SYNOPSYS®

Verification Continuum™ - TLM Library

DWC_LPDDR5_MCTL

Version U-2023.03-1
July 2023

Copyright Notice and Proprietary Information

Contents

Preface

The preface of the *DWC_LPDDR5_MCTL* describes:

- About This Manual
- Documentation Conventions
- Terminology
- References
- Customer Support
- Synopsys Statement on Inclusivity and Diversity

About This Manual

This manual describes the DWC_LPDDR5_MCTL, Product Version U-2023.03-1.

We are in the process of updating the screenshots of this manual to the latest user interfaces of Virtualizer Studio. You may find mismatch in the look and feel of some of the screenshots in this release.

This manual is organized as follows:

- Release Notes contains release information.
- Package Overview describes the content of the DWC_LPDDR5_MCTL package.
- Use Model describes the DWC_LPDDR5_MCTL use model.
- Programming Model describes the programming model for DWC_LPDDR5_MCTL.
- Analysis View describes the DWC_LPDDR5_MCTL specific analysis views.
- Creating Starting Point Platform and Generating DDR Targeted Workloads describes the process to create design with `LPDDR5_MCTL`.
- Example Platform for LPDDR5 Block describes the `LPDDR5ExamplePlatform` for the `LPDDR5` block.
- RTL CoSimulation for LPDDR5 Block describes how to run RTL CoSimulation for `LPDDR5` block.

Documentation Conventions

This section lists and explains the documentation conventions used throughout this manual.

Convention	Description and Examples
italic	Is used in running text for: • GUI elements. For example: The *Enumeration* field contains a space-separated list of values. • New terms. For example: A *protocol library* is a collection of protocol definitions. • Web sites. For example: For more information, see *www.eclipse.org*. • E-mail addresses. For example: Contact customer support via e-mail at *vp_support@synopsys.com*. • Manual names. For example: The preface of the *Analysis Manual* describes:

`courier`	Is used for:
	• Code text. For example:
	`list_library_configurations`
	`myConfig`
	In this example, `myConfig` is used.
	• System messages. For example:
	`JVM not found.`
	• Text you must type literally. For example:
	At the prompt, type `go`.
	• Names (of environment variables, commands, utilities, prompts, paths, macros, and so on). For example:
	The `build-options` command sets build parameters.

`courier italic`	Indicates variables. For example: *`scope`* specifies a module, a channel, or a refined port.
bold	Serves to draw your attention to the text in question. For example: `coreId = cwrSAGetCoreId("`**`mycore`**`");`

`[ ]`	Square brackets enclose optional items. For example:
	`clean [-pch]`
	If you must type a square bracket as part of the syntax, it is enclosed in single quotes. For example:
	`'['--use-vector']'`

Convention	Description and Examples

{ }	Braces enclose a list from which you must choose one or more items. For example:
	`add {signalPattern | portPattern} ID`
	If you must type a brace as part of the syntax, it is enclosed in single quotes. For example:
	`DECLARE '{'`
	`Item1`
	`Item1`
	`'}'`

|	A vertical bar separates items in a list of choices. For example: `autoflush {on | off}`
>	A right angle bracket separates menu commands. For example: The *Project > Update System Library* menu command is available.

...	A horizontal ellipsis in syntax indicates that the preceding expression may have zero, one, or more occurrences. For example:
	`build-options -option optionArgs ...`
	A horizontal ellipsis in examples and system messages indicates material that has been omitted. For example:
	`::scsh> dtrace add top1.signal_* $t1`
	`::scsh> dtrace add top1.clk_* $t1`
	`...`
	`::scsh> dtrace flush *`

Terminology

CAS	The CAS Latency of the device
CS	Chip Select
DMA	Direct Memory Access
EOF	End Of File
FCFS	First-come, First-served
GUI	Graphical User Interface
HPR	High Priority Reads
IDE	Integrated Development Environment
IPE	Interrupt-Priority Encoder
JEDEC	JEDEC Solid State Technology Association, formerly known as the Joint Electron Devices Engineering Council (JEDEC). It is an independent semiconductor engineering trade organization and standardization body.
LPR	Low Priority Reads
MPMC	Multi-Ported Memory Controller
PV	Programmer's View
RRB	Read Reorder Buffer
SCML	SystemC Modeling Library
VPR	Variable Priority Reads
VPW	Variable Priority Writes

References

The below listed manuals belong to Platform Architect documentation set and are referenced in this manual. Their inline references are redirected here.

Synopsys Virtual Prototyping Product Installation Guide

VP Explorer User Guide

Customer Support

For technical support (regarding license keys, IP downloads, Host ID, Project ID, documentation or general support), contact the Support Center with a description of your question and supplying the debug information, using one of the following methods:

- Go to *https://solvnetplus.synopsys.com* and sign-in with your Synopsys SolvNetPlus credentials. Select *Cases* from the menu bar, and select *Create a New Case*. Provide the requested information, including:
  - **Product L1:** Virtual Prototyping.

- **Product L2:** Select the product type that closest matches yours.
- **Case Type**: Select the case type from the drop-down menu.
- **Case Severity:** Select the case severity from the drop-down menu.
- **Subject:** Provide a brief summary of the issue or list the error message you have encountered.
- **Description:** For simulation issues, include the timestamp of any signals or locations in waveforms that are not understood.

After creating the case, attach the debug files you have created, if any.

- Or, send an e-mail message to *vp_support@synopsys.com*. (your email will be queued and then, on a first-come, first-served basis, manually routed to the correct support engineer):
  - Include the Product name, Sub Product name, and Tool Version number in your email; so that it can be routed correctly.
  - For simulation issues, include the timestamp of any signals or locations in waveforms that are not understood.
  - Attach any debug files you created in the previous step.
- Or, telephone your local support center:
  - North America:

    Call 1-800-245-8005 from 7:00 AM to 5:30 PM Pacific time, Monday through Friday.
  - All other countries:

    https://www.synopsys.com/support/global-support-centers.html

Synopsys Statement on Inclusivity and Diversity

Synopsys is committed to creating an inclusive environment where every employee, customer, and partner feels welcomed. We are reviewing and removing exclusionary language from our products and supporting customer-facing collateral. Our effort also includes internal initiatives to remove biased language from our engineering and working environment, including terms that are embedded in our software and IPs. At the same time, we are working to ensure that our web content and software applications are usable to people of varying abilities. You may still find examples of non-inclusive language in our software or documentation as our IPs implement industry-standard specifications that are currently under review to remove exclusionary language.

Chapter 1
Release Notes

This chapter describes:

1.1 What's New

This section lists the main changes and enhancements in this release of the DWC_LPDDR5_MCTL Library.

1.1.1 U-2023.03-1

The DWC_LPDDR5_MCTL has been enhanced for the following features.

- Supports only valid speed bins according to the selected device type, as per the *JEDEC standards*.
- Display a warning message before changing the memory device, which resets the address mapping to `default` value automatically.

1.1.2 U-2023.03

The DWC_LPDDR5_MCTL has been enhanced for the following features.

- Supports `PA OPT TYPE` value as `2` for 1 cycle idle latency arbitration scheme.
- The *Refresh* feature is enhanced to correlate with RTL IP version `1.40a-lca00`.
- A new package has been created by which RTL CoSimulation flow can run in an automated way. To run this flow, RTL design should be available in your working directory and the script will generate an `XML` file and the generated file can be run with Platform Architect.

1.2 Tool and Platform Version Information

This version of the DWC_LPDDR5_MCTL Library is compliant with Product Version U-2023.03-1 of Platform Architect and Virtualizer. For an overview of the supported platforms and their versions, see "Supported Platforms and Compilers" in the *Synopsys Virtual Prototyping Product Installation Guide*, which is available with the Platform Architect and Virtualizer documentation set.

- Windows platform is not supported.
- The `DWC_LPDDR5_MCTL` library should be installed at *`installDir`*`/IP`. For details, see the Synopsys Virtual Prototyping Product Installation Guide. To install the library at any other location, an environment variable `CWR_DWC_LPDDR5_MCTL` should be set as path of *`installDir`*`/DWC_LPDDR5_MCTL`.

1.3 Compatibility Information

This section lists compatibility issues that you must take into account in this release of the DWC_LPDDR5_MCTL.

1.3.1 U-2023.03-1

There are no compatibility issues in this release of DWC_LPDDR5_MCTL.

1.3.2 U-2023.03

There are no compatibility issues in this release of DWC_LPDDR5_MCTL.

1.4 Fixed Problems

This section lists problems that were found in the previous release of the DWC_LPDDR5_MCTL Library that have now been fixed.

If the problem has been assigned a Synopsys ID in the Synopsys defect tracking system, this number is also listed; otherwise, *No ID* is specified.

1.4.1 U-2023.03-1

There are no fixed problems in this release of DWC_LPDDR5_MCTL.

1.4.2 U-2023.03

- P10006271-44073: `LPDDR5` model stopped issuing write commands.

  Description: In the DWC_LPDDR5_MCTL model, pending write transaction was coming in `LPDDR4` memory in case of narrow transaction when `RMW` was enabled.

  Resolution: This problem has been fixed for `LPDDR4` memory.

- P10006271-44074: Per-bank Refresh issues in `LPDDR4`.

  Description: Per bank refresh timing `tRFC` was not getting reflected correctly in `LPDDR4` memory.

  Resolution: This problem has been fixed for per bank refresh timing.

- P10006271-45005: `LPDDR5` signal `11` due to Enable `opt_hit_gt_hpr` feature.

  Description: In the DWC_LPDDR5_MCTL model, simulation was suspended when `opt_hit_gt_hpr` was enabled and only low priority read requests were valid.

  Resolution: This problem has been fixed.

- P10006271-44224 `DDR5` controller with `CHI` interface uses wrong target ID for read data packets.

  Description: In the DWC_LPDDR5_MCTL model, the target ID was being set incorrectly for read data packets in `CHI` interface.

  Resolution: This problem has been fixed.

1.5 Limitations

This section lists limitations of this release of the DWC_LPDDR5_MCTL Library.

- The registers which impacts performance are only supported and documented in model.
- Following feature show significant timing deviation with respect to the RTL IP when enabled:

- Inline ECC - Enabling this feature can result in AXI port latency for Read and Write signals from the initiator in the range of 0 to 15%.
- Rank Interleaving - For AXI port latencies, deviation from RTL varies from -7 to +50%.
- QOS feature is still under verification.

- Combinatorial arbitration (0 cycle of idle latency) is not supported in model.
- Disable speculative activate is not supported in model.

1.6 License Requirements

Note In order to use the LPDDR5 Controller Library, you need to procure the product license of DW LPDDR5 Exploration Pack. For further queries, contact *vp_support@synopsys.com*.

Chapter 2
Package Overview

This chapter describes:

- General Package Information
- Supported Features
- Unsupported Features
- Timing Accuracy for Optimization
- Directory Structure
- Analysis

2.1 General Package Information

The DesignWare DDR LPDDR5_MCTL model (DWC_LPDDR5_MCTL) is a fast and accurate TLM model of Synopsys RTL IP DesignWare Cores LPDDR5/4 Memory Controller (DWC LPDDR5/4/4X Controller). The LPDDR5_MCTL RTL IP is a flexible and advanced solution for ASIC and System-on- Chip (SoC) designers who need very low power while achieving industry leading high-efficiency, low-latency and high-performance from their memory interface. The LPDDR5_MCTL TLM model enables system designers to quickly and efficiently explore large number of configuration parameters and optimize the performance of the model in the context of their design. The library enables the system designers to validate the system configurations having LPDDR5_MCTL model. The TLM model provides detailed analysis views and enables system designers to identify hot spots in the memory path, and optimize these.

2.2 Supported Features

The following list provides feature supported by the TLM model corresponding to this version of IP.

Note The current version of DWC_LPDDR5_MCTL Library is compatible with version 1.40a-lca00 of DWC LPDDR5/4/4x Controller RTL.

- Compliant to FT - AXI semantics on application side.
- Configurable number of application side interfaces (1-16).
- Supports LPDDR5 and LPDDR4 memory type.
- Support for AXI3 and AXI4 protocol.
- 1:4 (CK:WCK) frequency ratio architecture.
- 1:2 (CK:WCK) frequency ratio architecture for LPDDR5-3200 Speedbin.
- Data width conversion and clock domain conversion for all application ports.
- Advanced QoS pins.
- High Priority Reads (HPR) and Low Priority Reads (LPR) queues to avoid head of line blocking.

- External port priorities.
- Dual arbiter for read and write channels.
- Configurable memory ranks.
- Open page and Intelligent close-page with page close timer as `0`.
- Read reorder buffer (`RRB`).
- External RAM for `RRB`.
- Dynamic mapping of virtual channels.
- Arbiter supports port aging and Variable Priority Reads (`VPR`) and Variable Priority Writes (`VPW`).
- Scheduler implements a credit mechanism for optimal scheduling of transactions.
- Programmable register interface at `TLM2_GP` interface.
- Configurable address mapping scheme.
- Supports address collision handling.
- Supports Write combine to allow multiple writes to the same address to be combined into a single write to SDRAM.
- Supports Variable Priority Reads (`VPR`) and Variable Priority Writes (`VPW`).
- Programmable support for all of the following SDRAM data-bus widths:
  - Full data-bus width
- Supports a new mechanism to improve `RD/WR` switching.
- Supports `LPDDR5X` memory type for `Speedbin 8533` only.
- Supports `mem_load` and `mem_save` functionalities for `bin` and `elf` formats.

2.3 Unsupported Features

The following features are not supported by TLM model:

- Quarter Bus Width mode
- Half Bus Width mode
- `2T` timings
- Sideband `ECC`
- Bypass path
- Port Throttling
- Dynamic `BSM`
- Fast frequency change and implementation of `FREQ1/2/3` registers.
- Dynamic re-programming of quasi-dynamic registers.
- `LPDDR5 x8` devices
- Dual Channel
- Periodic memory and Phy maintenance/retraining features:
  - `ZQ` calibration
  - Controller assisted drift tracking

    - PPT2
- RMW bypass path
- Per bank refresh optimization

The following RTL hardware parameters are not supported by TLM model:

- UMCTL2_FAST_FREQUENCY_CHANGE
- MEMC_REG_DFI_OUT
- UMCTL2_XPI_USE_WAR
- UMCTL2_XPI_USE_RDR
- UMCTL2_PARTIAL_WR
- UMCTL2_WDATA_EXTRAM
- UMCTL2_RRB_EXTRAM_RETIME
- UMCTL2_RETRY_WDATA_EXTRAM
- UMCTL2_INCL_ARB

2.4 Timing Accuracy for Optimization

2.4.1 DWC_LPDDR5_MCTL Model Vs LPDDR5/4 Memory Controller RTL

- Transaction counts for the entire simulation are the same.
- Average AXI throughput (MB/sec) and Average AXI channel utilization (% active over time) are very similar and show high correlation between the TLM model and the RTL simulations for the features which are claimed to be cycle-accurate.
- Average duration of reads and writes are within +/-20%, depending on the interval for the features which are claimed to be cycle-accurate.

2.4.2 Recommended Use Case is Optimization

Simulation of the TLM model produces correct trends in LPDDR5_MCTL performance usable for optimization of uMCTL2 configurations. Validation of performance must still be done using the RTL IP.

2.5 Directory Structure

The installation of the DWC_LPDDR5_MCTL Library creates files in the following directory:

installDir/IP/DWC_LPDDR5_MCTL

From now on, references are made starting from the above directory.

The following files/directories are created in *installDir*/IP/DWC_LPDDR5_MCTL:

- Documentation

  This directory contains the DWC_LPDDR5_MCTL Library documentation.
- ConvergenSC

  This directory contains the Platform Creator block library file DWC_LPDDR5_MCTL.xml.
- SystemC/include

This directory contains the SystemC include files that are required to instantiate the blocks from the DWC_LPDDR5_MCTL Library.

- `Internal/script`

  This directory contains Platform Creator scripts developed for the DWC_LPDDR5_MCTL module.

- `Internal/lib`

  This directory contains the SystemC library files that are required while linking. Based on operating system and compiler version, different sub-directories are present with different versions of the library.

2.6 Analysis

DWC_LPDDR5_MCTL model is instrumented to provide detailed analysis views. This enables you to analyze the memory timings in order to improve the latency caused by memory accesses.

Chapter 3
Use Model

This chapter describes:

- Overview of Library in Platform Creator
- Opening the Library
- Configuring the Model
- Specifying Memory Options

3.1 Overview of Library in Platform Creator

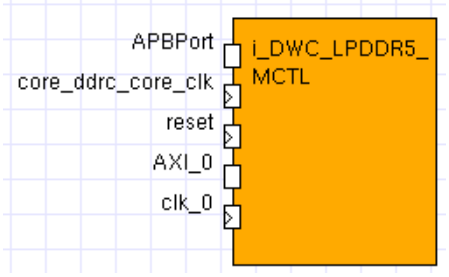
The DWC_LPDDR5_MCTL Library contains the `DWC_LPDDR5_MCTL` block which is the Platform Creator block that can be instantiated and configured in context of the design.

The `DWC_LPDDR5_MCTL` block has the following ports:

- `AXI_0`, `AXI_1` ... `AXI_15`: These are configurable number of ports which are used to communicate with the rest of the SoC. These are the application ports of the controller. The number of ports is determined by the value of `Parameter/MultiPort/number_of_host_ports` on the `DWC_LPDDR5_MCTL` block instance. All these ports are at TLM2 FT-AXI interface.
- `APBPort`: This is the register programming port of `DWC_LPDDR5_MCTL` block instance. This port is at TLM2-GP abstraction level.
- `core_ddrc_core_clk`: The `core_ddrc_core_clk` is main clock for the LPDDR5_MCTL. This is used to derive the clock period on which the memory controller operates. Operating frequency of the memory interface can be in a ratio of 1:4 of this clock.
- `clk_0`, `clk_1` ... `clk_15`: All application port clocks (`aclk_n`) are independently configurable as asynchronous or synchronous with respect to the main LPDDR5_MCTL clock (`core_ddrc_core_clk`). An application port's clock (`aclk_n`) is considered synchronous when it is phase aligned and equal frequency to the LPDDR5_MCTL `core_ddrc_core_clk`.
- `reset`: This port is the reset port of the DWC_LPDDR5_MCTL block instance. The reset port should be driven to value `1` only after register programming is complete.

The figure below shows the `DWC_LPDDR5_MCTL` block.

Figure 3-1 DWC_LPDDR5_MCTL Block



3.1.1 Ports

The following table describes the ports of the DWC_LPDDR5_MCTL block.

Table 3-1 Ports of the LPDDR Block

Name	Protocol	Description	Type
AXI	TLM2 FT-AXI	The array of up to 16 ports to which the FT AXI initiators/bus can connect.	InOut - Slave
APBPort	TLM2-GP	Register configuration interface port of memory controller.	InOut - Slave
clk	CLOCK	Clock port for the memory controller. The memory controller operates on the positive edge of the clock.	InOut - Slave
reset	RESET	Reset port for the memory controller. The reset port should be asserted only after the register programming is complete.	InOut - Slave

3.2 Opening the Library

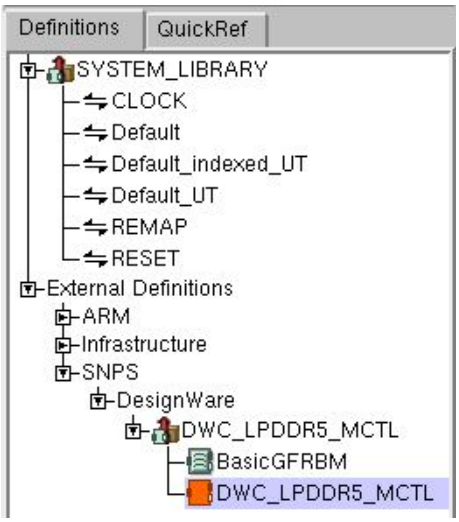
Before you can use the DWC_LPDDR5_MCTL Library in Platform Creator, you need to open the DWC_LPDDR5_MCTL Library.

To open the library:

Do either of the following depending on whether or not the library is available in the Library Browser:

- If the library is available in the Library Browser (see figure 3-2), expand the library folder.
- If the library is not available in the Library Browser, perform the following steps:
  - From the menu bar of the Platform Creator window, select *Project* > *Open Library File*. The open Library dialogue box appears.
  - Browse to the *installDir*/DWC_LPDDR5_MCTL/ConvergenSC/DWC_LPDDR5_MCTL.xml file. The DWC_LPDDR5_MCTL Library appears in the Library Browser (see figure 3-2).

Figure 3-2 The DWC_LPDDR5_MCTL Library



3.2.1 Instantiating

After opening the DWC_LPDDR5_MCTL Library, you can instantiate the DWC_LPDDR5_MCTL block.

To instantiate the DWC_LPDDR5_MCTL block:

Drag the DWC_LPDDR5_MCTL block from the Library Browser into the System Diagram.

3.2.2 Configuring DWC_LPDDR5_MCTL

After instantiation, the DWC_LPDDR5_MCTL block instance needs to be configured for desired operation. There are memory controller static configurations that need to be done on the block, the specific port configurations need to be done on the port and the runtime configurations are done using register programming.

3.2.2.1 Configuring the Block Instance Properties in Platform Creator

The block parameters are of following types:

- Hardware Parameters
- Programming Parameters

3.2.2.1.1 DWC_LPDDR5_MCTL Model Hardware Parameters

This set of parameters represents the hardware configuration parameters on the DWC_LPDDR5_MCTL block instance. The parameters for this category are grouped under /DDRC, /MultiPort and /AXI parameter sets.

Some of the parameters in these sets are read-only and these are for information purposes only, for example /DDRC/enable_bypass.

Following table provides a complete list of parameters, their valid values and description.

Table 3-2 Hardware Parameters

Property Name	Property Type	Default Value	Valid Values	Read Only	Description
/DDRC/RTL_version	string	1.00a-lca01	1.00a-lca01	Yes	Specifies the compatible version of RTL.
/DDRC/memory_type	string	LPDDR5	LPDDR5, LPDDR4	Yes	Specifies the type of memory to be supported. Needed for specification of memory options.
/DDRC/frequency_ratio	integer	4	2, 4	No	LPDDR4: Defines the frequency ratio between the LPDDR5_MCTL clock and the DRAM clock. LPDDR5: Defines the WCK:CK ratio. It is the ratio of the command clock to the data clock.

Property Name	Property Type	Default Value	Valid Values	Read Only	Description
/DDRC/memory_data_width	integer	16	8, 16, 32, 64	No	Specifies the memory data width. This is the width in bits of the DQ signal to SDRAM.
/DDRC/number_of_ranks_supported	integer	1	1, 2	No	Specifies the maximum number of ranks supported by model.
/DDRC/CAM_depth	integer	32	4, 8, 16, 32, 64	No	Specifies the CAM depth (number of entries).
DDRC/programmable_frequency_ratio	integer	0	0, 1	No	Specifies if the frequency ratio is programmable with the help of the software.
/DDRC/burst_length_supported	string	BL16	BL8, BL16	Yes	Specifies the size of a transaction on the host interface (HIF). This can be equivalent to a SDRAM burst length of either 4 or 8.
/DDRC/Enhanced_CAM_pointer_mechanism	boolean	true	true, false	No	Enables the CAM supports out-of-order pushing as well as out-of-order popping.
/DDRC/Enhanced_RD_WR_switching_mechanism	boolean	true	true, false	No	Enables Enhanced Read/Write switching mechanism.
/DDRC/Enhanced_RD_WR_switching_policy_selectable	boolean	true	true, false	No	Enables Enhanced Read/Write switching policy selectable.
/DDRC/enable_link_ecc	boolean	false	true, false	No	Enables linked ECC.
/DDRC/enable_rfm	boolean	false	true, false	No	Enables refresh management.
/DDRC/enable_rfm_sbc	boolean	false	true, false	No	Enables single bank mode for RFM.
/DDRC/rfm_level	integer	0	0,1,2,3	No	Selects RFM level for ARFM. You need to program RFM registers accordingly.
/DDRC/bank_hashing_enable	boolean	false	true, false	No	Enables the bank hashing feature.
/MultiPort/number_of_host_ports	integer	1	1..16	No	Specifies the number of AXI ports in the design.
/Multiport/event_ports	integer	0	0, 1	No	If sets to 1, enables support for custom block to fetch scheduling information.

Property Name	Property Type	Default Value	Valid Values	Read Only	Description
/MultiPort/application_address_width	integer	32	32..60	No	Specifies the address width of each AXI port.
/MultiPort/application_id_width	integer	8	1..32	No	Specifies the Trans ID width of each AXI port.
/MultiPort/application_burst_length_width	integer	4	4..8	No	Specifies the width of application burst length. Should be 8 for AXI4.
/MultiPort/pa_opt_type	integer	1	1..2	No	Specifies the type of optimization required for the Port Arbiter block. The options are: • Two-cycle arbitration (1 cycle of idle latency). • Combinatorial (0 cycle of idle latency).
/DDRC/enable_dual_HIF	boolean	false	true, false	No	This feature converts HIF single command channel into separate HIF command channels for Read and Write commands.
/DDRC/enable_variable_priority_read_write_feature	boolean	false	false	No	Enable variable priority read feature.
/DDRC/enable_refresh_on_reset	integer	0	0..1	No	When enabled, refresh starts on reset. This is for debugging and accuracy purpose.
/MultiPort/enable_XPI_read_address_input_retime	boolean	false	true, false	No	Retime block at XPI input for read address channel.
/MultiPort/enable_external_port_priorities	boolean	false	true, false	No	Enables dynamic setting of port priorities externally through the AXI QoS signals.
/MultiPort/enable_port_arbiter_page_match_feature	boolean	true	true, false	No	Enables the Port Arbiter (PA) PageMatch feature in the hardware.
/ReliabilityFeatures/ECC_Supported	integer	0	0, 1	No	Enables the ECC support. This feature is available only when the DRAM bus width is 16, 32, or 64 bits.
/ReliabilityFeatures/Enable_RMW	boolean	false	true, false	No	Enables read-modify-write commands. By default, this is set for ECC configurations, and unset for non-ECC configurations.

Property Name	Property Type	Default Value	Valid Values	Read Only	Description
/ReliabilityFeatures/Enable_Inline_ECC	integer	0	0, 1	No	Enables Inline ECC. When enabled does not requires an additional data bus for ECC, so the actual DRAM data width is equal to MEMC_DRAM_DATA_WIDTH. ECC parity is stored with the data without using a dedicated sideband memory device.
/ReliabilityFeatures/Block_Interleaving_Depth	integer	4	4, 8, 16, 32	No	Block Interleaving depth. Indicates the number of blocks that can be interleaved at DDRC input (HIF). Enabled in Inline ECC mode.
dis_mem_alloc	boolean	false	false	No	If set true, no SCML memory is allocated and no data accuracy is there.

3.2.3 LPDDR5_MCTL Model Programming Parameters

This set of parameters represents the register configurations of the LPDDR5_MCTL block. These registers are initialized with the value represented in this tab at the time of simulation. The parameters for this category are grouped under /Programming sets.

The following table provides a complete list of programming parameters, their valid values and description.

Table 3-3 Programming Parameters

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/MSTR0/data_bus_width	integer	0x10000	0	0 1	0	Selects proportion of DQ bus width that is used by the SDRAM. 00 - Full DQ bus width to SDRAM.
/Programming/DDRC/MSTR0/burst_rdwr	integer	0x10000	4	1, 2, 4, 8	0	SDRAM burst length used (4 - Burst Length of 8, 8 - Burst Length of 16).

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/FREQ/TMGCFG/frequency_ratio	integer	0xd00	0	0,1	0	Selects the frequency ratio as: • Value 0: - LPDDR4 - 1:2 mode - LPDDR5 - 1:1:2 mode • Value 1: - LPDDR4 - 1:4 mode - LPDDR5 - 1:1:4 mode Enabled only when /DDRC/programmable_frequency_ratio = 1
/Programming/DDRC/RANKCTL/max_rank_rd	integer	0x10c90	15	0..15	0	Maximum number of reads that can be scheduled consecutively to the same rank.
/Programming/DDRC/RANKCTL/max_rank_wr	integer	0x10c90	0	0..15	0	Maximum number of writes that can be scheduled consecutively to the same rank.
/Programming/DDRC/RFMMOD0/rfmth_rm_thr	integer	0x10220	0x1f	0..0xffff	0	Threshold of RM (Refresh Multiplier) to disable RFM command.
/Programming/DDRC/RFMMOD0/raadec	integer	0x10220	0	0...3	0	RAADEC: RAA count decrement per RFM command. • 0: RAAIMT • 1: RAAIMT * 1.5 • 2: RAAIMT * 2 • 3: RAAIMT * 4 (LPDDR5 only)
/Programming/DDRC/RFMMOD0/raamult	integer	0x10220	0	0...3		RAAMULT: Rolling Accumulated ACT Multiplier as: • 0: 2X • 1: 4X • 2: 6X • 3: 8X

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/RFMMOD0/raaimt	integer	0x10220	1	1....31	0	RAAIMT: Rolling Accumulated ACT Initial Management. Threshold as programmed: 0: Invalid 1: 8 2: 16 30: 240 31: 248
/Programming/DDRC/RFMMOD0/rfmsbc	integer	0x10220	0	0,1	0	Enables single bank mode.
/Programming/DDRC/RFMMOD0/rfm_en	integer	0x10220	0	0,1		Enables RFM.
/Programming/DDRC/RFMMOD1/init_raa_cnt	integer	0x10224	0xc	0... 0xffff	0	Initial RAA count.
/Programming/DDRC/SCHED0/prefer_write	integer	0x10380	0	0..1	0	If set then the bank selector prefers writes over reads.
/Programming/DDRC/SCHED0/pageclose	integer	0x10380	0	0..1	0	Bank is kept open only while there are page hit transactions available in the CAM to that bank.
/Programming/DDRC/SCHED0/rdwr_switch_policy_sel	integer	0x10380	1	0 1	0	Selects read write switching policy. 00 - Full DQ bus width to SDRAM.
/Programming/DDRC/SCHED0/opt_wrcam_fill_level	integer	0x10380	1	0 1	0	Enables the feature of optimized write CAM fill level by switching to write when write CAM reaches certain fill level set in SCHED03.
/Programming/DDRC/SCHED0/dis_opt_ntt_by_act	integer	0x10380	0	0 1, 0	0	Disable optimized NTT update by Activate command.
/Programming/DDRC/SCHED0/dis_opt_ntt_by_pre	integer	0x10380	0	0 1	0	Disable optimized NTT update by Precharge command.
/Programming/DDRC/SCHED0/lpr_num_entries	integer	0x10380	16	0..64	0	Number of entries in the low priority transaction store is this value +1.

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/SCHED0/lpddr4_opt_act_timing	integer	0x10380	0	0 1	0	Optimized ACT timing control for LPDDR4.
/Programming/DDRC/SCHED0/prefer_read	integer	0x10380	0	0 1		Sets the bank selector to prefer reads over writes.
/Programming/DDRC/SCHED0/go2critical_hysteresis	integer	0x10380	0	0..256	0	This parameters is not being used as of now.
/Programming/DDRC/SCHED0/dis_speculative_act	integer	0x10380	0	0 1		Enables and disables speculative activate feature.
/Programming/DDRC/SCHEDTMG0/rdwr_idle_gap	integer	0xc00	0x00	0..127	0	When the preferred transaction store is empty for these many clock cycles, switch to the alternate transaction store if it is non-empty.
/Programming/DDRC/SCHED0/opt_act_lat	integer	0x10380	0	0,1	0	Optimizes the best case latency for ACT from HIF to DFI. When the register is set to 1, it may reduce ACT latency by one DFI clock cycle.
/Programming/DDRC/SCHED1/delay_switch_write	integer	0x10384	2	0...15	0	Indicates the number of cycles to delay switching read to write mode when write page-hit request is there and no read page-hit request is there. • 0 - no delay • 1 - two cycles delay • 2 - four cycles delay • 3 - six cycles delay • 4 - eight cycles delay • ... 15- 30-cycles delay
/Programming/DDRC/SCHED1/visible_window_limit_wr	integer	0x10384	0	0...4	0	Visible window limiter for write to prevent extreme starvation against other entries within a CAM. • 0 - Disable this feature • 1 - 31 • 2 - 63 • 3 - 127 • 4 - 255

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/SCHED1/visible_window_limit_rd	integer	0x10384	0	0...4	0	Visible window limiter for read to prevent extreme starvation against other entries within a CAM. • 0 - Disable this feature • 1 - 4 • 2 - 8 • 3 - 16 • 4 - 32
/Programming/DDRC/SCHED1/page_hit_limit_wr	integer	0x10384	0	0...4	0	Page-Hit limiter for write. • 0 - Disable this feature. • 1 - 4 • 2 - 8 • 3 - 16 • 4 - 32
/Programming/DDRC/SCHED1/page_hit_limit_rd	integer	0x10384	0	0...4	0	Page-Hit limiter for read. • 0 - Disable this feature • 1 - 4 • 2 - 8 • 3 - 16 • 4 - 32
/Programming/DDRC/SCHED1/opt_hit_gt_hpr	integer	0x10384	0	0...1	0	Optimize the priority between Page-hit LPR and Page-miss HPR. • 0 - Page-miss HPR has priority (default). • 1 - Page-hit LPR has priority. This is to trade-off between HPR latency and total utilization. If set to 0, HPR latency can be better than 1 because HPR has priority over LPR. If set to 1, DRAM utilization can be better than 0 because number of ACT-PRE is reduced. When this register is set to 1. It is recommend to enable page-hit limiter so that once page-hit limiter is expired, HPR can have priority. Enabled when Enhanced CAM pointer mechanism is enabled.

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/SCHED3/wrcam_lowthresh	integer	0x1038c	8		0	The low threshold used in optimized write CAM fill level.
/Programming/DDRC/SCHED3/wrcam_highthresh	integer	0x1038c	2		0	The high threshold used in optimized write CAM fill level.
/Programming/DDRC/SCHED3/wr_pghit_num_thresh	integer	0x1038c	4		0	Switch to read mode once number of read page-hit request exceeds the threshold set in the register during waiting tW2R.
/Programming/DDRC/SCHED3/rd_pghit_num_thresh	integer	0x1038c	4		0	Switch to write mode once number of write page-hit request exceeds threshold set in this register during waiting delay_switch_write timeout.
/Programming/DDRC/SCHED4/rd_act_idle_gap	integer	0x10390	16		0	Indicates the number of cycles when read direction has no request to start preparing bank for write direction.
/Programming/DDRC/SCHED4/wr_act_idle_gap	integer	0x10390	8		0	Indicates the number of cycles when write direction has no request to start preparing bank for read direction.
/Programming/DDRC/SCHED4/rd_page_exp_cycles	integer	0x10390	64		0	Indicates the number of cycles to keep the bank opened for read direction in write mode when both directions has request to the bank.
/Programming/DDRC/SCHED4/wr_page_exp_cycles	integer	0x10390	8		0	Indicates the number of cycles to keep the bank opened for write direction in read mode when both directions has request to the bank.
/Programming/DDRC/SCHED5/wrecc_cam_lowthresh	integer	0x10394	0x4		0	The low threshold used in optimized write ECC CAM fill level.
/Programming/DDRC/SCHED5/wrecc_cam_highthresh	integer	0x10394	0x2		0	The high threshold used in optimized write ECC CAM fill level.

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/SCHED5/dis_opt_loaded_wrecc_cam_fill_level	integer	0x10394	0x0		0	Should be set as 0.
/Programming/DDRC/SCHED5/dis_opt_valid_wrecc_cam_fill_level	integer	0x10394	0x0		0	Should be set as 0.
/Programming/DDRC/DBICTL/dm_en	integer	0x10c94	1	0 1	0	Indicates the DM enable signal in DDRC.
/Programming/FREQ/PERFHPR1/hpr_max_starve	integer	0xc80	0x01	0x0..0xFFFF	0	Number of clocks that the HPR queue can be starved before it goes critical.
/Programming/FREQ/PERFHPR1/hpr_xact_run_length	integer	0xc80	0xf	0x0..0xFF	0	Number of transactions that are serviced once the HPR queue goes critical is the smaller of this number of Number of transactions available.
/Programming/FREQ/PERFLPR1/lpr_max_starve	integer	0xc84	0x7f	0x0...0xFFFF	0	Number of clocks that the LPR queue can be starved before it goes critical.
/Programming/FREQ/PERFLPR1/lpr_xact_run_length	integer	0xc84	0xf	0x0..0xFF	0	Number of transactions that are serviced once the LPR queue goes critical is smaller of Number of transactions available.
/Programming/FREQ/PERFWR1/w_max_starve	integer	0xc88	0x7f	0x0..0xFFFF	0	Number of clocks that the WR queue can be starved before it goes critical.
/Programming/FREQ/PERFWR1/w_xact_run_length	integer	0xc88	0xf	0x0...0xFF	0	Number of transactions that are serviced once the WR queue goes critical is smaller than the number of transactions available.
/Programming/DDRC/DBG0/dis_wc	integer	0x130050	1	0, 1	0	When the value is 1, disable write combine.

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/DBG0/dis_collision_page_opt	integer	0x130050	1	0, 1	0	When this is set to 0, auto-precharge is disabled for the flushed command in a collision case. Collision cases are write followed by read to same address, read followed by write to same address, or write followed by write to same address with DBG0.dis_wc bit = 1 (where same address comparisons exclude the two address bits representing critical word).
/Programming/DDRC/ECCCFG0/ecc_mode	integer	0x10600	0	0..7	0	ECC mode indicator. • 000 - ECC disabled • 100 - ECC enabled - SEC/DED over 1 beat • 101 - ECC enabled - Advanced ECC All other settings are reserved for future use. Set it to 4 for enabling Inline ECC.
/Programming/DDRC/ECCCFG0/ecc_region_map	integer	0x10600	127	0..127	0	Selectable Protected Region setting. Memory space is divided to 8/16/32/64 regions which is determined by ECCCFG1.ecc_region_map_granu

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/ECCCFG0/blk_channel_idle_time_x32	integer	0x10600	0	0..2147483647	0	Indicates the number of cycles on HIF interface with no access to protected regions which will cause flush of all the block channels. In order to flush block channel, LPDDR5_MCTL injects write ECC command (when there is no incoming HIF command) if there is any write in the block and then stop tracking the block address. • 0 indicates no timeout (feature is disabled, not supported with this version) • 1 indicates 32 cycles • 2 indicates 2*32 cycles, and so on. Unit: Multiples of 32 DFI clock cycles.
/Programming/DDRC/ECCCFG0/ecc_region_map_granu	integer	0x10600	0	0..3	0	Granularity of Selectable Protected Region. Define one region size for ECCCFG0.ecc_region_map • 0 – 1/8 of memory spaces • 1 – 1/16 of memory spaces • 2 – 1/32 of memory spaces • 3 – 1/64 of memory spaces
/Programming/DDRC/ECCCFG1/active_blk_channel	integer	0x10604	0	0..16	0	Number of active block channels. Total number of ECC block channels are defined by MEMC_NO_OF_BLK_CHANNEL hardware parameter. This register can limit the number of available channels. For example, if set to 0, only one channel is active and therefore block interleaving is disabled.

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/ECCCFG1/blk_channel_active_term	integer	0x10604	1	0..1	1	Block Channel active terminate enable.If enabled, block channel is terminated when full block write or full block read is performed (all address within block are written or read) • 0 - Disable (only for debug purpose) • 1 - Enable (default) This is debug register, and this must be set to 1 for normal operation.
/Programming/DDRC/ECCCFG1/ecc_region_waste_lock	integer	0x10604	0	0..1	1	Locks the remaining waste parts of the ECC region (hole) that are not locked by ecc_region_parity_lock. • 1- Locked; if this region is accessed, error response is generated. • 0- Unlocked; this region can be accessed normally, similar to non-ECC protected region.
/Programming/DDRC/ECCCFG1/ecc_region_parity_lock	integer	0x10604	0	0..1	1	Locks the parity section of the ECC region (hole) which is the highest system address part of the memory that stores ECC parity for protected region. • 1- Locked, if this region is accessed, error response is generated. • 0 - Unlocked, this region can be accessed normally, similar to non-ECC protected region.
Programming/DDRC/LNKECCCTL0/rd_link_ecc_enable	boolean	0x10980	0	0..1	0	Enables rd link ECC, this needs to be enabled along with Enable_link_ecc.
/Programming/DDRC/LNKECCCTL0/wr_link_ecc_enable	boolean	0x10980	0	0..1	0	Enables wr link ECC, this needs to be enabled along with Enable_link_ecc.
/Programming/DDRC/RFSHMOD0/per_bank_refresh	boolean	0x10200	0	0..1	1	Enables per bank refresh when selected = 1.

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/RFSHMOD0/refresh_burst	integer	0x10200	0	0..7	0	The programmed value +1 is the number of refresh timeouts that is allowed to accumulate before traffic is blocked and the refreshes are forced to execute.
/Programming/DDRC/RFSHCTL0/rank_dis_refresh	integer	0x10208	0	0..1	0	In case of 1 rank, value can be 0 or 1. In case of two ranks, value can be 00, 01,10 or 11.
/Programming/DDRC/RFSHCTL0/dis_auto_refresh	boolean	0x10208	1	0..1	0	Disable auto refreshes.
/Programming/ARB/PCCFG/pagematch_limit	integer	0x20000	0	1	0	If set to 1, limits the number of consecutive same page DDRC transactions that can be granted by the Port Arbiter to four when Page Match feature is enabled. If set to 0, there is no limit imposed on number of consecutive same page DDRC transactions.
/Programming/ARB/PCCFG/go2critical_en	integer	0x20000	0	1	0	If set to 1 (enabled), sets co_gs_go2critical_wr and co_gs_go2critical_lpr/co_gs_go2critical_hpr signals going to DDRC based on urgent input (awurgent, arurgent) coming from AXI master. If set to 0 (disabled), co_gs_go2critical_wr and co_gs_go2critical_lpr/co_gs_go2critical_hpr signals at DDRC are driven to 0.
/Programming/DDRC/OPCTRL0/dis_max_rank_rd_opt	boolean	0x10b80	0		0	Disable optimized max_rank_rd and max_logical_rank_rd feature. This register is debug purpose only.

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/DDRC/OPCTRL0/dis_max_rank_wr_opt	boolean	0x10b80	0		0	Disable optimized max_rank_wr and max_logical_rank_wr feature. This register is debug purpose only.
/Programming/DDRC/MSTR4/wck_on	boolean	0x10010	0	0..1	0	WCK always ON mode. • 0: WCK Always ON mode disabled. • 1: WCK Always ON mode enabled.

The table below lists the address programming parameters.

Table 3-4 Address Programming Parameters

Property Name	Property Type	Offset	Bit Field	Default Value	Valid Values	Read Only	Description
/Programming/ADDR/ADDRMAP0/addrmap_dch_bit0	integer	0x30000	0:5	0	0...63	0	Selects the HIF address bit used as data channel address bit 0.
/Programming/ADDR/ADDRMAP1/addrmap_cs_bit0	integer	0x30004	0:5	0	0...63	0	Selects the HIF address bit used as rank address bit 0/1/2/3.
/Programming/ADDR/ADDRMAP1/addrmap_cs_bit1			8:13				
/Programming/ADDR/ADDRMAP1/addrmap_cs_bit2			16:21				
/Programming/ADDR/ADDRMAP1/addrmap_cs_bit3			24:29				
/Programming/ADDR/ADDRMAP2/addrmap_cid_b0	integer	0x30008	0:5	0	0...63	0	Selects the HIF address bit used as chip id bit 0/1.
/Programming/ADDR/ADDRMAP2/addrmap_cid_b1			8:13				

Property Name	Property Type	Offset	Bit Field	Default Value	Valid Values	Read Only	Description
/Programming/ADDR/ADDRMAP3/addrmap_bank_b0	integer	0x3000c	0:5	0	0...63	0	Selects the HIF address bits used as bank address bit 0/1/2.
/Programming/ADDR/ADDRMAP3/addrmap_bank_b1			8:13				
/Programming/ADDR/ADDRMAP3/addrmap_bank_b2			16:21				
/Programming/ADDR/ADDRMAP4/addrmap_bg_b0	integer	0x30010	0:5	0	0...63	0	Selects the HIF address bits used as bank group address bit 0/1/2.
/Programming/ADDR/ADDRMAP4/addrmap_bg_b1			8:13				
/Programming/ADDR/ADDRMAP4/addrmap_bg_b2			16:21				
/Programming/ADDR/ADDRMAP5/addrmap_col_b7	integer	0x30014	0:4	0	0...31	0	Selects the HIF address bit used as column address bit 7 to bit 10 respectively.
/Programming/ADDR/ADDRMAP5/addrmap_col_b8			8:12				
/Programming/ADDR/ADDRMAP5/addrmap_col_b9			16:20				
/Programming/ADDR/ADDRMAP5/addrmap_col_b10			24:28				

Property Name	Property Type	Offset	Bit Field	Default Value	Valid Values	Read Only	Description
/Programming/ADDR/ADDRMAP6/addrmap_col_b3	integer	0x30018	0:3	0	0...15	0	Selects the HIF address bit used as column address bit 3. MEMC_BURST_LENGTH=16, it is required to program this to 0, hence register does not exist in this case.
/Programming/ADDR/ADDRMAP6/addrmap_col_b4			8:11				Selects the HIF address bit used as column address bit 4/5/6.
/Programming/ADDR/ADDRMAP6/addrmap_col_b5			16:19				
/Programming/ADDR/ADDRMAP6/addrmap_col_b6			24:27				
/Programming/ADDR/ADDRMAP7/addrmap_row_b14	integer	0x3001C	0:4	0	0...31	0	Selects the HIF address bit used as row address bit 14 to bit17 respectively. If unused, set to 31 and then row address bit is set to 0.
/Programming/ADDR/ADDRMAP7/addrmap_row_b15	integer	0x3001C	8:12	0	0...31	0	Selects the HIF address bit used as row address bit 14 to bit17 respectively. If unused, set to 31 and then row address bit is set to 0.
/Programming/ADDR/ADDRMAP7/addrmap_row_b16			16:20				
/Programming/ADDR/ADDRMAP7/addrmap_row_b17			24:28				

Property Name	Property Type	Offset	Bit Field	Default Value	Valid Values	Read Only	Description
/Programming/ADDR/ADDRMAP8/addrmap_row_b10	integer	0x30020	0:4	0	0...15	0	Selects the HIF address bits used as row address bit 10.
/Programming/ADDR/ADDRMAP8/addrmap_row_b11			8:12				Selects the HIF address bit used as row address bit 11. If unused, set to 15 and then row address bit is set to 0.
/Programming/ADDR/ADDRMAP8/addrmap_row_b12			16:20		0..31		Selects the HIF address bit used as row address bit 12 to bit13 respectively. If unused, set to 31 and then row address bit is set to 0.
/Programming/ADDR/ADDRMAP8/addrmap_row_b13			24:28				
/Programming/ADDR/ADDRMAP9/addrmap_row_b6	integer	0x30024	0:4	0	0...15	0	Selects the HIF address bits used as row address bit 6 to bit 9 respectively.
/Programming/ADDR/ADDRMAP9/addrmap_row_b7			8:12				
/Programming/ADDR/ADDRMAP9/addrmap_row_b8			16:20				
/Programming/ADDR/ADDRMAP9/addrmap_row_b9			24:28				
/Programming/ADDR/ADDRMAP10/addrmap_row_b2	integer	0x30028	0:4	0	0...15	0	Selects the HIF address bits used as row address bit 2 to bit 5 respectively.
/Programming/ADDR/ADDRMAP10/addrmap_row_b3			8:12				
/Programming/ADDR/ADDRMAP10/addrmap_row_b4			16:20				
/Programming/ADDR/ADDRMAP10/addrmap_row_b5			24:28				

Property Name	Property Type	Offset	Bit Field	Default Value	Valid Values	Read Only	Description
/Programming/ADDR/ADDRMAP11/addrmap_row_b0	integer	0x3002c	0:4	0	0...15	0	Selects the HIF address bits used as row address bit 0 to 1.
/Programming/ADDR/ADDRMAP11/addrmap_row_b1			8:12				
/Programming/ADDR/ADDRMAP12/bank_hash_en	boolean	0x30030	4	false	true false	0	Enables the bank hashing feature.

3.2.4 Configuring the Block Port Properties in Platform Creator

The Block Port parameters represent the various configuration parameters for the individual ports of LPDDR5_MCTL model.

The AXI port parameters are of following types:

- Hardware Parameters
- Programming Parameters

3.2.4.1 AXI Port Hardware Parameters

This set of parameters represent the hardware configuration of parameters on the LPDDR5_MCTL AXI port.

Some of the parameters in these sets are read only and these are for information purposes only. For example, preserve_read_write_transaction_order.

Following table provides a complete list of parameters, their valid values and description.

Table 3-5 AXI Port hardware Parameters

Property Name	Property Type	Default Value	Valid Values	Read Only	Description
port_number	integer	0	0..15	No	Unique number for each port. Is set automatically by the design creation flow.
protocol	string	AXI3	AXI3, AXI4	No	Defines the interface type for the LPDDR5_MCTL application port. This is usually consistent for all ports.
static_virtual_channel_mapping	boolean	false		No	Enables static virtual channels mapping for this port.
dual_read_address_queues	boolean	false	true, false	No	Enables the dual read address queue for the LPDDR5_MCTL application port n. Each dual address queue XPI consumes two consecutive PA ports.
number_of_virtual_channels	integer	1	1..64	No	Defines the number of virtual channels for this port.

Property Name	Property Type	Default Value	Valid Values	Read Only	Description
AXI_read_address_queue_depth	integer	4	2..32	No	Determines how many AXI addresses can be stored in the read address buffer of port.
AXI_write_address_queue_depth	integer	4	2..128	No	Determines how many AXI addresses can be stored in the write address buffer of port.
AXI_read_data_queue_depth	integer	10	2..128	No	Determines how many AXI burst beats can be stored in the read data buffer of port. This allows the LPDDR5_MCTL to store enough read data in the read data buffer such that the LPDDR5_MCTL does not stall a continuous stream of read commands to wait for data.
AXI_write_data_queue_depth	integer	10	2..64	No	Determines how many AXI burst beats can be stored in the write data buffer of port. This allows the LPDDR5_MCTL to store enough write data in the write data buffer such that the LPDDR5_MCTL does not stall a continuous stream of write commands to wait for data.
AXI_write_response_queue_depth	integer	10		No	Determines how many AXI write responses can be stored in the write response buffer of port.
read_data_interleaving	boolean	true		No	Enables the interleaving of the read data of transactions with different ARID fields.
data_width	integer	64	32, 64, 128, 256, 512	No	Data width of each port.
number_of_synchronizers	integer	2	2 3 4	No	Number of synchronization stages for asynchronous FIFO of port.
is_sync_clock	boolean	false		No	Defines port n clock to be synchronous or asynchronous with respect to the LPDDR5_MCTL core_ddrc_core_clk.
enable_external_ram_for_rrb	boolean	false		No	Enables external RAM for Read Reorder Buffer (RRB) of port.
preserve_read_write_transaction_order	integer	0	0, 1	Yes	

3.2.4.2 AXI Port Programming Parameters

This set of parameters represent the register configurations of the LPDDR5_MCTL AXI ports. These registers are initialized with the value represented in this tab at the time of simulation. The parameters for this category are grouped under /Programming set.

Following table provides a complete list of programming parameters, their valid values and description.

Table 3-6 AXI Port Programming Parameters

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/ARB/PCFGR/rd_port_priority	integer	0x20004	0x1F	0..0x3FF	No	Determines the initial load value of read aging counters.
/Programming/ARB/PCFGR/rd_port_bypass_en	integer	0x20004	0	0	Yes	If set to 1, read transactions with ID not covered by any of the virtual channel ID mapping registers are not reordered.
/Programming/ARB/PCFGR/rd_port_aging_en	integer	0x20004	1	0 1	No	If set to 1, enables aging function for the read channel of the port.
/Programming/ARB/PCFGR/rd_port_urgent_en	integer	0x20004	0	0	Yes	If set to 1, enables the AXI urgent sideband signal (arurgent).
/Programming/ARB/PCFGR/rd_port_pagematch_en	integer	0x20004	1	0 1	No	If set to 1, enables the Page Match feature.
/Programming/ARB/PCFGR/rd_port_ordered_en	integer	0x20004	0	0	Yes	If set to 1, preserves the ordering between read transaction and write transaction issued to the same address, on a given port.

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/ARB/PCFGR/rrb_lock_threshold	integer	0x20004	0x4	0..15	No	Specifies the RRB lock threshold in configurations that disable read data interleaving. Threshold is specified in terms of the HIF bursts that belong to the same AXI transaction. RRB locks onto VC only when this specified number of HIF bursts are returned by DDRC. RRB lock occurs earlier in cases where the AXI transaction itself is shorter and the total number of corresponding HIF bursts are below the programmed threshold and all of them are returned by DDRC. When N is programmed in this field, the threshold will be set to N+1 bursts. Max thresholding is up to 16 bursts. Value After Reset: 0x4 Exists: UMCTL2_READ_DATA_INTERLEAVE_EN == 0.
/Programming/ARB/PCFGW/wr_port_priority	integer	0x20008	0x1F	0x3FF	No	Determines the initial load value of write aging counters.
/Programming/ARB/PCFGW/wr_port_aging_en	integer	0x20008	1	0 1	No	If set to 1, enables aging function for the write channel of the port.
/Programming/ARB/PCFGW/wr_port_urgent_en	integer	0x20008	0	0 1	No	If set to 1, enables the AXI urgent sideband signal (awurgent).
/Programming/ARB/PCFGW/wr_port_pagematch_en	integer	0x20008	1	0 1	No	If enabled, once a requesting port is granted, the port is continued to be granted if the following immediate commands are given to the same memory page (same bank and same row).

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/ARB/PCFGQOS0/rqos_map_level1	integer	0x20094	0	0..13	No	Separation level1 indicates the end of region0 mapping; start of region0 is 0.
/Programming/ARB/PCFGQOS0/rqos_map_level2	integer	0x20094	14	0..15	No	Separation level2 indicates the end of region1 mapping; start of region1 is (level1 + 1).
/Programming/ARB/PCFGQOS0/rqos_map_region0	integer	0x20094	0	0..2	No	This bitfield indicates the traffic class of region 0. 0: LPR, 1: VPR, 2: HPR.
/Programming/ARB/PCFGQOS0/rqos_map_region1	integer	0x20094	0	0..2	No	This bitfield indicates the traffic class of region1. 0: LPR, 1: VPR, 2: HPR.
/Programming/ARB/PCFGQOS0/rqos_map_region2	integer	0x20094	2	0..2	No	This bitfield indicates the traffic class of region 2. 0: LPR, 1: VPR, 2: HPR
/Programming/ARB/PCFGQOS1/rqos_map_timeoutb	integer	0x20098	0	0x0..0x7FF	No	Specifies the timeout value for transactions mapped to the blue address queue.
/Programming/ARB/PCFGQOS1/rqos_map_timeoutr	integer	0x20098	0	0x0..0x7FF	No	Specifies the timeout value for transactions mapped to the red address queue.
/Programming/ARB/PCFGWQOS0/wqos_map_level1	integer	0x2009c	0	0..13	No	Specifies the end of region0 mapping when the start of the region is taken as 0.
/Programming/ARB/PCFGWQOS0/wqos_map_level2	integer	0x2009c	14	0..14	No	Separation level2 indicating the end of region1 mapping; start of region1 is (level1 + 1). Possible values for level2 are (level1 + 1) to 14 which corresponds to awqos.
/Programming/ARB/PCFGWQOS0/wqos_map_region0	integer	0x2009c	0	0 1	No	This bitfield indicates the traffic class of region0. Valid values are: 0: NPW 1: VPW

Property Name	Property Type	Offset	Default Value	Valid Values	Read Only	Description
/Programming/ARB/PCFGWQOS0/wqos_map_region1	integer	0x2009c	0	0 1	No	This bitfield indicates the traffic class of region1. Valid values are: 0: NPW 1: VPW
/Programming/ARB/PCFGWQOS0/ wqos_map_region2	integer	0x2009c	0	0 1	No	This bitfield indicates the traffic class of region1. Valid values are: 0: NPW 1: VPW
/Programming/ARB/PCFGWQOS0/wqos_map_timeout1	integer	0x2009c	0	0x0..0x7FF	No	Specifiers the timeout value for the write transactions in region 0 and 1.
/Programming/ARB/PCFGWQOS0/ wqos_map_timeout2	integer	0x2009c	0	0x0..0x7FF	No	Specifiers the timeout value for the write transactions in region 2.

3.2.4.3 Configuring Phy Static Latencies

This section explains how to program the phy static latencies in the TLM model using the RTL simulation.

1. phy_static_read_latency

   Number of clock cycles between Read command being issued by the Controller and read data arriving at the Controller, phy_static_read_latency corresponds to tphy_rdlat.

Figure 3-3 Waveform Sample

- tphy_rdlat: Specifies the maximum number of cycles allowed from the assertion of the dfi_rddata_en signal to the assertion of the dfi_rddata_valid signal.

  For example, as shown in the waveform illustration above, the tphy_rdlat = 18, so the phy_static_read_latency comes to 18.

2. phy_static_write_latency

Number of clock cycles from the write command to write data enable (dfi_wrdata_en). This corresponds to the DFI timing parameter, tphy_wrlat. In the RTL model, it corresponds to DFITMG0.dfi_tphy_wrlat.

For LPDDR4, tphy_wrlat = tphy_wrlat +3.

Note

- If (*Enable_Dual_Channel_support* == 1), then:
  - For DDRC Channel 0: For programming parameters phy_static_write_latency and phy_static_read_latency, RTL signals reg_ddrc_dfi_tphy_wrlat_dch0 and reg_ddrc_dfi_t_rddata_en_dch0 should be used in above calculations.
  - For DDRC Channel 1: For programming parameters phy_static_write_latency_chnl1 and phy_static_read_latency_chnl1, RTL signals reg_ddrc_dfi_tphy_wrlat and reg_ddrc_dfi_t_rddata_en should be used in above calculations.

3.3 Configuring the Model

To configure the LPDDR5_MCTL model instance in the Platform Creator, you may use any of the methods listed below.

- Configuring the Parameters in Platform Creator
- Importing CoreConsultant Configurations
- Importing Configuration from CoreConsultant Trace File

3.3.1 Configuring the Parameters in Platform Creator

Instantiate the model in Platform Creator and configure the parameters of the block that represents the model hardware configurations. All relevant registers that impact the timing of the model are represented on the block instance or on the block instance ports as explained. Valid ranges of these parameter and register values can be used in IMPO sweeping mechanism.

Note

In case, a register value is configured on the block as parameter and the same value is over-written using programming on the APB port, then the value written by the programming sequence applies and the value specified in Platform Creator parameter is ignored.

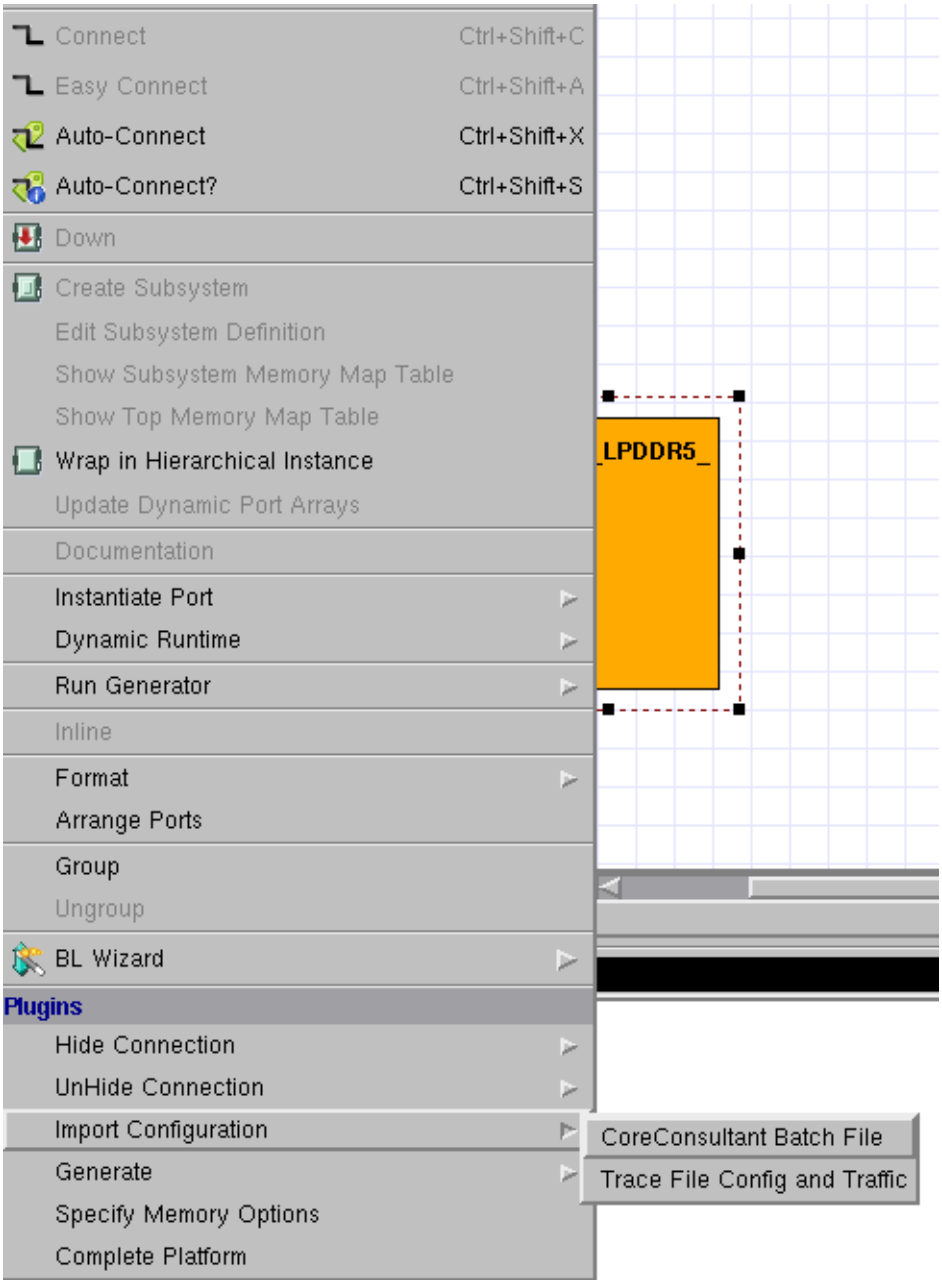
3.3.2 Importing CoreConsultant Configurations

You can apply a configuration of the model in CoreConsultant on a Platform Creator block instance.

To import the CoreConsultant configurations:

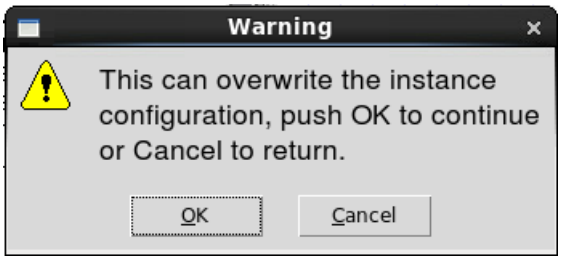
1 In the Design view of the Platform Creator, right-click on the block and from the context menu, under *Plugins* > *Import Configuration* > *CoreConsultant Batch File*.

Figure 3-4 Importing CoreConsultant Configurations



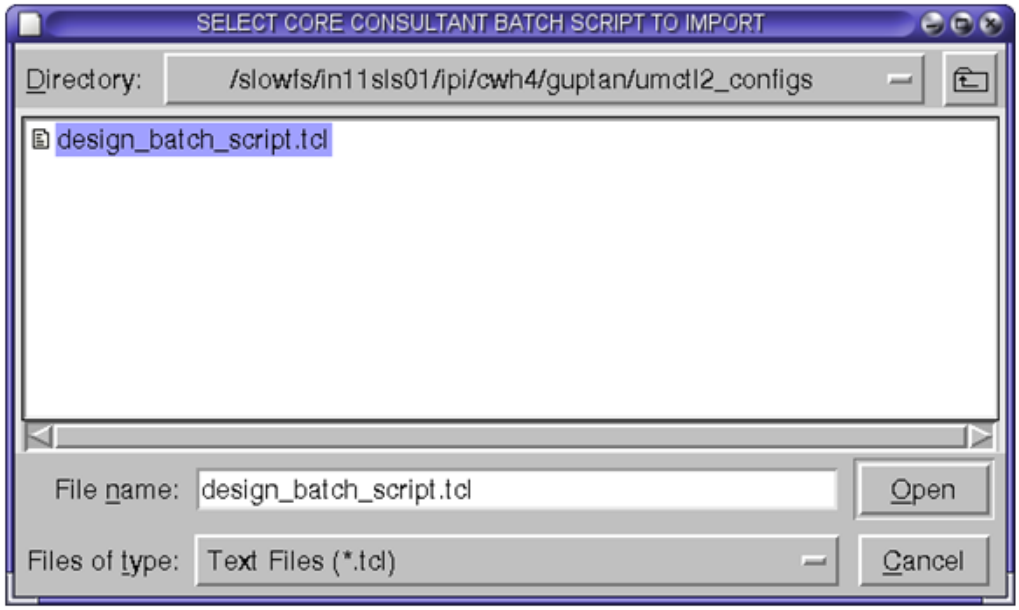
2 The tool prompts about the loss of current settings of the model as shown below.

Figure 3-5 Warning Message for Overwriting Instance Configuration



3 Press *OK*. In the dialog box that appears, browse to the directory where the CoreConsultant batch script is placed and select the script as shown below.

Figure 3-6 Selecting Core Consultant Batch Script



4 Click *Open* to apply the design configuration of the CoreConsultant batch script to the relevant parameters on the block.

> **Note** All CoreConsultant parameters are not supported in the TLM model. For parameter specifications that are not supported, warning message appears in the Platform Creator Console window. The default values in the CoreConsultant batch file are ignored by the utility.

3.3.3 Importing Configuration from CoreConsultant Trace File

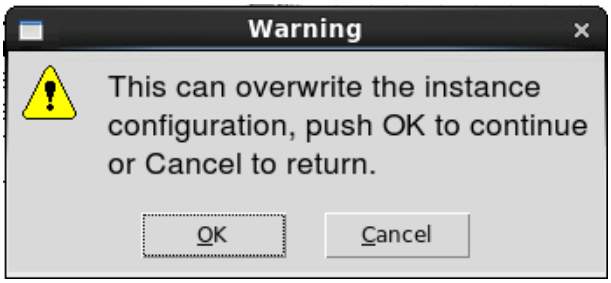
CoreConsultant trace files are used in CoreConsultant environment to fire stimuli to an RTL configuration. This file also contains values for registers and information about traffic at individual port of `LPDDR5_MCTL` instance. You can use the configuration as well as traffic specification in a CoreConsultant trace file to create an equivalent system in Platform Creator.

To import the configurations from CoreConsultant trace file:

1 In the Design view of the Platform Creator, right-click on the block and from the context menu, select *Plugins* > *Import Configuration* > *Trace File Config and Traffic*.

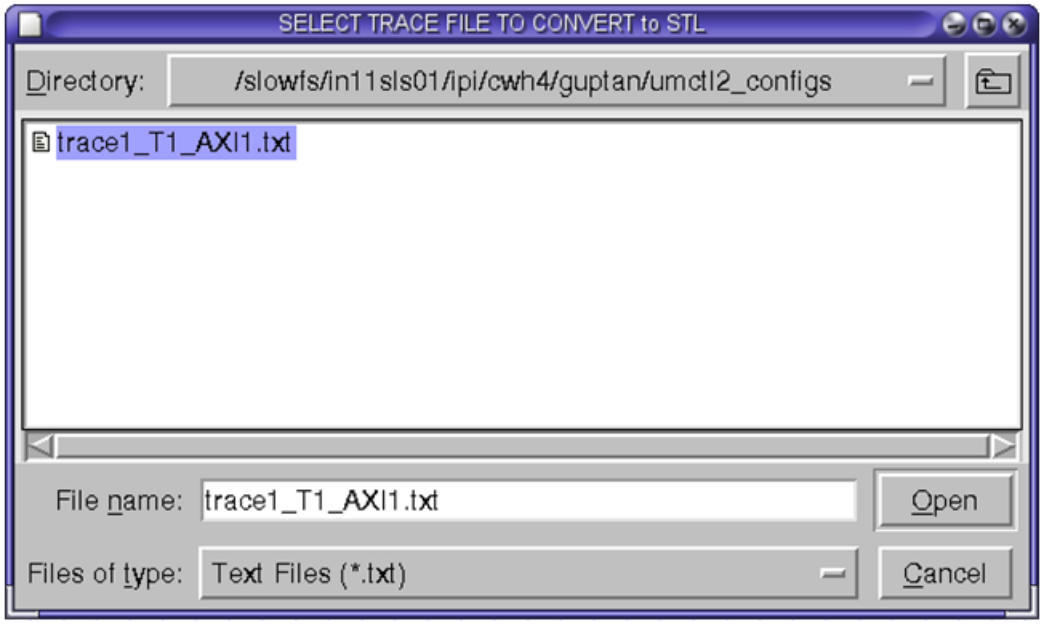
Once you invoke this plugin as shown in figure 3-4, the script prompts about the possible loss of existing configuration as shown below.

Figure 3-7 Warning Message for Overwriting Instance Configuration



2 Press *OK*. In the dialog box that appears, select the trace file to import configuration and convert trace traffic to GFRBM `STL` files, as shown below.

Figure 3-8 Selecting Trace File to Convert to STL



3 Click *Open* to apply the register settings from the trace files to the instances and to convert the trace file traffic to GFRBM `STL` files.

One `STL` file is created for each port and you are expected to connect one GFRBM instance to one port of `LPDDR5_MCTL` block instance for using this flow.

Note

All registers are not modeled in the TLM model and for the register settings that are not applied, explicit warning message is issued on Platform Creator console. For example:

```
Warning, cannot set parameter /HARDWARE/LPDDR5_MCTL
/Programming/DDRC/DFIUPD0/dis_auto_ctrlupd to 1, check line 5
```

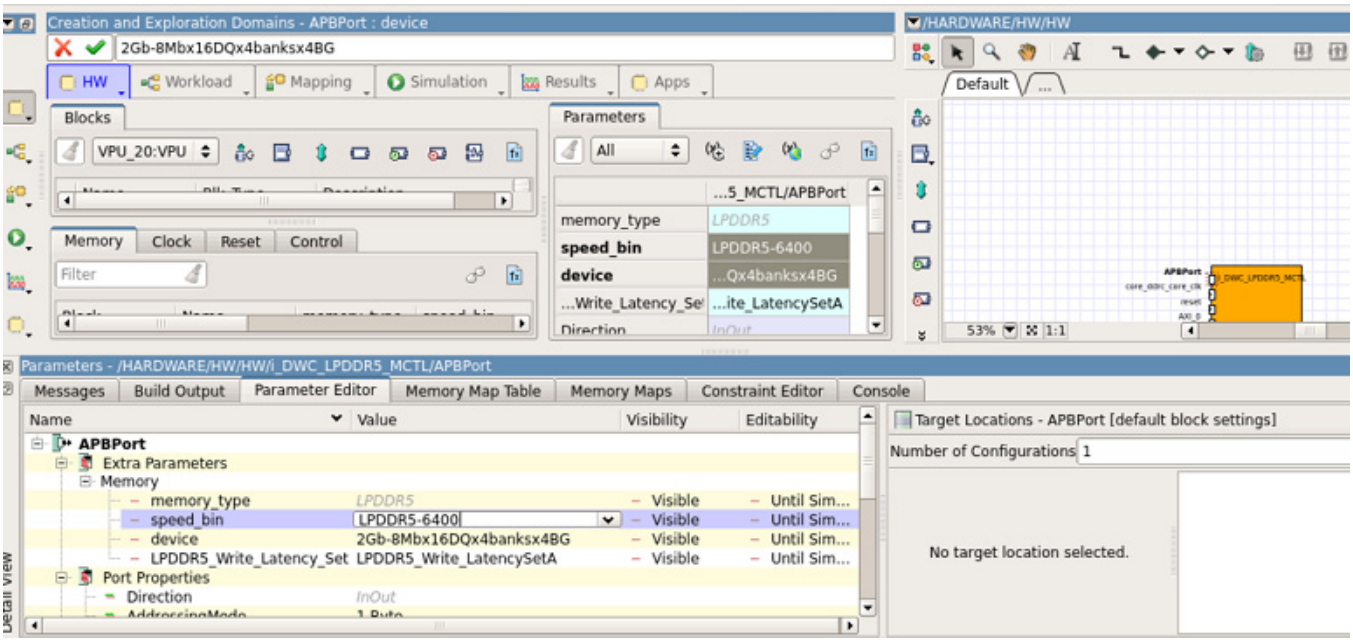
You are expected to review all warning messages from this utility.

The STL file for each port are named according to the name of the trace file. The port number for each port is added at the end to signify the port for which the STL file is generated.

3.4 Specifying Memory Options

You can specify the memory options on the block instance to represent the actual memory specifications by configuring memory options at `APBPort`. These memory options are dependent on the selection of the `/DDRC/memory_type` parameter on the instance. Typically, a user is interested in specification of memory speed bins, device type and additive latencies wherever possible. Based on the selection of memory speed bins the `DRAMTMG` registers are initialized to valid values as per `JEDEC` specifications. You are free to over-write any of these initialized values by programming the registers using `APBPort` on the block instance.

Figure 3-9 APBPort Creation and Exploration Domain



Chapter 4
Programming Model

This chapter describes the programming model for DWC_LPDDR5_MCTL which are not specified on LPDDR5_MCTL block but can be programmed through Register programming.

4.1 Programming Model for LPDDR5_MCTL

4.1.1 Register Memory Map

All registers are addressable on 32-bit boundaries, each unused bit or address location is reserved for future use and read back 0. The register access modes are as follows:

- R - read only
- W - write only
- R/W - Read/write

The DRAM timings provided are in units of DRAM clock cycle.

If you explicitly program the DRAM registers through the register port, than timings are applied as it is without any modification.

4.1.2 Register and Field Description

4.1.2.1 DRAMTMG0

- Size: 32-bits
- Offset: 0x0
- Memory Access: R/W
- Value after Reset: 0xf101b0f

Table 4-1 DRAMTMG0 Register Fields

Bits	Name	Memory Access	Description
31:24	wr2pre	R/W	Minimum time between write and precharge to same bank. Specifications: WL + BL/2 + tWR = approximately 8 cycles + 15 ns = 14 clocks @400MHz and less for lower frequencies; where: • WL= write latency • BL= burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM. BST (burst terminate) is not supported at present. • tWR = Write recovery time. This comes directly from the SDRAM specification. Add one extra cycle for LPDDR4 for this parameter. NOTE: Depending on the PHY, if using LRDIMM, it may be necessary to adjust the value of this parameter to compensate for the extra cycle of latency through the LRDIMM. • Unit: DRAM clock cycles. • Value After Reset: 0xf • Exists: Always • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4
23:16	t_faw	R/W	tFAW valid only when eight or more banks (or banks x bank groups) are present. In 8-bank design, at most four banks must be activated in a rolling window of tFAW cycles. In a 4-bank design, set this register to 0x1 independent of the 1:1/1:2 frequency mode. • Unit: DRAM clock cycles. • Value After Reset: 0x10 • Exists: Always • Programming Mode: Quasi-dynamic Group 2, Group 4
15:8	t_ras_max	R/W	tRAS (max): Maximum time between activate and precharge to same bank. This is the maximum time that a page can be kept open. Minimum value of this register is 1. Zero is invalid. t_ras_max should be set to RoundDown (tRAS(max)/tCK/1024). • Unit: Multiples of 1024 DRAM clock cycles. • Value After Reset: 0x1b • Exists: Always • Programming Mode: Quasi-dynamic Group 2, Group 4

Bits	Name	Memory Access	Description
7:0	t_ras_min	R/W	tRAS (min): Minimum time between activate and precharge to the same bank. When the controller is operating in 1:2 frequency mode, 1T mode, program this to tRAS(min)/2. No rounding up. When the controller is operating in 2T mode or LPDDR4 mode, program this to (tRAS(min)/2) and round it up to the next integer value. • Unit: DRAM clock cycles. • Value After Reset: 0xf • Exists: Always • Programming Mode: Quasi-dynamic Group 2, Group 4

4.1.2.2 DRAMTMG1

- Size: 32-bits
- Offset: 0x4
- Memory Access: R/W
- Value after Reset: 0x80414

Table 4-2 DRAMTMG1 Register Fields

Bits	Name	Memory Access	Description
31:24	t_rcd_write	R/W	tRCD: Minimum time from activate to same write command to the same bank in LPDDR5X mode. Unit: DRAM clock cycles. Value After Reset: 0x5 Exists: MEMC_LPDDR5X==1 Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4.
31:22			Reserved Field: Yes
15:8	rd2pre	R/W	tRTP: Minimum time from read to precharge of same bank. DDR4: Max of following two equations: tAL + max (tRTP, 4) or, RL + BL/2 - tRP (*). LPDDR4: BL/2 + max(tRTP,8) - 8 (*) When both DDR4 SDRAM and ST-MRAM are used simultaneously, use SDRAM's tRP value for calculation. When the controller is operating in 2T mode or LPDDR4 mode, divide the above value by two and round it up to the next integer value. • Unit: DRAM clock cycles. • Value After Reset: 0x4 • Exists: Always • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4

Bits	Name	Memory Access	Description
7:0	t_rc	R/W	tRC: Minimum time between activates to same bank. • Unit: DRAM clock cycles. • Value After Reset: 0x14 • Exists: Always • Programming Mode: Quasi-dynamic Group 2, Group 4

4.1.2.3 DRAMTMG2

- Size: 32-bits
- Offset: 0x8
- Memory Access: R/W
- Value after Reset: 0x0305060d

Table 4-3 DRAMTMG2 Register Fields

Bits	Name	Memory Access	Description
31			Reserved Field: Yes
30:24	write_latency	R/W	Set to WL Time from write command to write data on SDRAM interface. This must be set to WL. NOTE: Depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to adjust the value of WL to compensate for the extra cycle of latency through the RDIMM/LRDIMM. For all protocols, in addition to programming this register field, it is necessary to program DFITMG0 and DFITMG1 to control the read and write latencies. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4
23:22			Reserved Field: Yes
22:16	read_latency	R/W	Set to RL Time from read command to read data on SDRAM interface. This must be set to RL. NOTE: Depending on the PHY, if using RDIMM/LRDIMM, it may be necessary to adjust the value of RL to compensate for the extra cycle of latency through the RDIMM/LRDIMM. For all protocols, in addition to programming this register field, it is necessary to program DFITMG0 and DFITMG1 to control the read and write latencies • Unit: DRAM clock cycles. • Value After Reset: 0x5 • Exists: MEMC_DDR4_OR_LPDDR4==1 • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4

Bits	Name	Memory Access	Description
15:14			Reserved Field: Yes
13:8			• DDR4: RL + BL/2 + 1 + WR_PREAMBLE - WL • LPDDR4 (DQ ODT is Disabled): RL + BL/2 + RU(tDQSCKmax/tCK) + WR_PREAMBLE + RD_POSTAMBLE - WL • LPDDR4 (DQ ODT is Enabled): RL + BL/2 + RU(tDQSCKmax/tCK) + RD_POSTAMBLE - ODTLon - RU(tODTon(min)/tCK) + 1 • LPDDR5: RL + BL/n_max + RU(tWCKDQO(max)/tCK) - WL Minimum time from read command to write command. Include time for bus turnaround and all per-bank, per-rank, and global constraints. For details on what to include here, see the relevant PHY databook, where: • WL = write latency • BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM. • RL = read latency = CAS latency • WR_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble). This is unique to DDR4 and LPDDR4. • RD_POSTAMBLE = 0.5 (0.5tCK read postamble), 1.5 (1.5tCK read postamble). This is unique to LPDDR4. After PHY has completed training the value programmed may need to be increased. Refer to relevant PHY documentation. For LPDDR4, if derating is enabled (DERATEEN.derate_enable=1), derated tDQSCKmax should be used. NOTE: Depending on the PHY, if using LRDIMM, it may be necessary to adjust the value of this parameter to compensate for the extra cycle of latency through the LRDIMM. • Unit: DRAM clock cycles. • Value After Reset: 0x6 • Exists: Always • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4.
7:6			Reserved Field: Yes

Bits	Name	Memory Access	Description
7:0	wr2rd	R/W	• DDR4: CWL + PL + BL/2 + tWTR_L • LPDDR4: WL + BL/2 + tWTR + 1 • Others: CWL + BL/2 + tWTR In DDR4, minimum time from write command to read command for same bank group. In others, minimum time from write command to read command. Includes time for bus turnaround, recovery times, and all per-bank, per-rank, and global constraints. For details on what should be included here, see the relevant PHY databook, where: • CWL = CAS write latency • WL = Write latency • PL = Parity latency • BL = Burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM • tWTR_L = internal write to read command delay for same bank group. This comes directly from the SDRAM specification. • tWTR = internal write to read command delay. This comes directly from the SDRAM specification. After PHY has completed training the value programmed may need to be increased. Refer to relevant PHY documentation. Add one extra cycle for LPDDR4 operation. WTR_L should be increased by one if DDR4 2tCK write preamble is used. • Unit: DRAM clock cycles. • Value After Reset: 0xd • Exists: Always • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4.

4.1.2.4 DRAMTMG4

- Size: 32-bits
- Offset: 0x10
- Memory Access: R/W
- Value after Reset: 0x5040405

Table 4-4 DRAMTMG4 Register Fields

Bits	Name	Memory Access	Description
31:30			Reserved Field: Yes

Bits	Name	Memory Access	Description
30:24	t_rcd	R/W	tRCD - tAL: Minimum time from activate to read or write command to same bank. Minimum value allowed for this register is 1. • Unit: DRAM clock cycles. • Value After Reset: 0x5 • Exists: Always • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4.
23:22			Reserved Field: Yes
21:16	t_ccd	R/W	DDR4: tCCD_L: This is the minimum time between two reads or two writes for same bank group. Others: tCCD: This is the minimum time between two reads or two writes. • Unit: DRAM clock cycles. • Value After Reset: 0x4 • Exists: Always • Programming Mode: Quasi-dynamic Group 2, Group 4.
15:14			Reserved Field: Yes
13:8	t_rrd	R/W	DDR4: tRRD_L: Minimum time between activates from bank a to bank b for same bank group. Others: tRRD: Minimum time between activates from bank a to bank b. • Unit: DRAM clock cycles. • Value After Reset: 0x4 • Exists: Always • Programming Mode: Quasi-dynamic Group 2, Group 4.
7:6			Reserved Field: Yes
6:0	t_rp	R/W	tRP: Minimum time from single-bank precharge to activate of same bank. When the controller is operating in 1:1 frequency ratio mode, t_rp should be set to RoundUp (tRP/tCK). • Unit: DRAM clock cycles. • Value After Reset: 0x5 • Exists: Always • Programming Mode: Quasi-dynamic Group 2, Group 4.

4.1.2.5 DRAMTMG9

- Size: 32-bits
- Offset: 0x24
- Memory Access: R/W
- Value after Reset: 0x4040d

Table 4-5 DRAMTMG9 Register Fields

Bits	Name	Memory Access	Description
31:21			Reserved Field: Yes
20:16	t_ccd_s	R/W	tCCD_S: This is the minimum time between two reads or two writes for different bank group. For bank switching (from bank a to bank b), the minimum time is this value + 1. Present only in designs configured to support DDR4/5. • Unit: DRAM clock cycles. • Value After Reset: 0x4 • Exists: MEMC_DDR4_OR_DDR5==1 • Programming Mode: Quasi-dynamic Group 2, Group 4.
15:14			Reserved Field: Yes
13:8	t_rrd_s	R/W	tRRD_S: Minimum time between activates from bank a to bank b for different bank group. Present only in designs configured to support DDR4/5. • Unit: DRAM clock cycles. • Value After Reset: 0x4 • Exists: MEMC_DDR4_OR_DDR5==1 • Programming Mode: Quasi-dynamic Group 2, Group 4.
7:0	wr2rd_s	R/W	CWL + PL + BL/2 + tWTR_S Minimum time from write command to read command for different bank group. Includes time for bus turnaround, recovery times, and all per-bank, per-rank, and global constraints. Present only in designs configured to support DDR4/5, where: • CWL = CAS write latency • PL = Parity latency • BL = burst length. This must match the value programmed in the BL bit of the mode register to the SDRAM. • tWTR_S = internal write to read command delay for different bank group. This comes directly from the SDRAM specification. WTR_S should be increased by one if DDR4 2tCK write preamble is used. • Unit: DRAM clock cycles. • Value After Reset: 0xd • Exists: MEMC_DDR4_OR_DDR5==1 • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4.

4.1.2.6 DRAMTMG13

- Size: 32-bits
- Offset: 0x34
- Memory Access: R/W

- Value after Reset: 0x10

Table 4-6 DRAMTMG13 Register Fields

Bits	Name	Memory Access	Description
31			Reserved Field: Yes
23			Reserved Field: Yes
22:16	t_ccd_mw	R/W	LPDDR4: tCCDMW: This is the minimum time from write or masked write to masked write command for same bank. • Unit: DRAM clock cycles. • Value After Reset: 0x20 • Exists: MEMC_LPDDR4_OR_LPDDR5==1 • Programming Mode: Quasi-dynamic Group 2, Group 4.
15:4			Reserved Field: Yes
3:0	t_ppd	R/W	LPDDR4: tPPD: This is the minimum time from precharge to precharge command. • Unit: DRAM clock cycles. • Value After Reset: 0x4 • Exists: MEMC_LPDDR4_OR_LPDDR5_OR_DDR5==1 • Programming Mode: Quasi-dynamic Group 2, Group 4.

4.1.2.7 DRAMTMG23

- Size: 32-bits
- Offset: 0x5C
- Memory Access: R/W
- Value after Reset: 0x10

Table 4-7 DRAMTMG23 Register Fields

Bits	Name	Memory Access	Description
21:16	wckenl_rd	R/W	WCKENL (RD) Exists: MEMC_LPDDR5==1
5:0	wckenl_wr	R/W	WCKENL/tCK (WR) Exists: MEMC_LPDDR5==1

4.1.2.8 RANKTMG0

- Size: 32-bits
- Offset: 0xd04
- Memory Access: R/W
- Value after Reset: 0x606

Table 4-8 RANKTMG0 Register Fields

Bits	Name	Memory Access	Description
15:8	diff_rank_wr_gap	R/W	Indicates gap in data responses when performing consecutive writes to different ranks. Unit - DRAM clock cycles Exists: multirank configurations
7:0	diff_rank_rd_gap	R/W	Indicates gap in data responses when performing consecutive reads to different ranks. Unit - DRAM clock cycles Exists: multirank configurations

4.1.2.9 RANKTMG1

- Size: 32-bits
- Offset: 0xd08
- Memory Access: R/W
- Value after Reset: 0xff

Table 4-9 RANKTMG1 Register Fields

Bits	Name	Memory Access	Description
31:16			Reserved Field: Yes
15:8	rd2wr_dr	R/W	Minimum time from read command to write command for different rank. Includes time for bus turnaround, recovery time, and all per bank, per rank, and global constraints. The value must be larger than or equal to the value of DRAMTMG2. For LPDDR5, set to "JEDEC formula + tphy_wckcsgap + board delay". For the value of tphy_wckcsgap, see PHY databook. • Unit: DRAM clock cycles. • Value After Reset: 0xf • Exists: multirank configurations • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4.

Bits	Name	Memory Access	Description
7:0	wr2rd_dr	R/W	Minimum time from write command to read command for different rank. Includes time for bus turnaround, recovery time, and all per bank, per rank, and global constraints. For LPDDR5, set to "JEDEC formula + tphy_wckcsgap + board delay". For the value of tphy_wckcsgap, see PHY databook. • Unit: DRAM clock cycles. • Value After Reset: 0xf • Exists: multirank configurations • Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4.

4.1.2.10 RFSHTMG0

- Size: 32-bits
- Offset: 0x600
- Memory Access: R/W
- Value after Reset: 0x210

Table 4-10 RFSHTMG0 Register Fields

Bits	Name	Memory Access	Description
31	t_refi_x1_sel	R/W	Specifies whether t_refi_x1_x32 and refresh_to_x1_x32 register values are x1 or x32. • 0 - x32 register values are used, • 1 - x1 register values are used. This applies only when per-bank refresh is enabled (per_bank_refresh=1); if per-bank refresh is not enabled, the x32 register values are used and this register field is ignored. • Value After Reset: 0x0 • Exists: Always • Programming Mode: Dynamic - Refresh Related
30:28			Reserved Field: Yes
27:24	refresh_margin	R/W	Threshold value in number of DRAM clock cycles before the critical refresh or page timer expires. A critical refresh is to be issued before this threshold is reached. It is recommended that its default value is retained, currently it is 0x2. It must always be less than internally used t_refi/32. • Unit: Multiples of 32 DRAM clock cycles. • Value After Reset: 0x2 • Exists: Always • Programming Mode: Dynamic - Refresh Related

Bits	Name	Memory Access	Description
23:22			Reserved Field: Yes
21:16	refresh_to_x1_x32	R/W	If the refresh timer has expired at least once, that is >tREFI period elapses, and there are postponed refreshes, then a speculative refresh may be performed. A speculative refresh is a refresh performed at a time when refresh would be useful. When there are no transactions pending in the CAM for a period of time determined by this refresh_to_x1_x32 and the refresh timer has expired at least once since the last refresh, then a speculative refresh is performed. Speculative refreshes continues successively until there are no refreshes pending or until new reads or writes are issued to the DDRCTL. This is also used for non speculative refresh when LPDDR per-bank refresh (REFpb) is enabled. • Unit: DRAM clock cycles or multiples of 32 DRAM clock cycles, depending on t_refi_x1_sel. • Value After Reset: 0x10 • Exists: Always • Programming Mode: Dynamic - Refresh Related
15:12			Reserved Field: Yes
11:0	t_refi_x1_x32		Average time interval between refreshes per rank. Set this register to RoundDown(tREFI/tCK) if t_refi_x1_sel = 0, divide the above result by 32 and round down. - if using all-bank refreshes (per_bank_refresh = 0), use tREFIab in the above calculations. - if using per-bank refreshes (per_bank_refresh = 1), use tREFIpb in the above calculations. • Unit: DRAM clock cycles or multiples of 32 DRAM clock cycles, depending on t_refi_x1_sel. • Value After Reset: 0x62 • Exists: Always • Programming Mode: Dynamic - Refresh Related

4.1.2.11 RFSHTMG1

- Size: 32-bits
- Offset: 0x604
- Memory Access: R/W
- Value after Reset: 0x62008c

Table 4-11 RFSHTMG1 Register Fields

Bits	Name	Memory Access	Description
11:0	t_rfc_min	R/W	tRFC (min): Minimum time from refresh to refresh or activate. t_rfc_min should be set to RoundUp(tRFCmin/tCK). In LPDDR4/5 mode: • If using all-bank refreshes, the tRFCmin value in the above equations is equal to tRFCab • if using per-bank refreshes, the tRFCmin value in the above equations is equal to tRFCpb In DDR4 mode, the tRFCmin value in the above equations is different depending on the refresh mode (fixed 1X, 2X, 4X) and the device density. The user should program the appropriate value from the spec based on the refresh_mode and the device density that is used. • Unit: DRAM clock cycles. • Value After Reset: 0x8c • Exists: Always • Programming Mode: Dynamic - Refresh Related

4.1.2.12 RFSHTMG2

- Size: 32-bits
- Offset: 0x608
- Memory Access: R/W
- Value after Reset: 0x0

Table 4-12 RFSHTMG2 Register Fields

Bits	Name	Memory Access	Description
23:16	t_pbr2pbr	R/W	LPDDR4: tpbR2pbR Per-bank Refresh to Per-bank refresh different bank time. Program this to RoundUp(tpbR2pbR/tCK). The tpbR2pbR value in the above equations is different depending on the device density. Program the appropriate value from the spec. Register is valid only in LPDDR4 per-bank refresh mode (RFSHCTL0.per_bank_refresh_mode == 1). • Unit: DRAM clock cycles. • Value After Reset: 0x8c • Exists: MEMC_LPDDR4_OR_LPDDR5==1 • Programming Mode: Dynamic - Refresh Related
15:0			Reserved Field: Yes

4.1.2.13 RFSHTMG4

- Size: 32-bits
- Offset: 0x610
- Memory Access: R/W
- Value after Reset: 0x0

Table 4-13 RFSHTMG4 Register Fields

Bits	Name	Memory Access	Description
31:28			Reserved Field: Yes
27:16	refresh_timer1_start_value_x32	R/W	Refresh timer start for rank 1 (only present in multi-rank configurations). This is useful in staggering the refreshes to multiple ranks to help traffic to proceed. • Unit: Multiples of 32 DFI clock cycles. • Value After Reset: 0x0 • Exists: MEMC_NUM_RANKS>1 • Programming Mode: Dynamic - Refresh Related
15:12			Reserved Field: Yes
11:0	refresh_timer0_start_value_x32	R/W	Refresh timer start for rank 0 (only present in multi-rank configurations). This is useful in staggering the refreshes to multiple ranks to help traffic to proceed. • Unit: Multiples of 32 DFI clock cycles. • Value After Reset: 0x0 • Exists: MEMC_NUM_RANKS>1 • Programming Mode: Dynamic - Refresh Related

4.1.2.14 RFSHTMG5

- Size: 32-bits
- Offset: 0x614
- Memory Access: R/W
- Value after Reset: 0x0

Table 4-14 RFSHTMG5 Register Fields

Bits	Name	Memory Access	Description
31:28			Reserved Field: Yes

Bits	Name	Memory Access	Description
27:16	refresh_timer3_start_value_x32	R/W	Refresh timer start for rank 3 (only present in 4-rank configurations). This is useful in staggering the refreshes to multiple ranks to help traffic to proceed. • Unit: Multiples of 32 DFI clock cycles. • Value After Reset: 0x0 • Exists: MEMC_NUM_RANKS>2 • Programming Mode: Dynamic - Refresh Related
15:12			Reserved Field: Yes
11:0	refresh_timer2_start_value_x32	R/W	Refresh timer start for rank 2 (only present in 4-rank configurations). This is useful in staggering the refreshes to multiple ranks to help traffic to proceed. • Unit: Multiples of 32 DFI clock cycles. • Value After Reset: 0x0 • Exists: MEMC_NUM_RANKS>2 • Programming Mode: Dynamic - Refresh Related

Chapter 5
Analysis View

This chapter describes:

5.1 Introduction

The `LPDDR5_MCTL` library has inbuilt instrumentation for specific analysis views in the Synopsys analysis infrastructure. This instrumentation enables the recording of data during simulation, which is processed and presented in analysis views in VP Explorer. These analysis views help you to get a complete overview and insight on the performance of the `LPDDR5_MCTL` model in your design. To tune the model for optimal performance, iteratively modify the architecture parameters of the models or various register settings based on the configuration and traffic, and observe the results in the analysis views.

This chapter provides an overview on how to set up the design analysis and the features of the analysis infrastructure that help in the design analysis using the `LPDDR5_MCTL` library. It also describes some aspects of interpreting the data in the analysis views. A powerful `JEDEC` level analysis is provided with the `LPDDR5_MCTL` library to observe utilization of the memory interface, to study the root cause of memory access delays and improve the software efficiency. For more information on Synopsys analysis infrastructure, see “Tracing and Analysis” in the *VP Explorer User Guide*, which is available with the Platform Architect documentation set.

The `LPDDR5_MCTL` library provides support for reporting results in the following analysis views and charts in VP Explorer.

- *Bus Path Statistics* chart - The analysis view that records and displays statistical data over a time interval. *Bus Path Statistic* shows transaction count and delay information for all the internal components in the memory controller. This enables you to analyze and improve the design performance. For supported information, see “Path Statistics Chart” in the *VP Explorer User Guide*, which is available with the Platform Architect documentation set.
- *Bus Resource Statistics* chart - The analysis view that records and displays statistical data over a time interval. This view shows the utilization of various buffers, CAM depth utilization, contention information on the memory controller arbiter and enables you to analyze and improve the design performance. For supported information, see “Resource Statistics Chart” in the *VP Explorer User Guide*, which is available with the Platform Architect documentation set.

- *Memory Channel Utilization Stats* - This is a statistical view which records and displays memory command channel utilization data over a time interval. This analysis view gives a complete visualization of memory interface for the command channel. It also helps to know the root cause for any bottlenecks or limitations. For more information, see "Enable `JEDEC level analysis:` on page 70.
- *Memory Data Channel Utilization Stats* - This is a statistical view which records and displays memory data channel utilization data over a time interval. This analysis view gives a complete visualization of memory data channel interface activity. It also helps to know the root cause for any bottlenecks or limitations. For more information, see "Enable `JEDEC level analysis:` on page 70.
- *Hot Bit analysis* – This view helps a user to modify the address mapping information of the model to calculate the bit toggling count for a specific simulation run. For more information, see "Hot Bit Analysis" on page 79.

To create the *Bus Path Statistics*, *Bus Resource Statistics*, and *Transaction Trace* charts, the analysis infrastructure uses an analysis database from the simulation. The monitors that you attach to the simulation session enables data recording to this analysis database. For more information, see "Bus Debug Monitor" and "Bus Analysis Monitor" in the *VP Explorer User Guide*, which is available with the Platform Architect documentation set.

5.2 Using Analysis in a Design with LPDDR5_MCTL

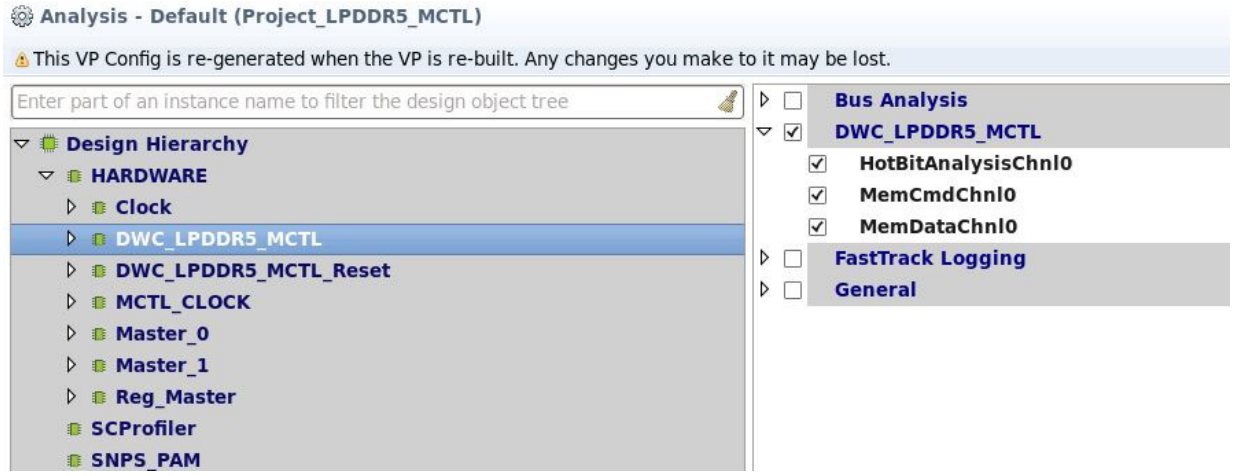
This section describes the steps to attach bus debug and analysis monitors to the `LPDDR5_MCTL` hierarchy and set up the display for corresponding analysis views. For further details on the bus analysis flow and the steps to perform analysis, see "Bus Analysis Monitor" in the *VP Explorer User Guide*, which is available with the Platform Architect documentation set.

To use the analysis on your design, export and built the simulation project of a design with the `LPDDR5_MCTL` model.

To enable analysis on your design with the LPDDR5_MCTL:

1. Start the simulation session in VP Explorer.
2. From the Analysis Configuration dialog box, enable bus analysis or bus debug. For details, see "Bus Debug Monitor Reference" and "Bus Analysis Monitor Reference" in the *VP Explorer User Guide*, which is available with the Platform Architect documentation set.
3. From the Configure Attributes dialog box, you can optionally set the attributes if bus analysis is enabled. For more details, see "Configuration" under "Bus Analysis Monitor Reference" in the *VP Explorer User Guide,* which is available with the Platform Architect documentation set.
4. Enable `JEDEC` level analysis:
   1. From the VP Explorer menu bar, select *Analysis* > *Configure*.
   2. From the Configure Analysis dialog box, select the checkboxes to enable the required Memory Channel and Memory Data Channel analysis views as shown below.

Figure 5-1 Enabling JEDEC Level Analysis



3. Click *OK* to save the changes.

5 You can also enable *HotBitAnalysis* in the Configure Analysis dialog box to optimize the address mapping settings.

6 Run the complete simulation.

7 Send the data for visualization in the Chart view using the Result view. For more details, see "Using the Results View" in the *VP Explorer User Guide*, which is available with the Platform Architect documentation set.

8 Typically, you will use the pop-up menu of the current simulation node in the Results view to launch the Bus Path or Resource Statistics view or the Transaction Trace view.

9 You can create multiple charts of the same type, like Bus Paths Statistics chart, in the Results view. Use this to simultaneously inspect related metrics at the specific node. For example, utilization and throughput.

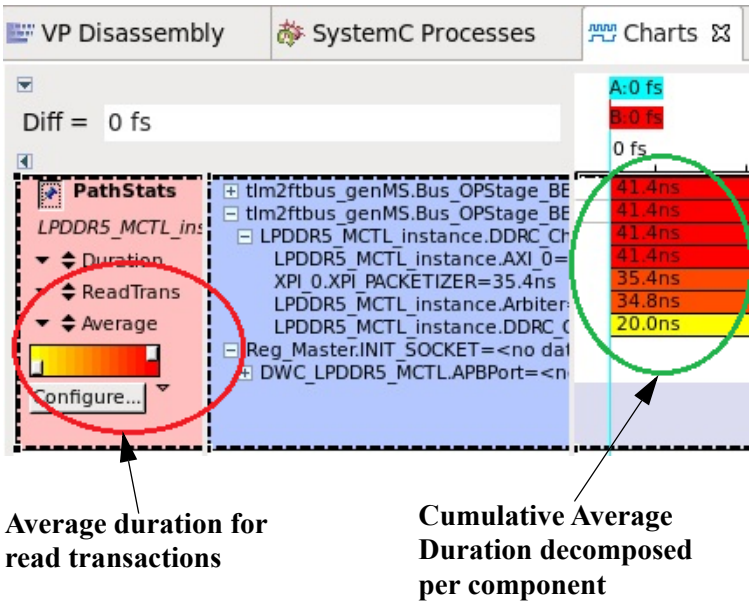
10 To configure the visualization further, use the Configure button supported by each of these charts. For more details, see "Configuration" under "Bus Analysis Charts" in the *VP Explorer User Guide*, which is available with the Platform Architect documentation set.

5.3 Bus Path Statistics

The Bus Path Statistics analysis view records and displays statistical data over a time interval. This view provides an end to end analysis of the entire data flow from a particular initiator to the end `SDRAM` memory. Thus gives `delay`, `throughput` and `utilization` statistics for every internal component inside the Memory Controller and helps in identifying the real bottleneck for Memory Controller optimization.

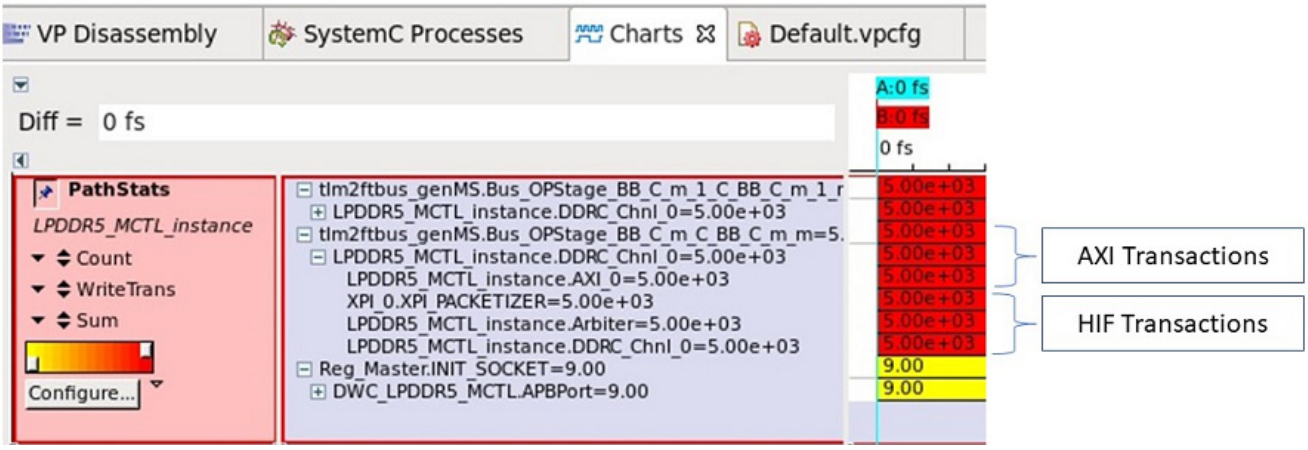
The figure below shows the Bus Path Statistics view, showing average delay in `read` transactions, decomposed over all the components in the entire data path from `CPU_CL0_0` to the memory interface of `DDRC` inside the `LPDDR5_MCTL` model.

Figure 5-2 Bus Path Statistics View



This analysis view helps the user to analyze `count`, `latency`, `throughput` and `utilization` characteristics of traffic from a particular initiator. This view also decomposes the `count`, `latency`, `throughput` and `utilization` statistics on every component of memory controller. For example, in the above figure, the average duration of the `read` transaction from initiator `CPU_CL0_0` is `410.392 ns`. You can visualize from the path view, the contribution of each component in the read latency. The average read latency at the `DDRC PHY` interface is `37.5 ns`. The latency between the arbiter and `DDRC` interface is `151.908 ns`. This includes the latency due to the `DDRC` interface. You can judge from this analysis that the read latencies are incurred primarily at the `XPI` input interface and the interface between Arbiter and `DDRC`.

Figure 5-3 Chart View Showing AXI Write Transactions



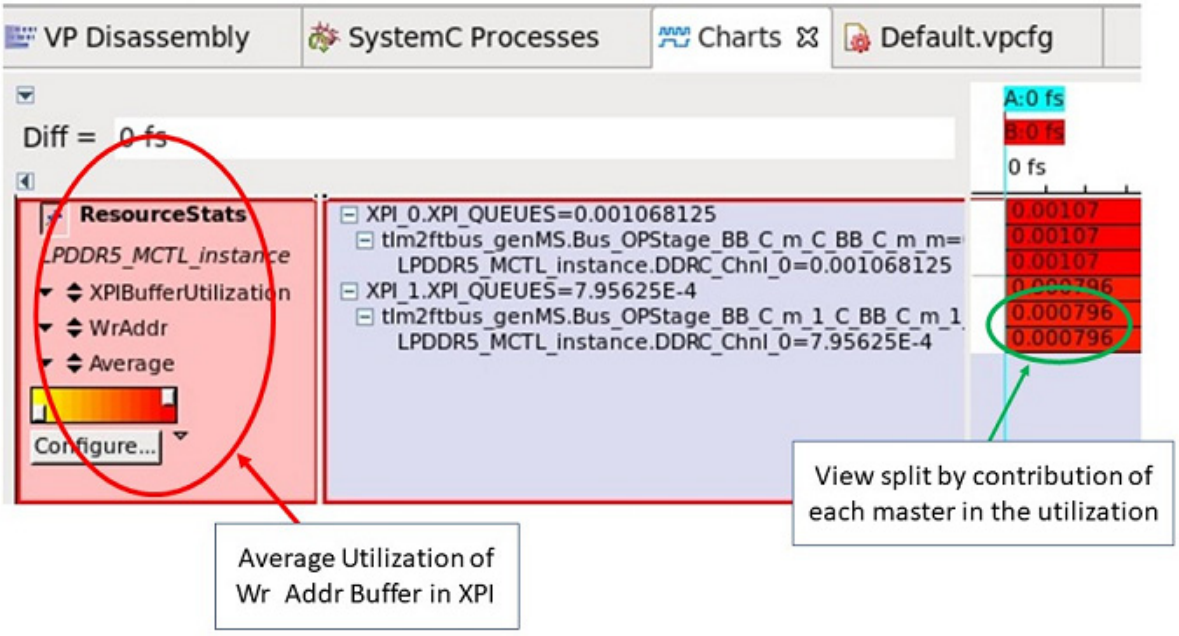
The above figure shows how the AXI `Write` transactions are broken into multiple HIF transactions and the total count of AXI transactions against total count of HIF transactions.

5.4 Resources Statistics

The Bus Resource Statistics provide insight into the utilization of various resources within the model. These resources can be the buffers or CAM storage inside the model; or the contention numbers as well as DDRC stalls. This view can be used to arrive at the optimal settings of the parameters and registers in context of the traffic. Every resource metric can be viewed as minimum, maximum and average value.

The figure below shows the Resource Statistic view of average utilization of Write Address Buffer in XPI blocks. The view can be expanded to find out the contribution by each initiator.

Figure 5-4 Resource Statistic View



Following metric in Resource Statistics views are available for LPDDR5_MCTL model.

- CAMUtilization
- DDRCStall
- XPI Buffer Utilization
- Contention
- RRBUtilization

5.4.1 CAMUtilization

The LPDDR5_MCTL model has a content addressable memory (CAM) inside DDRC that contains the memory transactions before they are scheduled on the memory interface. The size of CAM can impact the overall performance of the scheduler and hence this metric can be used to arrive at the optimal settings of CAM. There is one CAM for Write transactions and one for Read transactions. The Read transactions can be of type Low Priority and High Priority based on the register configurations and QoS values. The Read CAM can be divided for utilization of Low Priority Reads or High Priority Reads based on register settings (SCHED.lpr_num_entries).

This metric can be grouped as follows:

- WrAddr

- LoPrioRdAddr
- HiPrioRdAddr

5.4.2 DDRCStall

When there is a Read or Write transaction that collides with an existing Read or Write transaction in the CAM, then the DDRC block issues a stall command to the arbiter. On the assertion of this stall command, the transactions are not forwarded to the DDRC block and this can cause higher average latencies for the transactions. DDRC stall also causes inefficient usage of the memory and should be avoided. The average value of DDRC stall represents the duration when this signal is high.

5.4.3 XPI Buffer Utilization

The LPDDR5_MCTL model has buffers for every AXI channel on each port. These buffers are utilized to store the AXI transfers, before forwarding this to the other side. The Read address channel is further divided into a Red queue and a Blue queue based on the QoS values.

This metric can be grouped as follows:

- WrAddr
- WrData
- WrRsp
- RdAddrBlue
- RdAddrRed
- RdData

5.4.4 Contention

The LPDDR5_MCTL arbiter does arbitration for RdAddr and WrAddr requests coming from various ports on the block. In case, there are multiple requests at same time, the arbiter gives grant based on various policies that are internal to the model behavior. In case, a particular port does not get the grant from the arbiter, the port is said to be contending. This contention for each initiator can be viewed in the Resource Statistic view.

This metric has only one group named request.

5.4.5 RRBUtilization

The read data can be returned from the DDRC in a different order from which the read commands are forwarded from the XPI. A read reorder buffer is implemented in each AXI port to reorder the read data for that port, to the same order (per AXI ID) as the order of the AXI read commands. The read reorder buffer, SRAM, holds the same number of entries as the read CAM and each entry holds the read data corresponding to a DDR command. Storage for the reorder buffer is implemented internally as virtual channels.

This ResourceStats provide information on how many virtual channels are used.

5.5 Memory Channel Analysis and Memory Data Channel Analysis

One of the key requirements in exploration of the architecture of LPDDR5_MCTL is to get maximum possible utilization of the memory channel. The memory interface is divided into a command channel and a data channel. The command signals are signified by the selection of RAS, CAS and WE pins. The data channel is as wide as the memory data width parameter on the block. The utilization of the Memory

Data Channel should be maximized to get higher memory throughput. You can observe memory transactions on the memory channel using this metric and visualize the effect of any parameter or register changes on the memory utilization.

5.5.1 Memory Channel Analysis

The Memory Channel Analysis metric can be viewed by sending the results to the Memory Channel Utilization Statistics.

Utilization values are calculated as shown below:

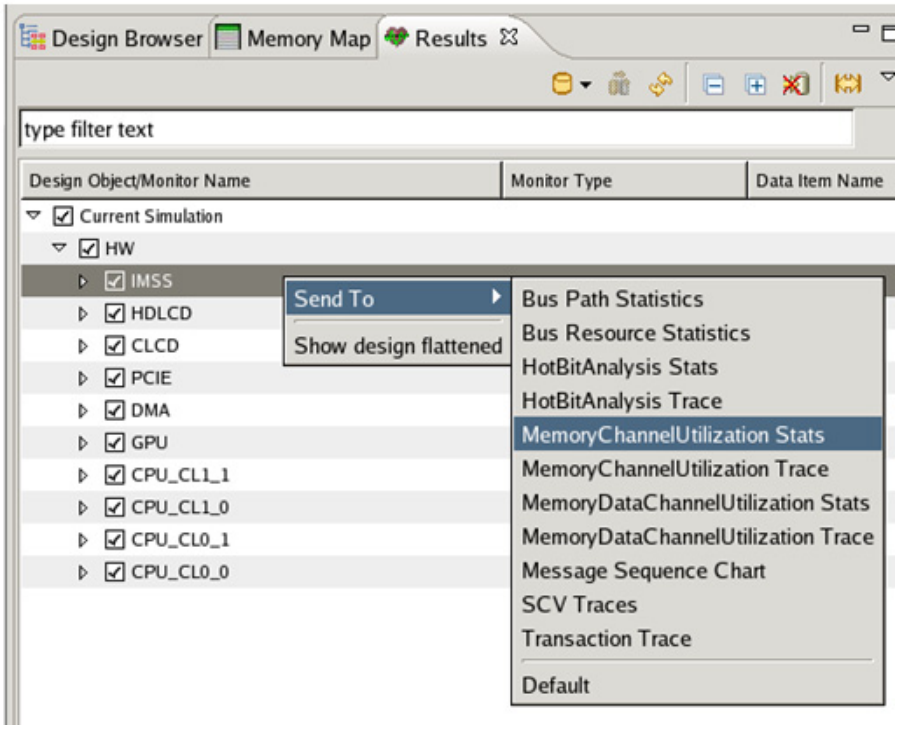
`IDLE` utilization = (time range of command channel is idle)/(interval time)

`CMD_SETUP` utilization = (time range when there were meaningful commands on the Memory Command Channel)/(interval time)

Sending results for memory channel analysis:

1. From the *Results* tab of the VP Explorer menu bar, select the check-box for an instance under *Design Object/Monitor Name.*
2. Right-click on the instance and from the contextual menu, select *Send To* > *MemoryChannelUtilization Stats,* as shown in the figure below.

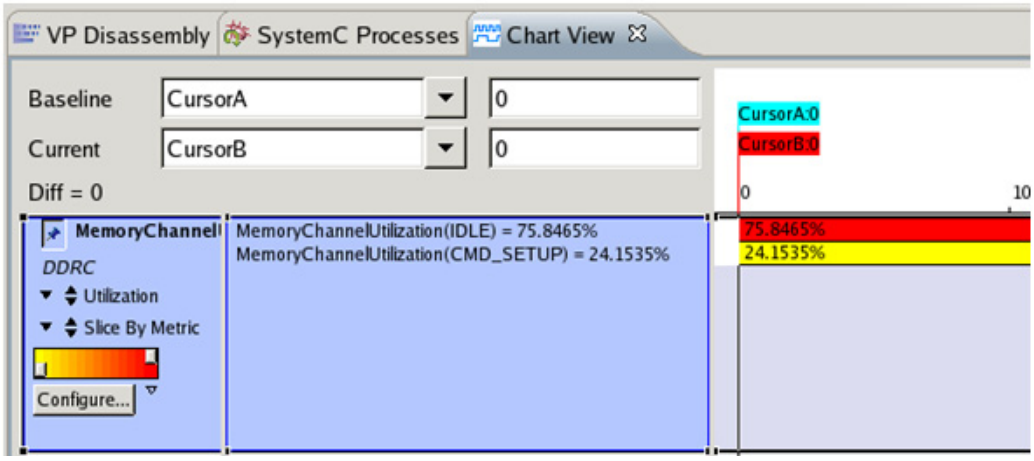
Figure 5-5 Memory Channel Utilization Statistics



The Chart view opens up for the selected instance.

Once you have the analysis database in Chart view, it reports the percentage of time when Memory Command Channel was idle and when there were meaningful commands on the Memory Command Channel as shown in figure below.

Figure 5-6 Chart View with Analysis Database

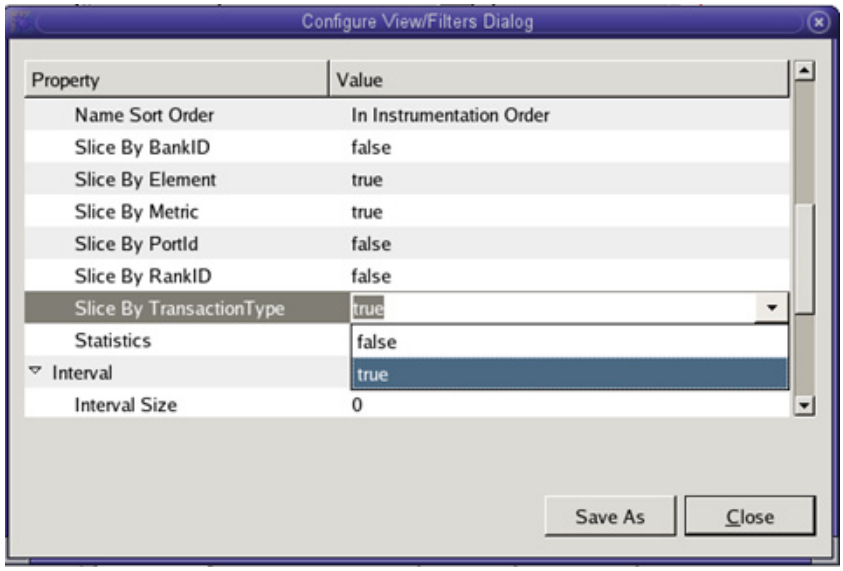


3 To find out more information about the commands, click the *Configure* button in the Chart View.

The Configure View/Filters dialog box pops up, where you can select the slicing of analysis data as per your requirements. The `slicing` operation further breaks the metric as per the available views. To do so:

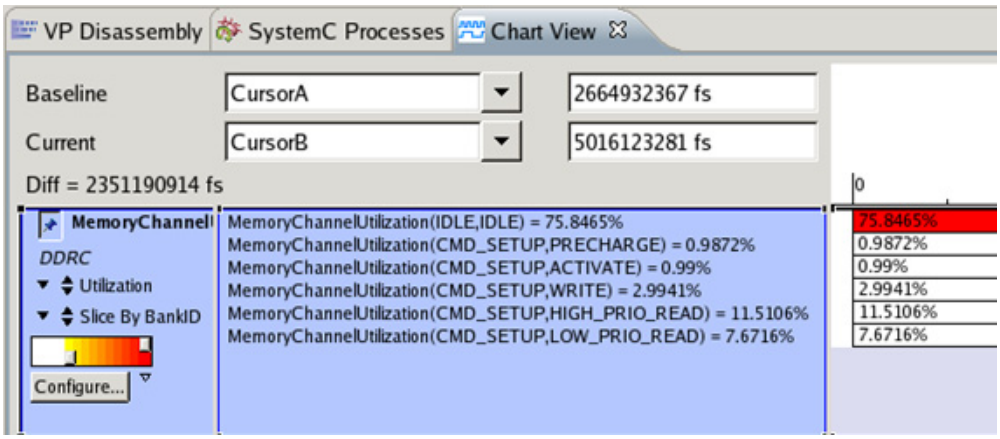
a. In the Configure View/Filters dialog box, select the value for property *Slice by TransactionType* as `true`, as shown in figure below.

Figure 5-7 Configure View /Filters Dialog Box



b. The view splits to show the various transactions that got scheduled on the Memory Command Channel, as shown in figure below:

Figure 5-8 Chart View for Slicing Based on Transaction Type



The analysis metric can be split according to following views:

- Transaction Type: Shows the various transaction types scheduled on the Memory Command channel. These are `Precharge`, `Activates`, `Write`, `Low Priority Reads` and `High Priority Reads`.
- Rank ID: Shows the commands as per the memory ranks.
- Bank ID: Shows the commands as per the memory banks.
- Port ID: Shows the commands as per the ports from which the corresponding AXI transaction was received.
- Transaction ID: Shows the commands as per the transaction IDs. Set environment variable `SNPS_ENABLE_ANALYSIS_SLICE_BY_TID` to `1` in VP Explorer to enable it.
- Add new point->
- `IdleCause`: Shows the cause of idle between commands. Causes are represented in terms of JEDEC defined timings, that is, `tCCD`, `tRCD`. See figure 5-10, currently this feature is protected by environment variable `SNPS_PAU_ENABLE_ANALYSIS_IDLECAUSE`.

You can select any combination of these views to get a complex view. For example, if you select *Slice By TransactionType* and *Slice By PortId* in Configure View/Filters dialog box (refer figure 5-7), you can find the number of `Low Priority Read Commands` scheduled on the memory interface that arrived on port `AXI[2]` of the memory controller, as shown in figure below.

Figure 5-9 Chart View Based on Transaction Type and Port ID

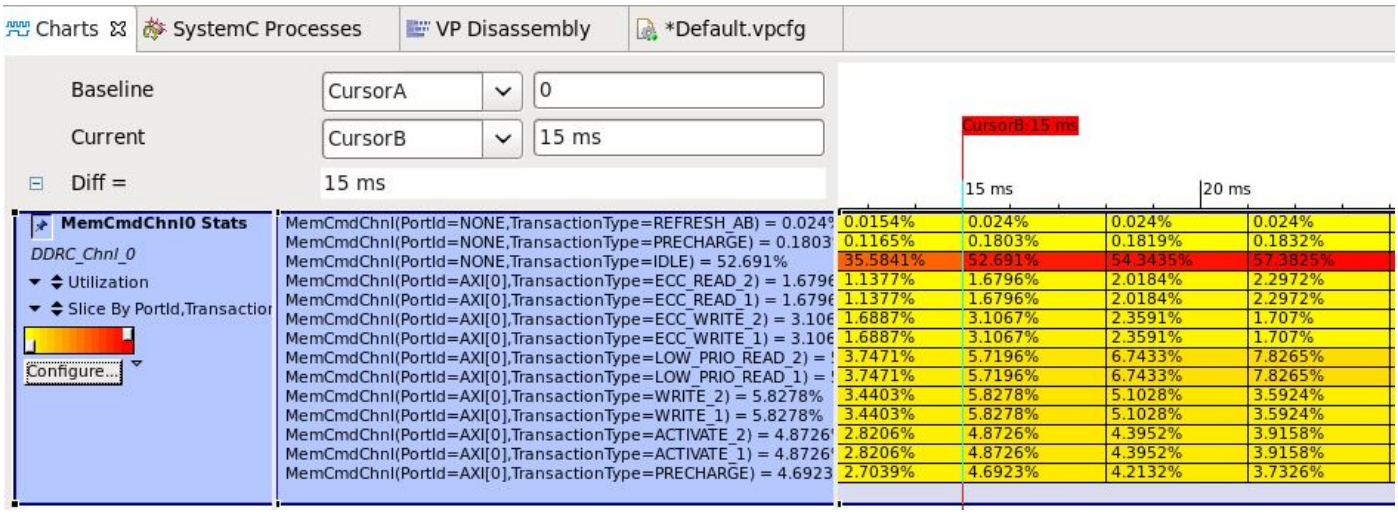
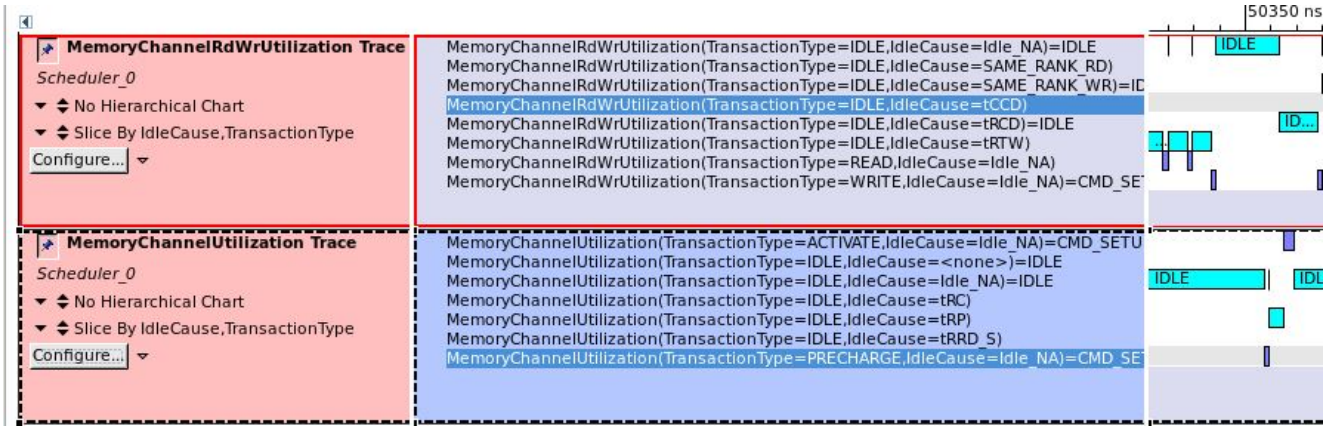


Figure 5-10 Chart View Based on Transaction Type and IdleCause



5.5.2 Memory Data Channel Analysis

The Memory Data Channel Analysis metric represents the utilization of the data channel on the memory interface. This metric can be viewed by sending the results to the Memory Data Channel Utilization Statistics.

Utilization values are calculated as shown below:

`IDLE` utilization = (time range of data channel is idle)/(interval time)

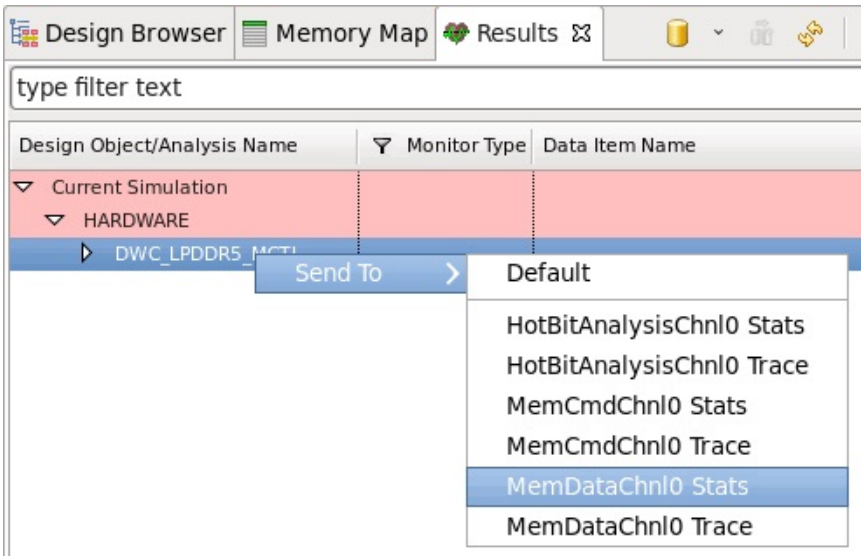
`WRITE` utilization = (time range when data channel is transferring write data)/(interval time)

`READ` utilization = (time range when data channel is transferring read data)/(interval time)

Sending results for memory data channel analysis:

1. From the *Results* tab of the VP Explorer menu bar, select the check-box for an instance under *Design Object/Monitor Name.*
2. Right-click on the instance and from the contextual menu, select *Send To* > *MemoryDataChannelUtilization Stats,* as shown in the figure below.

Figure 5-11 Memory Data Channel Utilization Statistics



The Memory Data Channel Utilization metric can then be viewed and sliced as explained for the Memory Channel Utilization in "Memory Channel Analysis" on page 75.

The various views on which the Memory Data Channel Analysis metric can be sliced are:

- Transaction Type: This view breaks the metric as per the data transfer types like `Write`, `Low Priority Reads` or `High Priority Reads`.
- Rank ID: Shows the data phase as per the memory ranks.
- Bank ID: Shows the data phase as per the memory banks.
- Port ID: Shows the data phase as per the ports from which the corresponding AXI transaction was received.
- Transaction ID: Shows the commands as per the transaction IDs. Set environment variable `SNPS_ENABLE_ANALYSIS_SLICE_BY_TID` to `1` in VP Explorer to enable it.

Similar to the Memory Channel Analysis, the views in Memory Data Channel Analysis can also be combined to get more complex views.

5.6 Hot Bit Analysis

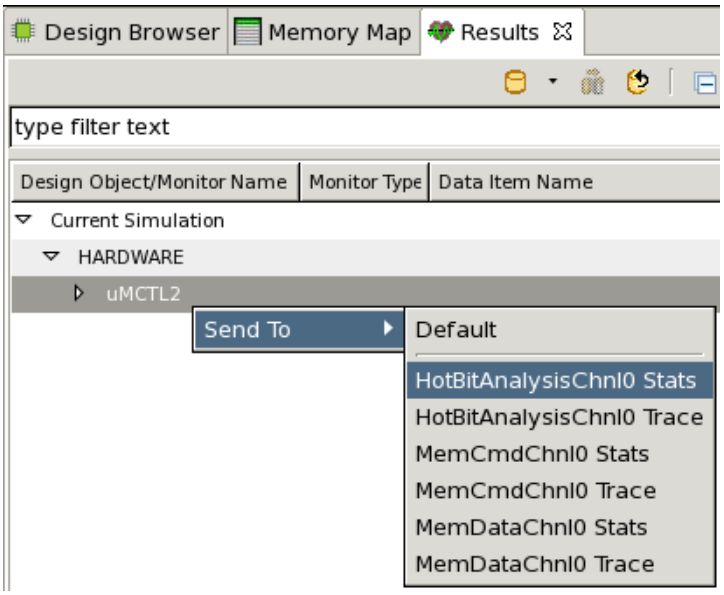
The Hot Bit Analysis view shows the count of transitions of bits for the HIF address. This information is very useful to arrive at an optimal setting of the Address Mapping Registers (`ADDRMAPx` registers). The analysis instrumentation is done after the arbiter and just before the scheduler, so that addresses from different AXI ports are combined to arrive at more logical hot bit analysis numbers.

To view the hot bit analysis numbers, send the results to the Hot Bit Analysis view as follows.

Sending results for hot bit analysis:

1. From the *Results* tab of the VP Explorer menu bar, select the check-box for an instance under *Design Object/Monitor Name.*
2. Right-click on the instance and from the contextual menu, select *Send To* > *HotBitAnalysis Stats,* as shown in the figure below.

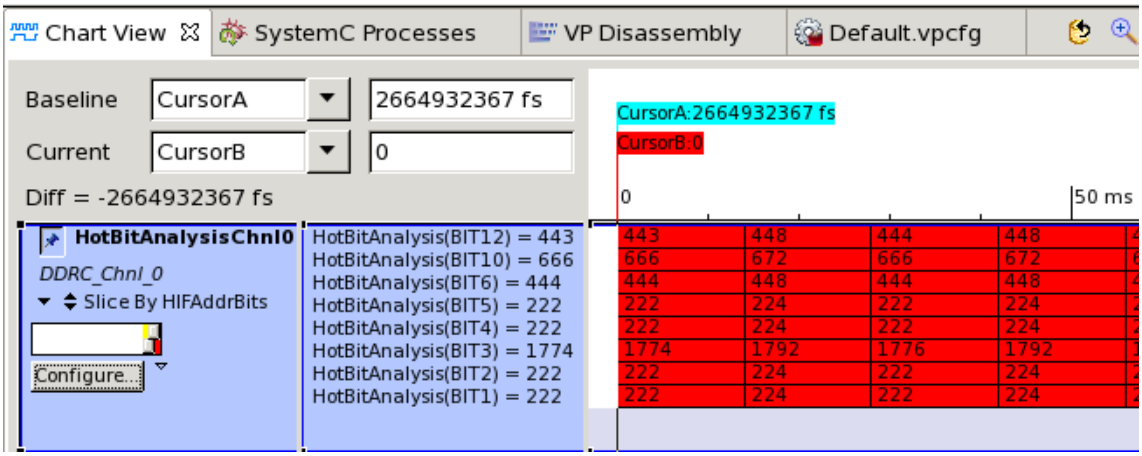
Figure 5-12 Hot Bit Analysis View



The Chart View for the selected instance opens up.

3 In the Chart View, slice the data by `HIFAddrBits` as shown in the figure below.

Figure 5-13 Chart View Showing HIFAddrBits



The view shows the number of times the bit has transitioned from `0` to `1` or from `1` to `0`. Any bit that is not transitioning at all, is not shown in the view and the number of transitions for this particular bit is assumed to be `0`.

5.7 Page Status Analysis

The Memory Page Status Analysis metric represents the data at memory interface. It gives details about `rd` and `wr` page hit and page miss.

- Page Hit: The page is already open, and `rd/wr` of same page is scheduled.
- Page Empty: The page is closed and needs to be opened before reading/writing.

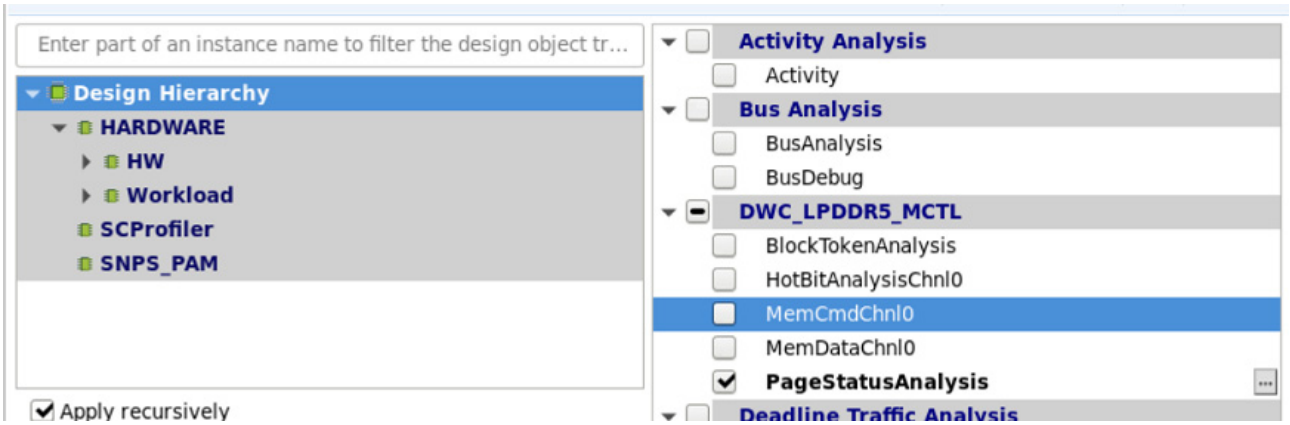
- Page Miss: The bank is open on a different page, and the controller needs to wait for the old page to be closed and the new row opened before the transaction can occur.

This metric can be viewed by sending the results to the Page Status Analysis.

To send results for page status analysis:

1 From the Results tab of the VP Explorer menu bar, select the check-box for an instance under *Design Object/Monitor Name*. Right-click on the instance, from the contextual menu, select *Send To* > *PageStatusAnalysis Stats*, as shown in the figure below.

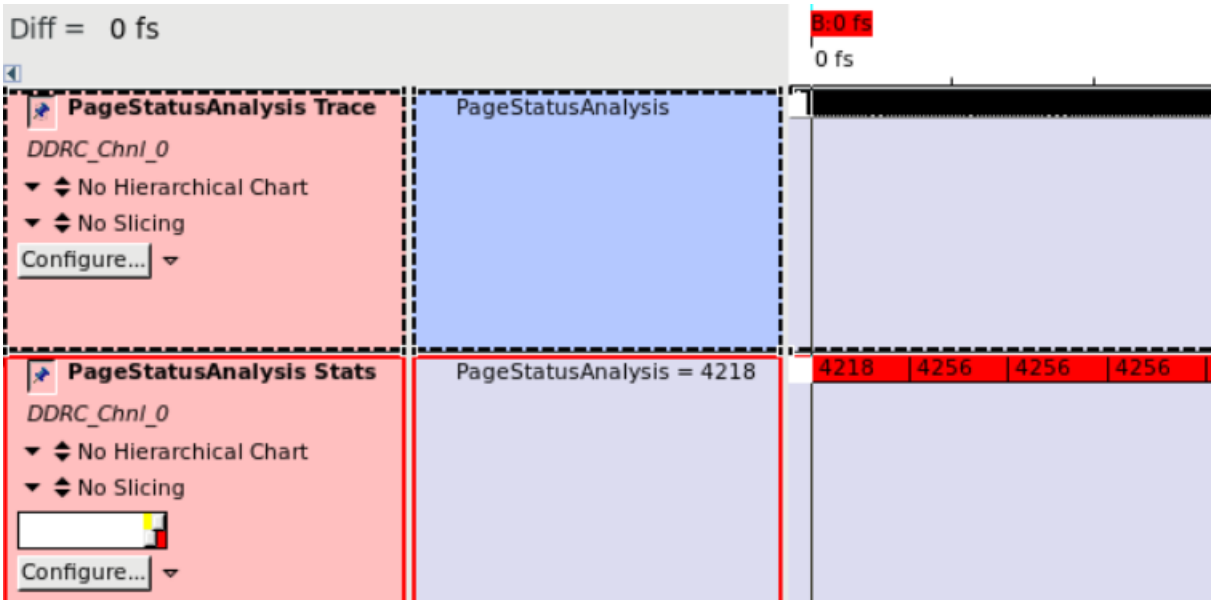
Figure 5-14 Page Status Analysis Statistics



The Chart view opens up for the selected instance.

2 Once you have the analysis database in Chart view, it reports the total number of times for page miss and page hit.

Figure 5-15 Chart View with Analysis Database

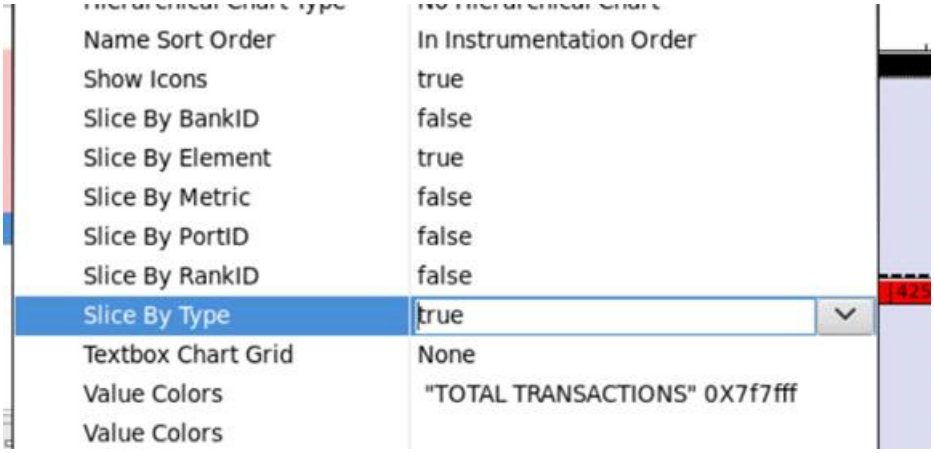


3 To find out more information about the commands, click the *Configure* button in the Chart View.

4 The Configure View/Filters dialog box pops up, where you can select the slicing of analysis data as required. The slicing operation further breaks the metric as per the available views. For this:

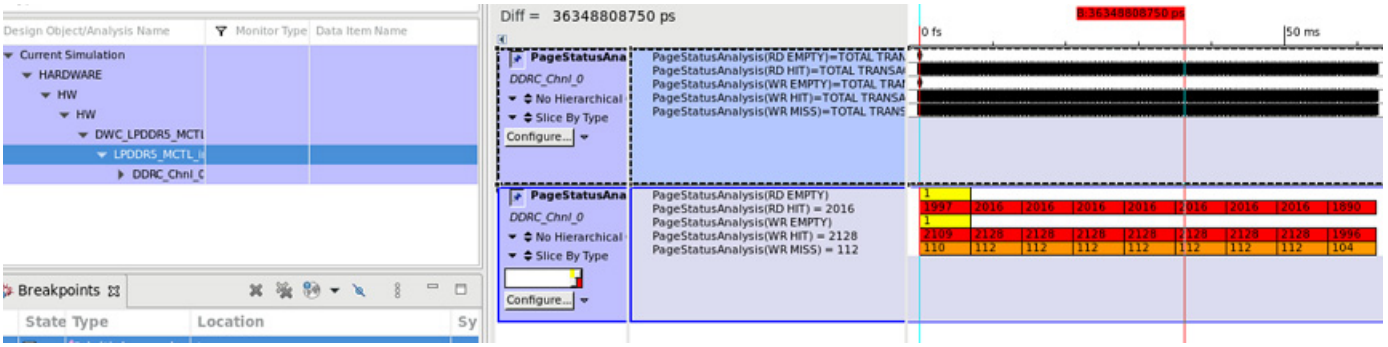
1. In the Configure View/Filters dialog box, select the value for *Slice by Type* as `true`, as shown below.

Figure 5-16 Configure View /Filters Dialog Box



2. The view splits to show the details about page hit and page miss, as shown in figure below:

Figure 5-17 Chart View for Slicing Based on Type



3. The analysis metric can be split according to following views:

- Type: Shows the various page hit and page miss status. These are `RD Empty`, `RD Page Hit`, `RD Page Miss`, `WR Empty`, `WR Page Hit`, `WR Page Miss`.
- Rank ID: Shows the page stat details as per the memory ranks.
- Bank ID: Shows the page stat details as per the memory banks.
- Port ID: Shows the page stat details as per the ports from which the corresponding AXI transaction is received.

  You can select any combination of these views to get a complex view. For example, if you select *Slice By Type* and *Slice By PortId* in Configure View/Filters dialog box. You can find the number of read and write page hits and page miss on port `AXI[2]` of the memory controller.

5.8 AutoPrecharge Analysis

The AutoPrecharge Analysis metric represents the precharge information issued for a required command. It gives details about `rd` and `wr` auto precharges.

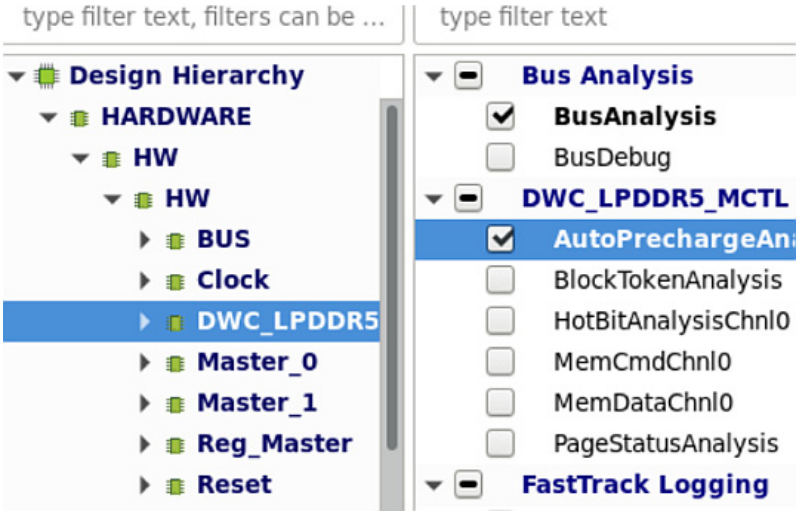
- Read AutoPrecharge: Read command is issued and the page is not already opened. So, precharge for read is issued.
- Write AutoPrecharge: Write command is issued and the page is not already opened. So, precharge for write is issued.

This metric can be viewed by sending the results to the AutoPrecharge Analysis.

To send results for AutoPrecharge Analysis:

1 From the Results tab of the VP Explorer menu bar, select the check-box for an instance under *Design Object/Monitor Name*. Right-click on the instance, from the contextual menu, select *Send To* > *AutoPrechargeAnalysis Stats*, as shown in the figure below.

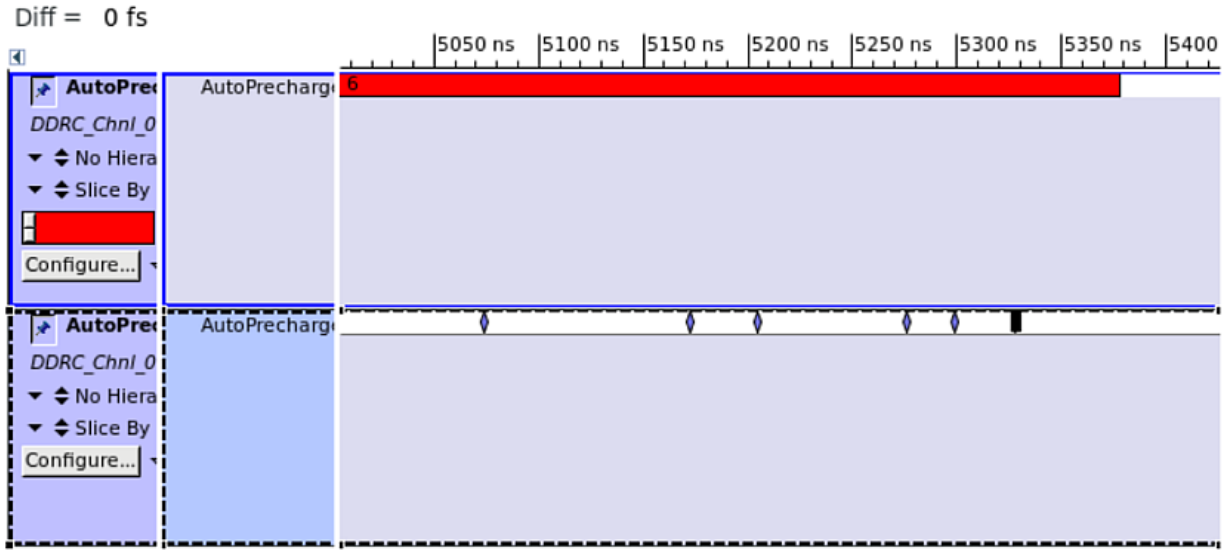
Figure 5-18 AutoPrecharge Statistics



The Chart view opens up for the selected instance.

2 Once you have the analysis database in Chart view, it reports the total number of times AutoPrecharge happened for read and write.

Figure 5-19 Chart View for AutoPrecharge

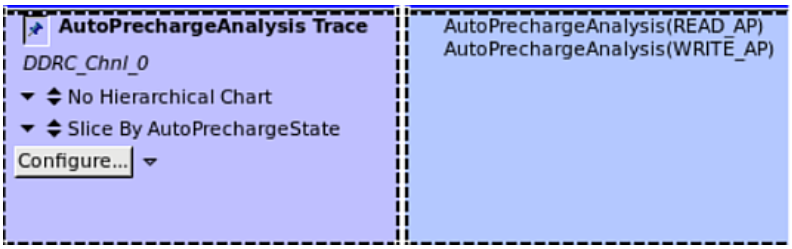


3 To find out more information about the commands, click the *Configure* button in the Chart View.

4 The Configure View/Filters dialog box pops up, where you can select the slicing of analysis data as required. The slicing operation further breaks the metric as per the available views. For this:

1. In the Configure View/Filters dialog box, select the value for *Slice* by *AutoPrechargeState* as `true`, as shown below.

Figure 5-20 Configure View/Filters Dialog Box



2. The view splits to show the details about read AutoPrecharge and write AutoPrecharge, as shown in figure below:

Figure 5-21 Chart View for Read and Write AutoPrecharge

Chapter 6
Creating Starting Point Platform and Generating DDR Targeted Workloads

This chapter describes the design creation flow for `LPDDR5_MCTL` model. This flow is useful to create an initial system with the `LPDDR5_MCTL` model and assumes that only a particular set of IPs demonstrates the various design considerations. Only a limited number of IPs can be connected to the `LPDDR5_MCTL` model or can be applied in the use cases of the model.

- Specifying Parameters
- Creating a Design
- Workload Description

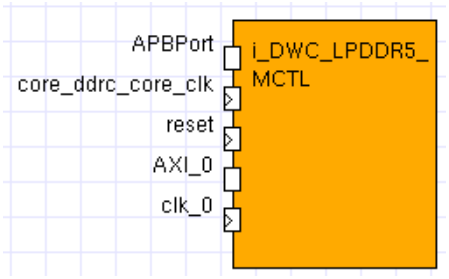
6.1 Specifying Parameters

This section explains how you can specify parameters to a `LPDDR5_MCTL` instance.

To specify the parameters to a LPDDR5_MCTL block instance:

1 From the Definition list in Platform Creator, instantiate `LPDDR5_MCTL` instance of the `LPDDR5_MCTL` model, as shown below.

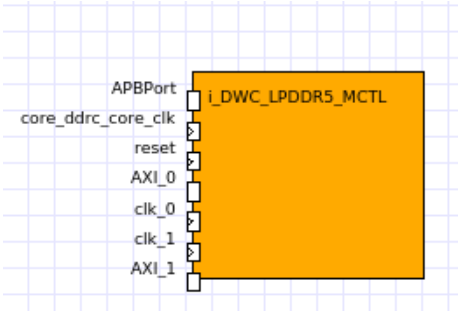
Figure 6-1 LPDDR5_MCTL Instance



2 Now you can either specify the parameters in Platform Creator or import configurations in Platform Creator from `coreConsultant` batch file or `CoreConsultant` trace file. For specific details on the import flow, see "Importing Configuration from CoreConsultant Trace File" on page 49.

Once the model configuration is done, the model represents the correct number of ports as per `/MultiPort/number_of_host_ports` parameter, as shown below.

Figure 6-2 LPDDR5_MCTL Ports



3 You can also specify the port parameters by selecting the ports and editing these parameters in the Parameter Editor tab.

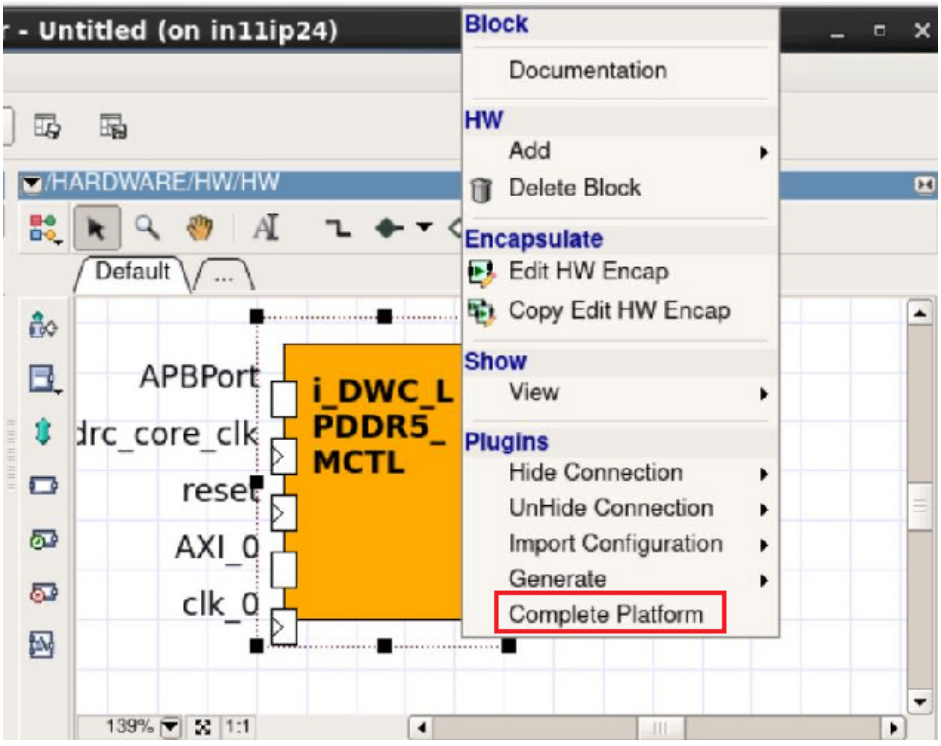
6.2 Creating a Design

Once the `LPDDR5_MCTL` instance is configured correctly, you can create a complete system around this block. In this example, VPUs with custom tasks and SBLTLM2FT bus instances are used to complete the system.

To complete a design using LPDDR5_MCTL:

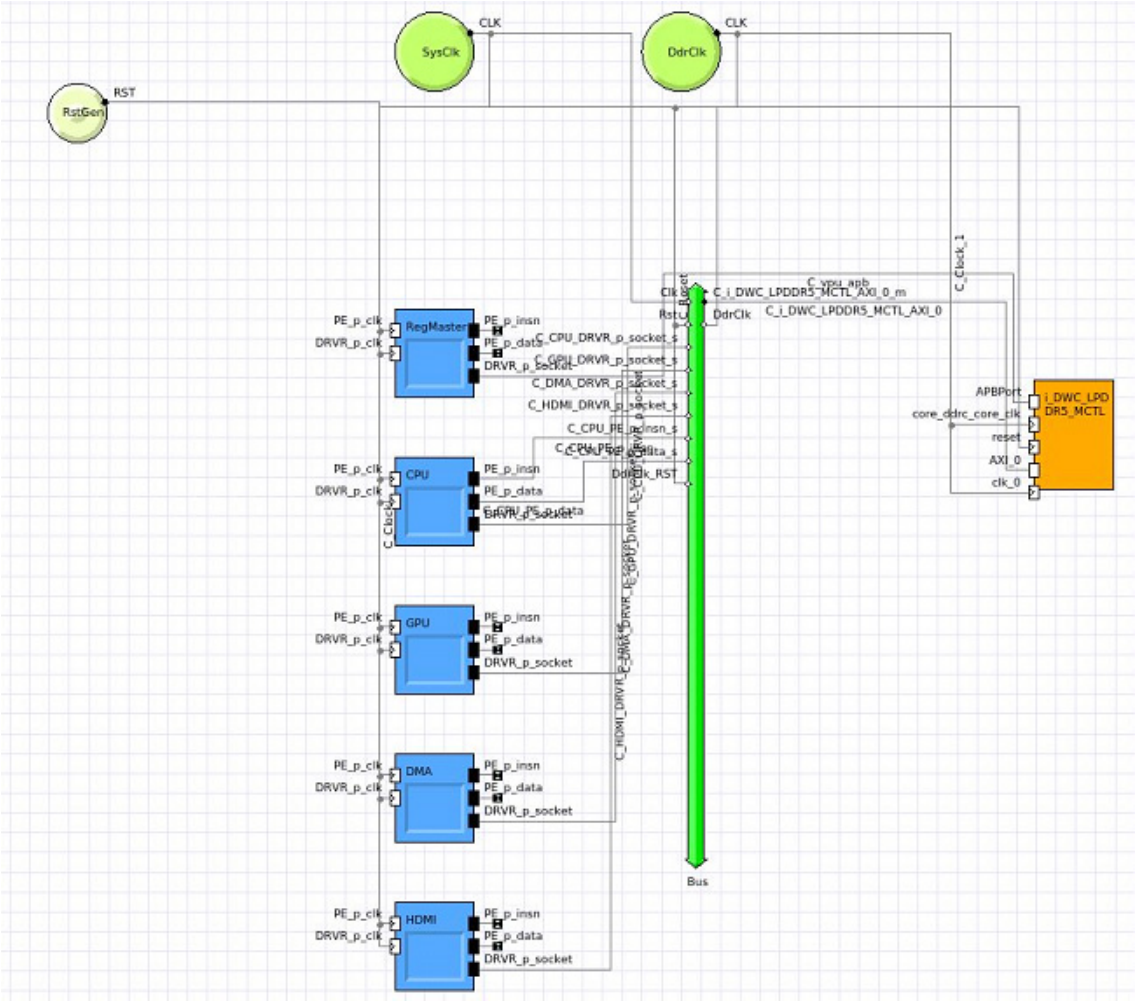
1 Right-click on `LPDDR5_MCTL` instance and select *Complete Platform*, as shown below.

Figure 6-3 LPDDR5_MCTL Block Configuration



A platform gets generated with five VPUs and one instance of SBLTLM2FT bus, connected to selected `LPDDR5_MCTL` instance, as shown in the figure below.

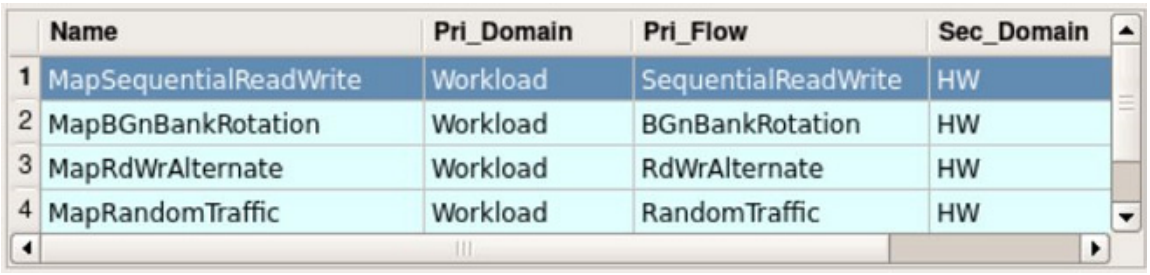
Figure 6-4 Generated Platform



6.2.1 Platform Description

- There are four VPUs (traffic initiators) which are mapped to four different workloads and one VPU that acts as Register Master. If you do not wish to do any register programming and rely on the register settings as parameters, ignore this VPU, else a `STL` task might be created and mapped to this VPU.
- All the VPUs and `SBLTLM2_FT` bus is configured with same data width as `port_data_width` of DDR controller.
- Clock is configured based on the selected Memory speed-bin and frequency ratio.
- Four different workloads are generated, as shown below. All these workloads are mapped to different VPUs.

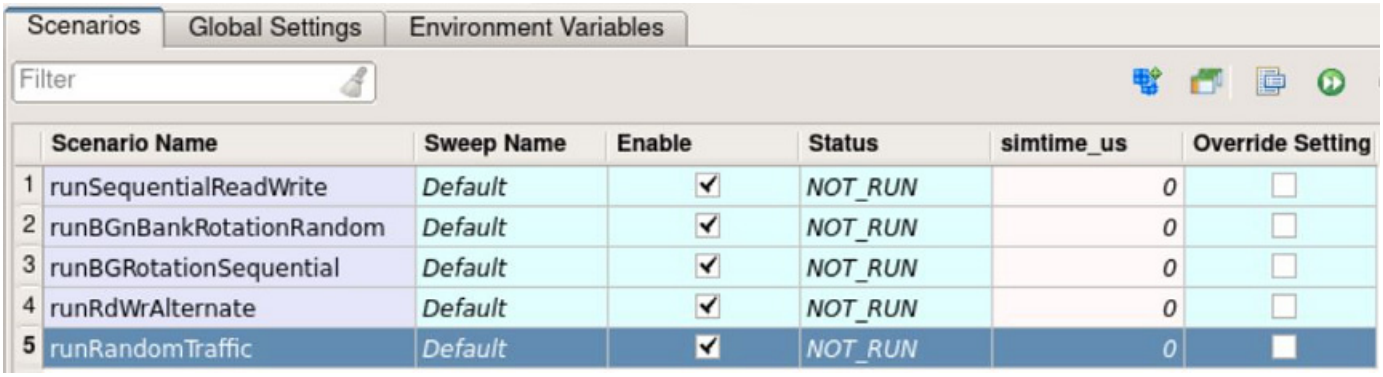
Figure 6-5 Workloads



	Name	Pri_Domain	Pri_Flow	Sec_Domain
1	MapSequentialReadWrite	Workload	SequentialReadWrite	HW
2	MapBGnBankRotation	Workload	BGnBankRotation	HW
3	MapRdWrAlternate	Workload	RdWrAlternate	HW
4	MapRandomTraffic	Workload	RandomTraffic	HW

- There are multiple scenarios readily available and distinct mapping is selected in each of the scenario, as shown in the figure below.

Figure 6-6 Mapping for Various Scenarios



	Scenario Name	Sweep Name	Enable	Status	simtime_us	Override Setting
1	runSequentialReadWrite	Default	☑	NOT_RUN	0	☐
2	runBGnBankRotationRandom	Default	☑	NOT_RUN	0	☐
3	runBGRotationSequential	Default	☑	NOT_RUN	0	☐
4	runRdWrAlternate	Default	☑	NOT_RUN	0	☐
5	runRandomTraffic	Default	☑	NOT_RUN	0	☐

6.2.2 Specifying Reset Settings

It is important to correctly set the reset value of the controller if register programming is done by external VPU using `STL` file, else it may be set as `0`. All register programming values are considered by the model only after the positive edge of reset generator. Hence, it is important that the reset port of the `LPDDR5_MCTL` instance gets a reset positive edge after all the register programing is done.

Note If any initiator in the system initiates traffic on the `LPDDR5_MCTL` instance without the positive edge of reset, the simulation will stop with an error message.

Hence, set the `start_time` parameter to `0` and duration to a value high enough so that programing of all the registers is already complete.

6.3 Workload Description

The workload models are generated and also the platform configuration is tailored to the Memory Block and its configuration. For example, taking into account the memory type, speedbin, number of ports, address mapping parameters and so on, to generate the right platform (ports and memory map set up correctly, DDR clock matching speedbin and frequency ratio) as well as workload models configured for the current Memory Block configuration (throughput/deadlines matching the expected bandwidth of the memory, memory testing like bank rotations matching the address mapping and so on).

The following workloads are provided with this utility, and these can be used for structural memory testing.

- `SequentialReadWrite` - This workload has two tasks Read, that generates around 1000 read transactions, followed by writes that generate around 1000 write transactions, with linear addresses. This workload may cater as the starting point to check the maximum throughput driven by DDR as in ideal cases. Traffic results into page-hits.Task used here are of `traffic_type` as Stochastic.
- `BGnBankRotation` - This workload generates AXI addresses, randomly or sequentially, by rotating Bank or BG bits depending upon address mapping applied in DDR controller. This workload can serve the use-cases where you want to traverse all the banks and achieve the benefits of low latencies among different banks.

  The workload uses task of type `SdramTestTraffic`, which is a custom DDR task and provide multiple parameters to generate the AXI transactions, based on DDR configuration.

  For this particular workload, `rotation_mode` in task can be set to `BGnBankRotation` or `BGRotation`. All parameters details of custom DDR task can be seen by hovering on that parameter.
- `RdWrAlternate` - This workload generates AXI transactions with alternate read and write. It uses task of type `SdramTestTraffic`, which is a custom DDR task and provides multiple parameters to generate the AXI transactions based on DDR configuration.

  For this particular workload, the parameters `readwrite_mix`/mode is set as deterministic and ratio as 0.5 to ensure that there is a read followed by write. Parameter throughput is also set as 12.8 GBps which restricts the traffic at this particular rate, else reads and writes would flow parallel.
- `RandomTraffic` - This workload generates pure random traffic such that every next address is not linear with previous address and order of reads/writes is also random.

  For this particular workload, the parameters branch/mode is set as randomized and ratio as one to ensure random addresses generation.

Chapter 7
Example Platform for LPDDR5 Block

This chapter describes the LPDDR5ExamplePlatform for the LPDDR5 block.

- Software Running on Platform
- Directory Structure
- Opening Example Platform
- Example Platform-LPDDR5ExamplePlatform

7.1 Software Running on Platform

The software floats the actual reads and writes of different access and burst sizes to each of the memory ranks and banks of SDRAM inside LPDDR5. Thus tests the basic functionality of the LPDDR5.

7.2 Directory Structure

Each example platform has following directories and files to create the platform.

- Scripts/directory contain scripts to create and export example platform.
- Scripts/1_open_libs.tcl script opens the required PSPs.
- Scripts/2_create_platform.tcl script instantiates and configure the blocks, and connects them together.
- Scripts/3_generate_framework.tcl script generates and configures the BLWizard framework.
- Scripts/4_export_system.tclscript exports the platform into a subdirectory.
- Scripts/createsys.tclscript calls the first three scripts.
- Scripts/exportsys.tcl script calls the remaining scripts.
- Software/directory contains the software code which is run on the example platform.
- Software/Reg.stl file contains the software that configures the LPDDR5 registers. This file is left blank intentionally because LPDDR5 auto programs itself. Though you can also program the controller.
- Software/Access_Master_0.stl file contains the software that floats the read/write transactions to different memory ranks and banks.

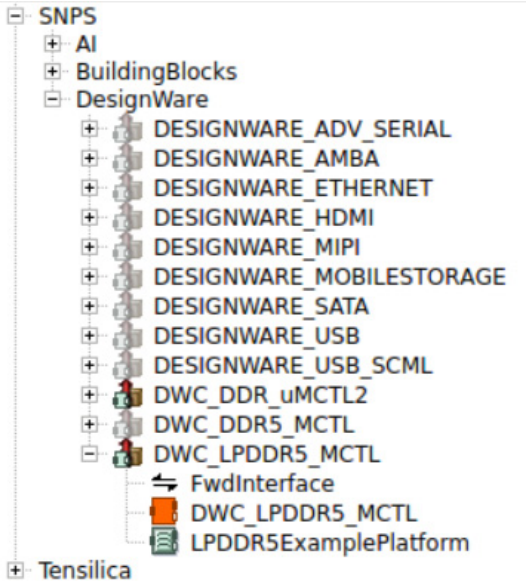
You can find the above listed files at:

- For LPDDR5ExamplePlatform, at installDir/IP/DWC_LPDDR5_MCTL/Examples/LPDDR5ExamplePlatform/Scripts.

7.3 Opening Example Platform

The DWC_LPDDR5_MCTL library can be opened to see the example platforms as shown in the figure below.

Figure 7-1 DWC_LPDDR5_MCTL Library

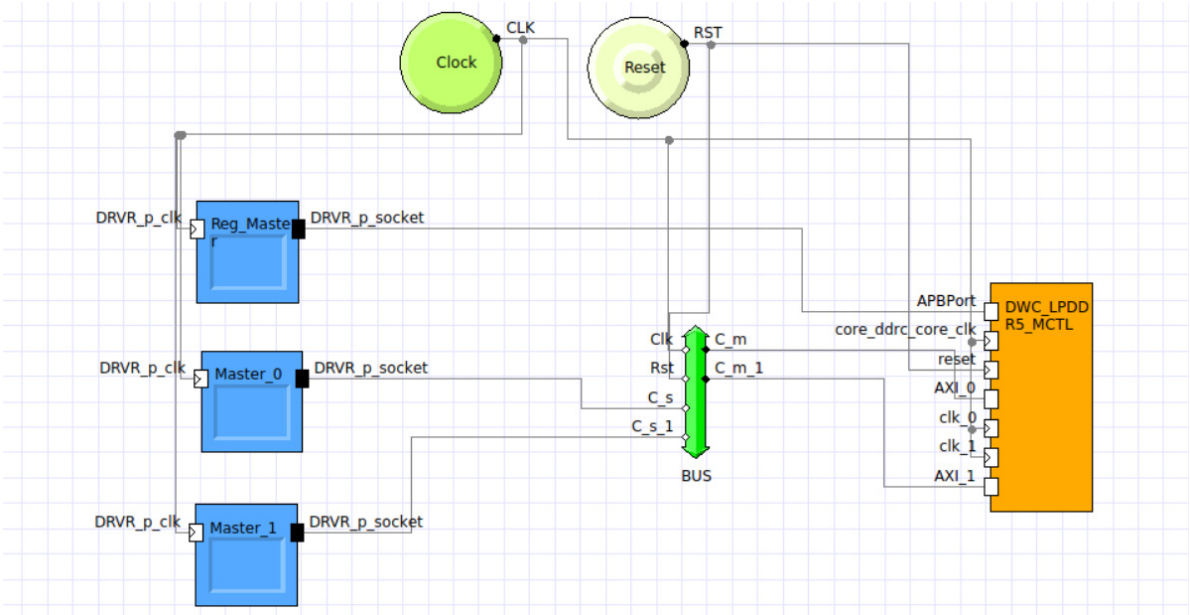


Double click on `LPDDR5ExamplePlatform` to open the platform.

7.4 Example Platform-LPDDR5ExamplePlatform

The figure below shows the `LPDDR5ExamplePlatform`.

Figure 7-2 LPDDR5ExamplePlatform



7.4.1 Configuration for LPDDR5ExamplePlatform

The figure below shows the usage of the default values for `LPDDR5` model.

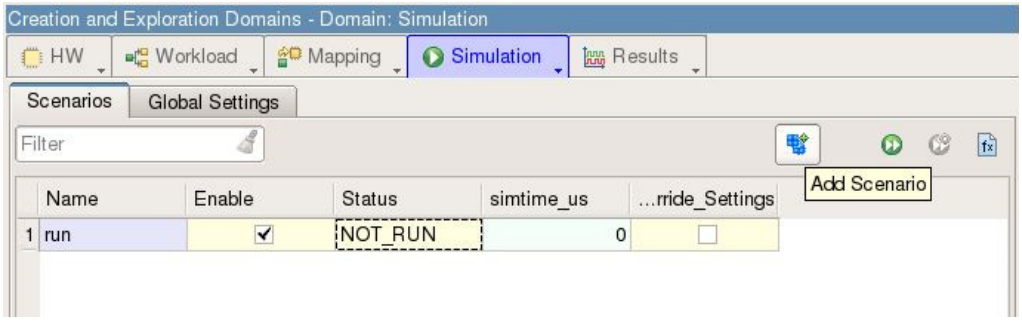
Figure 7-3 Parameter Editor Window

Block properties				
Name	DWC_LPDDR5_MCTL			
Extra properties				
Product				
DDR_Controller_Product	LPDDR5_MCTLP	Default	Visible	Until Simulation Start
DDRC				
RTL_version	1.10a-lca00	Default	Visible	Until Simulation Start
memory_type	LPDDR5	Default	Visible	Until Simulation Start
programmable_frequency_ratio	0	Default	Visible	Until Simulation Start
frequency_ratio	4	Default	Visible	Until Simulation Start
burst_length_supported	BL16	Default	Visible	Until Simulation Start
enable_dual_HIF	false	Default	Visible	Until Simulation Start
memory_data_width	16	Default	Visible	Until Simulation Start
number_of_ranks_supported	1	Default	Visible	Until Simulation Start
CAM_depth	32	Default	Visible	Until Simulation Start
enable_variable_priority_read_write_feature	false	Default	Visible	Until Simulation Start
Enhanced_CAM_pointer_mechanism	false	Default	Visible	Until Simulation Start
Enhanced_RD_WR_switching_mechanism	false	Default	Visible	Until Simulation Start
Enhanced_RD_WR_switching_policy_selectable	false	Default	Visible	Until Simulation Start
refresh_timeout	10000	Default	Visible	Until Simulation Start
enable_link_ecc	false	Default	Visible	Until Simulation Start
MultiPort				
number_of_host_ports	2	Default	Visible	Until Simulation Start
event_ports	0	Default	Visible	Until Simulation Start
application_address_width	32	Default	Visible	Until Simulation Start
application_id_width	8	Default	Visible	Until Simulation Start
application_burst_length_width	4	Default	Visible	Until Simulation Start
enable_XPI_read_address_input_retime	false	Default	Visible	Until Simulation Start
enable_external_port_priorities	false	Default	Visible	Until Simulation Start
enable_port_arbiter_page_match_feature	true	Default	Visible	Until Simulation Start
ReliabilityFeatures				
phy_static_read_latency	11	Default	Visible	Until Simulation Start
phy_static_write_latency	7	Default	Visible	Until Simulation Start
phy_static_read_latency_chnl1	13	Default	Visible	Until Simulation Start
phy_static_write_latency_chnl1	2	Default	Visible	Until Simulation Start
enable_register_log	false	Default	Visible	Until Simulation Start
enable_performance_log	false	Default	Visible	Until Simulation Start
Programming				
Generator parameters				

7.4.2 Running the Example Platform in Platform Architect

To build any platform, click on the *Add Scenario* tab in the Simulation Domain, as shown in the figure below.

Figure 7-4 Simulation Domain



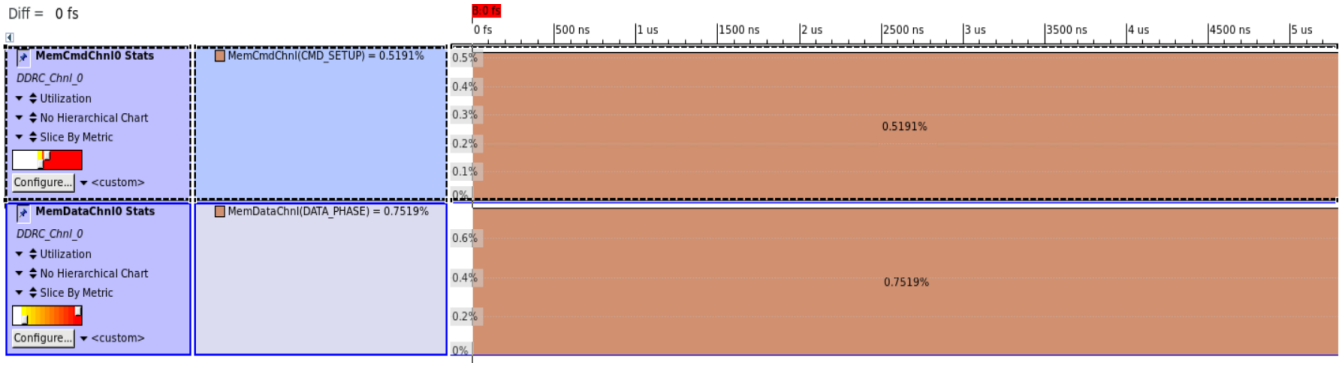
Now, right-click on the scenario and from the context menu, select *Run Selected Default*. This executes all the steps from exporting, building to running the platform.

To enable traces, from *Global Settings*, select the required traces before you run the scenario.

7.4.3 Analysis View

The `LPDDR5` is instrumented for the analysis. To enable Synopsys analysis on `LPDDR5`, see "Analysis View" on page 69. For more information on enabling analysis and viewing the results, see the *VP Explorer User Guide*, ["References" on page 10].

Figure 7-5 Analysis View for LPDDR5ExamplePlatform



Chapter 8
RTL CoSimulation for LPDDR5 Block

This chapter describes how to run RTL CoSimulation for `LPDDR5` block. There is a package which has top level script `runRTLCosimFlow.sh` that can be used to import the RTL design and create the platform.

The `ddr_cosim` package is available in the install directory at the below location:

```
<installDir>/IP/DWC_LPDDR5_MCTL/Internal/script/ddr_cosim
```

Steps to run the package:

1 RTL must be generated in *<test_dir>* to run this package. If RTL is not present, do the following:

```
source <test_dir>/coreConsultant_env_setup.csh
coreConsultant -f <test_dir/configuration_file> -shell
```

2 In test directory, the below steps generate the required XML file, which can be opened with Platform Architect. You can modify the default programming file `register_programming.stl` and traffic file `dtata.stl` provided with the package.

```
cd <test_dir>
source <pkg_dir>/setup.csh
<pkg_dir>/cleanup.csh
<pkg_dir>/runRTLCosimFlow.sh
pct RTLProject.xml &
```

Index

W