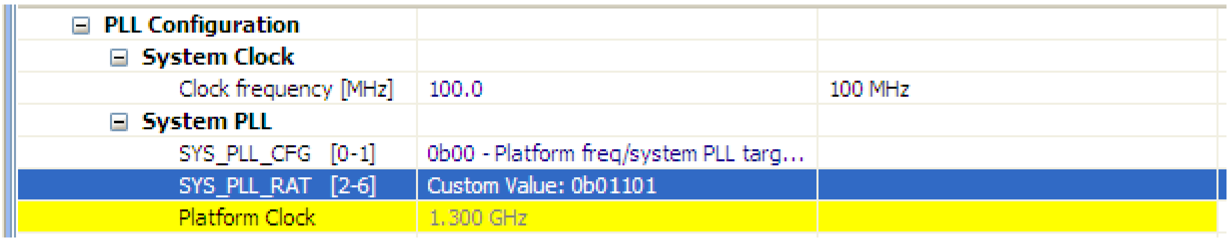
2. Select the required property. For example, *SYS_PLL_RAT [2-6]*.
3. If the field has discrete values, a graphical editor will appear to the right that shows those values and it also shows a **Custom Bitfield Value** field. This field allows you to enter any value (that fits, based on the size of the field).
4. Enter the prefixed custom value in the **Custom Bitfield Value** text box. Prefix with *0b* for binary and *0x* for hex.



System PLL multiplier/ratio.
Bits 2-6
For additional information see description of the SYS_PLL_RAT field in device documentation.
This item modifies SYS_PLL_RAT[29:25] bit field in the RCWSR1 register.
Version specific item: Settings supported only for devices using Reset Configuration Word (RCW).

Figure 8. Setting property in graphical mode

5. Click **Apply**.
6. View the *SYS_PLL_RAT [2-6]* property updated with the custom value. This custom value will be propagated into the PBL output.



PLL Configuration		
System Clock		
Clock frequency [MHz]	100.0	100 MHz
System PLL		
SYS_PLL_CFG [0-1]	0b00 - Platform freq/system PLL targ...	
SYS_PLL_RAT [2-6]	Custom Value: 0b01101	
Platform Clock	1.300 GHz	

Figure 9. SYS_PLL_RAT [2-6] property updated

NOTE

You need to be careful when changing RCW data using custom value. Setting RCW to an unsupported bit value(s) may lead to unpredictable behavior or may break the boot process completely and hang the device.

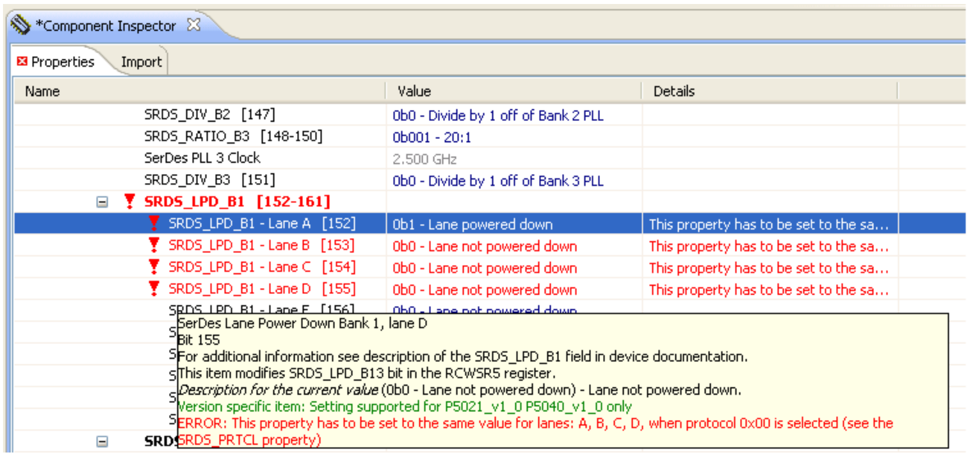
1.1.1.2.2 Suppression of non-critical errors in PBL settings

You can suppress all errors except critical errors in the PBL tool settings.

Critical errors signify contradictory requests for the generated output that prevent the PBL tool from generating code.

To suppress non-critical errors in tool settings, click **View** menu in the **Component Inspector** and select the **Ignore Constraints and Non-critical Errors** option (Figure 7. Enabling Ignore Constraints and non-Critical Errors option on page 10).

For example, to suppress the SerDes lanes power-down setting errors, as shown in the following figure, select **View** > **Ignore Constraints and Non-critical Errors** option in the **Component Inspector** view.

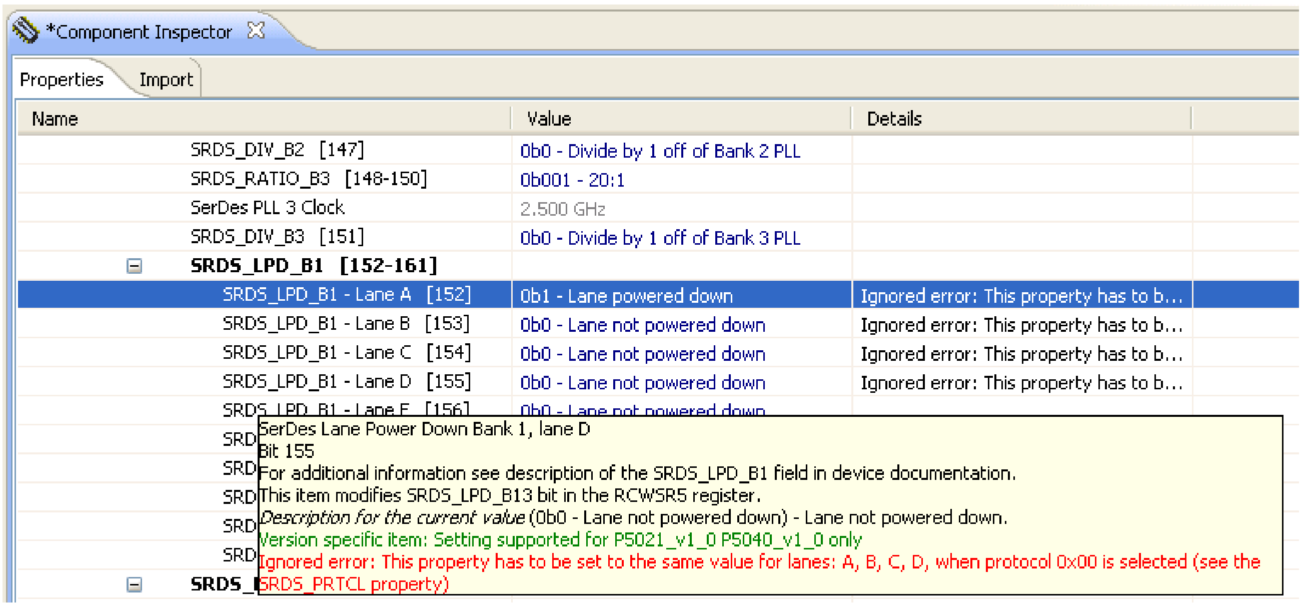


Name	Value	Details
SRDS_DIV_B2 [147]	0b0 - Divide by 1 off of Bank 2 PLL	
SRDS_RATIO_B3 [148-150]	0b001 - 20:1	
SerDes PLL 3 Clock	2.500 GHz	
SRDS_DIV_B3 [151]	0b0 - Divide by 1 off of Bank 3 PLL	
SRDS_LPD_B1 [152-161]		
SRDS_LPD_B1 - Lane A [152]	0b1 - Lane powered down	This property has to be set to the sa...
SRDS_LPD_B1 - Lane B [153]	0b0 - Lane not powered down	This property has to be set to the sa...
SRDS_LPD_B1 - Lane C [154]	0b0 - Lane not powered down	This property has to be set to the sa...
SRDS_LPD_B1 - Lane D [155]	0b0 - Lane not powered down	This property has to be set to the sa...
SRDS_LPD_B1 - Lane E [156]	0b0 - Lane not powered down	

SerDes Lane Power Down Bank 1, lane D
Bit 155
For additional information see description of the SRDS_LPD_B1 field in device documentation.
This item modifies SRDS_LPD_B13 bit in the RCWSR5 register.
Description for the current value (0b0 - Lane not powered down) - Lane not powered down.
Version specific item: Setting supported for P5021_v1_0 P5040_v1_0 only
ERROR: This property has to be set to the same value for lanes: A, B, C, D, when protocol 0x00 is selected (see the SRDS_PRTCL property)

Figure 10. Suppressing SerDes lanes power-down settings errors

The error messages change to warnings as shown in the figure below.



Name	Value	Details
SRDS_DIV_B2 [147]	0b0 - Divide by 1 off of Bank 2 PLL	
SRDS_RATIO_B3 [148-150]	0b001 - 20:1	
SerDes PLL 3 Clock	2.500 GHz	
SRDS_DIV_B3 [151]	0b0 - Divide by 1 off of Bank 3 PLL	
SRDS_LPD_B1 [152-161]		
SRDS_LPD_B1 - Lane A [152]	0b1 - Lane powered down	Ignored error: This property has to b...
SRDS_LPD_B1 - Lane B [153]	0b0 - Lane not powered down	Ignored error: This property has to b...
SRDS_LPD_B1 - Lane C [154]	0b0 - Lane not powered down	Ignored error: This property has to b...
SRDS_LPD_B1 - Lane D [155]	0b0 - Lane not powered down	Ignored error: This property has to b...
SRDS_LPD_B1 - Lane E [156]	0b0 - Lane not powered down	
SRD		
SRD		
SRD		
SRD		
SRD		
SRDS_		

SerDes Lane Power Down Bank 1, lane D
Bit 155
For additional information see description of the SRDS_LPD_B1 field in device documentation.
This item modifies SRDS_LPD_B13 bit in the RCWSR5 register.
Description for the current value (0b0 - Lane not powered down) - Lane not powered down.
Version specific item: Setting supported for P5021_v1_0 P5040_v1_0 only
Ignored error: This property has to be set to the same value for lanes: A, B, C, D, when protocol 0x00 is selected (see the SRDS_PRTCL property)

Figure 11. SerDes lanes power-down warning message

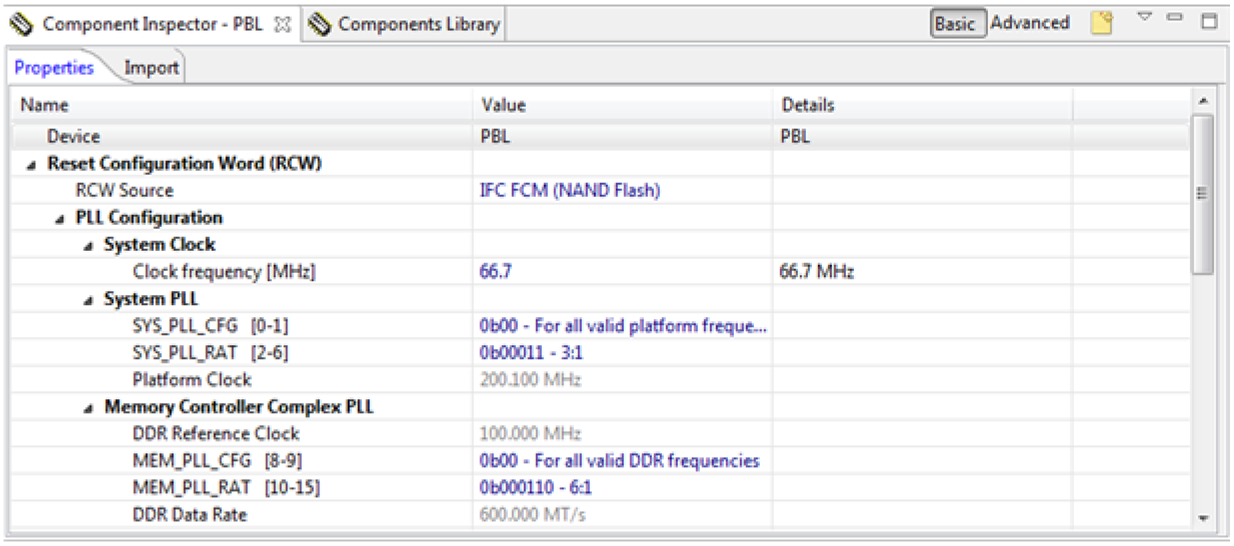
1.1.2 Generating output from PBL component

The PBL configuration tool generates pre-boot initialization data that encapsulates into a PBL image file.

NOTE

In QCVS, **Generate Processor Expert Code** option does not generate code for a PBL component. For other components in QCVS, it does.

After you have configured the RCW and/or PBI data, you can produce a file that contains the same configuration. The most commonly used output format is Binary, which is an image suitable for storing into a non-volatile memory device.



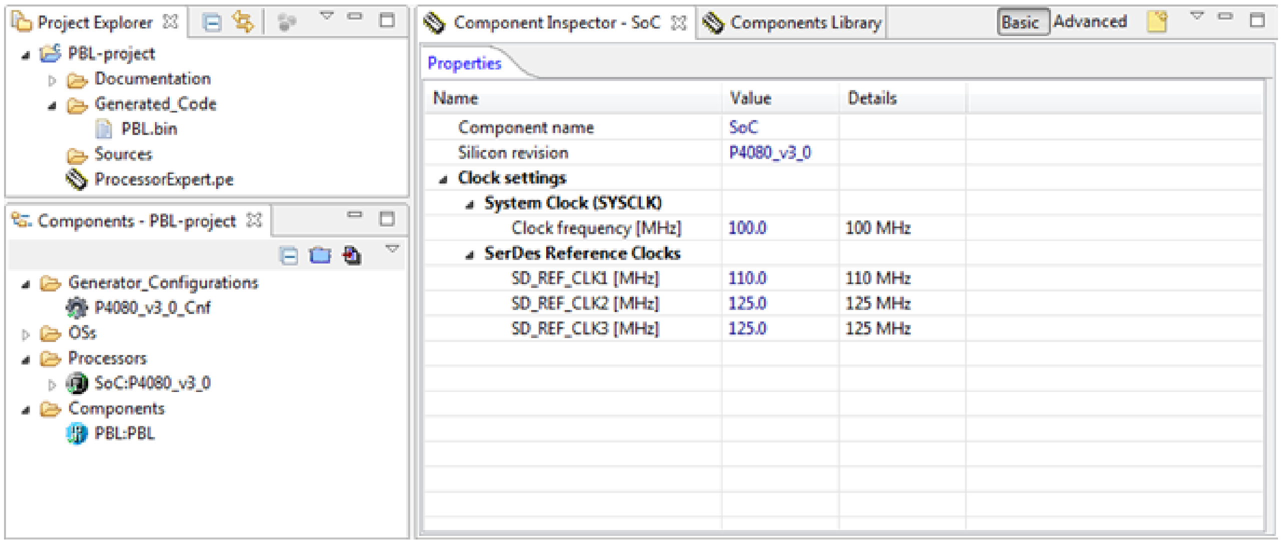
Name	Value	Details
Device	PBL	PBL
Reset Configuration Word (RCW)		
RCW Source	IFC FCM (NAND Flash)	
PLL Configuration		
System Clock		
Clock frequency [MHz]	66.7	66.7 MHz
System PLL		
SYS_PLL_CFG [0-1]	0b00 - For all valid platform freque...	
SYS_PLL_RAT [2-6]	0b00011 - 3:1	
Platform Clock	200.100 MHz	
Memory Controller Complex PLL		
DDR Reference Clock	100.000 MHz	
MEM_PLL_CFG [8-9]	0b00 - For all valid DDR frequencies	
MEM_PLL_RAT [10-15]	0b000110 - 6:1	
DDR Data Rate	600.000 MT/s	

Figure 12. Configuration page for QorIQ LS series processors

The format of the output generated is based on the properties set in the **Component Inspector** view. Output is generated only by request.

Perform any of the following steps to generate output:

- Click the **Generate Processor Expert Code** toolbar button in the **Components** view as shown in the following figure.



Name	Value	Details
Component name	SoC	
Silicon revision	P4080_v3_0	
Clock settings		
System Clock (SYSCLK)		
Clock frequency [MHz]	100.0	100 MHz
SerDes Reference Clocks		
SD_REF_CLK1 [MHz]	110.0	110 MHz
SD_REF_CLK2 [MHz]	125.0	125 MHz
SD_REF_CLK3 [MHz]	125.0	125 MHz

Figure 13. Generate Processor Expert Code icon

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

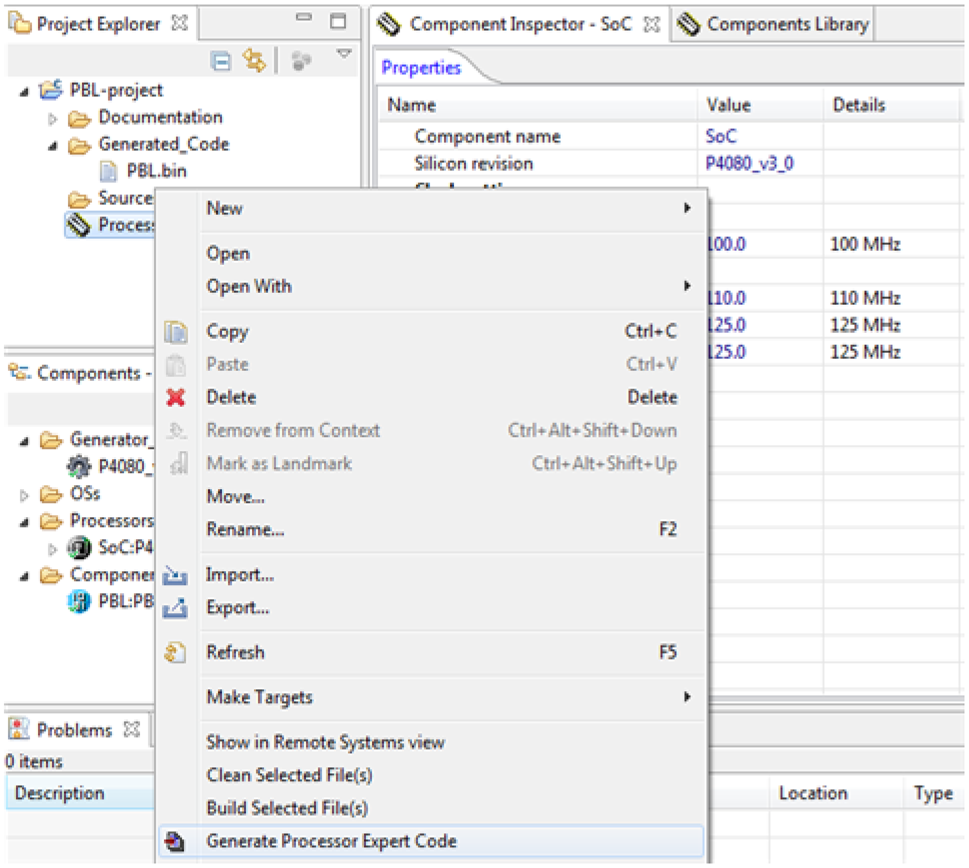


Figure 14. Generate Processor Expert code

- Select the **Project > Generate Processor Expert Code** option from the Eclipse IDE menu.

The **Generating code** dialog appears.

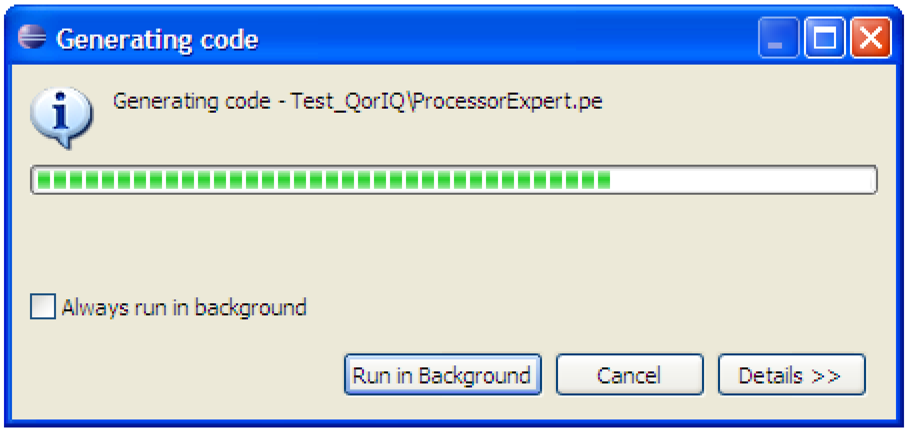


Figure 15. Code generation

The output file is generated and added to the **Project Explorer** in the **Generated_Code** folder. The output file is automatically named after the PBL component. You can change the name of the component, but not the output file directly. The name of the component appears in the **Components** view, as shown in the figure below.

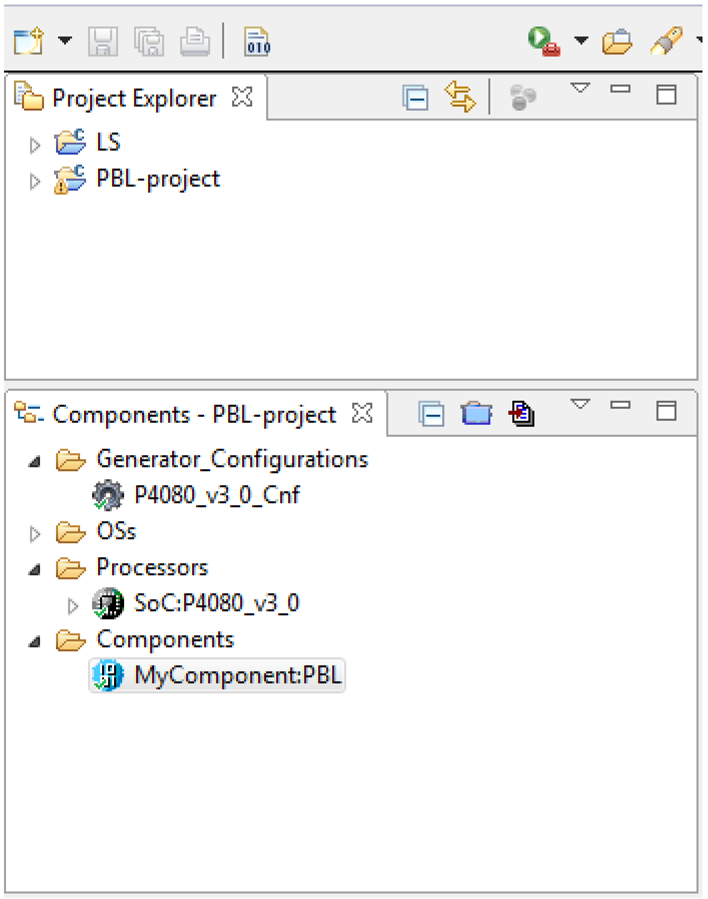
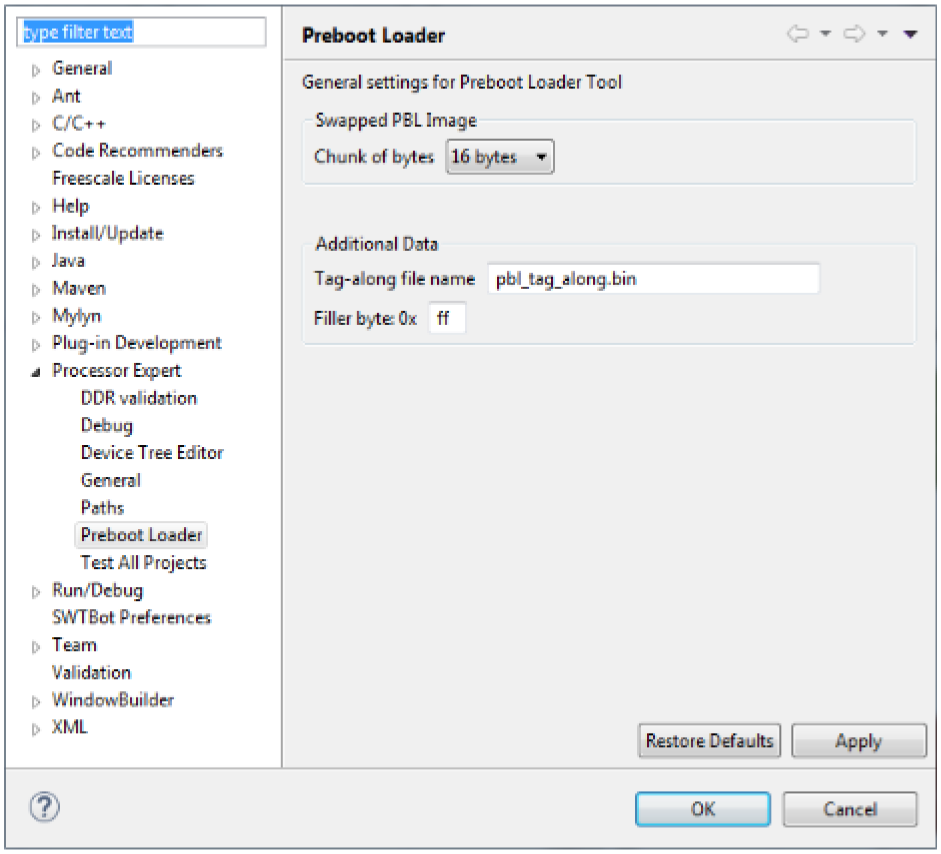


Figure 16. PBL generated code file in Project Explorer

The PBL tool generates a swapped PBL image each time the RCW source is set as QSPI or Serial NOR. As you can see in Figure 12. Configuration page for QorIQ LS series processors on page 12, the RCW source displays in the main tree and can be changed from there. The size of the swapped chunks (part of the final image) can be configured through a system property, **Chunk of bytes**. This property is set per Eclipse instance and can be found using the following steps:

1. Choose **Window > Preferences**. The **Preferences** dialog appears.
2. Expand the **Processor Expert** node and click **Preboot Loader**. The **Preboot Loader** page appears displaying the **Chunk of bytes** property, as shown in the figure below.



Preboot Loader

General settings for Preboot Loader Tool

Swapped PBL Image

Chunk of bytes: 16 bytes

Additional Data

Tag-along file name: pbl_tag_along.bin

Filler byte: 0x ff

Figure 17. PBL setting for swapped PBL image

The **Chunk of bytes** property indicates in how many bytes the final PBL image will be split before the effective swapping process. The generated binary file is found under the same directory, but has a different file name, `PBL_swapped.bin`. Note that this feature only works for binary format.

1.1.3 Importing PBL configuration

You can configure PBL data by importing a configuration from an existing PBL image file, available in several supported formats.

The following subsections explain importing PBL configuration with some use cases:

- Importing PBL configuration during project creation on page 15
- Importing PBL configuration for existing PBL project on page 16

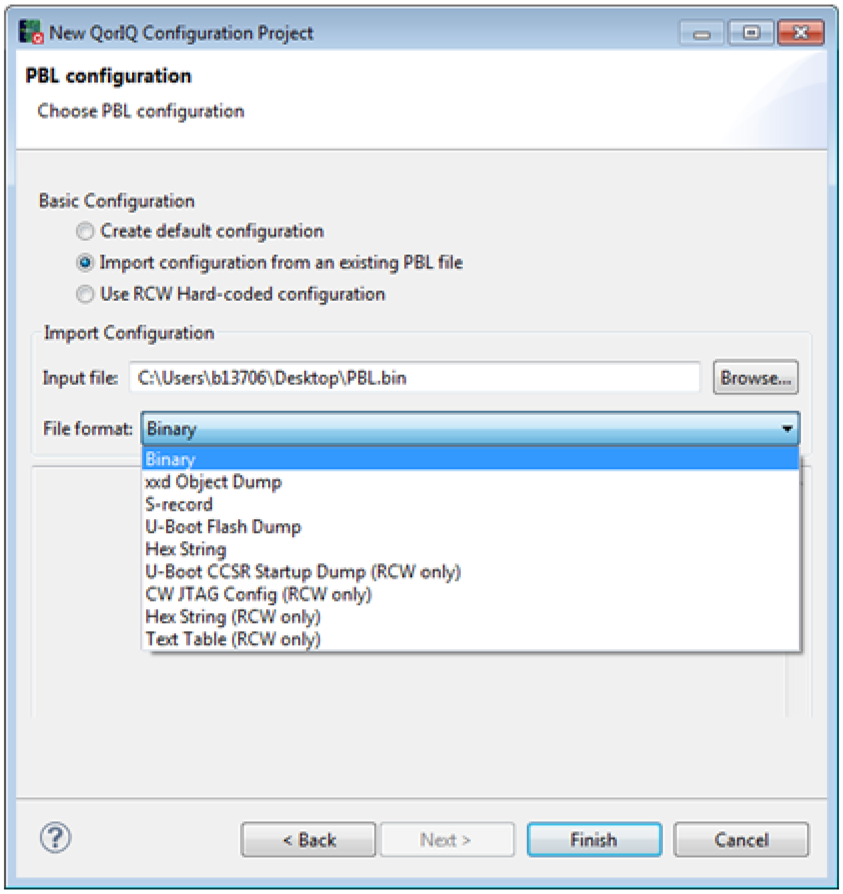
1.1.3.1 Importing PBL configuration during project creation

This section explains how to import a PBL configuration during project creation.

Perform the following steps to import a PBL configuration as part of project creation:

1. In the **PBL Configuration** page of the **New QorIQ Configuration Project** Wizard, select the **Import configuration from an existing PBL file** option.

2. Specify the file to be imported in the **Input file** text box. Use the **Browse** button to locate the file.
3. Choose the format of the file you are importing from the **File format** list.



New QorIQ Configuration Project

PBL configuration

Choose PBL configuration

Basic Configuration
- Create default configuration
- (●) Import configuration from an existing PBL file
- Use RCW Hard-coded configuration

Import Configuration

Input file: C:\Users\b13706\Desktop\PBL.bin

File format: Binary

Figure 18. Selecting input file - PBL configuration

4. Click **Finish**.

The PBL project is created using imported configuration.

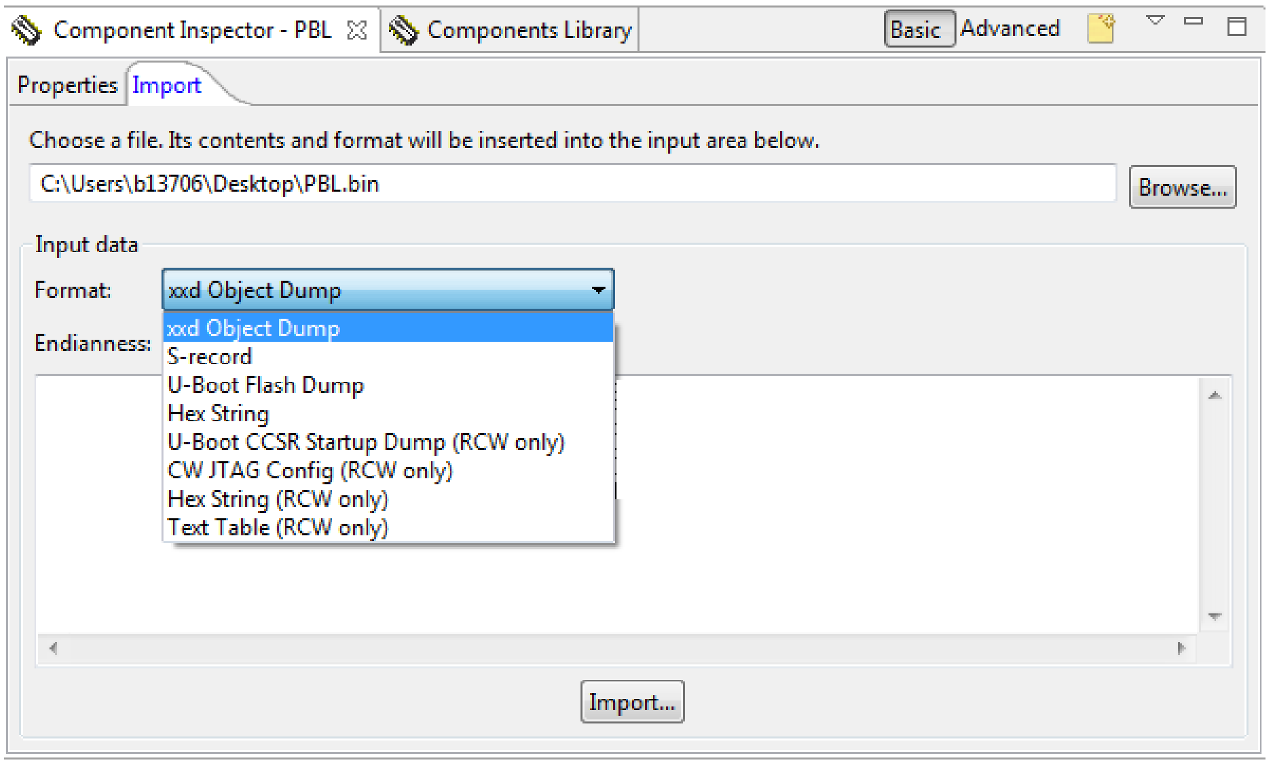
1.1.3.2 Importing PBL configuration for existing PBL project

This section explains how to import a PBL configuration into an existing PBL component.

All the settings in the imported configuration are applied to the component. In other words, the existing configuration is replaced. You can import from a file or use copy and paste. In either case, you can manually modify the incoming data before applying it to the component.

Following are the steps to import a PBL image:

1. Open the **Component Inspector** view for PBL component in the existing project.
2. Select the **Import** tab in the **Component Inspector** view. The following fields are available:
   - **Input File** – Specifies the input configuration file for import
   - **Format** – Specifies the input configuration file format
   - **Input Data** – Displays the content of the selected input file



Choose a file. Its contents and format will be inserted into the input area below.

C:\Users\b13706\Desktop\PBL.bin

Input data

Format: xxd Object Dump

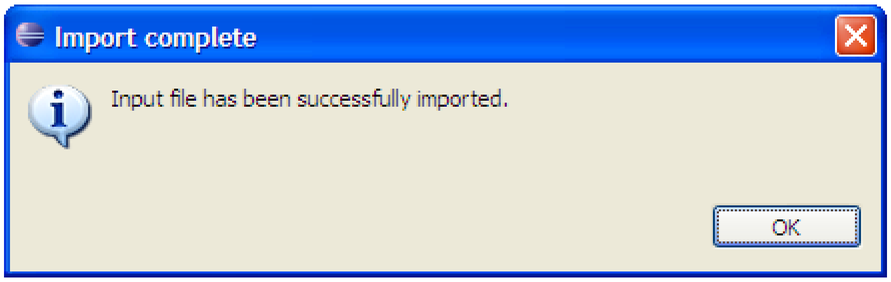
Endianness:

- xxd Object Dump
- S-record
- U-Boot Flash Dump
- Hex String
- U-Boot CCSR Startup Dump (RCW only)
- CW JTAG Config (RCW only)
- Hex String (RCW only)
- Text Table (RCW only)

Figure 19. Import tab - Specifying import parameters

3. Either specify the file to be imported, in the **Input File** text box, or put the input data directly into the **Input data** text box (by copying and pasting from another source).
4. Specify the format of the file in the **Input Format** field.
5. Click **Import**.

   The data is imported and the **Import Complete** dialog appears.

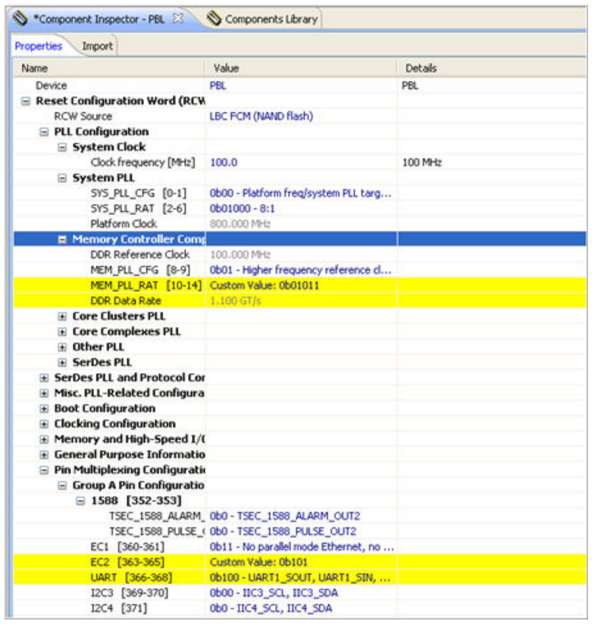


Input file has been successfully imported.

Figure 20. Import complete dialog

6. Click **OK**.

   The changed properties are highlighted as shown in the figure below.



Name	Value	Details
Device	PBL	PBL
Reset Configuration Word (RCW		
RCW Source	LBC FCM (NAND flash)	
PLL Configuration		
System Clock		
Clock frequency [MHz]	100.0	100 MHz
System PLL		
SYS_PLL_CFG [0-1]	0b00 - Platform freq/system PLL targ...	
SYS_PLL_RAT [2-6]	0b01000 - 8:1	
Platform Clock	800.000 MHz	
Memory Controller Comp		
DDR Reference Clock	100.000 MHz	
MEM_PLL_CFG [8-9]	0b01 - Higher frequency reference cl...	
MEM_PLL_RAT [10-14]	Custom Value: 0b01011	
DDR Data Rate	1.100 GT/s	
Core Clusters PLL		
Core Complexes PLL		
Other PLL		
SerDes PLL		
SerDes PLL and Protocol Cor		
Misc. PLL-Related Configura		
Boot Configuration		
Clocking Configuration		
Memory and High-Speed I/(		
General Purpose Informatio		
Pin Multiplexing Configurati		
Group A Pin Configuratio		
1588 [352-353]		
TSEC_1588_ALARM_	0b0 - TSEC_1588_ALARM_OUT2	
TSEC_1588_PULSE_(	0b0 - TSEC_1588_PULSE_OUT2	
EC1 [360-361]	0b11 - No parallel mode Ethernet, no ...	
EC2 [363-365]	Custom Value: 0b101	
UART [366-368]	0b100 - UART1_SOUT, UART1_SIN, ...	
I2C3 [369-370]	0b00 - IIC3_SCL, IIC3_SDA	
I2C4 [371]	0b0 - IIC4_SCL, IIC4_SDA	

Figure 21. Changed properties

This is how you import a PBL image.

This section contains the following subsections:

- Importing PBL configuration from file on page 18
- Importing PBL configuration from target on page 18

1.1.3.2.1 Importing PBL configuration from file

This section explains how to import PBL configuration from a file.

This section contains the following subsection:

- Importing swapped PBL images on page 18

1.1.3.2.1.1 Importing swapped PBL images

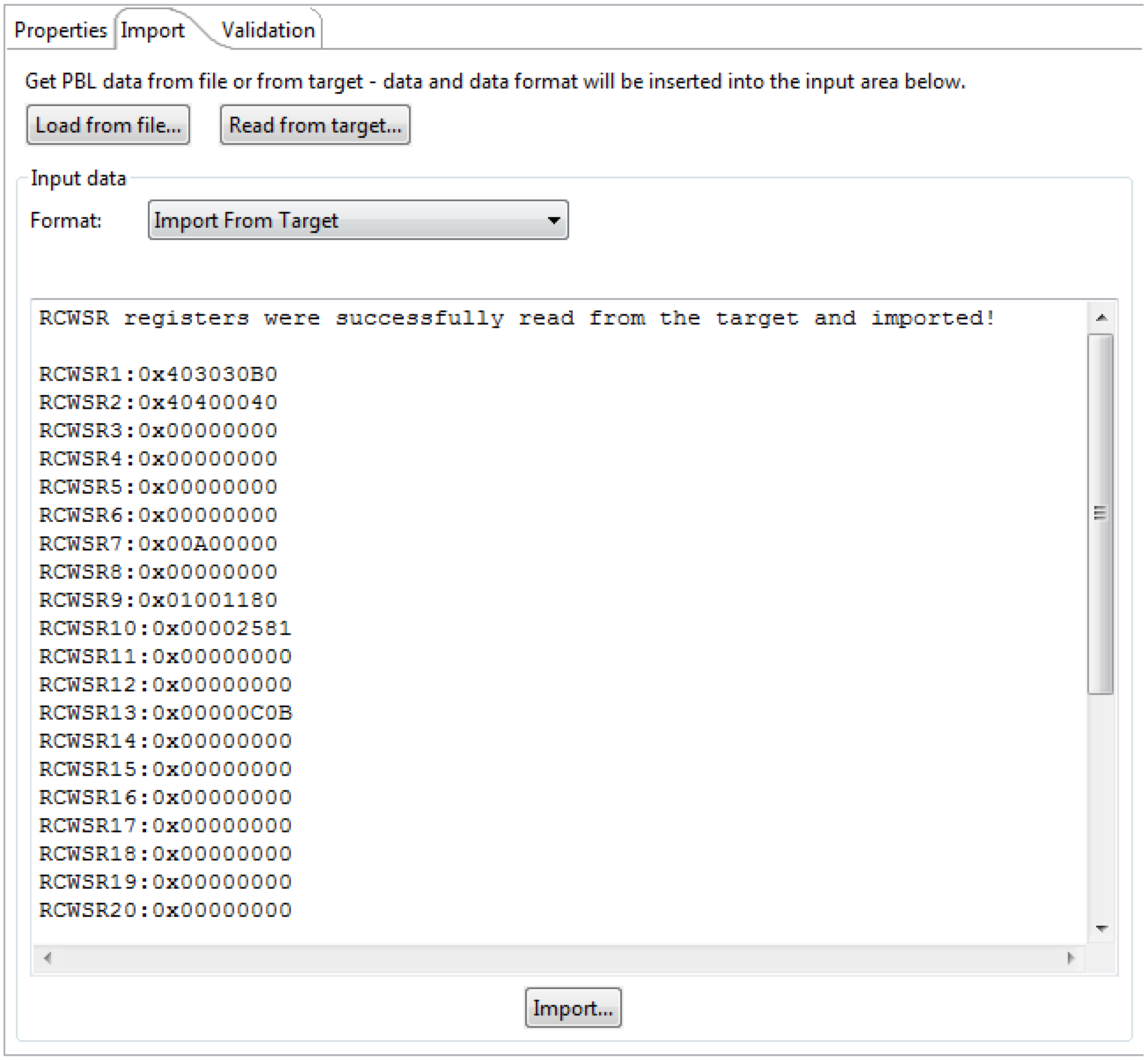
This section explains importing swapped PBL images.

The PBL tool allows you to import swapped PBL images. In this case, the size of a swapped chunk is calculated automatically and it can be 8, 16, 32, or 64 bytes. This type of PBL image files are imported as binary files, using the same user interface used to import other types of PBL image files.

1.1.3.2.2 Importing PBL configuration from target

This section explains how to import PBL configuration from the target.

The PBL tool allows you to import the RCWSR registers from the target. The import operation can only be performed if the active target connection is working. If the RCWSR registers are imported successfully from the target, then the active PBL component is populated automatically and the RCWSR values that were read from the board are displayed in the Component Inspector view, as shown in the figure below.



```
RCWSR registers were successfully read from the target and imported!

RCWSR1:0x403030B0
RCWSR2:0x40400040
RCWSR3:0x00000000
RCWSR4:0x00000000
RCWSR5:0x00000000
RCWSR6:0x00000000
RCWSR7:0x00A00000
RCWSR8:0x00000000
RCWSR9:0x01001180
RCWSR10:0x00002581
RCWSR11:0x00000000
RCWSR12:0x00000000
RCWSR13:0x00000C0B
RCWSR14:0x00000000
RCWSR15:0x00000000
RCWSR16:0x00000000
RCWSR17:0x00000000
RCWSR18:0x00000000
RCWSR19:0x00000000
RCWSR20:0x00000000
```

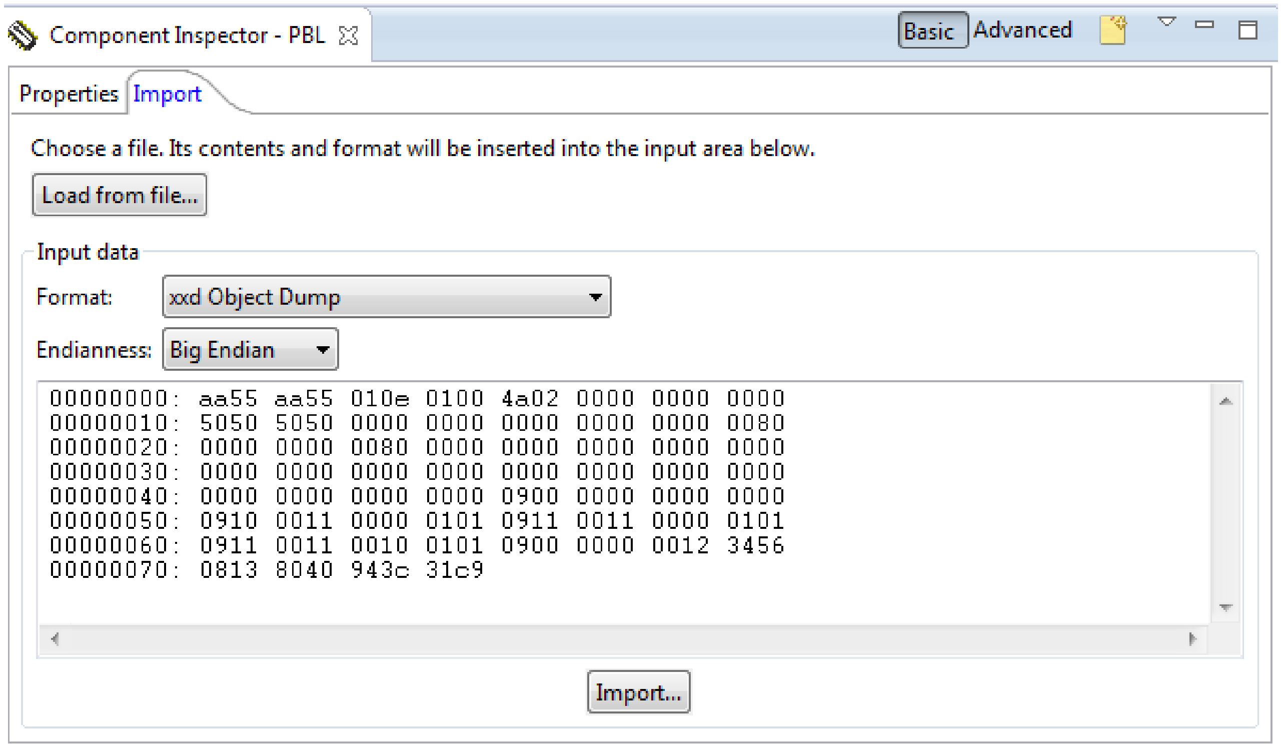
Figure 22. Registers imported from target

1.1.4 PBL component import file types

This section describes the PBL input file formats supported by the PBL tool import functionality.

The various PBL formats available for the import configuration are shown in the following figures.

- xxd Object Dump: This is a hex dump of a full PBL image (includes the preamble, RCW, and any PBI commands). The hex dump can come from any number of sources, for example, U-Boot's xd command, or the 'hex dump' Linux command. For this format, you can specify the endianness of the data, which is meaningful only when the data is organized in multi-byte words. The endianess option is automatically set to the endianness of the general purpose processor.


Component Inspector - PBL
Basic
Advanced
Properties
Import
Choose a file. Its contents and format will be inserted into the input area below.
Load from file...
Input data
Import...

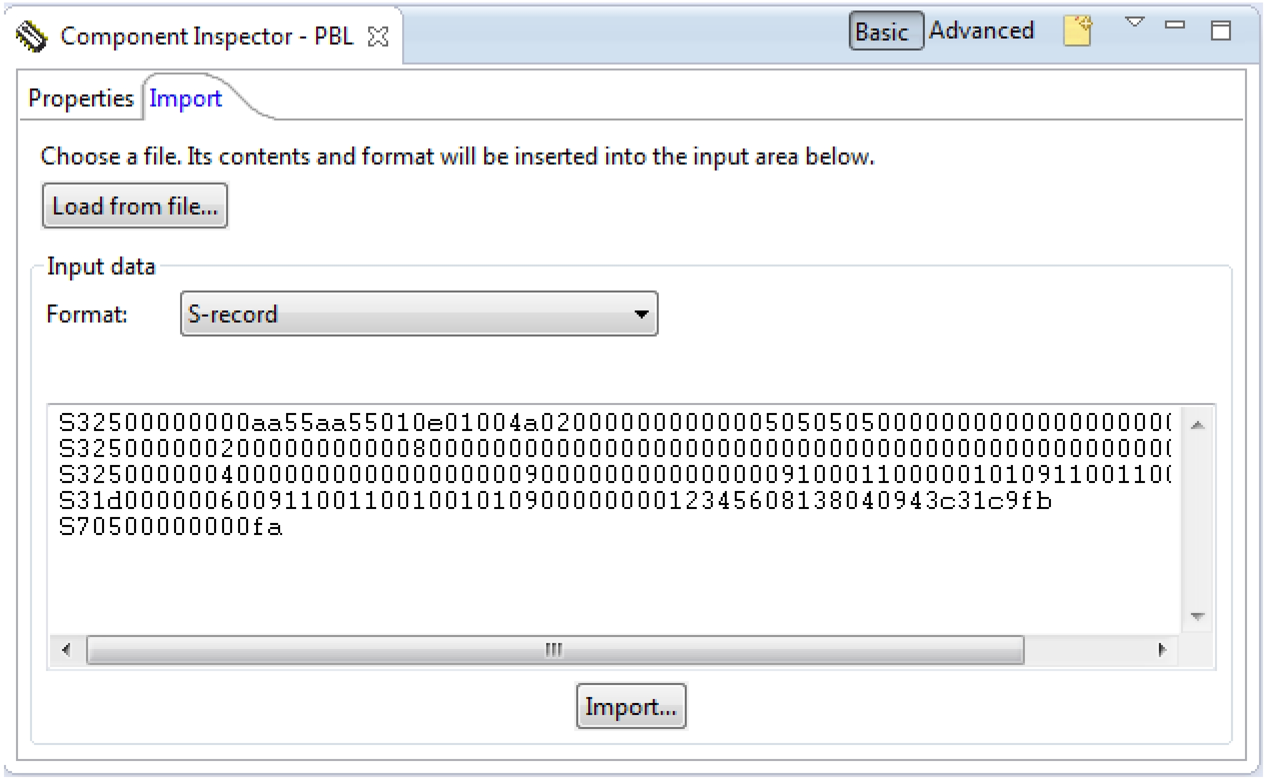
Format: xxd Object Dump

Endianness: Big Endian

```
00000000: aa55 aa55 010e 0100 4a02 0000 0000 0000
00000010: 5050 5050 0000 0000 0000 0000 0000 0080
00000020: 0000 0000 0080 0000 0000 0000 0000 0000
00000030: 0000 0000 0000 0000 0000 0000 0000 0000
00000040: 0000 0000 0000 0000 0900 0000 0000 0000
00000050: 0910 0011 0000 0101 0911 0011 0000 0101
00000060: 0911 0011 0010 0101 0900 0000 0012 3456
00000070: 0813 8040 943c 31c9
```

Figure 23. xxd Object Dump format

- S-record: You can find detailed information about the Motorola S-record format in *MC68000 Family Programmer's Reference Manual* at: http://www.freescale.com/files/archives/doc/ref_manual/M68000PRM.pdf.

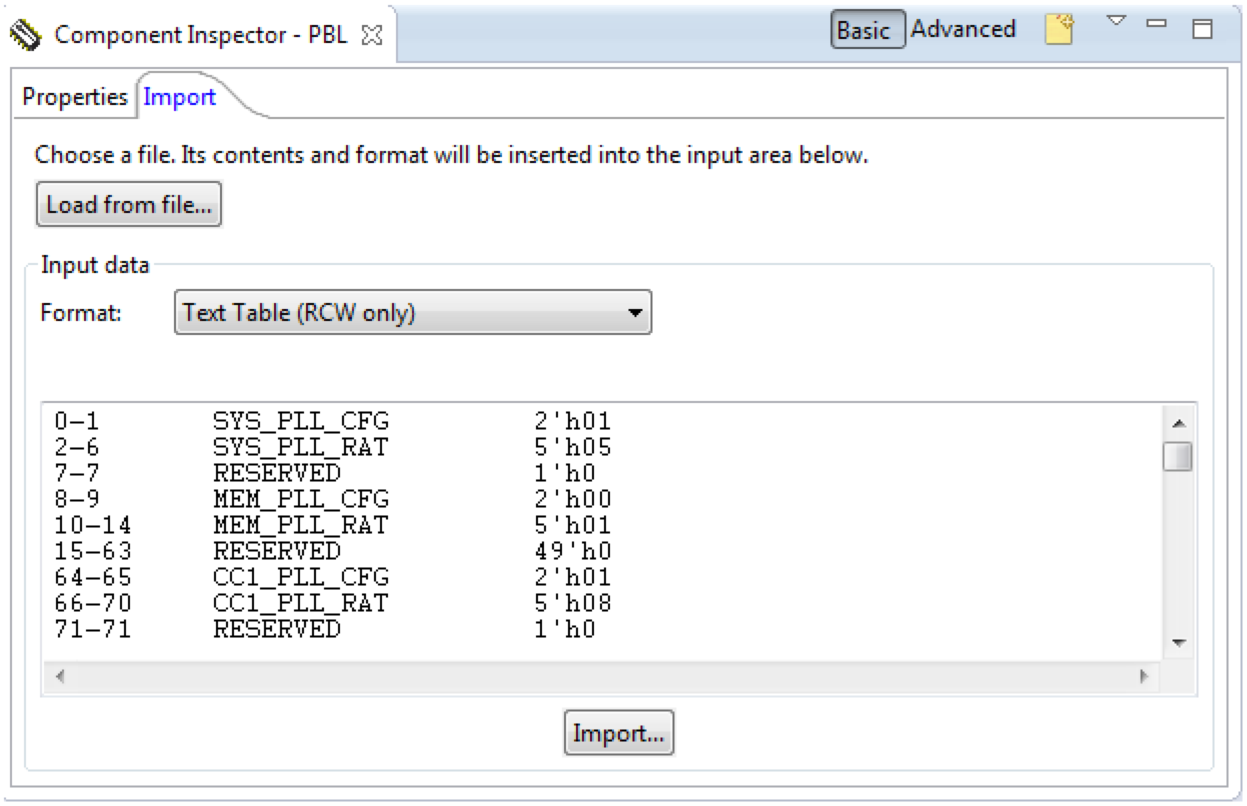

Component Inspector - PBL
Basic
Advanced
Properties
Import
Choose a file. Its contents and format will be inserted into the input area below.
Load from file...
Input data
Import...

Format: S-record

```
S32500000000aa55aa55010e01004a020000000000005050505000000000000000000(
S325000000200000000000800000000000000000000000000000000000000000000000(
S32500000040000000000000000009000000000000000910001100000101091100110(
S31d000000600911001100100101090000000012345608138040943c31c9fb
S70500000000fa
```

Figure 24. S-record format

- Text Table (RCW only) format: This file format is directly generated from the PBL tool and supports user comments within the data. The format lists every RCW field and its value in a descriptive, human readable form. This format is particularly useful when you want to compare two RCW definitions, the text differentiator tool can be used to easily identify the fields that differ between the two definitions. Because the PBL tool supports both generating and importing this format, this is a useful format when revisioning RCWs, as they can be stored in a Software Configuration Management System (such as GIT or CVS) and revisions can be compared.



Choose a file. Its contents and format will be inserted into the input area below.

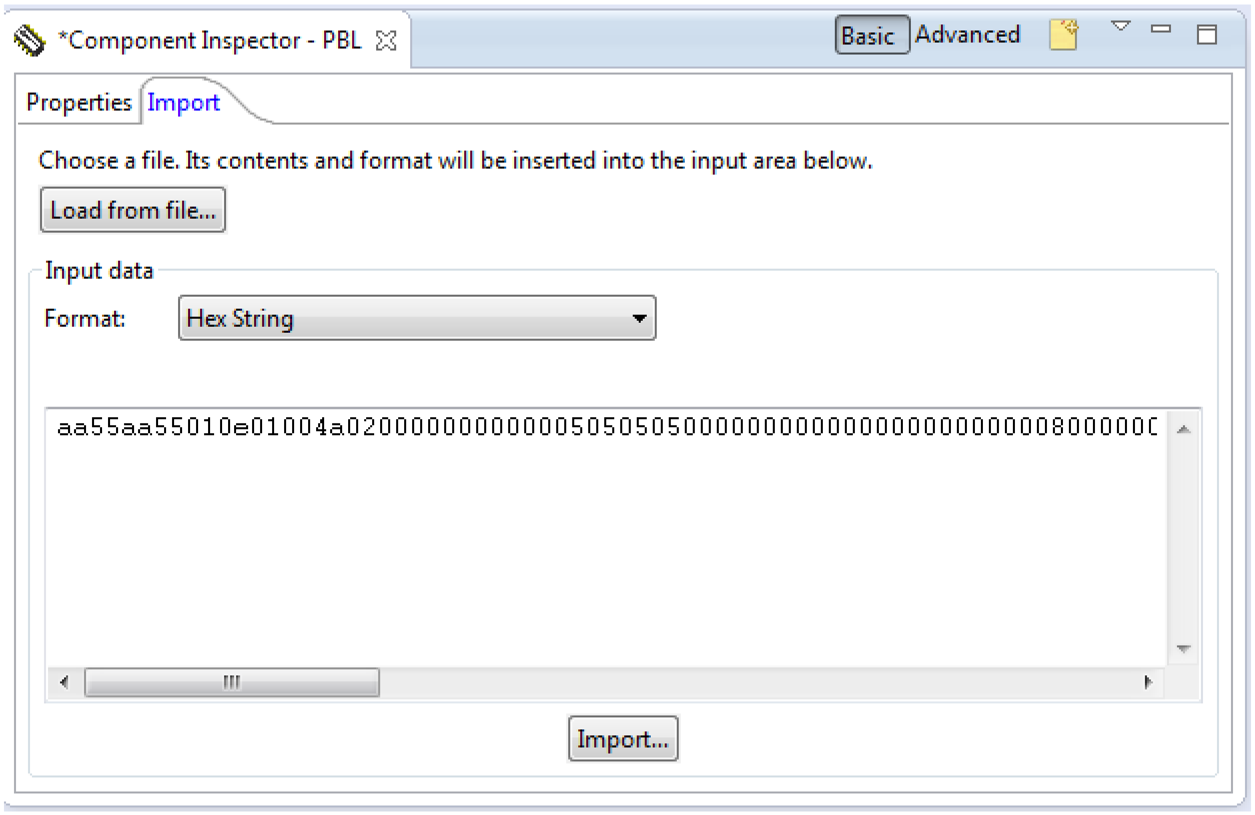
Input data

Format: Text Table (RCW only)

```
0-1         SYS_PLL_CFG           2'h01
2-6         SYS_PLL_RAT           5'h05
7-7         RESERVED              1'h0
8-9         MEM_PLL_CFG           2'h00
10-14       MEM_PLL_RAT           5'h01
15-63       RESERVED              49'h0
64-65       CC1_PLL_CFG           2'h01
66-70       CC1_PLL_RAT           5'h08
71-71       RESERVED              1'h0
```

Figure 25. Text Table (RCW only) format

- Hex String: This format renders each byte of the data in hexadecimal. It contains the entire PBL image, from the preamble to the PBI commands.



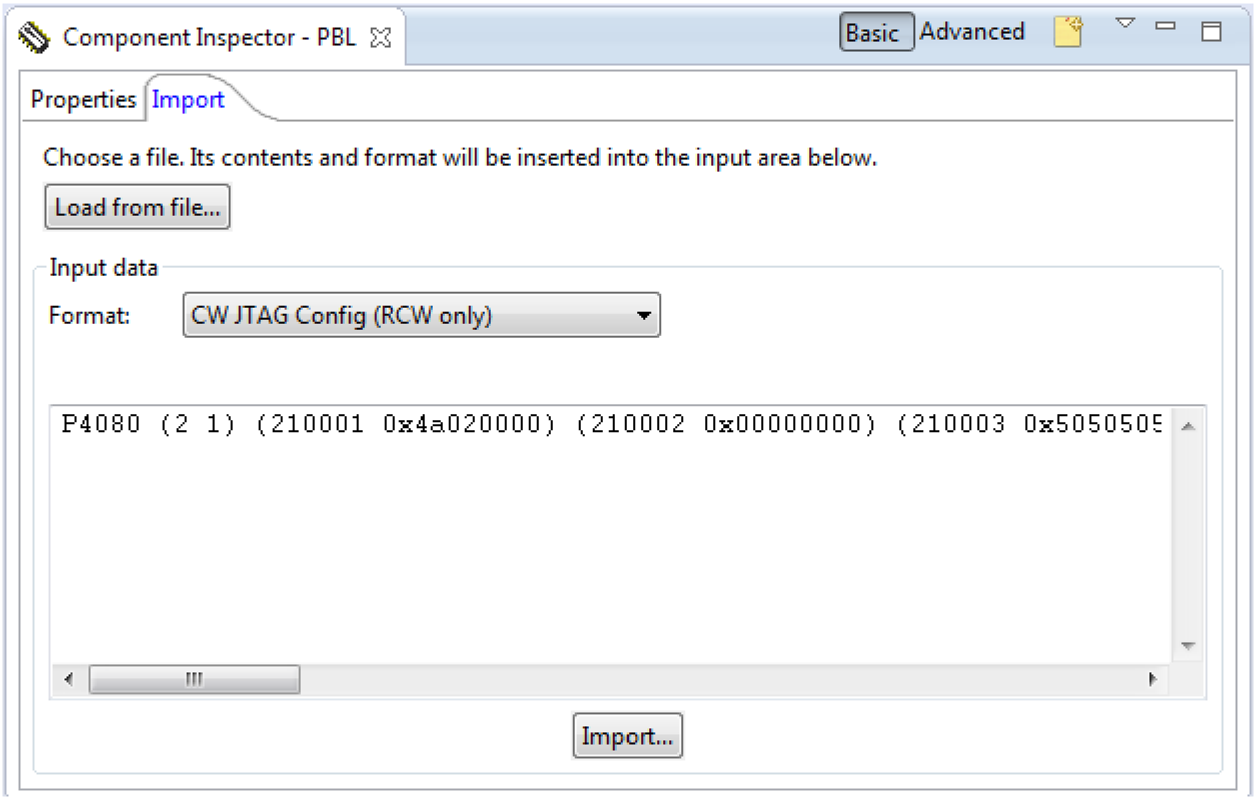
Choose a file. Its contents and format will be inserted into the input area below.

Format: Hex String

```
aa55aa55010e01004a020000000000005050505000000000000000000000000008000000
```

Figure 26. Hex String format

- CW JTAG Config (RCW only): This file format is used with the CodeWarrior debugger to override the target's RCW out of reset. For QorIQ LS series processors, hardcoded RCW value can be used in CodeWarrior JTAG file. This eliminates the need to have all RCW registers in the JTAG file. With this new functionality, the PBL tool is now able to import newer file formats exposed by CodeWarrior.



Choose a file. Its contents and format will be inserted into the input area below.

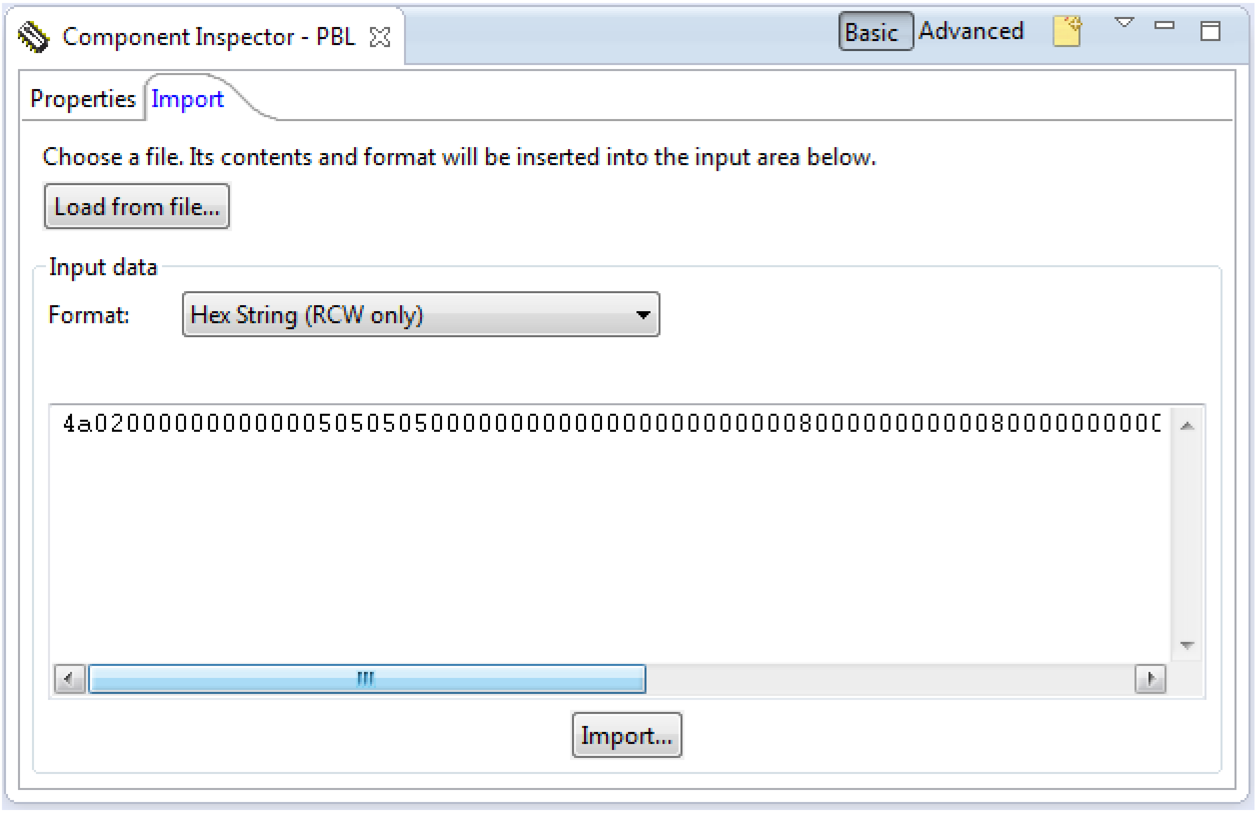
Input data

Format: CW JTAG Config (RCW only)

```
P4080 (2 1) (210001 0x4a020000) (210002 0x00000000) (210003 0x5050505
```

Figure 27. CW JTAG Config (RCW only) format

- Hex String (RCW only): This format renders each byte of the RCW data in hexadecimal. It contains only the RCW bits.



Choose a file. Its contents and format will be inserted into the input area below.

Input data

Format: Hex String (RCW only)

```
4a0200000000000050505050000000000000000000000000800000000000008000000000C
```

Figure 28. Hex String (RCW only) format

- Binary: A file in this format is what's stored in the target's non-volatile memory and what is used by the SoC out of reset.

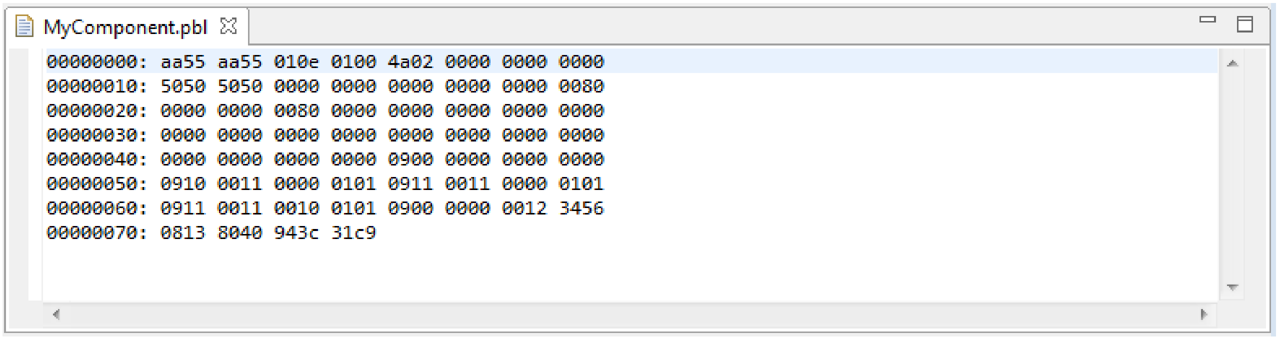
1.1.5 PBL component output file types

This section describes the PBL output file formats supported by the PBL tool code generation.

NOTE

PBL supports different set of input and output file formats although some of the file formats are supported for both.

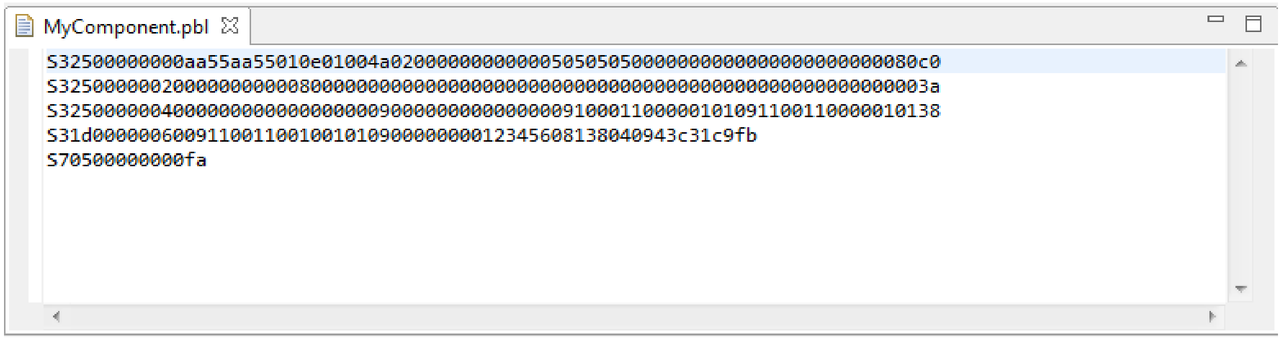
- xxd Object Dump: This is a hex dump of a full PBL image (includes the preamble, RCW, and PBI commands).



```
00000000: aa55 aa55 010e 0100 4a02 0000 0000 0000
00000010: 5050 5050 0000 0000 0000 0000 0000 0080
00000020: 0000 0000 0080 0000 0000 0000 0000 0000
00000030: 0000 0000 0000 0000 0000 0000 0000 0000
00000040: 0000 0000 0000 0000 0900 0000 0000 0000
00000050: 0910 0011 0000 0101 0911 0011 0000 0101
00000060: 0911 0011 0010 0101 0900 0000 0012 3456
00000070: 0813 8040 943c 31c9
```

Figure 29. xxd Object Dump format

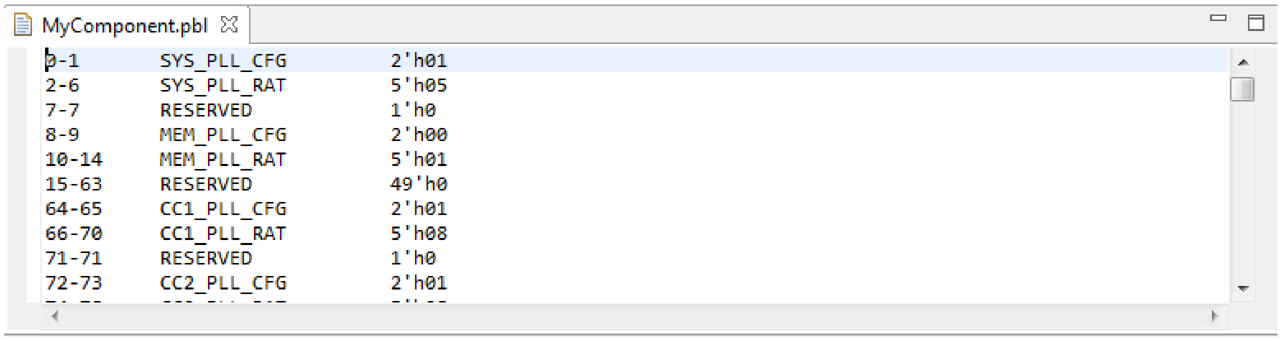
- S-Record: You can find detailed information about the Motorola S-record format in *MC68000 Family Programmer's Reference Manual* at http://www.freescale.com/files/archives/doc/ref_manual/M68000PRM.pdf. This file format is directly supported for PBL input also.



```
S32500000000aa55aa55010e01004a0200000000000050505050000000000000000000000080c0
S32500000020000000000080000000000000000000000000000000000000000000000000000003a
S32500000040000000000000000009000000000000000910001100000101091100110000010138
S31d000000600911001100100101090000000012345608138040943c31c9fb
S70500000000fa
```

Figure 30. S-Record format

- Text Table (RCW only): The format lists all RCW fields and their values in a descriptive, human readable form. This format is useful when you want to compare two RCW definitions. A text differentiator tool can be used to identify the fields that differ between the two definitions. Because the PBL tool supports both generating and importing this format, this is a useful format when revisioning RCWs, as they can be stored in a Software Configuration Management System (such as GIT or CVS) and revisions can be compared.

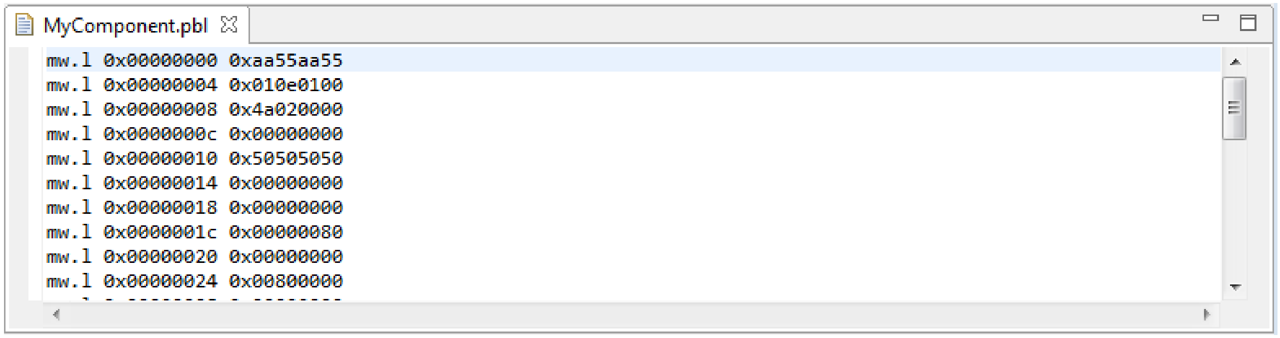


MyComponent.pbl

```
0-1      SYS_PLL_CFG     2'h01
2-6      SYS_PLL_RAT     5'h05
7-7      RESERVED        1'h0
8-9      MEM_PLL_CFG     2'h00
10-14    MEM_PLL_RAT     5'h01
15-63    RESERVED        49'h0
64-65    CC1_PLL_CFG     2'h01
66-70    CC1_PLL_RAT     5'h08
71-71    RESERVED        1'h0
72-73    CC2_PLL_CFG     2'h01
```

Figure 31. Text Table (RCW only) format

- U-Boot Commands: This format is a sequence of U-Boot `mw.l` commands that will write the entire PBL image (preamble, RCW and PBI commands). This format is used to put a PBL image on the board in two steps. The first step is to write the image to RAM; the second is to copy the data from RAM to flash memory, making use of U-Boot's flashing capabilities built into its copy command.

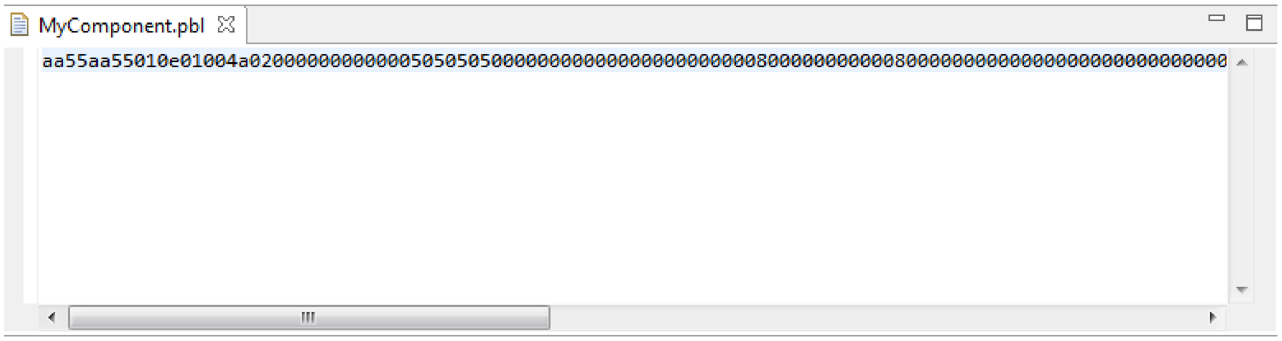


MyComponent.pbl

```
mw.l 0x00000000 0xaa55aa55
mw.l 0x00000004 0x010e0100
mw.l 0x00000008 0x4a020000
mw.l 0x0000000c 0x00000000
mw.l 0x00000010 0x50505050
mw.l 0x00000014 0x00000000
mw.l 0x00000018 0x00000000
mw.l 0x0000001c 0x00000080
mw.l 0x00000020 0x00000000
mw.l 0x00000024 0x00800000
```

Figure 32. U-Boot Commands file format

- Hex String: This format renders each byte of the data in hex. It contains the entire PBL image, from the preamble to the PBI commands.

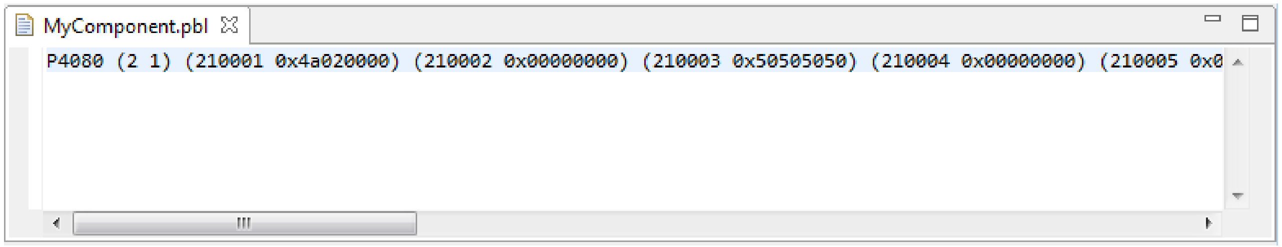


MyComponent.pbl

```
aa55aa55010e01004a020000000000005050505000000000000000000000008000000000008000000000000000000000000000
```

Figure 33. Hex String format

- CW JTAG Config (RCW only): Files of this format are distributed with CodeWarrior, and are used by the debugger to override the RCW on reset.

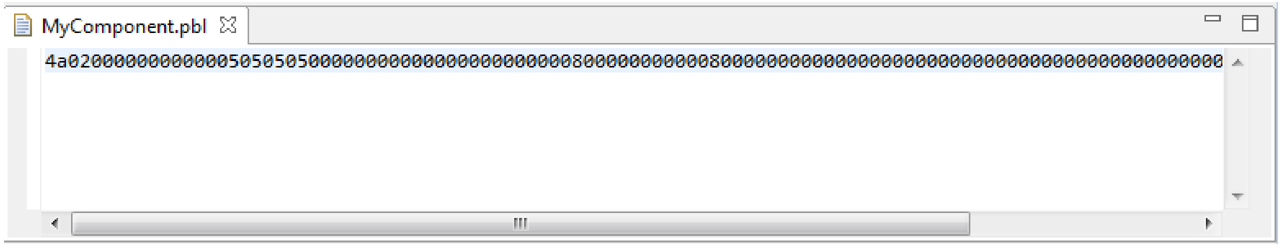


MyComponent.pbl

```
P4080 (2 1) (210001 0x4a020000) (210002 0x00000000) (210003 0x50505050) (210004 0x00000000) (210005 0x0
```

Figure 34. CW JTAG Config (RCW only) format

- Hex String (RCW only): This format renders each byte of the RCW data in hexadecimal. It contains only the RCW bits.

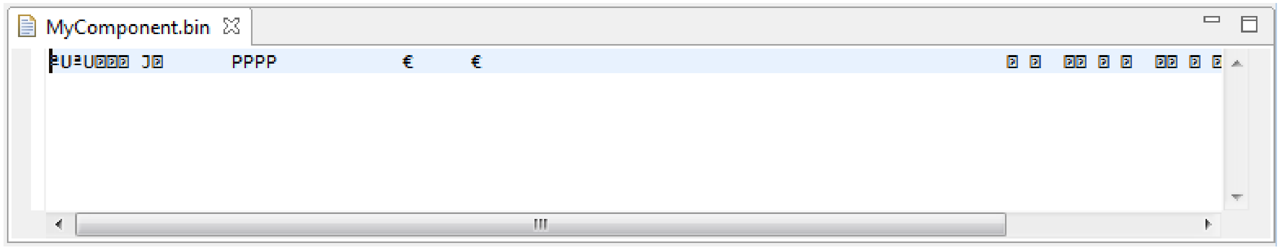


MyComponent.pbl

```
4a020000000000005050505000000000000000000000008000000000008000000000000000000000000000000000000000000000000
```

Figure 35. Hex String (RCW only) format

- Binary: The file format is what you will store in the target's non-volatile memory and what will be used by the SoC out of reset. This format cannot be viewed in a QCVS editor. It is typically viewed with an external hex editor. This file format is directly supported for PBL input also.



MyComponent.bin

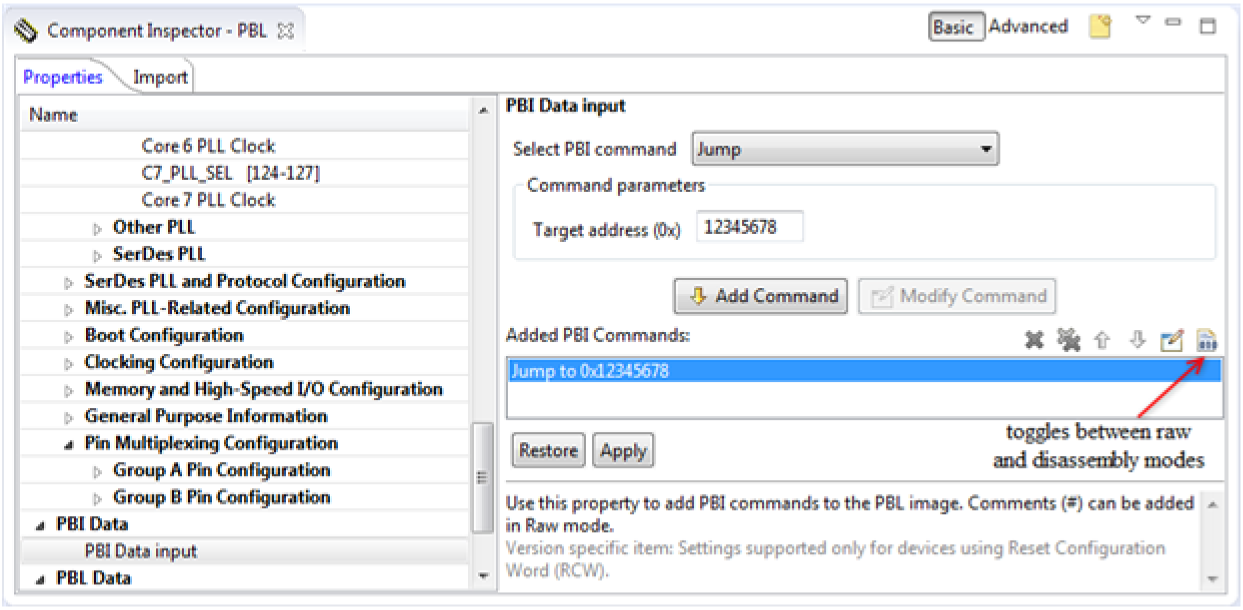
```
▯U=U▯▯▯ J▯       PPPP          €      €                                                    ▯ ▯  ▯▯ ▯ ▯  ▯▯ ▯ ▯
```

Figure 36. Binary file format

1.1.6 PBI commands

This section describes how to add, modify, and view PBI commands.

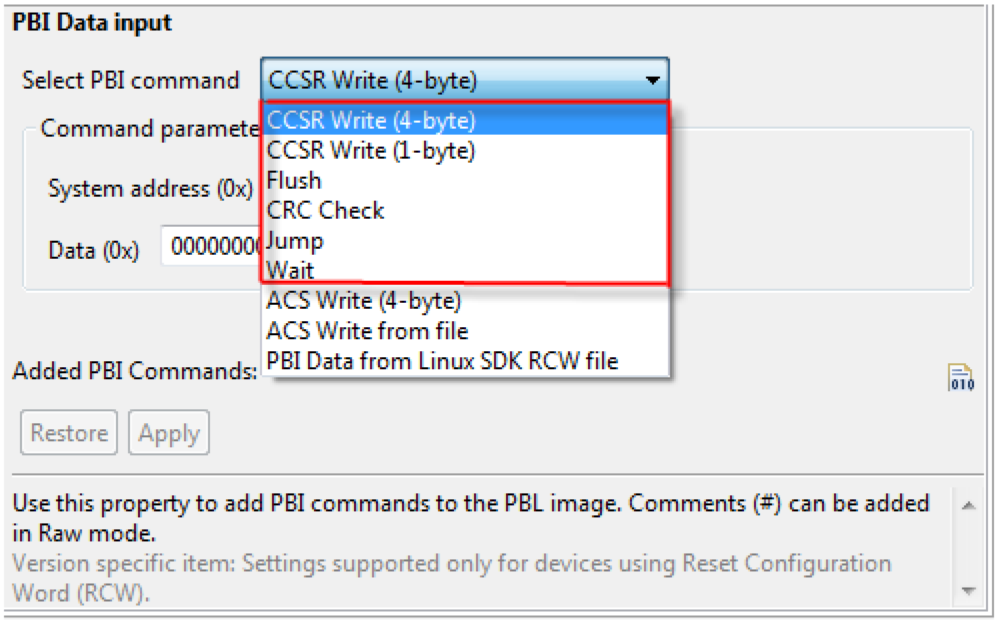
The UI described here is in the editor for the PBI Data input property in the **Component Inspector** view. PBI commands has two view modes, Raw and Disassembly, which can be toggled using a toolbar button as shown below.



Use this property to add PBI commands to the PBL image. Comments (#) can be added in Raw mode.
Version specific item: Settings supported only for devices using Reset Configuration Word (RCW).

Figure 37. PBI command types

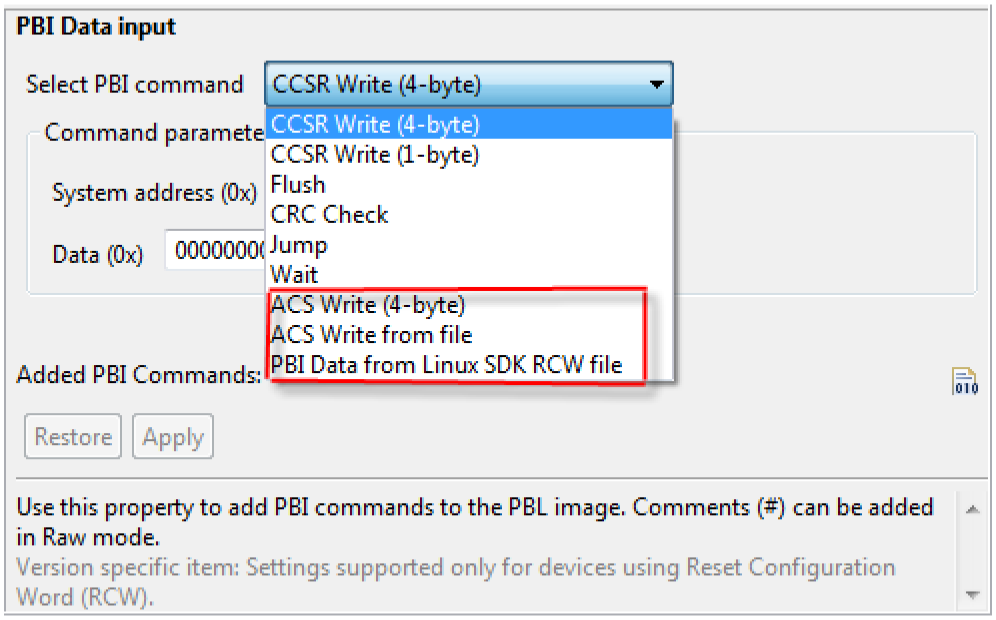
- PBI command to write CCSR data of 4-bytes or 1-byte
- PBI command to write ACS data of 4-bytes
- PBI command to Flush, CRC Check, Jump or Wait



Use this property to add PBI commands to the PBL image. Comments (#) can be added in Raw mode.
Version specific item: Settings supported only for devices using Reset Configuration Word (RCW).

Figure 38. Simple PBI command types

The complex set of PBI data read from user-defined file is shown in the following figure.



Use this property to add PBI commands to the PBL image. Comments (#) can be added in Raw mode.
Version specific item: Settings supported only for devices using Reset Configuration Word (RCW).

Figure 39. Complex set of PBI command data

- PBI data to initialize only ACS space read as data from an xxd Object Dump or from a binary file processed as chunks of data. For example, to initialize ACS address space by a user-defined boot loader or OS kernel that will take care of device booting after a PBL hardware. Data read from the selected file are processed by the PBL tool to set the largest possible data chunks, maximum of 64-bytes as supported by the PBL hardware. The PBL tool properly accepts only xxd Object Dump or binary formatted data for this kind of ACS space initialization.
- PBI data to initialize memory space (CCSR or ACS) read as PBI data from Linux SDK RCW text file and processed to a set of separate PBI commands. For example, device errata workaround or single register initialization through additional PBI commands during boot process.

  You can select the input file which is then processed by the PBL tool with respect to a simple pre-processing and recognition of basic PBI command types as supported by a QorIQ Linux SDK (since version 1.3.1). The following commands are recognized in .pbi sections within the input RCW text file.

  - `write <a> <n>`... write PBI command that writes to CCSR address `<a>` value of `<n>`
  - `awrite <a> <n>`... write PBI command that writes to ACS address `<a>` value of `<n>`
  - `wait <n>`... wait PBI command for defined value of `<n>`
  - `flush` ... flush PBI command

  The following pre-processing statements are supported within the input file.

  - `#include <file-name>` or `#include "file-name"`...include specified file content
  - `#define` ...define either symbol for conditions or symbol with value
  - `#ifdef-#endif`... conditional statement applied to inner content

1.2 PBL validation

This section describes how to work with the PBL validation tool (PBLv) to validate a RCW generated by the tool.

PBLv is a licensed product; therefore, it requires a valid license. See *QCVS DDR Tool User Guide* for more information related to licensing.

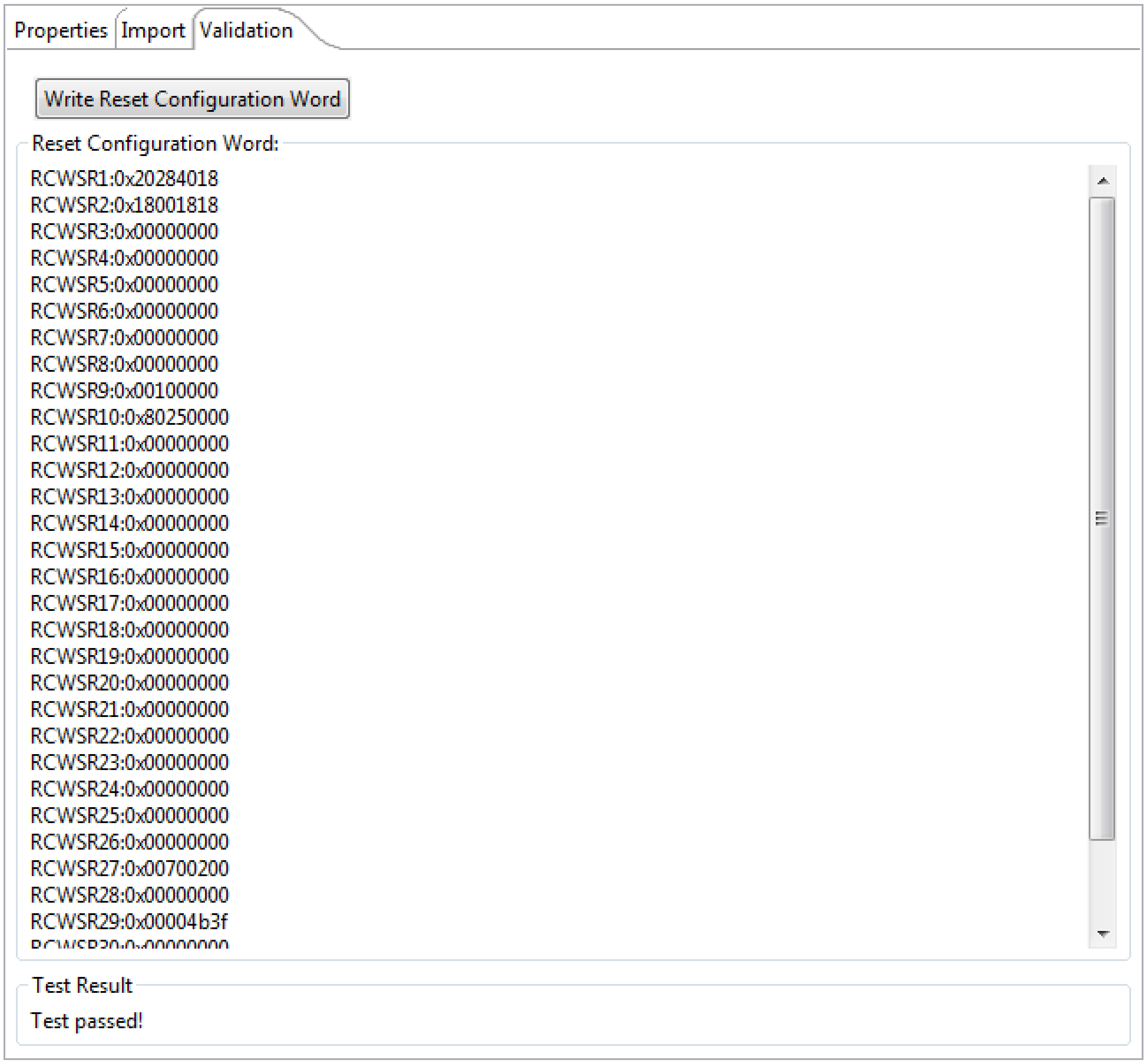
This section contains the following subsection:

- Using PBL validation tool on page 29

1.2.1 Using PBL validation tool

The PBL validation tool can be used for RCW validation, after a PBL configuration is created using the PBL configuration tool.

The **Component Inspector** view displays a page, Validation, which represents the GUI of the PBL validation tool. To use the PBL validation tool, select the **Validation** tab, as shown in the figure below.



```
RCWSR1:0x20284018
RCWSR2:0x18001818
RCWSR3:0x00000000
RCWSR4:0x00000000
RCWSR5:0x00000000
RCWSR6:0x00000000
RCWSR7:0x00000000
RCWSR8:0x00000000
RCWSR9:0x00100000
RCWSR10:0x80250000
RCWSR11:0x00000000
RCWSR12:0x00000000
RCWSR13:0x00000000
RCWSR14:0x00000000
RCWSR15:0x00000000
RCWSR16:0x00000000
RCWSR17:0x00000000
RCWSR18:0x00000000
RCWSR19:0x00000000
RCWSR20:0x00000000
RCWSR21:0x00000000
RCWSR22:0x00000000
RCWSR23:0x00000000
RCWSR24:0x00000000
RCWSR25:0x00000000
RCWSR26:0x00000000
RCWSR27:0x00700200
RCWSR28:0x00000000
RCWSR29:0x00004b3f
```

Test Result

Test passed!

Figure 40. RCW validation using PBL validation tool

If you click the **Write Reset Configuration Word** button, it will override the RCW on target and will perform a target reset. If the processor core is working correctly and it does not return any error, then the test will pass.

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

QCVS_PBL_User_Guide
Rev. 4.x
06/2017