QCVS DDR Tool User Guide



Contents

Chapter 1 DDR Configuration and Validation	3
1.1 DDR configuration	
1.1.1 Using DDR configuration tool	
1.1.1.1 Create a new QorlQ configuration project	
1.1.1.2 Configure DDR controller	
1.1.1.3 Generate code from DDR component	
1.1.2 Importing DDR configuration	
1.1.2.1 Memory dump formats	
1.1.2.2 Get memory dump formats	
1.1.3 Exporting DDR configuration	21
1.1.4 Advanced DDR configuration options	
1.1.4.1 DDR configuration by reading SPD	22
1.1.4.2 DDR configuration by importing from target	24
1.2 DDR validation	28
1.2.1 Overview of DDR validation UI	28
1.2.1.1 Validation scenarios panel	29
1.2.1.2 Validation control panel	
1.2.1.3 Test configuration panel	
1.2.1.4 Scenario results panel	30
1.2.1.5 Validation page toolbar	32
1.2.2 Using DDR validation tool	
1.2.2.1 Choose validation scenarios and tests	33
1.2.2.2 Customize validation tests	
1.2.2.3 Run validation	36
1.2.2.4 Report validation results	
1.2.2.5 Set DDR validation preferences	
1.2.3 Advanced DDR validation operations	
1.2.3.1 DDR validation using margin scenarios	
1.2.3.2 DDR validation using stressing scenarios	
1.2.3.3 DDR validation with warm boot	
1.2.3.4 Validation code customization	
1.2.4 Licensing	48

Chapter 1 DDR Configuration and Validation

This document introduces the double data rate (DDR) RAM configuration and validation tool, which is an embedded component of QorIQ Configuration and Validation Suite (QCVS).

The DDR tool has two components: DDR configuration tool and DDR validation tool. The DDR configuration tool allows you to create a configuration for the DDR component and the DDR validation tool allows you to validate the DDR configuration using various validation scenarios.

This chapter is divided into the following sections:

- DDR configuration
- DDR validation

1.1 DDR configuration

This section describes how to create a new QorIQ configuration project, how to configure the DDR component, and how to perform some basic operations using the DDR configuration tool.

The DDR configuration tool inside the QCVS configures the DDR controller embedded in various supported QorlQ/Qonverge device families.

The DDR configuration tool provides a user-friendly graphical interface to configure the DDR controller. It can be used for tweaking some of the configuration parameters when you want to use different DDR Dual In-line Memory Modules (DIMMs) than the ones received with the board or when you want to optimize the DDR configuration.

While configuring DDR, you can view register values of the DDR controller. The register values will be exported by the DDR tool into C, TCL, or GDB code. After configuring the DDR controller, you can generate code that will fit easily into your software environment in the following ways:

- · OS-agnostic low-level C source code that configures the DDR controller(s) registers' values.
- C source code that outputs the register values in the format required by U-Boot. This file can replace an existing file inside uBoot.
- Source code in DDR related part of the CodeWarrior target initialization file *.tcl. The code in this file can be used inside CodeWarrior projects.

This section contains the following subsections:

- Using DDR configuration tool
- Importing DDR configuration
- Exporting DDR configuration

1.1.1 Using DDR configuration tool

The DDR configuration tool helps you configure the DDR controller in a QorIQ processor, based on the DDR implementation (discrete or DIMM) used in the processor.

The DDR implementation may differ by various parameters, such as number of chip selects, memory size, or ranks interleaving. To configure a new DDR controller, you need to use the specifications from the DDR manufacturer data sheet and fill the values in the interface. The DDR parameters are grouped together by functionality.

This section contains the following subsections:

- Create a new QorIQ configuration project
- Configure DDR controller
- · Generate code from DDR component

1.1.1.1 Create a new QorlQ configuration project

To use the DDR configuration tool, you first need to create a QorIQ configuration project with DDR configuration.

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

- 1. Select File > New > QorlQ Configuration Project. Follow the steps in the New QorlQ Configuration Project wizard.
- 2. Enter project name.
- 3. Select the target SoC.
- 4. On the Toolset selection page, select the DDR Memory Controller Configuration checkbox.

ſ			
	P New QorlQ Configuration Project	0 X	
	Toolset selection		
	Choose what do you want to configure		
	Components to be selected		
	type filter text		
	✓ □ Components		
	DDR Memory Controller Configuration PBL - Preboot Loader RCW configuration		
	SERDES Configuration Tool		
	The DDR Memory Controller configuration tool supports the specific settings for a custom DDR based o manufacturers data sheet and includes optional clock, bus and DMA settings.	n the	
	? < gack Next > Einish	Cancel	
L			
Figure 1. Toolset select	tion page		

- 5. Click **Next**. The **DDR configuration** page appears.
 - To configure the DDR with DIMM settings, select the **DIMM** option.

Configuration mode Auto configuration From memory file Read SPD From target Discrete DRAM Image: DIMM Read PLL Probe: 10.10.10.10 Clock settings	
O Discrete DRAM © DIMM Read PLL Probe: 10.10.10.10	
Read PLL Probe: 10.10.10.10	
Clock settings	¥
crock scange	
Input DDR clock: 100.0 MHz V DDR PLL Multiplexer: N/A Output data rate: 2100 MT/s	-
DDR Controller DRAM Settings	
Type DDR.4 V DRAM configuration per device 16Gb: 2Gb x8	~
Rank/Chip select 1 V DIMM speed rating PC4-25600 (3200)	~
Data bus width 64 bits \sim	
tCL 15 clocks Clock Quad-ranked DIMM Mirrored DIMM	
ECC enabled Quad-ranked DIMM Mirrored DIMM	
Skews	
CLK to DQS 0 0 0 0 0 0 0 0 0 mm	
Lead-in 0 mm	
Import from file	
Input file Browse	
File format CodeWarrior Register Text	
1080000	1
Endian mode Little Endian Start address of 1st DDR controller	
Addressable Size 1 byte 0 Use default	

Lead-in skew is the difference between the A/C bus trace length and data bus trace length from the controller pins to the DIMM connector pins.

• For designs that utilize embedded, discrete devices, select the **Discrete DRAM** option.

Configure: 1st DDR Controller	~
Configuration mode	
Auto configuration O From	n memory file O Read SPD O From target
Discrete DRAM O DIMM	
Read PLL Probe:	10.10.10.10
Clock settings	
-	PLL Multiplexer: N/A Output data rate: 2100 MT/s 🗸
DDR Controller	DRAM Settings
Type DDR 4 ~	DRAM configuration per device 16Gb: 2Gb x8 ~
Rank/Chip select 1 ~	DRAM speed rating 2133 MT/s \checkmark
Data bus width 64 bits \vee	
tCL 15 clocks ~	Quad-ranked DIMM
ECC enabled DBI enabled	
Skews	
	0 0 0 0 0 mm
Lead-in 0 mm	
Import from file	
Input file 🧧	Browse
File format CodeWarrior Register Te	
Endian mode Little Endian	1080000 Start address of 1st DDR controller
Addressable Size 1 byte	0 Use default
	Import
	mpore

6. Click Finish to complete project creation.

This creates a QorIQ configuration project for DDR configuration.

1.1.1.2 Configure DDR controller

After creating a QorIQ configuration project with the DDR toolset, you can configure the DDR controller using the DDR configuration tool.

Perform the following steps to configure DDR controller:

1. Select the DDR component under the **Components** folder in the **Components** view, as shown in the following figure. The properties of the DDR component opens in the **Component Inspector** view.

NOTE If the Component Inspector view is closed, you can open it by right-clicking the DDR component in the Components view and choosing the Inspector option from the shortcut menu.

🖉 eclipse-workspa	ce 26 10 - Cod	leWarrior Developme	nt Studio for Oor	IQ LS series - ARM V8 IS	54				- 0	×
										~
<u>File Edit N</u> avigate									o ' =	
	: 💊 🔻 🔛	🔗 🕶 📮 🔛 👻		· • • 🗹					Q : 🖽	' EC 🚺
Project Explorer	🛄 Connectio	ons View 🛛	- 8	🚫 Component Inspe	ector - DDR_mc1	🔀 🚫 Compo	onents Library Basic	Advanced	000	
Target connections	s			Properties Import	Export Validati	ion				
🔮 💷 🗙			() X	Name		Value		Details		^
				Device		DDR_Controller	_1	DDR_Controller	1	
Processor	Probe Type	Probe Address		Memory type		DDR 4				
LX2160A	cwtap			DRAM type		Module				
				DDR Bus Clock		1050 MHz		DDR Data Rate:	2100 MT/	
				DIMM type		Unbuffered DIM	1Ms			
				Bus mode		64-bit bus				
				V SDRAM Control	-					
				✓ Control Config SDBAM and		Fredelad				
			0		f refresh during s Checking and Co					
€. Components - Ix2	2160 🖂	🖻 🖆 🐴	°		ower manageme					
✓				Timing mod	-	1T Timing				
🎆 LX2160A_v	2_0_Cnf			I/O Driver Ir		Full Strength				
> 🗁 OSs					ed ECC enable	no				
V > Processors				✓ Control Config	guration 2					
> 💮 SoC:LX2160	0A_v2_0			DLL Reset	-	no				
✓ ▷→ Components				DQS Config	guration	Use differential	DQS signals			
🌐 DDR_mc1:E	JUK			ODT Config			ternal IOs only dur			
				Number of	posted refreshes	1 refresh is issue	ed at a time			
				x4 DRAM er		x8 or x16				
				Use quad-ra	anked DIMM	no				¥
				<					>	
🖹 Problems 🔀								7 5	80	
0 items	~				1					
Description			Resource	Path	Location	Туре				
251M of 393M	Ē						8			
Figure 4. Sele	ect DDR	components	in Compor	ients view						

NOTE

You can add multiple DDR components to a project. Each DDR component maps to one DDR controller of the SoC. For example, LS2088A SoC has two embedded DDR controllers, so if you want to configure both of them, add two DDR components from the **Components Library** to the project, one component for each controller. Right-click the DDR component in the **Categories** tab and select **Add to project with Wizard** from the shortcut menu, as shown in the following figure.

Categories	Alphabetical A	ssistant Processors	
Component	Co	mponent Level	
	guration Tool:		
		inharal Initiali	
🚾 🖻	Add to project	245 MG1	
() 🔮	Add to project	with Wizard	
	Expand all		
	Collapse all		
8	Refresh		F5
×	Delete	Del	ete
	Help on Comp	onent	

- 2. Modify the following basic DDR parameters according to your requirements:
 - Memory type: Specify the DDR memory type, for example, DDR 4 or DDR 3L

Memory type	DDR 4	~
DRAM type	DDR 4	
DDR Bus Clock	[!] DDR 3L	

Figure 6. Set memory type

• DDR Bus Clock: For each DDR bus clock, there is a specific bus clock frequency that measures the duration of a bus clock cycle

DDR Bus Clock	1050 MHz	~
DIMM type	733 MHz	^
Busmode	750 MHz	
SDRAM Control Continuization	800 MHz 900 MHz	
Control Configuration 1	933 MHz	
SDRAM self refresh during s	1000 MHz	
ECC (Error Checking and Cc	1050 MHz	
Dynamic power manageme	1066 MHz	
Timing mode	1100 MHz 1200 MHz	
VO Driver Inspedence	1300 MHz	
Accumulated ECC enable	1333 MHz	\sim

• DIMM type: Specify the DIMM type: unbuffered or registered

DIMM type	Unbuffered DIMMs 🗸
Bus mode	Unbuffered DIMMs
SDRAM Control Configuration	Registered DIMMs
Figure 8. Set DIMM type	

• **Timing Configuration 1**: Set four basic DDR timings: *tCL, tRCD, tRP, tRAS* specified usually as *tCL-tRCD-tRP-tRAS*. The timings are measured in bus clock cycles.

CAS Latency (tCL)	15 clocks	14.3 ns
Activate to read/write interval (tRCD)	15 clocks	14.3 ns
Precharge-to-activate interval (tRP)	15 clocks	14.3 ns
Activate to precharge interval (tRAS)	40 clocks	38.1 ns
Refresh recovery time (tRFC)	473	D 450.5 ns
Last data to precharge minimum interval (tWR)	16 clocks	15.2 ns
Activate-to-activate interval (tRRD)	6 clocks	5.7 ns
Last write data pair to read command issue interval (tWTR)	3 clocks	2.9 ns

• Chip Select 0, 1, 2, 3: Specify the number of physical ranks or chip selects. A DIMM can have 1, 2, or 4 ranks and are specified in the basic DIMM information as 1R ... or 2R... or 4R. For this, from the four chip selects, enable the ones that correspond to the ranks (each chip select corresponds to one rank). Also, set size of the DIMM. The size is configured inside each chip select. For example, a 32 GB dual-ranked DIMM has its size split evenly between the two ranks such that each chip select is of 16 GB size.

Auto-adjust chip select addressing	yes	
Chip Select 0	Enabled	
✓ Memory Bounds		
Start Address	0	H
Size (in MB)	16384	D 16 GB
> Configuration		
Chip Select 1	Enabled	
✓ Memory Bounds		
Start Address	40000000	H
Size (in MB)	16384	D 16 GB
> Configuration		
Chip Select 2	Enabled	
✓ Memory Bounds		
Start Address	80000000	H
Size (in MB)	16384	D 16 GB
> Configuration		
Chip Select 3	Enabled	
✓ Memory Bounds		
Start Address	C0000000	H
Size (in MB)	16384	D 16 GB
> Configuration		

3. Set rest of the parameters as per the datasheet of the DDR device. The datasheet can be obtained from the DDR manufacturer's website or directly from the DDR manufacturer.

For each step of the configuration process, you might receive validation errors for some specific parameters along with information on how to fix the validation conflict.

You can import register values of the DDR component from a memory dump, containing the values of the DDR memory-mapped registers. Memory dumps can be obtained from U-Boot (using the md command) or CodeWarrior (using the Export registers/ memory feature).

1. Select the Import tab in the Component Inspector view.

roperties Import	Export Validation		
Memory dump se			
File format	Import From Target	~	
Addressable size	1 byte	\sim	
Endianness	Big Endian	~	
Address informat	ion		
First address from	memory dump file	1080000	
Start address of D	DR memory controller	1080000	🗹 Use defau
		Import	

2. Specify the location of the input file in the Input file field. The red mark denoting an error/conflict disappears.

NOTE

The red marks indicate errors; therefore, before clicking the Import button, ensure that no red mark is displaying.

- 3. Choose an appropriate file format for the input file from the **File format** menu. If the file content is not in accordance with the chosen format, the import operation will fail and an error message will pop up.
- 4. Specify the number of bytes by choosing an option from the Addressable size menu.
- 5. Specify byte order by choosing an option from the Endianness menu.
- 6. Specify valid address information of the memory dump file in the First address from memory dump file field.
- 7. Specify the beginning address of the memory-mapped DDR registers inside the memory dump in the Start address of DDR memory controller field. The First address from memory dump file and Start address of DDR memory controller fields are enabled for editing only after deselecting the Use default checkbox.
- 8. Click Import. The input file is imported.
- 9. Click **OK** in the **Import complete** dialog (shown below).

🏴 Imj	port complete	×	
1	Input file has been imported		
		ОК	
ıre 12. Import	complete dialog		

10. Open the **Problems** view to see reported errors if any.

NOTE

You can also configure the DDR component while creating the QorIQ project. On the **DDR Configuration** page (see Figure 3), select the appropriate option, **Auto configuration** or **From memory file**, and provide the necessary details corresponding to the selected option.

1.1.1.3 Generate code from DDR component

After you finish configuring the DDR controller, you can generate code from DDR component to get updated output of the component.

The source code generated from the DDR component helps you configure the DDR controller registers and it can be used in the CodeWarrior projects.

The code generation from the DDR component is based on the properties set in the **Component Inspector** view. You can use any of the following options to generate DDR component code:

Ð

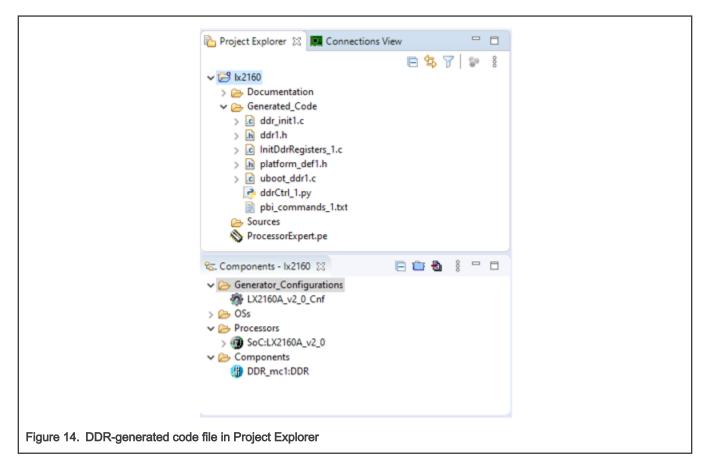
· Click the Generate Processor Expert Code icon

in the **Components** view, as shown in the following figure.

- Right-click **ProcessorExpert.pe** in the **Project Explorer** and choose **Generate Processor Expert Code** from the shortcut menu.
- Choose the Project > Generate Processor Expert Code option from the Eclipse IDE menu.

arget connections Properties Import Export Value Processor Probe Type Probe Address DRAM DRAM DRAM VX2160A cwtap 10.171.77.80 DRAM Memory type DOR 4 DRAM type Module DDR 4 DDR 4 DDR 4 DRAM type Module DDR 4 DDR 4 DRAM type Unbuffered DIMMs Bus mode 64-bit bus SDRAM control Configuration v SDRAM self refresh during sleep Enabled © Generator_Configurations U/O Driver Impedance Full Strength Accumulated ECC enable no © Os SociLV2160A,v2.0_Crif Vo Driver Impedance Full Strength Accumulated ECC enable no © Os ODT configuration 2 Numeer opostel refreshes 1 refresh is submote opostel refreshes 1 refresh is submote opostel refreshes 1 refresh is subonty c		-	🔗 ▼ 📮 🖢 ▼			ent Increastor - DDP	1 1 1 1 1	m non onte Liberra	Basic Advanc	ed 🌱 8	. : 🖻 8 — न	• E C
Image: Second particular processor Value Processor Probe Type DDR 4 Device DDR 4 Devide Drantification Components - W2160 (3) Image: Configuration 1 Second Configurations Image: Configuration 1 Second Configurations Image: Configuration 2 Processor Processor Expert Code hg mode IT Timing Processor Processor Processor ODT Configuration 2 <								mponents Library	Dusic		•	
Processor Probe Type Probe Address Device DDR 4 LX2160A covtap 10.171.77.80 DRAM type DDR 4 DRAM type DRAM type DDR 4 DRAM type Unbuffered DIMMs Bus Clock 1050 MHz DIMM type Unbuffered DIMMs Bus Clock 1050 MHz DIMM type Unbuffered DIMMs Bus mode 64-bit bus < SDRAM Control Configuration	arget connectio	-			Properties	Import Export Valida	ition				_	_
Processor Probe Type Probe Address DR.4 LX2160A cvrtap 10.171.77.80 DRAM type Module DRAM type Unbuffered DIMMs 64-bit bus 64-bit bus Components - tx2160 Control Configuration 64-bit bus 64-bit bus Components - tx2160 Central Configuration Drawer management mode Enabled Components - tx2160 Central Configuration 1 Drawer management mode Enabled Components - tx2160 Generate Processor Expert Code bg mode 111 Timing OS: VO Driver Impedance Full Strength Processors VO Oriver Impedance Full Strength Components Use differential DQS signals OC Configuration 2 no Components Use quad-ranked DIMM no VS Probetisor Scrift Configuration Use quad-ranked DIMM no VS BockLX2160A_V2_0 VS Probetisor 1refresh is issued at time x4 DRAM enable x8 or x16 VS quad-ranked DIMM no VS quad-ranked DIMM no VS quad-ranked DIMM no VS quad-ranked DIMM no VS quad-ranked DIMM	🖹 🗟 🗶			0 💋	Name				Value		^	
IX2160A cvrtap 10.171.77.80 DRAM type Module DDR Bus Clock 1050 MHz Components - k2160 13 Image: Strength OSs Control Configuration 1 Drocessors VO Driver Impedance Followard Full Strength Accumulated ECC enable no VO Driver Impedance Full Strength OS Configuration Assert ODT to internal IOS only c DLR Reset no Components 11 effesh is issued at a time W DDR_mc1:DDR Number of posted refreshes Image: Strength No V DRAM Configuration Assert ODT to internal IOS only c Image: Strength No V DRAM Configuration Assert ODT to internal IOS only c Image: Strength No V DRAM Configuration Assert ODT to internal IOS only c Image: Strength No Image: Strength No Image: Strength No <		_			Device				_	ler_1		
DNR Total Total DNR Mice 1050 MHz DDR Bus Clock 1050 MHz DDR Bus Clock DDR Mus DDR Bus Clock DDR Mus DDR Bus Clock 04-bit bus Sortin Total 64-bit bus Sortin Total Sortin Total Components - Ix2160 S Sortin Total Generator Configurations Sortin Total Sortin Total Dynamic power management mode DDR Sortin Total Dynamic power management mode Processors Sortin Total Sortin Total Sortin Total <t< td=""><td>Processor</td><td>Probe Type</td><td>Probe Address</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></t<>	Processor	Probe Type	Probe Address									
DIMM type Unbuffered DIMMs Bus mode 64-bit bus SDRAM Control Configuration SDRAM Control Configuration 1 Socratic Configurations S ■ ■ SCRETC Checking and Correction) Disabled Dynamic power management mode Enabled Socratic Configurations Generate Processor Expert Code by mode 11 Timing YO Driver Impedance Full Strength no Socratic Configuration Vice Ontrol Configuration 2 no Socratic Configuration Vice Ontrol Configuration 2 no Socratic Configuration Use differential DOS signals ODT Configuration Socratic Configuration Use differential DOS signals ODT Configuration Assert ODT to internal DOS signals ODT Configuration Use quad-ranked DIMM no Image: Strength Image: Strength Problems: S Resource Path Location Type ens Image: Strength Image: Strength Image: Strength Image: Strength Mode Path Location Type Image: Strength Image: Strength Image: Strength Image: Strength Image: Strength Image: Stre	LX2160A	cwtap	10.171.77.80			•						
Bus mode 64-bit bus SDRAM Control Configuration SDRAM self refresh during sleep Enabled Dynamic power management mode Enabled Dynamic power power management mode Enabled Dynamic power												
SDRAM Control Configuration Image: Control Configuration 1 Components - kx2160 % Image: Control Configuration 1 Generator, Configurations ECC (Error Checking and Correction) Disabled Mainter Configurations Image: Control Configuration 2 Image: Control Configuration 2 OSs Components NO Processors Soc Configuration 2 Image: Configuration 2 Image: Configuration 3 Image: Configuration 2 Image: Configuration 2 Image: Configuration 4 Image: Configuration 2 Image: Configuration 2 Image: Components Image: Configuration 2 Image: Configuration 2 Image: Control Configuration 3 ODT Configuration 2 Image: Configuration 3 Image: Control Configuration 4 ODT Configuration 3 Assert ODT to internal IOs only configuration 4 Image: Control Configuration 5 Image: Configuration 4 Image: Configuration 4 Image: Control Configuration 6 Image: Configuration 4 Image: Configuration 4 Image: Control Configuration 7 Use differential DQS signals ODT Configuration 7 Image: Control Configuration 7 Image: Control Configuration 7 Image: Control Configuration 7 Image: Control Configuration 7 <						•				IMMs		
Components - Ix2160 % Control Configuration 1 SDRAM self refersh during sleep Enabled Disabled Disabled									64-bit bus			
SDRAM self refresh during sleep Enabled Components - lx2160 (2) Image: Strength of the strength						-						
Components - tx2160 % Configurations Generator_Configurations LX2160A_v2_0_Cnf OSs Processors Soci_X2160A_v2_0 Components DDR_mc1:DDR Components						-	cleen		Enabled			
Components Configurations Generate Processor So Components DDR_mc1:DDR Problems S2 Problems S2 Path Location Type	-											
Generator_Configurations I X2160A_v2_0_Cnf Processors Processors Soc.LX2160A_v2_0 Components DQS Configuration Use differential DQS signals ODT Configuration Assert ODT to internal IOS only c Number of posted refreshes 1refresh is issued at a time x4 DRAM enable x8 or x16 Use quad-ranked DIMM	Components -	lx2160 🔀										
Image: Display and the second sec				Generate Proces			ient mode					
		_v2_0_Cnf	I I									
SoC:LX2160A_v2_0 Components DDR_mc1:DDR DLL Reset									-			
> Soc:LX21604_v2_0 ▷ Components □ DDR_mc1:DDR □ DR_mc1:DDR □ DR_mc1:DDR □ DR_mc1:DDR □ DR_mc1:DDR <t< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></t<>												
Image: Configuration Use configuration Obs configuration ODD Configuration Assert ODT to insure IN Obs signals ODT Configuration Assert ODT to insure IN Obs signals Number of posted refreshes 1 refresh is sued at a time x4 DRAM enable x8 or x16 Use quad-ranked DIMM no Image: Configuration Image: Configuration Problems S3 Image: Configuration Image: Configuration Image: Configuration Image: Configur						-			no			
Problems ⊠					DQ	S Configuration			Use differenti	al DQS sign	als	
x4 DRAM enable x8 or x16 Use quad-ranked DIMM no v < <p>Problems ⊠ ems escription Resource Path Location Type</p>	B DDR_mc	1:DDR			OD	T Configuration			Assert ODT to	internal IO	s only d	
Use quad-ranked DIMM no Problems ⊗ > ems > escription Resource Path Location Type					Nu	mber of posted refresh	es		1 refresh is iss	ued at a tin	ne	
Image: Contract of the second of the sec					x4	DRAM enable			x8 or x16			
Problems X Problems X ems escription Resource Path Location Type					Use	e quad-ranked DIMM			no		~	
ems escription Resource Path Location Type					<						>	
Ascription Association Resource Path Location Type	Problems 🔀								7	7 50 8	8 - 8	1
escription Resource Path Location Type		~										_
	scription			Resource	Path	Location	Туре					
21584 - 6 20284												
21584 - 6 20284												
	21514 - 62021	1						8.				

The output file is generated and placed in the **Generated_Code** folder of the **Project Explorer**. The output file is named according to the DDR component name used in the QorlQ Configuration project, as shown in the figure below.



1.1.2 Importing DDR configuration

The DDR tool allows you to import a memory dump, which can be in a format, such as U-Boot, S-Record, AnnotatedHex, Signed/Unsigned decimal, CodeWarrior Regs, Hex, or Raw Binary.

The memory dump is decoded into a DDR configuration, which then can be visualized, validated, and adjusted, if required.

To import a DDR configuration, a QorIQ Configuration project must be created. You can import the DDR configuration during project creation or later for any component already added to the project. Perform the following steps:

- 1. On the **DDR Configuration** page of the **New QorlQ Configuration Project** wizard (shown below), select the **From memory file** option.
- 2. Specify the memory dump file to be imported in the Input file text box.
- 3. Based on file content, choose a file format for the input file from the File format menu.
- 4. Choose byte order from the Endian mode menu.
- 5. Click Import and wait to see if the import was done correctly. The Finish button will be enabled.

Configure:	1st DD	R Controller						~
Configuration	mode							
OAuto	config	uration	From	m memory file	Read SPE) OF	rom target	
O Discrete DR	M	DIMM						
Read		Probe:						
		Probe:						¥
- Clock settings		00.0 MHz 🔍	DDR	PLL Multiplexer: N/A	Outp	ut data rate:	1300 MT/s	\sim
DDR Controll	er			DRAM Settings				_
Туре		DDR 4	\sim	DRAM configuration p	er device	2Gb: 512Mb	×4	\sim
Rank/Chip se	lect	4	\sim	DIMM speed rating		PC4-12800 ((1600)	\sim
Data bus widt	h	64 bits	\sim					
tCL		14 clocks	\sim	Registered DIMM		SO-DIMN		
ECC enabl				Quad-ranked DIMN	1	Mirrored	DIMM	
DBI enable	d							
Skews								
CLK to DQS	0		0	0 0 0	0	0 mm		
Lead-in		mm						
Import from fi	le							_
Input file C	:\Users	\nxa11975\Desk	top\D[DR\LX2160A_v2_0_annota	ted_hex.b	đ	Brows	se
File format	D	Annotated Hex T	ext.	~	First add	iress from me	mory dump	file
Endian mod		Big Endian		~	108000	0		
Addressable				~		dress of 1st DI		
					108000	0 10	Use default	
				Import				

- 6. Click Finish. The DDR project is created.
- 7. Open Component Inspector for the DDR project.
- 8. Select the **Import** tab and set the following parameters as shown in the next figure:
 - Input file: Specify the file that encapsulates the memory dump
 - File format: Choose a file format for the input file based on the file content. If the file content is not in accordance with the chosen format, the import operation will fail and an error message will pop up.

You can also import a DDR configuration from the target board by choosing **Import from target** from the **File format** menu. This action only works if an active and working connection is configured in the **Connections View** (see *QCVS Getting Started Guide* for more information). If such a connection exists, the DDR tool reads the DDR registers from

the board and populates the active component with the data read. Note that this type of action is relevant only if the DDR registers were previously configured (for example, by U-Boot).

- · Addressable size: Specify the number of bytes
- Endianness: Specify byte order
- · First address from memory dump file: Specify valid address information of the memory dump file
- Start address of DDR memory controller: Specify the beginning address of the memory-mapped DDR registers inside the memory dump. The First address from memory dump file and Start address of DDR memory controller fields are enabled for editing only after deselecting the Use default checkbox.

roperties Import	Export Validation						
Memory dump s	ettings						
Input file	C:\Users\nxa11975\De	sktop\DDR\l	LX2160A_v2_0)_annotate	d_hex.txt	Bro	wse
File format	Annotated Hex Text	\sim					
Addressable size	1 byte	\sim					
Endianness	Big Endian	~					
Address informat	tion						
First address from	n memory dump file	read from r	memory dum	p file			
Start address of D	DR memory controller	1080000				🗹 Use d	efault
		Impo	ort				

NOTE

In case the input file does not contain the DDR data rate value on the first line, you will be prompted to choose a data rate value.

🔑 DDR DataRate Input		×
DDR data rate is invali	id!	
 Please select a DDR da 	ta rate value	
Gelect DDR Data Rate	2100	
		ОК

This is how a DDR configuration can be imported.

While performing the import operation during project creation, you can configure all the DDR controllers. Choose the current DDR controller from the **Configure** list on the **DDR Configuration** page (see Figure 15). If you import from Component Inspector for DDR, then the import operation will be done only for the selected DDR controller.

This section contains the following subsections:

- Memory dump formats
- Get memory dump formats

1.1.2.1 Memory dump formats

This section describes the different memory dump formats available for importing.

The various memory dump formats available for importing are shown in the following figures.

- · AnnotatedHex: In this format:
 - The first line contains the corresponding data rate
 - The second line contains the start address of DDR memory controller
 - Each line contains four registers, each of them is a 32-bit register
 - First four registers are: CS0_BNDS, CS1_BNDS, CS2_BNDS, CS3_BNDS

🔚 LX2160A_v2_0_annotated_hex.txt 🗵

1	##2100
2	01080000: →000003FF·00000000·040007FF·00000000· ?
3	01080010: →08000BFF·00000000·0C000FFF·00000000· ??
4	01080020: →00000000.0000000.0000000.0000000
5	$01080030: \longrightarrow 00000000 \cdot 00000000 \cdot 00000000 \cdot 000000$
6	$01080040: \longrightarrow 00000000 \cdot 00000000 \cdot 00000000 \cdot 000000$
7	$01080050: \longrightarrow 00000000 \cdot 00000000 \cdot 00000000 \cdot 000000$
8	01080060: →00000000 · 0000000 · 0000000 · 0000000 ·
9	01080070: →00000000.0000000.0000000.0000000
10	$01080080: \longrightarrow 80340522 \cdot 80340522 \cdot 80340522 \cdot 80340522 \cdot ?4."?4."?4."?4."$
11	01080090: →00000000.0000000.0000000.0000000
12	010800a0: →00000000·0000000·0000000·0000000·
13	010800b0: →00000000.0000000.0000000.0000000
14	010800c0: →00000000.0000000.0000000.0000000
15	$010800d0: \longrightarrow 00000000 \cdot 00000000 \cdot 00000000 \cdot 000000$



S-Record: You can find detailed information at http://en.wikipedia.org/wiki/SREC_(file_format)

1	##2100
2	S0030000FC
3	s3250108000000003FF0000000040007FF000000008000BFF0000000000000FFF00000000
4	s32501080020000000000000000000000000000000
5	s325010800400000000000000000000000000000000
6	s325010800600000000000000000000000000000000
7	s3250108008080340522803405228034052280340522000000000000000000000000000000000
8	\$325010800a00000000000000000000000000000000
9	\$325010800c00000000000000000000000000000000
10	s325010800e00000000000000000000000000000000
11	s32501080100021D11009A770010F8Fc10630059119A45240000004010300101063004100000DF
12	s3250108012000000001ffe07ffDeADBEEF00000000000000000000000000000000000
13	s325010801400000000000000000000000000000000
14	s32501080160000000206401400000000255400008A090705000000000000000000000000000000000
15	s32501080180000000000000000000000000000000
16	\$325010801a00000000000000000000000000000000
17	s325010801c00000000000000000000000000000000
18	s325010801e00000000000000000000000000000000

• Signed/Unsigned Decimal: The difference between Signed and Unsigned is that the former has a decimal number with a sign while the latter contains a decimal number without a sign.

Signed	Decimal	Unsigne	d_Decimal
260	0	260	0
261	64	261	64
262	51	262	51
263	1	263	1
264	4	264	4
265	111	265	111
266	107	266	107
267	70	267	70
268	68	268	68
269	15	269	15
270	-88	270	168
271	-111	271	145
272	-112	272	144
273	-9	273	247
274	20	274	20
275	0	275	0
276	0	276	0

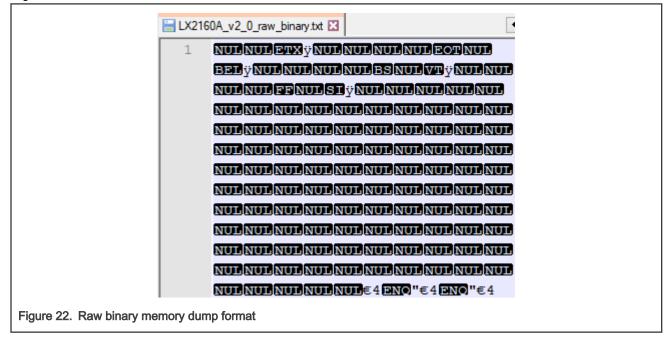
 CodeWarrior Regs: It is the CodeWarrior-specific memory dump format. Such a dump can be obtained from a CodeWarrior product (PA, ARMv7, or ARMv8) by exporting the DDR registers (using the Register View dialog) into XML format. Г

4	🔚 LX21	60A_v2_0_codewanior_regs.regs 🔀
10000	1	xml·version="1.0"·encoding="UTF-8"·standalone="no"?
	2	<pre>G<register-groups></register-groups></pre>
	3	<board-config></board-config>
	4	···· <ddrrate·value="2100"></ddrrate·value="2100">
10000	5	-···
1000	6	<register-group.name="ddr.memory.controller.1"></register-group.name="ddr.memory.controller.1">
1000	7	<register.name="ddr1_timing_cfg_8"></register.name="ddr1_timing_cfg_8">
1000	8	····· <value>0x05226800</value>
1000	9	····· <location>0x01080250</location>
1000	10	····· <custom-groups></custom-groups>
	11	-····
	12	<register.name="ddr1_timing_cfg_7"></register.name="ddr1_timing_cfg_7">
2000	13	····· <value>0x25540000</value>
1000	14	····· <location>0X0108016C</location>
1000	15	····· <custom-groups></custom-groups>
1000	16	-····
10000	17	: <register name="DDR1_TIMING_CFG_5"></register>
1000	18	····· <value>0x06401400</value>
	19	····· <location>0x01080164</location>
1000	20	····· <custom-groups></custom-groups>
	21	-····
F	igure 20.	CodeWarrior Regs memory dump

• Hex: It contains a list of hex-formatted values where four consecutive lines form a register value.

🔚 LX21	60A_v2_0_hex_text.txt 🔀
1	##2100
2	0
3	0
4	3
5	FF
6	0
7	0
8	0
9	0
10	4
11	0
12	7
13	FF
14	0
15	0
16	0
17	0
18	8
19	0
20	В
21	FF
e 21. Hex memory dump format	

• Raw binary: It is a binary file. It represents a stream of hex values in a binary format, and four consecutive values form a register value.



• PHY parameters: This format is only available for LX series targets. Each line contains PHY parameters values. These parameters are part of the input for the "Full Init" performed by the firmware.

	"ddr_phy_init".:.{
	····"dram_type"········:·"DDR·4",
	"dimm_type":."Unbuffered.DIMMs",
	····"hard_macro_ver"·····:•"10",
	····"num_dbyte"······:'"0x9",
	<pre>"num_active_dbyte_dfi0".:."0x9", "</pre>
	····"num_anib"······:·"0xC",
	····"num_rank_dfi0"······:·"0x2",
	····"num_pstates"······:'"0x1",
	····"frequency"······: "1450",
	····"pll_bypass"······:·"0x0",
	····"dfi_freq_ratio"······:·"0x1",
	····"dfi1_exists"·······:"0x0",
	····"dram_data_width"·····::"8",
	····"dram_byte_swap"······:·"0x0",
	····"ext_cal_res_val"·····::"0x0",
	<pre>''''tx_slew_rise_dq"''''''''''''''''''''''''''''''''''''</pre>
	<pre>'''tx_slew_fall_dq"''''''''''''''''''''''''''''''''''''</pre>
	····"tx_slew_rise_ac"·····:·"0xF",
	····"tx_slew_fall_ac"·····:"0xF",
	····"odt_impedance"······:"48",
	····"tx_impedance"······:'48",
	····"atx_impedance"·······:"30",
	····"mem_alert_en"············
	····"mem_alert_pu_imp"·····:"0x5",
	<pre>''''mem_alert_vref_level"''''''''''''''''''''''''''''''''''</pre>
	<pre>''''mem_alert_sync_bypass'' : '''0x0'', ''''''''''''''''''''''''''''''</pre>
	····"cal_interval"······::"0x9",
	"cal_once":."0x0",
	····"dis_dyn_adr_tri"·····::"0x0",
	····"is2t_timing"······:·"0x0",
Figure 23. PHY part	rameters
-	

1.1.2.2 Get memory dump formats

Using CodeWarrior, you can get a memory dump format from the DDR controller registers of a board.

The DDR configuration tool allows you to import, view, and then configure the DDR controller registers of a board. The DDR controller registers can be downloaded from the board in a memory dump format using a dump utility provided by NXP's CodeWarrior Development Studio.

To get a dump from a memory controller using Code Warrior, follow these steps:

1. Open Memory view at runtime, and select Export option, as shown in the figure below.

(x)= Vari 💁 Bre 👷 Exp	o 🎁 Peri 📋 Me	🛛 🛄 Targ	🛋 Mod		
Monitors 🛛 🛖 💥 🍇	0x10:0x10 <hex> 😒</hex>	🚽 New Rend			
Ox10	Address	0 - 3	4 - 7	8 - B	c ^
	00000000000000010	3E000094	29000094	14000094	
	000000000000000000000000000000000000000	49000094	207B00B4	00001FD6	1

The **Export Memory** dialog appears, as shown in the figure below.

🥦 Export Memory	– 🗆 X
Format: Plain Text \checkmark	
Start address: 0x10 End address: 0x3f8	Length: 1000
File name: C:\Users\nxa11975\Desktop\DDR\annotated_hex.txt	Browse
?	OK Cancel
5. Export Memory dialog	

- 2. Specify the start and end memory addresses from where registers will be read, the number of registers, and the name and location of file where the registers will be dumped.
- 3. Click **OK**. The CodeWarrior memory dump file will be generated. For more information about the export memory dump, see ARMv8 Targeting Manual.

1.1.3 Exporting DDR configuration

The DDR configuration tool allows you to export memory dump, which can be in a format, such as S-Record, AnnotatedHex, Signed/Unsigned decimal, CodeWarrior Regs, Hex, or Raw Binary.

The DDR configuration is exported as a memory dump that can be shared with others and/or imported into the tool later.

To export a DDR configuration, create a QorlQ Configuration project. You can export the DDR configuration for any component already existing in the project. Click the **Export** tab and set the following parameters:

- · Output file: Specify the file where you will write the memory dump
- · File format: Choose the supported format of the memory dump into which you want to export the DDR configuration
- Endianness: Specify byte order

	mc1 🕴 🕎 Components Library Basic Advanced	
	dation	
Output file C:\Users\nxa11975	\Desktop\DDR\LX2160A_v2_0_annotated_hex.txt	Browse
File format Annotated Hex Text	t ~	
Endianness Big Endian	~	
	Export	

1.1.4 Advanced DDR configuration options

This section provides some advanced configuration options to quickly configure DDR controller.

The section contains the following subsections:

• DDR configuration by reading SPD

• DDR configuration by importing from target

1.1.4.1 DDR configuration by reading SPD

DDR controller can be configured quickly using the **Read SPD** option available on the **DDR configuration** page of the **New QorlQ Configuration Project** wizard.

This option allows you to perform DDR configuration according to the DIMM vendor recommendation. It helps you to determine an optimized DDR configuration in less time.

Configuring DDR controller using serial presence detect (SPD) involves the following steps:

- 1. Select an update package from the **Configure** drop-down list.
- 2. Select the Configuration mode as Read SPD.

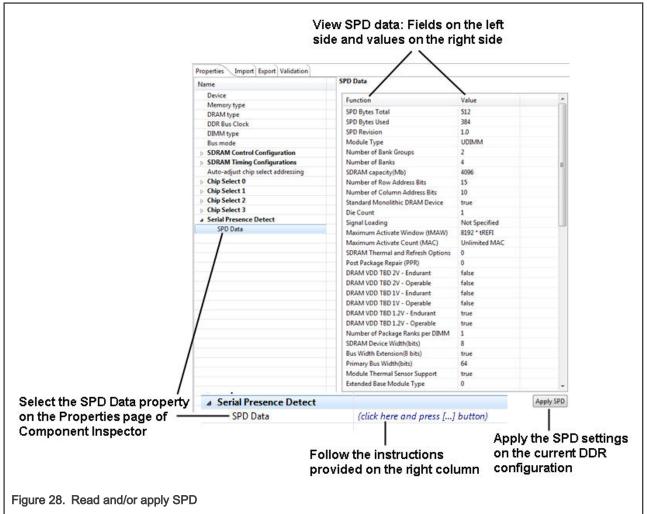
DDR configuration Configured device L								
Configure: 1st D	DR Controller]~
Configuration mod	le							
O Auto con	figuration	O From	memory file	۲	Read SPD	0) From target	
O Discrete DRAM	DIMM							
Read SPD	Probe:						10.171.77.115	*
Clock settings								
Input DDR clock:	100.0 MHz 🛛 🗸	DDR PI	LL Multiplexer:	29:1	Output	data rate:	2900 MT/s	1
DDR Controller			DRAM Setting	s				
Туре	DDR 4	\sim	DRAM configu	ration pe	r device	16Gb: 2G	8×8	\sim
Rank/Chip select	2	\sim	DIMM speed r	ating		PC4-1280	0 (1600)	\sim
Data bus width	64 bits	\sim						
tCL CC enabled DBI enabled	20 clocks	~	Quad-rank			SO-DIN	d DIMM	
Skews								
CLK to DQS 0	0 0	0	0 0	0	0	0 mm		
Lead-in 0	mm							
Import from file								
Input file 🖁							Brows	se
File format	CodeWarrior Reg	ister Text	•	~	First add	lress from	memory dump t	file
Endian mode	Little Endian			\sim	108000	0		
Addressable Size				\sim	Start ad		t DDR controller	
			Import					
Unsupported raw ca	rd revision. Set the	CLK to [QS skew values	manually				
?			< <u>B</u> ack	Nex		<u>F</u> inis		ancel

- 3. Choose the debug device from the **Probe** menu and specify the probe ID or IP address in the text field.
- 4. Click the **Read SPD** button. A green tick mark sign appears next to the **Probe** text box if SPD is read successfully. Disabled parameters are read from SPD.

5. Click the Finish button. This button is only enabled if SPD is read successfully.

The read SPD information is used to create the DRAM part of a DDR configuration. Later on, you can view what has actually been read from SPD using the following steps:

- 1. Finish creating the DDR component using the Read SPD configuration mode.
- 2. Go to the DDR component and open it in **Component Inspector**.
- 3. Select Serial Presence Detect > SPD Data on the Properties page.
- 4. Click where you are instructed to (go to the second table column, and then click the (...) button) and a new window opens on the right side. In that window, complete information about SPD is displayed, with the SPD fields on the left column and their values on the right column, as shown in the figure below. For some fields, multiple values have been provided by the vendor; such values are separated by a comma.



5. Click the Apply SPD button to apply the SPD settings on the current DDR configuration.

1.1.4.2 DDR configuration by importing from target

Another option to configure DDR controller quickly is by copying the DDR configuration of an already configured working target.

You can use this configuration option either while creating a new QorIQ configuration project or after creating the project. Both approaches are explained below:

• While creating a new QorIQ configuration project:

- 1. Initiate creating a new QorIQ configuration project with DDR component and go to the **DDR configuration** page of the **New QorIQ Configuration Project** wizard.
- 2. Choose the DDR controller from the Configure menu on the DDR configuration page.
- 3. Select From target as the configuration mode.
- 4. Choose the probe type and probe ID from the Probe menu and specify the probe ID or IP address in the text field.
- 5. Click the **Read from target** button. A green tick mark sign appears next to the **Probe** text box if SPD is read successfully, as shown in the figure below.

Configure: 1st D	DR Controller							\sim
Configuration mod	e							
O Auto con	figuration	O From	memory file	0	Read SPD		From target	
Discrete DRAM	DIMM							
Read from targ	et Probe:						10.171.77.115	~
Clock settings								
Input DDR clock:	100.0 MHz 🗸 🗸	DDR PLL	Multiplexer:	29:1	Output da	ata rate:	2900 MT/s	-
DDR Controller			DRAM Setti	ngs				
Туре	DDR 4	\sim	DRAM conf	iguration	per device	2Gb: 512	2Mb x4	\sim
Rank/Chip select	2	\sim	DIMM spee	d rating		PC4-128	00 (1600)	\sim
Data bus width	64 bits	\sim						
tCL ECC enabled DBI enabled	22 clocks	~		ed DIMM nked DIMI	М	SO-DI	MM red DIMM	
Skews								
CLK to DQS 0 Lead-in 0	0 0 mm	0	0 0	0	0 0	mm		
Import from file								
Input file 🔍							Brov	vse
City Connect	Code Marcine Dec			~	First add	ess from	memory dump	file
File format Endian mode	CodeWarrior Reg	ister lext		×	1080000			
Addressable Size	Little Endian 1 byte			× ×	Start add		t DDR controlle	
			Import					

- 6. Click the Finish button. This button is only enabled if the target is read successfully.
- After creating a QorIQ configuration project:

- 1. Select the DDR component and go to the Import tab of Component Inspector.
- 2. Choose Import From Target from the File format menu.
- 3. Click the **Import** button. The import operation starts and the import progress is displayed in a progress window, as shown in the figure below.

irget connecti	-		0 ø	Properties Import Export Valia	dation	Basic	Advanced 🎽 🖇
Processor	Probe Type	Probe Address		Memory dump settings			
LX2160A	cwtap	10.171.77.80					
				File format Import From T	arget 🗸 🗸		
				Addressable size 1 byte	\sim		
				Endianness Big Endian	~		
				Address information			
Components -	lx2160a 🔀		🖻 🖆 🝓 🕴 🗖 🗖	First address from memory dum	p file 1080000		
👝 Generator_(🎆 LX2160A	Configurations			Start address of DDR memory co	ntroller 1080000		✓ Use default
> Componen DDR_mc			Progress Information Import from target				
Problems 🔀						7	' ॄ 8 □ □
ms			-		Cancel	Path	Location
scription						Path	Location
							>

Eventually, a status window appears indicating whether the import was successful or not, as shown in the figure below. If the import was successful, then the new values are added directly to the DDR component. The registers that change will turn yellow in the **Configuration Registers** view.

DDR Configuration and Validation

Project Explore	r 🛄 Connectio	ns View 🔀		🗞 *Component Inspector - DDR_mc1 🛛	🕎 Components Library		- 8
Target connectio	ons					Basic Advan	ced 🎽 🖇
🔮 🗉 🗙			0 \$	Properties Import Export Validation			
				Memory dump settings			
Processor	Probe Type	Probe Address 10.171.77.80					
LX2160A	cwtap	10.171.77.80		File format Import From Target	\sim		
				Addressable size 1 byte	~		
				Endianness Big Endian	~		
					· ·		
				Address information			
Components -			🖻 💼 월 🕴 🗖 🗖	First address from memory dump file	1080000		
Generator_C 💮 🦢				Start address of DDR memory controller	1080000		Use default
B OSs	_v2_0_Cm				Import		
Processors							
> 💮 SoC:LX21 Component 🗁			Market Complete		×		
😗 DDR_mc			- import complete				
			Input file has been in	nported			
					ОК		
Problems 🔀						7 8	8
tems			~				
escription					Resource P	ath	Location
				· · · · · · · · · · · · · · · · · · ·			>
				8 200M of 2	256M 🔟		
		eration co					

In each of the two approaches mentioned above, if the controller being read is not initialized, then the data is not imported and an error message is displayed.

1.2 DDR validation

This section describes the DDR validation (DDRv) tool and explains how to use the tool to validate and optimize a DDR configuration using different validation scenarios.

The DDR validation tool can be used to validate a DDR configuration either created using memory specifications or read from a memory dump or target. The validation process is performed in stages, by gradually refining an initial DDR configuration up to an optimal configuration. Each validation step is responsible with the verification and optimization of specific DDR parameters. The outcome of a validation stage represents the input to the next one; therefore, gradually refining the DDR settings.

DDRv is a licensed product; therefore, it requires a valid license. See Licensing for more information related to licensing.

This section explains:

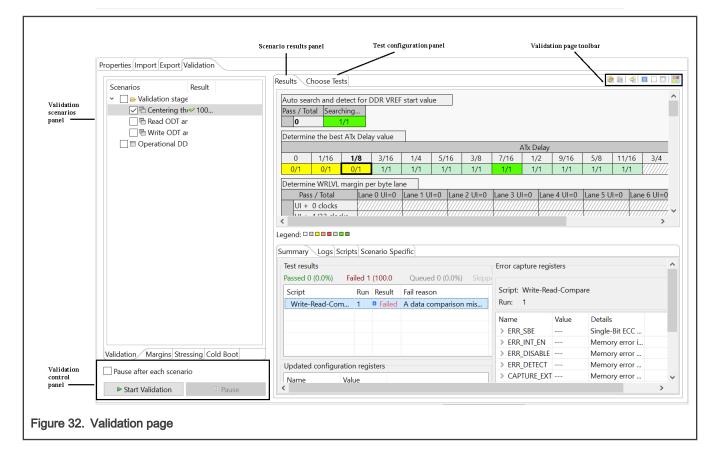
- Overview of DDR validation UI
- Using DDR validation tool
- Advanced DDR validation operations
- Licensing

1.2.1 Overview of DDR validation UI

This section describes the different elements of the DDR validation user interface.

The **Component Inspector** view displays a dedicated page, named **Validation**, for the validation of a DDR component. The **Validation** page, shown in the figure below, represents the GUI of the DDR validation tool.

A working target connection is mandatory to run DDR validation. For instructions to set up a target connection, see QCVS Getting Started Guide.



NOTE

To validate DDR using an existing QorlQ configuration project that does not have a DDR component, add a new DDR component to the project from the Components Library.

The subsections below explain the different panels of the Validation page:

- Validation scenarios panel
- Validation control panel
- Test configuration panel
- Scenario results panel
- · Validation page toolbar

1.2.1.1 Validation scenarios panel

The validation scenarios panel lists the predefined DDR validation scenarios.

A scenario represents a test approach for testing DDR. A scenario takes a DDR configuration as input, applies changes to it, verifies each change against several tests, and delivers an optimal DDR configuration. A particular case is the Operational DDR tests scenario, which takes the newly created DDR configuration as input and verifies it against several tests without applying any changes to it. The scenarios are grouped into: Validation, Margins, and Stressing. For LX series targets, there is one more scenario: Cold Boot.

Usually, a scenario takes the following form:

- It modifies the input DDR configuration by programming the DDR controller memory-mapped registers with a different set of values. This results in a set of derived DDR configurations.
- Each derived DDR configuration is tested against several types of tests. Depending on the scenario, you can select the tests from a predefined list or the tests are automatically selected by the tool.
- The test results are displayed in a tabular format, with each cell in the table representing a derived DDR configuration. The results indicate which parameters are varied and how they are varied as well as an overview of the passed/failed DDR configurations.
- After completion of all the tests, an optimal DDR configuration is determined that represents the input to the next scenario

To access any of the derived configurations (including the optimal configuration), click the corresponding cell. This loads the corresponding DDR configuration. You can export the DDR configuration to a file or obtain DDR initialization code out of it.

1.2.1.2 Validation control panel

The validation control panel allows you to control the validation process.

You can control the validation process to start, stop, pause, or resume it. This is performed by two buttons, **Start Validation** and **Pause** that are present above the target connection panel.

1.2.1.3 Test configuration panel

The test configuration panel allows you to configure the validation tests.

The DDRv tool provides several predefined tests, some of them are configurable. To achieve the optimal DDR configuration for each scenario, a test can be run multiple times. You can choose and configure the tests you want to run for a scenario, using the **Choose Tests** page (some scenarios do not have configurable tests to choose from). You can add your own tests using **Add custom test** button located in the right upper side of the tests list. When you add a new test to the list, a default test is created using a template that you have to fill in with your own DDR testing code. The code has to be written in Python syntax.

You can remove or rename the user-defined (custom) tests using a context menu of each test. Any test from the list can be viewed by double-clicking it. The test content is displayed in a new window. You can modify the test content and also see the DDR default test scripts.

1.2.1.4 Scenario results panel

The scenario results panel summarizes the results of the validation tests.

It is located in the center of the **Validation** page inside the **Results** page. It is organized into two sections: high-level summary and detailed summary. The high-level summary is organized into one or more tables that show the following:

- All the variations of the scenario's input DDR configuration. Each input derivative is abstracted as a cell.
- · The validation progress while it is running

A variation table is a visual representation of how the input DDR configuration for a scenario varies while the scenario runs. The validation progress is displayed using colors and numbers. The color code helps in visualizing the results of the executed scenario. The colors and numbers are decoded as follows:

- The current cell that is being tested flashes
- The cells, which are being validated displays the number in fraction. The numerator represents how many tests have passed, while the denominator represents how many tests were run for that cell
- Each cell is colored based on the validation result (see the figure below):
 - Red: Indicates that validation failed for the cell due to target connection issues or unknown causes
 - Yellow: Indicates that validation failed for the cell because it could not pass some tests

- Orange: Indicates that validation failed for the cell due to incorrect DDR configuration
- Light green: Indicates that the cell passed all tests
- Bright green: Indicates that the cell has optimal DDR configuration

In case of multiple errors, the color code displayed for a cell corresponds to the most critical error occurred for the cell during the test iterations. The order of criticality from high to low is: red, yellow, orange.

	ermine best v		rite levelin	ig start val	ues											
	s / Total Se															
	0 1/1															
Dete	ermine the b	est clock a	djust valu	e												
								CLK_AD	J							
(0 1/16	1/8	3/16	1/4	5/16	3/8	7/16	1/2	9/16	5/8	11/16	3/4	13/16	7/8	15/16	Т
0/3	0/3	0/3	0/3	0/3	0/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	0
Det	ermine WRL\	/ margin	oer hute l	100												
	ass / Total	Lane 0	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8 (EC	0					
	1/8 clocks	conco	Corre a	COTTO	cones	Corre 4	come o	conco	corre /	conco (co	<u>c</u> ,					
	1/4 clocks	<u> </u>	<u> </u>								-					
	3/8 clocks	-	-								-					
	1/2 clocks	0/3	-								-					
Ì	5/8 clocks	0/3	0/3								-					
	3/4 clocks	3/3	0/3	0/3												
Ī	7/8 clocks	3/3	3/3	0/3	0/3											
	1 clocks	3/3	3/3	3/3	0/3					0/3						
	9/8 clocks	3/3	3/3	3/3	3/3					0/3						
	5/4 clocks	3/3	3/3	3/3	3/3	0/3				3/3						
																•
	d: 🗆 🕄 🗖 🗖		-													

The detailed summary displays detailed information about the cell selected in the high-level summary. The information is organized into three pages:

- Summary page: Shows the DDR configuration register values that were updated for the selected cell along with the detailed status (queued, passed, failed, or skipped) and error register values for each test execution
- Logs page: Displays the execution log for each test execution, redirected from CDDE log. You can use this log to understand why a test failed or passed.
- Scripts page: Shows the exact Python scripts that were executed on the target using CDDE. You can open a script in a separate editor by clicking the **Open in editor** button, located at the top-right corner of the **Scripts** page.
- Scenario Specific page: Shows the write-leveling start value for each data strobe (DQS). It is available only for the **Determine** the best clock adjust value table.

The figure below shows the detailed summary.

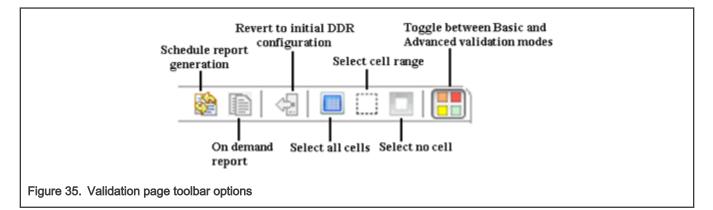
etermi	ine best wo	orking wri	te levelir	ng start va	ues										
	otal Searc	ching		-											
0	1/1														
etermi	ine the bes	t clock ad	ljust valu	ie											
								CLK_AD)]						
0	1/16	1/8	3/16		5/16	3/8	7/16	1/2	9/16	5/8	11/16 3/4			15/1	6
/3	0/3	0/3	0/3	0/3	0/3	3/3	3/3	3/3	3/3	3/3	3/3 3/3	3/3	3/3	3/3	
etermi	ine WRLVL	margin p	er byte l	ane											
	(Total	Lane 0	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8 (ECC)				
1/8	clocks										7				
	clocks														
	clocks										_				
		0/3									_				
			0/3	0.0							_				
			0/3 3/3	0/3 0/3	0/3						-				
-		-	3/3	3/3	0/3					0/3					
	ioeks j.														
end: 🗆 mmary	l 🖾 🗖 🖡			a/a	3/3	<u> </u>		Erro	·	n/3	1				,
end: 🗆 mmary Test res	l 🖾 🗖 🖡	Scripts S] cenario S	opecific	1 A A A A A A A A A A A A A A A A A A A) Skippe	m ed 0 (0.0%)		pr capture		1				,
end: 🗆 mmary Test res	Logs	Scripts S] cenario S	opecific	3/3			Sc	or capture ript: Writ		pare				,
end: mmary Test res Passed : Script	Logs	Scripts S] cenario () (0.0%)	Specific Queu	3/3 ed 0 (0.0%)			Sc	or capture	registers	pare				
end: mmary Test res Passed : Script Write-] 🖾 🗖 🔲 [Logs] sults 3 (100.0%)	Scripts S Failed (] cenario 9 0 (0.0%) Run 1	Specific Queu Result	3/3 ed 0 (0.0%)			Sc	or capture ript: Writ	registers	pare	Details			,
end: nmary Test res Passed : Script Write- Write-	Logs	Scripts S Failed (mpare mpare] cenario S 0 (0.0%) Run 1	Specific Queu Result Passed	3/3 ed 0 (0.0%)			Sc Ru Ni	or capture rript: Writ un: 1 ame	registers e-Read-Com	Value	Details			
end: nmary Test res Passed : Script Write- Write-	Logs sults 3 (100.0%) Read-Cor -Read-Cor	Scripts S Failed (mpare mpare	2 (0.0%) Run 1 2	Specific Queu Result Passed Passed	3/3 ed 0 (0.0%)			Sc Ru Ni	or capture cript: Writ un: 1 ame ERR_1	registers e-Read-Com SBE	Value 0x00010000	Details			
end: nmary Test res Passed : Script Write- Write-	Logs sults 3 (100.0%) Read-Cor -Read-Cor	Scripts S Failed (mpare mpare	2 (0.0%) Run 1 2	Specific Queu Result Passed Passed	3/3 ed 0 (0.0%)			Sc Ru Ni	or capture ript: Writ un: 1 ame ERR_ & ERR_1	registers e-Read-Com SBE INT_EN	Value	Details			
end: nmary Test res Passed : Script Write- Write-	Logs sults 3 (100.0%) Read-Cor -Read-Cor	Scripts S Failed (mpare mpare	2 (0.0%) Run 1 2	Specific Queu Result Passed Passed	3/3 ed 0 (0.0%)			Sc Ru Ni	or capture ript: Writ un: 1 ame ERR_ & ERR_ ERR_	registers e-Read-Com SBE	Value 0x00010000 0x0000001d	Details			
end: nmary Test res Passed : Script Write- Write-	Logs sults 3 (100.0%) Read-Cor -Read-Cor	Scripts S Failed (mpare mpare	2 (0.0%) Run 1 2	Specific Queu Result Passed Passed	3/3 ed 0 (0.0%)			Sc Ru Ni D	or capture ript: Writ un: 1 ame ERR_ ERR_ ERR_ ERR_	registers e-Read-Com SBE INT_EN DISABLE DETECT	Value 0x00010000 0x0000001d 0x00000000	Details			
end: nmary Test res Passed : Script Write- Write-	Logs sults 3 (100.0%) Read-Cor -Read-Cor	Scripts S Failed (mpare mpare	2 (0.0%) Run 1 2	Specific Queu Result Passed Passed	3/3 ed 0 (0.0%)			Sc Ru Ni D	or capture ript: Writ un: 1 eme ERR_ ERR_ ERR_ CAPT	registers e-Read-Com SBE INT_EN DISABLE DETECT	Value 0x00010000 0x0000001d 0x0000000 0x0000000	Details			
end: nmary Test res Passed : Script Write- Write-	Logs sults 3 (100.0%) Read-Cor -Read-Cor	Scripts S Failed (npare npare npare	I cenario 5 0 (0.0%) Run 1 2 3	Specific Queu Result Passed Passed	3/3 ed 0 (0.0%)				or capture ript: Writ un: 1 erre ERR_ ERR_ ERR_ CAPT CAPT	registers e-Read-Com SBE INT_EN DISABLE DETECT FURE_EXT_AD FURE_ECC	Value 0x00010000 0x0000001d 0x0000000 0x0000000 0x00000000	Details			
end: mmary Test res Passed : Script Write- Write- Write- Update	Logs sults 3 (100.0%) Read-Cor Read-Cor Read-Cor	Scripts S Failed (npare npare npare	(0.0%) 0 (0.0%) Run 1 2 3 3	Specific Queu Result Passed Passed	3/3 ed 0 (0.0%)			Sc Rt D	or capture ript: Writ ame ERR_1 ERR_ ERR_ CAPT CAPT CAPT	registers e-Read-Com SBE INT_EN DISABLE DETECT FURE_EXT_AE FURE_ECC FURE_DATA_1	Value 0x00010000 0x0000001d 0x0000000 0x0000000 0x00000000 0x000000	Details			
nmary Passed 3 Script Write- Write- Update	Logs sults 3 (100.0%) Read-Cor Read-Cor Read-Cor	Scripts S Failed (npare npare npare ration reg	Cenario S 0 (0.0%) Run 1 2 3 3 isters Value	Queu Queu Result Passed Passed Passed	3/3 ed 0 (0.0%)			Sc Rt D D D D D D D D D D D D D D D D D D	or capture rript: Writ ame ERR_ ERR_ ERR_ CAPT CAPT CAPT CAPT	registers e-Read-Com SBE INT_EN DISABLE DETECT FURE_EXT_AE FURE_EXT_AE FURE_EATA_I	Value 0x00010000 0x000001d 0x0000000 0x0000000 0x0000000 0x0000000 0x0000000 0x0000000	Details			
end: mmary Test res Passed 3 Script Write- Write- Write- Updates Name SDRAH	Logs sults 3 (100.0%) Read-Cor Read-Cor Read-Cor	Scripts S Failed (npare npare npare ration reg	Cenario S 0 (0.0%) Run 1 2 3 3 isters Value 0x02c00	Queu Queu Result Passed Passed Passed	3/3 ed 0 (0.0%)			Sc Rt D D D D D D D D D D	or capture rript: Writ ame ERR_ ERR_ ERR_ CAPT CAPT CAPT CAPT CAPT	registers e-Read-Com SBE INT_EN DISABLE DETECT FURE_EXT_AE FURE_EXT_AE FURE_DATA_I FURE_DATA_I	Value 0x00010000 0x000001d 0x0000000 0x0000000 0x0000000 0x0000000 0x0000000 0x0000000 0x0000000 0x0000000	Details			
end: mmary Test res Passed : Script Write- Write- Update Name SDRAN WRLVI	Logs sults 3 (100.0%) Read-Cor Read-Cor Read-Cor	Scripts S Failed (npare npare npare ration reg	Cenario S 0 (0.0%) Run 1 2 3 3 isters Value	Queu Queu Result Passed Passed Passed D000 0609	3/3 ed 0 (0.0%)			Sc Rt N. D D D D D D D D D D D D D D D D D D	or capture rript: Writ ame ERR_ ERR_ ERR_ CAPT CAPT CAPT CAPT CAPT	registers e-Read-Com SBE INT_EN DISABLE DETECT FURE_EXT_AE FURE_EXT_AE FURE_DATA_I FURE_DATA_I	Value 0x00010000 0x000001d 0x0000000 0x0000000 0x0000000 0x0000000	Details			

The high-level and detailed scenario execution summaries are available during and after the scenario has executed. The information is updated as the scenario execution progresses.

1.2.1.5 Validation page toolbar

The Validation page toolbar allows you to perform different operations before or during DDR validation.

The figure below shows the options on the Validation page toolbar.



You can perform the following operations using the Validation page toolbar options:

- Schedule report generation: Using a toolbar option, you can make validation reports generated based on a schedule. This option is available only before the validation starts.
- Revert to the initial DDR configuration: Using a toolbar option, you can revert the DDR configuration to the last known default configuration used in a previous validation. This option is available only before the validation starts. It is useful when running the validation multiple times over the same DDR configuration.
- Toggle between the Basic and Advanced validation modes: Using a toolbar option, you can toggle between the Basic and Advanced validation error view modes. This provides improved viewing of the validation failures that helps in determining the failure cause quickly.
- Choose the DDR parameters to test: You can choose which DDR parameters to test using one of the three icons on the toolbar: choose all, an area, or none. This option is useful when you want to perform a quick test of a specific variation without waiting for the full set to be tested.

1.2.2 Using DDR validation tool

This section explains how to use the DDR validation tool to perform validation of a DDR component.

Performing DDR validation involves the following operations:

- Choose validation scenarios and tests
- Customize validation tests
- Run validation
- · Report validation results
- Set DDR validation preferences

1.2.2.1 Choose validation scenarios and tests

After the tool is successfully connected to a validation server, you can choose the validation scenarios and tests that will be run during the validation of the DDR component.

You can choose:

- Which scenarios will be run: You can select the scenarios you want to run, using the left pane of the **Validation** page of Component Inspector (see the figure below). At least, one scenario needs to be selected. If multiple scenarios are selected, then they will run, one by one.
- Which tests will be run for a scenario: After selecting a scenario, you can select the tests you want to run for that scenario, using the **Choose Tests** page on the right pane of the **Validation** page (see the figure below)
- Which of the derived DDR configurations will be tested (all, some, or none)

Scenarios Result	Results Choose Tests				1	
✓ □	Select which tests the scen					
✓ ☐ Centering the 100	Sync selection across a	all scenarios	Add custom test	Write-Read-C	ompare parameters	
🗌 🖶 Read ODT ar	Test	Repet		Start address	0x0	hex
Write ODT ar	BIST-1Write-1Rea	0		Size	1	KB 🔻
🗌 🗉 Operational DD	BIST-2Write-2Rea	0		Use rando	m pattern	
	BIST-4Write-4Rea	0		Pattern	m pattern	
	BIST-Write-then-R	0			0.55 0.00 0.01 0.02 0.45 0.57 0	er i
	DMA Test	0			0xCC, 0xDD, 0x01, 0x23, 0x45, 0x67, 0 xFF, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA,	arr, ^
	✓ Write-Read-Comp	1			0xDD, 0x01, 0x23, 0x45, 0x67, 0xFF, 0x	FF,
	Walking Ones	0			xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xBB,	
	Walking Zeros	0			0x01, 0x23, 0x45, 0x67, 0xFF, 0xFF, 0x	÷F,
					0xAA, 0xAA, 0xAA, 0xAA, 0xBB, 0xCC, 0x23, 0x45, 0x67, 0xFF, 0xFF, 0xFF, 0xF	c
				0xDD, 0x01, 0xAA, 0xAA,		г,
						\sim
				Test description	on	
				Checks DDR	reads versus writes performed using D	MA
					quentially writes into a DDR area a patt	
					ully covered. Then reads back from the	
					d compare with what has been written. art and size of DDR area and pattern t	
alidation Margins Stressing Cold Boot					t fails after the first identified memory	o be
Pause after each scenario				mismatch.	e lans alter die mist laenanda memory	
Start Validation						

1.2.2.2 Customize validation tests

While selecting tests to run for a validation scenario, you can customize settings related to these tests.

To access the tests, go to the **Choose Tests** tab (see the figure below) located on the **Validation** page of Component Inspector. In the **Choose Tests** tab, you can:

- · Select or deselect what tests will be run for a selected scenario
- Update the number of repetitions for each test (by selecting the test and typing a new value in the cell under the **Repetition** column)
- Add custom tests
- · Remove or rename a custom test (by right-clicking the test)
- · Edit the existing test script content (by double-clicking a test)

Scenarios Result	t Results Choose Tes	its				🕸 🗈 🝕 🗖		
 Validation stage 		the scenario will rur	n, and how many times					
Centering the clock) %	across all scenarios	Add custom test	Write-Read-C	Compare parameters			
🗌 🖷 Read ODT and driver	Test	Repet		Start address	0x0000000		hex	
🗌 🖻 Write ODT and driver	BIST-1Write-			Size	1	KB	-	
🗌 🗏 Operational DDR tests	BIST-2Write-					KB		
	BIST-4Write-	4Rea 0		Use random pattern				
	BIST-Write-th	ien-R 0		Pattern				
	DMA Test	0		1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	0xCC, 0xDD, 0x01, 0x23, 0		^	
	✓ Write-Read-C			0xFF, 0xFF, 0xFF, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xBB, 0xCC, 0xDD, 0x01, 0x23, 0x45, 0x67, 0xFF,				
	Walking Ones 0		0xFF, 0xFF, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xBB,					
	🗌 Walking Zero	s 0			0x01, 0x23, 0x45, 0x67, 0			
					0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0			
				0xDD, 0x01, 0xAA, 0xAA,	0x23, 0x45, 0x67, 0xFF, 0x	FF, UXFF, UXFF,		
							\sim	
				Warm boo	ot			
				Test description	on			
				Checks DDR	reads versus writes perfor	med using DMA		
					quentially writes into a DD			
					ully covered. Then reads b d compare with what has			
idation Margins Stressing Cold Boot					tart and size of DDR area a			
					st fails after the first identif			
Pause after each scenario				mismatch.				
► Start Validation ^{III} Pause								

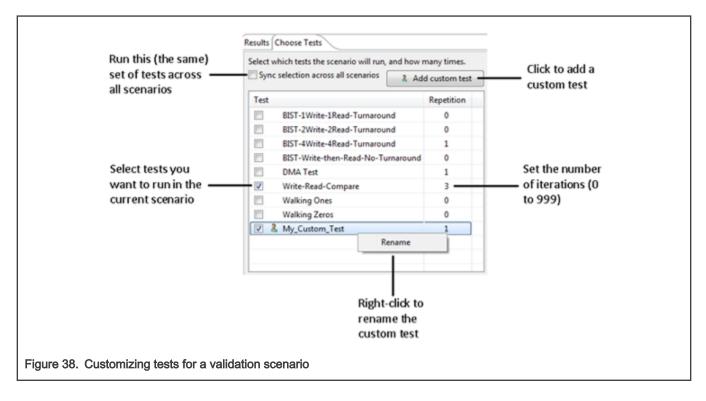
Some tests can only be selected for some particular scenarios. The set of tests for a validation scenario depends on:

- The validation scenario
- The device whose DDR is being validated

Some of the tests have customizable parameters. Those parameters depend on the test type, and you can modify them within several constraints. Whenever a constraint is broken, you will be notified about it using an error decorator on the UI.

Besides the standard tests provided by the tool, you can add your custom tests to the list of tests (see the figure below). A custom test consists of a template (a Write-Read-Compare template) that you can further enhance. Similar to standard tests, custom tests can be added, removed, or selected to be run for a validation scenario. The DDR tool provides the following additional options related to tests:

- · You can select each test (standard or custom) to be run multiple times for a scenario
- · You can use for all scenarios the set of tests you selected for a scenario



1.2.2.3 Run validation

When one or more test scenarios are selected, the Start Validation button gets enabled. You can click this button to start the validation of the DDR component.

A validation starts executing various configurations with each combination represented as a cell in the **Results** table as shown below. For example, you select the **Centering the clock** scenario and **BIST** and **Walking Ones** as tests. The **Centering the clock** scenario varies the DDR controller parameters, **CLK_ADJ** and **WRLVL_START**. When this scenario runs, the chosen tests are run for each of the two parameter variations.

	nine best w Total Sea		te leveling	y start valu	105												
eterm	nine the be	st clock ad	ljust value														
			·					CLK_AD	J								1
0	1/16	1/8	3/16	1/4	5/16	3/8	7/16	1/2	9/16	5/8	11/16	3/4	13/16	7/8	15/16	1	1
]
)eterm	nine WRLV	margin r	er hyte la	ne													
	/ Total	Lane 0	Lane 1	Lane 2	Lane 3	Lane 4	Lane 5	Lane 6	Lane 7	Lane 8 (ECC	3						
1/	8 clocks										-						
1/4	4 clocks										1						
3/0	8 clocks																
1/2	2 clocks																
-	8 clocks																
_	4 clocks																
_	8 clocks																
	clocks										_						
	8 clocks										_						
	4 clocks										-						
	/8 clocks																

When the validation step for the scenario is finished, the **Results** table shows whether or not the test passed. In addition, the bright green cell, which corresponds to the optimal DDR configuration, is automatically selected.

Before proceeding to the next validation step (that is the next test scenario), the tool applies the DDR configuration corresponding to the bright green cell of the last executed test scenario. In case none of the cells has passing configurations, then you should restart the validation step. In case there are few passing cells but you are not satisfied with the automatic selection of the bright green cell, then you can select a different passing cell to have its configuration applied for the next validation step.

1.2.2.4 Report validation results

After or during the validation, you can save the validation results of a scenario in a format, such as PDF, XLS, or CSV.

To print the results in a format, use the action buttons shown in figure below.

Results Choose Tests	🏽 📄 🔄 🔲 🔅 🗖
Determine best working write leveling start values Pass / Total O	
Figure 40. Report validation results	

In addition, you can set up a reporting job to auto-generate reports at different events, such as scenario completion or periodically after some time.

[Auto-generate DDRV Report	×
	Enable automatic generation of DDRV reports	
	File C:\Users\nxa11975\Downloads\hhh_new\ddrv_a Browse	e
	Type HTML web page \sim	
	Auto-generate every 10 Minutes	~
	Report Options	
	✓ Initial DDR configuration ✓ Detailed cell results	
	Cell updated registers	
	Error capture registers	
	OK Cancel	
Figure 41. Auto-generate I	DDRv report	

The report is generated periodically with all the validations executed until that time. Adjust the report generation settings, if needed.

	Generate DDRV Report X]
	File C:\Users\nxa11975\Downloads\hhh_new\ddrv_r Browse	
	Type HTML web page $$	
	Report Options	
	Initial DDR configuration	
	Detailed cell results	
	Cell updated registers	
	Error capture registers	
	OK Cancel	
Figure 42. On demand rep	port	

1.2.2.5 Set DDR validation preferences

This section explains you how to specify your preferences for using the DDR validation tool.

The **DDR validation** page of the **Preferences** dialog allows you to specify your preferences for using the DDR validation tool. To open the **DDR validation** page, follow these steps:

- 1. Choose **Window > Preferences**. The **Preferences** dialog appears.
- 2. Expand the **Processor Expert** node and click **DDR validation**. The **DDR validation** page appears, as shown in the figure below.

Preferences		— 🗆 X
type filter text	DDR Validation	
 General C/C++ ChangeLog Debug Server Connection Help 	General settings for DDR validation tool Additional Data Connection timeout (s) Test execution timeout (s)	30
> Install/Update > Library Hover > Mylyn	Write leveling searching method Note: This method ignores the user entered skews	Auto search & detect for write leveling start values \sim
NXP Licenses > Oomph > Processor Expert	Options Always perform reset between test iterations Generate random pattern for BIST tests	
DDR Validation General Paths Preboot Loader SerDes Validation	Constant of the second	
 Remote Development Remote Systems RPM Run/Debug Software Analysis 	Example: 1:0x50304040,2:0x50500050,3:0x00000000 RCW Example: 0x9F),4:0×0000000,5:0×0000000,6:0×000
 > Team > Terminal > Tracing Validation > XML 		Restore <u>D</u> efaults <u>Apply</u>
? 🎽 🖉 🔘		Apply and Close Cancel
Figure 43. DDR validatio	n page	

1.2.3 Advanced DDR validation operations

This section explains some advanced DDR validation operations using some example use cases.

The section is divided into the following subsections:

- DDR validation using margin scenarios
- DDR validation using stressing scenarios
- DDR validation with warm boot
- Validation code customization

1.2.3.1 DDR validation using margin scenarios

The DDR validation tool provides two scenarios for validating data bus read/write margin: read margin and write margin.

Using read margin or write margin scenario, you can perform the following two types of validation tests:

- 1D margin test
- 2D margin test

1.2.3.1.1 1D margin test

1D margin test is used for DDR validation when DDR data rate of the processor is less than or equal to 1600 MT/s.

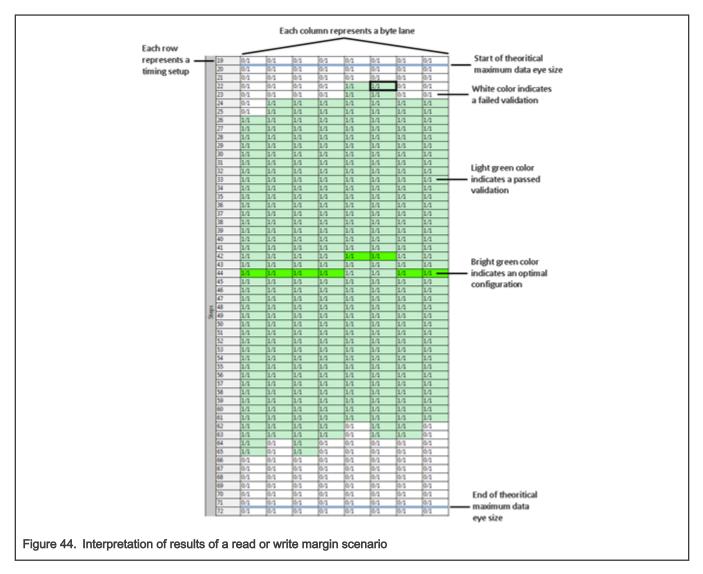
The validation process for read/write data bus margin scenario using 1D margin test involves the following operations:

- 1. The read or write margin validation scenario allows the memory controller to complete its normal initialization process and perform the normal operation.
- 2. The timing location of each strobe corresponding to its data byte lane (golden value) is stored.
- 3. The scenario sweeps the strobe signal in the data eye for each byte lane, using small timing steps. At each timing step during the sweep:
 - Randomly generated data is transferred via DMA engine from one memory space in DDR to another and values at the destination are compared with generated data to determine if the test is passed or failed.
 - When the DMA test is passed, the corresponding cell/step is marked as passed and is displayed as green cell in the generated margin table.
 - When an error is detected, the cell is marked as failed and is displayed as white cell in the margin table.
- 4. After all data byte lanes have gone through this process and have determined the passed and failed cells, then the location of strobe where memory controller was selected during its initialization process (that is, golden value) is added to the margin table.

NOTE

Not all QorlQ families support this feature. If the scenarios described here are not visible in your DDRv GUI, then it implies that the DDR controller in the processor you are working with does not support this feature (the support has not yet been added to QCVS for these scenarios for that particular processor). The QorlQ processors (B4/G4/T) were the first ones to support this feature.

Every time the read or write margin test is run, the memory controller is initialized at the start. The resulting display is a reconstructed margin data eye, including the strobe signal crossing location within the data eye. A simple visual inspection of location of strobe with the reconstructed margin data eye will show the number of timing steps before a failure could occur.



In the figure above, each cell in each lane column shows a combination of two digits. The first digit indicates the number of tests passed. The second digit indicates the total number of tests conducted.

The size of each step can be calculated by dividing the data rate unit interval by the number of cells between the two blue lines that indicate the start and end of the theoritical maximum data eye size.

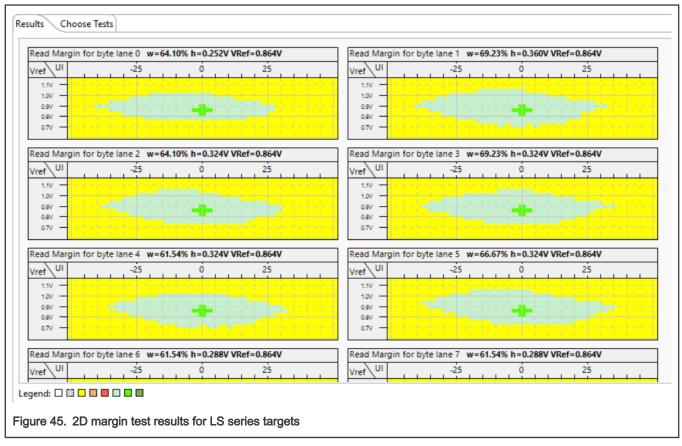
For additional details about the margin scenarios, see *QCVS FAQ Guide*.

1.2.3.1.2 2D margin test

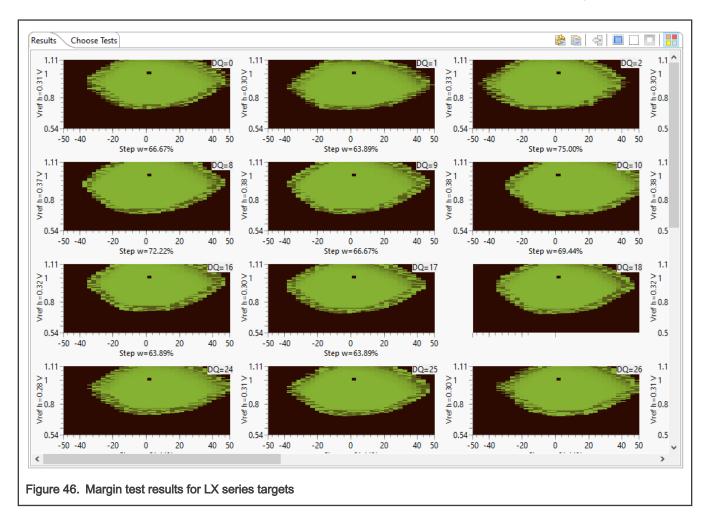
2D margin test is used for DDR validation when DDR data rate of the processor is greater than 1600 MT/s.

2D margin test is based on data eye margin, which is a post-operational reconstruction of data eye. In 2D margin test, DMA test is used to define pass and fail for each timing/voltage step. The strobe is shifted in steps to traverse the data unit interval (x-axis). For each timing step, V_{REF} is shifted to traverse between the available range of V_{REF} values (y-axis) and GVDD. The data eye reconstruction is done for each byte lane. The results are oscilloscope like reconstructed data eye margins for each byte lane and the location of trained DQS within that data eye.

The figure below shows the results of performing 2D margin test on LS series targets.



The figure below shows the results of performing margin test on LX series targets using a diagnosis tool from Synopsys.



1.2.3.2 DDR validation using stressing scenarios

The DDR validation tool provides two scenarios for checking the memory subsystem for faults: Stress tests and MemTester. You can run these scenarios individually or sequentially. Each scenario contains several stress tests that run back-to-back on the target system. You can modify source address and size for each test.

The figure below shows the different tests available for Stress tests scenario.

Scenarios Result	Results Choose Tests				8 9 4 5 5 5
Stress tests	Select which tests the scenario will r		s.		
memTester	Sync selection across all scenario	Add custom test	Data is Address	parameters	
☐	Test	0x0	hex		
	Data is Address	1	Size	32 MB	~
	Row hop read				
	SSN memcpy x32	1	Warm boot		
	Byte SSN memcpy x32				
	Memcpy pseudo random d	1			
	✓ IRAM to DDR x32	1			
	✓ IRAM to DDR v2 x32	1			
			Test description		
			Stress Test Suite	9	
alidation Margins Stressing Cold Boot					
Pause after each scenario					
▶ Start Validation □ Pause					
Touse Touse					

The figure below shows the different tests available for MemTester scenario.

Scenarios Result	Results Choose Tests				🕸 🕼 🖪 🗖 🗖 🗖
Stress tests	Select which tests the scenario wi				
MemTester	Sync selection across all scena	rios Add custom test	Stuck Address T	lest parameters	
MemTester	Test	Repetition	Source address	he	
	Stuck Address Test	1	Size	1	КВ 🕶
	Random Value Test	Random Value Test 1			KD
	Compare XOR Test	1	Warm boot		
	Compare Sub Test	1			
	Compare Mul Test	1			
	Compare Div Test	1			
	Compare Or Test	1			
	Compare And Test	1			
	Sequential Increment Test	1			
	Solid Bit Test				
	Block Sequential Test	1			
	Checkerboard Test	1	Test description		
	Bit Spread Test	1	Memtester test	suite	
	Bit Flip Test	1			
	Walkbits1 Test	1			
	Walkbits0 Test	1			
lidation Margins Stressing Cold Bo	pot				
Pause after each scenario					
▶ Start Validation	JSe				

1.2.3.3 DDR validation with warm boot

This section contains the following subsections:

- Running cold boot scenario test
- Running a test with warm boot option

1.2.3.3.1 Running cold boot scenario test

The DDR validation tool provides one cold boot scenario: Save PHY registers. This scenario is only available for LX series targets.

The Save PHY registers scenario has one test, which performs a cold boot, and saves the PHY registers right after the firmware training is executed. It also saves the DDR registers. All the registers are saved in a dump file within the workspace.

The figure below shows the test available for Save PHY registers scenario.

Scenarios	Result	Results Choose Tests		🗞 🖻 🗟 🗖 🗆 🗆
Save PHY registe		Select which tests the scenario will ru	n, and how many times	
		Sync selection across all scenarios	⁴ Add custom test	Test description
		Test	Repetition	Performs PHY training and saves the values to a dump file.
		Cold Boot - Save Registers	1	Saves also the DDR registers.
alidation Margins Stressi Pause after each scenari	0			
▶ Start Validation	III Pause			

NOTE

If the active DDR configuration has changed (for example, by importing new configuration from target, by running a validation test, or by changing DDR settings within Component Inspector) or PORESET was asserted, then the cold boot scenario test should be run again to update the dump file. If you try to run a validation test with warm boot ON without running cold boot again, then you will get a warning message, as shown in the figure below.

Warm boot is ON! Warm boot is ON! Make sure Cold-boot is executed after each PORESET! In case the DDR configuration changed, run Code-boot again! Press OK to continue with validation or Cancel to skip!	
Ok Cancel	

1.2.3.3.2 Running a test with warm boot option

The warm boot option is only displayed for LX series targets. This option is available for tests of many validation scenarios, such as Operational DDR tests, Stress tests, and MemTester. It is not available for tests of other scenarios, such as Centering the clock.

When a validation test is run with warm boot option selected, the PHY training is skipped, and the DDR controller and PHY are initialized using a dump file that was created earlier while running the cold boot scenario test. If the cold boot scenario test was

not run before (the dump file with registers was not found in the workspace), the warm boot option is disabled, as shown in the figure below.

Scenarios Result	Results Choose Tests			10 I	s 🗖		
Validation stage	Select which tests the scenario will run	, and how many times					
Centering the clock	Sync selection across all scenarios	^{&} Add custom test	Write-Read-Co	ompare parameters		^	
Operational DDR tests BIST-2Write-2Read-1 BIST-Write-4Read-1 BIST-Write-then-Rea DMA Test	BIST-1Write-1Read-Turnaro BIST-2Write-2Read-Turnaro BIST-4Write-4Read-Turnaro BIST-Write-then-Read-No-T	Repetition 0 0 0 0 0		1	KB		
	Write-Read-Compare	3	0xBB, 0xCC, 0	0xDD, 0x01, 0x23, 0x45, 0x67, 0xFF,	0xFF,		
	Walking Zeros	0	0xFF, 0xFF, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xBB, 0xCC, 0xDD, 0x01, 0x23, 0x45, 0x67, 0xFF, 0xFF, 0xI 0xFF, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xBB, 0xCC, 0xDD, 0x01, 0x23, 0x45, 0x67, 0xFF, 0xFF, 0xFF, 0xFF 0xAA, 0xAA, 0xAA, 0xAA				
			Warm boo	t			
			transfers. Seq until area is fu DDR area and	reads versus writes performed usin uentially writes into a DDR area a p Illy covered. Then reads back from I compare with what has been writt	oattern, the same ten. User	2	
lidation Margins Stressing Cold Boot				art and size of DDR area and patter t fails after the first identified memo			
Pause after each scenario			mismatch.	rais arter the hist dentined menn	.,,		
Start Validation						~	

If you need to run a test with warm boot option selected after the DDR configuration was changed (for example, by importing new configuration from target, by running a validation test, or by changing DDR settings within Component Inspector), then run the cold boot scenario test first to ensure that you run the warm boot test with active configuration.

NOTE The cold boot test must be run before running a warm boot test if PORESET was asserted or if DDR configuration was changed.

1.2.3.4 Validation code customization

This section describes how to customize the code used by the DDR validation tool.

The DDR validation tool communicates with the target using GTA. The actual communication is facilitated by GDB extensions that allow to execute target operations using Python scripts.

The DDR validation tests run as applications on the device cores; therefore, their customization is hidden. However, the pre- and post-validation operations (for example, target reset, applying the DDR configuration, and clearing the error debug registers) can be customized as they are written in Python. Such customizations could take the form of additional operations required at reset.

If needed, you can make such customizations (for example, performing an RCW override operation before target reset happens) by modifying the default code of the DDR validation tool. The Python scripts used by DDRv are located at *<INSTALL_DIR>ICommonIQCVSIOptimizationIresourcesIQorIQIARMv8IcommonI** and *<INSTALL_DIR>ICommonIQCVSIOptimizationIresourcesIQorIQIARMv8IcommonI**.

1.2.4 Licensing

The DDR validation tool is a licensed software.

When QCVS is installed over CodeWarrior for PA/ARMv7/ARMv8, the CodeWarrior license file is used.

In case a license file is not found, the needed license key does not exist or has expired, or the license server is not responding, then such details are displayed on the **Validation** page of **Component Inspector** for the DDR component (see the figure below).

ties Import Export Validation	n														
DR Validation is a licensed produ	uct. A valid license v	vas neither fo	ound in t	he installa	tion layou	t nor borr	owed fro	m a licens	e server.						
cense issues details:															
annot find license file.															
he license files (or license server	r system network ad	ldresses) atte	empted ar	e											
sted below. Use LM_LICENSE_FI		license file,													
or contact your software provide															
eature: DDR_Validation_Qorl					1.5										
lename: C:\Freescale\CW4N															
cense path: C:\Freescale\CW4N				\Comm	on\license	.dat;									
lexNet Licensing error:-1,359. Sy or further information, refer to the second se															
or further information, refer to ti vailable at "www.flexerasoftware		g documenta	ation,												
remove at www.nexerasontware	accord a														
lease click HERE in order to requ	iest a license.														
lease click <u>riete</u> in order to requ	rescu meense.			-											
you already have a license, click	on Browse button	to apply it:	Browse.												
												94 (m.)	i ma p		
Scenarios	Result	ts Choose T	ests								Í	8 🗈 🗐 🗖	0] 📑	
	Result	Its Choose T		TART / CLK	_ADJUST con	mbination	7				Í	8 🗈 <			
Scenarios a Validation stage Validation tage	Result	etermine the be	est WRLVL_S		-		CLK_AI					8 (B) <			
a 📝 🗁 Validation stage	Result D V100 %	etermine the be Pass / Total	est WRLVL_S	1/8 clocks	s 1/4 clocks	s 3/8 clock	s 1/2 clo	cks 5/8 cloc			ks 1 clocks) 			
✓ ☑ ➢ Validation stage ✓ I Centering the clock	Result ✓ 100 % ✓ 100 %	etermine the be Pass / Total 0 clocks	0 clocks	1/8 clock	s 1/4 clocks 0/1	s 3/8 clock 0/1	s 1/2 clos 0/1	cks 5/8 cloc 0/1	0/1	0/1	ks 1 clocks 0/1	8 @ 4 ■			
Validation stage Validation stage Centering the clock Pead ODT and driver	Result ✓ 100 % ✓ 100 %	etermine the be Pass / Total 0 clocks 1/8 clocks	est WRLVL_S 0 clocks 0/1 0/1	1/8 clock 0/1 0/1	s 1/4 clocks 0/1 0/1	s 3/8 clock 0/1 0/1	s 1/2 clos 0/1 0/1	cks 5/8 cloc 0/1 0/1	0/1 0/1	0/1 0/1	ks 1 clocks 0/1 0/1	2 (n) <			
	Result ✓ 100 % ✓ 100 %	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks	est WRLVL_S 0 clocks 0/1 0/1 0/1	1/8 clock 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1	s 1/2 clos 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1	0/1 0/1 0/1	0/1 0/1 0/1	ks 1 clocks 0/1 0/1 0/1	2 () 		- -	
■ ♥ ⇒ Validation stage ♥ ➡ Centering the clock ♥ ➡ Read ODT and driver ♥ ➡ Write ODT and driver	Result ✓ 100 % ✓ 100 %	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks 3/8 clocks	est WRLVL_S 0 clocks 0/1 0/1 0/1 0/1	1/8 clock 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1 0/1 0/1	s 1/2 cloo 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1	0/1 0/1 0/1 1/1	0/1 0/1 0/1 0/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1	8 m =			
	Result ✓ 100 % ✓ 100 %	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks	est WRLVL_S 0 clocks 0/1 0/1 0/1	1/8 clock 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1	s 1/2 clos 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1	0/1 0/1 0/1	0/1 0/1 0/1	ks 1 clocks 0/1 0/1 0/1	8 in 2 •			
	Result ✓ 100 % ✓ 100 %	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks 3/8 clocks 1/2 clocks	0 clocks 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clock 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1 0/1 0/1 0/1	s 1/2 clos 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1	0/1 0/1 0/1 0/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1	ð (r) < (
	Result ✓ 100 % ✓ 100 %	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks 3/8 clocks 1/2 clocks 5/8 clocks 5/8 clocks 5/8 clocks 7/8 clocks	est WRLVL_S 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	5 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/2 clos 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	2 (m) < (m)			
■ ♥ ⇒ Validation stage ♥ ➡ Centering the clock ♥ ➡ Read ODT and driver ♥ ➡ Write ODT and driver	Result ✓ 100 % ✓ 100 %	Pass / Total 0 clocks 1/8 clocks 1/8 clocks 1/4 clocks 3/8 clocks 1/2 clocks 5/8 clocks 5/8 clocks 7/8 clocks 1/2 cl	est WRLVL_S 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/2 clos 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				
■ ♥ ⇒ Validation stage ♥ ➡ Centering the clock ♥ ➡ Read ODT and driver ♥ ➡ Write ODT and driver	Result	etermine the be Pass / Total 0 clocks 1/8 clocks 1/8 clocks 3/8 clocks 5/8 clocks 5/8 clocks 5/8 clocks 5/8 clocks 7/8 clocks 1 clocks 9/8 clocks	est WRLVL_S 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clock: 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/2 clo 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				
	Result	etermine the be Pass / Total 0 clocks 1/8 clocks 3/8 clocks 3/8 clocks 5/8 clocks 1/2 clocks 5/8 clocks 1/8 clocks 1/8 clocks 1/8 clocks 5/8 clocks 5/8 clocks 5/8 clocks 5/8 clocks 5/8 clocks 5/8 clocks 5/8 clocks	est WRLVL_5 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clock: 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/2 clo 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				
	Result	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks 3/8 clocks 1/2 clocks 3/8 clocks 3/4 clocks 3/4 clocks 3/4 clocks 5/4 clocks 5/4 clocks 5/4 clocks 5/4 clocks	est WRLVL_S 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/2 clov 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				
■ ♥ ⇒ Validation stage ♥ ➡ Centering the clock ♥ ➡ Read ODT and driver ♥ ➡ Write ODT and driver	Result	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks 3/8 clocks 1/2 clocks 5/8 clocks 3/4 clocks 7/8 clocks 1/4 clocks 5/8 clocks 5/8 clocks 5/8 clocks 5/4 clocks 1/8 clocks 5/4 clocks 1/8 clocks 5/4 clocks 1/8 clocks 5/4 clocks 1/8 clocks 5/4 clocks 1/8 clocks 5/8 clocks 5	est WRLVL_S 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clock: 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	3/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/2 clos 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				
	Result	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks 3/8 clocks 1/2 clocks 3/4 clocks 3/4 clocks 9/8 clocks 1/8 clocks 1/8 clocks 3/4 clocks 3/2 cl	est WRLVL_S 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 3/8 clock 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/2 clov 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				
	Result	etermine the be Pass / Total 0 clocks 1/4 clocks 1/4 clocks 3/8 clocks 1/2 clocks 5/8 clocks 5/8 clocks 5/4 clocks 1/8 cl	est WRLVL_5 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	3/8 clock 0/1	s 1/2 doi 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				
Velidation stage Centering the clock Perform Read ODT and driver Virte ODT and driver Operational DDR tests	Result	etermine the be Pass / Total 0 clocks 1/8 clocks 1/4 clocks 3/8 clocks 1/2 clocks 3/4 clocks 3/4 clocks 9/8 clocks 1/8 clocks 1/8 clocks 3/4 clocks 3/2 cl	est WRLVL_5 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	3/8 clock 0/1	s 1/2 doi 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				
	Result	etermine the be Pass / Total 0 clocks 1/4 clocks 1/4 clocks 3/8 clocks 1/2 clocks 5/8 clocks 5/8 clocks 5/4 clocks 1/8 cl	est WRLVL_S 0 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	1/8 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	s 1/4 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	3/8 clock 0/1	s 1/2 doi 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1	cks 5/8 cloc 0/1 0/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	0/1 0/1 0/1 1/1 1/1 1/1 1/1 1/1	ks 1 clocks 0/1 0/1 0/1 0/1 0/1 0/1 0/1 0/1				

In case you get a license error, you can perform one of the following steps:

- Get a new license. Use the provided link to go to the appropriate web page and request for a license.
- Apply an existing license. Click the Browse button, choose the license, and apply it. You might be asked to restart the tool.

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Limited warranty and liability — 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.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP, the NXP logo, Freescale, the Freescale logo, CodeWarrior, Layerscape, PowerQUICC, Processor Expert, QorlQ, QorlQ Qonverge, and QUICC Engine are trademarks of NXP B.V. All other product or service names are the property of their respective owners. Arm, CoreLink, CoreSight, Cortex, and TrustZone are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© NXP B.V. 2021.

All rights reserved.

For more information, please visit: http://www.nxp.com For sales office addresses, please send an email to: salesaddresses@nxp.com

> Date of release: 01/2021 Document identifier: QCVS_DDR_User_Guide