SYNOPSYS®

DesignWare ® Cores LPDDR5/4/4X PHY CTB and Verification

Application Note

DWC LPDDR5/4/4X PHY
CTB Version: C-2021-10

CTB Version: C-2021-10
Document Version: 1.03a
October 13, 2021

Copyright Notice and Proprietary Information

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
www.synopsys.com

Contents

Revision History

Note

- Links and references to section, table, figure, and page numbers in this table are only assured to be valid for the version in which the change is made
- In some instances, documentation-only updates occur. The DesignWare IP product information (https://www.synopsys.com/designware-ip.html) has the latest documentation

Document Version	Date	Description
1.03a	October 13, 2021	Updated: ■ "Preface" on page 7 ■ "Unsupported Features" on page 17 ■ "Test Case Description" on page 23 ■ "Configurable Test Options" on page 28
1.02a	June 21, 2021	Updated: ■ "Overview" on page 11 ■ "Supported Features" on page 15 ■ "Test Case Description" on page 23 ■ "Configurable Test Options" on page 28 ■ "demo ate" on page 51 ■ "demo ate" on page 57 ■ "Create Timing Disable List for SDF GateSim" on page 74
1.01a	May 10, 2021	Updated: ■ Table "Configurable Test Options" on page 28
1.00a	April 23, 2021	Initial release.

Preface

This databook describes the LPDDR5/4/4X CTB and Verification Application Note.

LPDDR5/4/4X CTB and Verification Application Note Organization

The chapters of this Application Note are organized as follows:

- Chapter 1, “Overview”
- Chapter 2, “Customer Testbench”
- Chapter 3, “Behavioral Verification”
- Chapter 4, “GateSim”

Web Resources

- DesignWare IP product information: https://www.synopsys.com/designware-ip.html
- Your custom DesignWare IP page: https://www.synopsys.com/dw/mydesignware.php
- Documentation through SolvNetPlus: https://solvnetplus.synopsys.com (Synopsys password required)
- Synopsys Common Licensing (SCL): https://www.synopsys.com/keys

STAR on the Web (SotW)

- You must review all Stars on the Web (SotWs) associated with your product. SotWs are considered a part of the Synopsys documentation suite, and show critical information related to your product. To review product SotWs, refer to the DesignWare IP product information: https://www.synopsys.com/designware-ip.html

Reference Documentation

- *DesignWare Cores LPDDR54 PHY Databook Utility Block*, Synopsys, Inc
- *DesignWare Cores LPDDR54 PHY Databook*, Synopsys, Inc
- *DesignWare Cores LPDDR54 PHY Training Firmware Application Note*, Synopsys, Inc
- *DesignWare Cores LPDDR54 PHY ATE Firmware Application Note*, Synopsys, Inc
- *DesignWare Cores LPDDR54 PHY Quickboot Firmware Application Note*, Synopsys, Inc

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.

Customer Support

To obtain support for your product, prepare the required files and contact the support center using one of the methods described:

- Prepare the following debug information, if applicable:
  - For environment set-up problems or failures with configuration, simulation, or synthesis that occur within coreConsultant or coreAssembler, select the following menu:

    File > Build Debug Tar-file

    Check all the boxes in the dialog box that apply to your issue. This option gathers all the Synopsys product data needed to begin debugging an issue and writes it to the <*core tool startup directory*>/debug.tar.gz file.
  - For simulation issues outside of coreConsultant or coreAssembler:
    - Create a waveforms file (such as VPD or VCD).
    - Identify the hierarchy path to the DesignWare instance.
    - Identify the timestamp of any signals or locations in the waveforms that are not understood.
- *For the fastest response*, enter a case through SolvNetPlus:

a. https://solvnetplus.synopsys.com

> **Note** SolvNetPlus does not support Internet Explorer. Use a supported browser such as Microsoft Edge, Google Chrome, Mozilla Firefox, or Apple Safari.

b. Click the **Cases** menu and then click **Create a New Case** (below the list of cases).

c. Complete the mandatory fields that are marked with an asterisk and click **Save**.

Make sure to include the following:

- **Product L1:** DesignWare Cores
- **Product L2:** LPDDR54

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

For more information about general usage information, refer to the following article in SolvNetPlus:

https://solvnetplus.synopsys.com/s/article/SolvNetPlus-Usage-Help-Resources

- Or, send an e-mail message to support_center@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 L1 and Product L2 names, and Version number in your e-mail so 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.
- Or, telephone your local support center:
  - North America:

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

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

1 Overview

Note This application note pertains to LPDDR5/4/4X CTB Version: C-2021.06. There may be variations in operation, interface, and usage procedure, depending on the CTB version.

This application note has 3 sections demonstrating behavioral and gate level simulation, so that user can take it as a starting point to create SoC specific simulations and learning how the PHY works. Each section is explained in more detail in dedicated chapters later in this document.

- "Customer Testbench" on page 13
  - Demonstration of some LPDDR PHY capabilities.
- "Behavioral Verification" on page 61
  - Guidance for running behavioral verification.
- "GateSim" on page 69
  - Guidance for Gate level simulations of the digital sections of the PHY.

2 Customer Testbench

The Customer TestBench (CTB) environment provides a demonstration of external stimuli to simulate some of the LPDDR PHY capabilities. Customers can see how the PHY works and use the simulation environment as a starting point to create SoC specific simulations. The CTB is meant as a demonstration and not a verification example.

2.1 Introduction

The Customer Testbench (CTB) enables LPDDRn PHY IP customers to:

- Check basic DDR IP functionality in the stand-alone environment
- Learn about DDR IP functionality in the dynamic working scenes.
- Validate the correctness of IP hardware and phyinit/firmware configuration prior to using it at the SoC level.

The Customer Testbench environment is intended to reduce the ramp-up time, making DDR IP integration process easier and faster.

2.2 Supported Features

The following features are supported by this version of the Customer Testbench (CTB):

- PHY initialization and control
- Supports skip-training mode
- Supports devinit
- LPDDR4 support
- LPDDR4X support
- LPDDR5 support
- Channel support
  - Support for 1 or 2 independent LPDDR4/4X/5 channels
  - Supports 16-bit data path widths per channel
  - Total up to 32 bits data path using two channels
- Different memory rank number support
  - Supports 1, 2 memory ranks for LPDDR4.
  - Supports 1, 2 memory ranks for LPDDR4X.
  - Supports 1, 2 memory ranks for LPDDR5.
- Fixed SDRAM device type
  - X16 SDRAM device type for LPDDR4
  - X16 SDRAM device type for LPDDR4X
  - X16 SDRAM device type for LPDDR5
- Supports different SDRAM speed grades
  - LPDDR4/ LPDDR4X supported SDRAM Clock(ck) frequency: 266 MHz, 333 MHz, 400 MHz, 533 MHz, 667 MHz, 800 MHz, 933 MHz, 1066 MHz, 1200 MHz, 1333 MHz, 1466 MHz, 1600 MHz, 1866 MHz, or 2133 MHz; Default frequency setting for LPDDR4 tests is 800 MHz.
  - LPDDR5 supported SDRAM Clock(ck) frequency: 7 MHz, 13 MHz, 67 MHz, 133 MHz, 200 MHz, 267 MHz, 344 MHz, 400 MHz, 467 MHz, 533 MHz, 600 MHz, 688 MHz, 750 MHz, or 800 MHz; Default frequency setting for LPDDR5 tests is 800 MHz.
- Frequency ratio
  - CK:WCK = 1:4/1:2, DFI:CK=1:1 for LPDDR5
  - DFI 1:2 and 1:4 mode support for LPDDR4/LPDDR4X
- Pll bypass mode
- Different PState(0,1) for frequency change feature
- DFI low power mode
- Synopsys DFI/DDR memory VIP support

- APB interface to configuration registers
- TDR interface to configuration registers
- DM/DBI
- UPF power aware simulation
- ATE FW
- LPDDR4 Training FW
- LPDDR5 Training FW
- LPDDR4 quickboot
- LPDDR5 quickboot

Table 2-1 Tested Configurations

Num of Channel	DWC_DDRPHY_LPDDR5_ENABLED	DWC_DDRPHY_NUM_DBYTESPER_CHANNEL	DWC_DDRPHY_DBYTE_DMI_ENABLED	Supported Rank Define	Define File CONFIG
1	defined	2	defined	DWC_DDRPHY_NUM_RANKS_1	lp54cs1dq18ch1
1	defined	2	undefined	DWC_DDRPHY_NUM_RANKS_1	lp54cs1dq16ch1
1	defined	2	defined	DWC_DDRPHY_NUM_RANKS_2	lp54cs2dq18ch1
1	defined	2	undefined	DWC_DDRPHY_NUM_RANKS_2	lp54cs2dq16ch1
2	undefined	2	defined	DWC_DDRPHY_NUM_RANKS_1	lp4cs1dq18ch2
2	defined	2	defined	DWC_DDRPHY_NUM_RANKS_1	lp54cs1dq18ch2
2	defined	2	undefined	DWC_DDRPHY_NUM_RANKS_1	lp54cs1dq16ch2
2	defined	2	defined	DWC_DDRPHY_NUM_RANKS_2	lp54cs2dq18ch2
2	defined	2	undefined	DWC_DDRPHY_NUM_RANKS_2	lp54cs2dq16ch2

2.3 Unsupported Features

The following features are not supported in this version of Customer Testbench (CTB) and may be supported in subsequent releases (this list should not be considered a definitive or exhaustive list of unsupported features):

- 2D training
- DFI Controller Update and more DFI Sideband interfaces
- It is recommended that Quickboot be run >= 3200M data rate

Note

1. This list should not be considered a definitive or exhaustive list of unsupported features.
2. Some of the features supported in coreConsultant may not be supported by CTB.

2.4 Installation

2.4.1 Installation Procedure

The following procedure outlines the Customer Testbench (CTB) installation process:

1. Configuring the PHY:

   The configuration of the PHY is based on the package components:

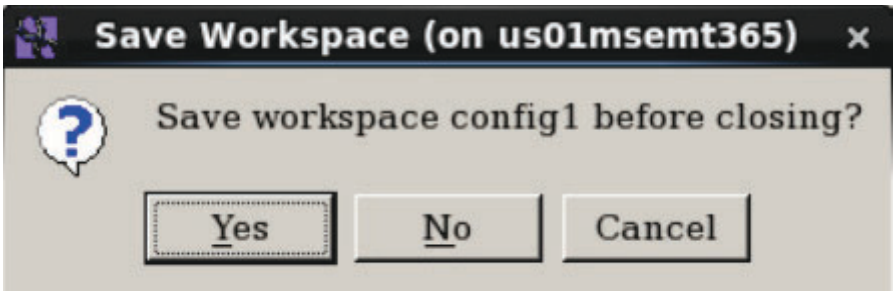
   - If the PHY release package includes an installed_coreKit component, then user must install the Synopsys coreConsultant tool and configure the PHY using this tool. For more details, see "Configure the PHY using the coreConsultant tool" in Designware Cores LPDDR5/4/4X PHY Implementation Guide databook.

     After configuring the PHY using the coreConsultant tool and doing an export, the CTB area from release package will be copied into <WORKSPACE>/ctb.

     Note:

     If "Setup and Run CTB" activity is available in the GUI, it is strongly recommended to run CTB in the coreConsultant GUI. See section "Simulating the Core" in the "DesignWare Cores LPDDR5/4/4X PHY coreKit User Guide" for details.

     If user expect to run CTB in <WORKSPACE>/ctb out of coreConsultant GUI, make sure the coreConsultant is closed and the workspace is saved. Click "Yes" on the button when closing the coreConsultant tool.



- If the PHY release package includes pub+macro components, then configure the PHY as described in the "DDRPHY Top Level" section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook.

  Note:

  a. Contact "Customer Support" to ensure you choose the right configuration for build.

  b. CTB area will be resident in release package <synopsys/<ip_name>/Latest/ctb >

2. Set VCS, Verdi tool versions to the compatible versions listed in the Customer Testbench(CTB) readme file in the IP package.
3. Download and Install Synopsys Verification IP(VIP) of SDRAM memory.

   The Customer Testbench (CTB) uses Synopsys SDRAM memory VIP. If you do not have supported versions of Synopsys memory VIP installed, you must install them by performing the following steps:

   a. Select "VC VIP Memory Portfolio"or "VC VIP Library" from the SolvNet Download Center, at: https://solvnet.synopsys.com/DownloadCenter/dc/product.jsp

   b. Select "VC VIP Memory Portfolio" or "VC VIP Library" version specified in CTB readme file and download the .run files.

   c. Install the Synopsys memory VIP following the instructions in Synopsys_VC_VIP_Memory_Portfolio_README.txt.

      i. Make sure $DESIGNWARE_HOME environment variable exists and points to the Synopsys Memory VIP installation directory.

      ii. Execute the *.run file by invoking its filename.

      iii. Make sure $CTB_VIP_HOME environment variable exists and points to a writable directory as the Synopsys memory VIP project directory specified in Synopsys_VC_VIP_Memory_Portfolio_README.txt.

      iv. Add VIP models required by CTB.

      1. DFI

      ```
      % $DESIGNWARE_HOME/bin/dw_vip_setup -path $CTB_VIP_HOME -add
      dfi_phy_svt dfi_agent_svt dfi_mc_agent_svt dfi_mc_svt
      ```

      2. LPDDR

      ```
      % $DESIGNWARE_HOME/bin/dw_vip_setup -path $CTB_VIP_HOME -add
      lpddr_agent_svt lpddr_controller_agent_svt lpddr4_env_svt
      lpddr5_env_svt lpddr_controller_agent_svt lpddr4_controller_env_svt
      lpddr5_controller_env_svt
      ```

Note

For issues with the licensing or installation of the Synopsys Verification IP (VIP) of SDRAM memory, contact your account/sales team. Additionally, a SolvNet may be filed with the following categories:

- Product: Verification IP
- Sub Product: VIP_title
- Tool Version: VIP_version
- Fill in the remaining fields according to your environment and your issue.

4. Source the boot environment to set up directory paths

Note	If user runs CTB in the coreConsutlant GUI, then this step can be skipped. Otherwise, follow this section to set up environment.

a. `% cd <WORKSPACE>/ctb/Latest/sim`

For PHY release package that does not include an installed_coreKit component, which includes pub+macro components:

`cd synopsys/<ip_name>/Latest/ctb/Latest/sim`

b. `% source bootenv`

For PHY release package that does not include an installed_coreKit component, the environment variable must be changed in bootenv before source bootenv:

`setenv CORETOOLS 0`

Note	The IP-XACT file is used in CTB simulation. For PHY release package that includes an installed_coreKit component, the IP-XACT file is resident in <WORKSPACE>/export/dwc_ddrphy_top.xml. The file is config-dependent and can be generated in the coreConsultant. If the file is already generated, the simulation will use it directly, otherwise, CTB will generate it automatically. When user change the PHY configuration using coreConsultant, make sure the IP-XACT file is re-generated with the new configuration.

2.4.2 Customize Script Interpreter

Table 2-2 lists the customer testbench scripts, including default interpreter path and customization options.

Table 2-2 Customer Testbench Scripts

Script	Default Interpreter Path	Customization Options
$CTB_HOME/sim/runtc	/usr/local/bin/bash	Can be customized by modifying the first line: #!/usr/local/bin/bash to user-defined path for bash script interpreter
$CTB_HOME/testbench/inc/gen_fw_cfg.tcl	/usr/local/bin/tclsh	Can be customized by modifying the first line: #!/usr/local/bin/tclsh to user-defined path for tcl script interpreter
$CTB_HOME/testbench/inc/ctb_utils.pl	/usr/bin/perl	Can be customized by modifying the first line: #!/usr/bin/perl to user-defined path for perl script interpreter

Table 2-2 Customer Testbench Scripts (Continued)

Script	Default Interpreter Path	Customization Options
$CTB_HOME/testbench/inc/postprocess.pl	/usr/bin/perl	Can be customized by modifying the first line: #!/usr/bin/perl to user-defined path for perl script interpreter
$CTB_HOME/testbench/inc/gen_override_c.pl	/usr/bin/perl	Can be customized by modifying the first line: #!/usr/bin/perl to user-defined path for perl script interpreter

2.5 Execution

If “Setup and Run CTB” activity is available in the GUI, it is recommended for user to run CTB in the coreConsultant GUI, then this section can be skipped.

Otherwise, if user expects to run CTB in <WORKSPACE>/ctb , or synopsys/<ip_name>/Latest/ctb/Latest/sim, or in any other writable directory as the simulation directory without CoreTool, out of coreConsultant GUI, follow this section to execute simulation.

2.5.1 Overview

Each test case run consists of two phases:

1. Generating dwc_ddrphy_phyinit_userCustom_overrideUserInput.c and firmware configuration.
2. Running the Test.

Note

1. Refer to the documentation of “PhyInit Software Application Note” for more details of dwc_ddrphy_phyinit_userCustom_overrideUserInput.c.
2. In CTB environment, dwc_ddrphy_phyinit_userCustom_overrideUserInput.c will be automatically generated by CTB script analyzing specified runtc command options, if override_c option is not specified in runtc command line.
3. This generated file of dwc_ddrphy_phyinit_userCustom_overrideUserInput.c can be overridden by specifying runtc command option of override_c=<path of your new dwc_ddrphy_phyinit_userCustom_overrideUserInput.c >

2.5.2 Using Combined Command

The following steps can be followed to run a single command to execute above 2 phases.

1. User must enter the simulation directory:
   - For PHY release package that does not include an installed_coreKit component, which PHY release package includes pub+macro components:

     User can enter into CTB sub-directory(synopsys/<ip_name>/Latest/ctb/Latest/sim) or another writable directory as the simulation directory.
   - For PHY release package that includes an installed_coreKit component, user must enter the simulation directory in the workspace created by coreConsultant with Installation Procedure Step1.(<workspace>/ctb/Latest/sim)
2. Run the following command to run the simulation and save simulation result files:

```
% runtc tc=<test_name> cfg=<cfg_name> dram=<dram_type> skip_train=<train_type>
hard_macro=<macro_type>
```

Refer to the "Configurable Test Options" on page 28" for more details of command line options.

For PHY release package that includes an installed_coreKit component, make sure the command switch <cfg_name> coincides with the generated PHY config, which can be found from the filename: <WORKSPACE>/src/dwc_ddrphy_lp*cs*dq*ch*_VDEFINES.v, the <cfg_name> must be lp*cs*dq*ch*.

The `runtc` command executes both phases of generating firmware configuration and running the test sequentially.

For example:

```
% runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333
freq_ratio0=1 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type> dump=FSDB
<=> equals to:
% runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333
freq_ratio0=1 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type> dump=FSDB -fw
% runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333
freq_ratio0=1 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type> dump=FSDB -sim
```

2.5.3 Generating Firmware Configuration

To execute only phase 1 of generating Firmware configuration and dwc_ddrphy_phyinit_userCustom_overrideUserInput.c, run the following command:

```
% runtc tc=<test_name> cfg=<cfg_name> dram=<dram_type> dimm=<dimm_type>
skip_train=<train_type> hard_macro=<macro_type> -fw
```

<macro_type> can be A/B, user must set it according to the PHY Databook.

For example, for LPDDR4 skip train and hardware configuration lp54cs1dq18ch1, run the following command:

```
% runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333
freq_ratio0=1 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type> -fw
```

The above runtc command generates the following files in the simulation directory:

- dwc_ddrphy_phyinit_userCustom_overrideUserInput.c, containing the override settings of Phyinit data structure and message block.
- **csr_defines.sv**, containing the macros for the registers name-to-address conversion
- **apb_config.sv**, containing APB configuration tasks to be invoked during DUT configuration stage
- **cfg_data.sv**, containing all the software configuration values. This file is required to synchronize the testbench behavior (for example, clock frequency) with given software configuration.

2.5.4 Running the Test

To execute only phase 2 of running the test, run the following command:

```
% runtc tc=<test_name> cfg=<cfg_name> dram=<dram_type> dimm=<dimm_type>
skip_train=<train_type> hard_macro=<macro_type> dump=<VPD or FSDB or NONE> -sim
```

For example, for LPDDR4 skip train and hardware configuration of lp54cs1dq18ch1, run the following command:

```
% runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333
freq_ratio0=1 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type> dump=FSDB -sim
```

The above command executes the test using given hardware and software configurations and writes simulation logs and waveforms into the simulation directory.

Use the "`runtc -help`" command to get help on `runtc` command options.

2.6 Test Cases

2.6.1 Overview

The test cases are organized under the <ctb/Latest/testbench/tc/demo> folder. The function of each test case corresponds to the file name:

- demo_basic.sv
- demo_lp.sv
- demo_ate.sv

2.6.2 Test Case Description

The following tests are supplied with the Customer Testbench(CTB):

2.6.2.1 Basic demo test examples

- LPDDR4, full training test

  ```
  % runtc cfg=lp54cs1dq18ch2 tc=demo_basic dram=lpddr4 skip_train=0 seqCtrl=0x121f
  freq0=2133 freq_ratio0=1 rank=1 pstates=1 dfi_mode=3 hard_macro=<macro_type>
  ```

- LPDDR4, skip training test

  ```
  % runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333 freq_ratio0=1
  rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type> -svt_dfi
  ```

- LPDDR4, quickboot test

  ```
  %runtc cfg=lp54cs1dq18ch2 tc=demo_basic dram=lpddr4 skip_train=0 seqCtrl=0x121f
  freq0=2133 freq_ratio0=1 rank=1 pstates=1 dfi_mode=3 hard_macro=<macro_type> -quickboot
  ```

- LPDDR4, jtag

  ```
  % runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333 freq_ratio0=1
  rank=1 pstates=1 dfi_mode=1 -jtag hard_macro=<macro_type> -svt_dfi
  ```

- LPDDR4, pll_bypass

```
% runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr4 skip_train=1 freq0=266 freq_ratio0=1
rank=1 pstates=1 dfi_mode=1 -pll_bypass0 hard_macro=<macro_type> -svt_dfi
```

- LPDDR4, dm

```
% runtc cfg=lp54cs1dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1 freq0=2133
freq_ratio0=1 rank=1 pstates=1 dfi_mode=3 hard_macro=<macro_type> -dm0
```

- LPDDR4, wdbi

```
% runtc cfg=lp54cs1dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1 freq0=2133
freq_ratio0=1 rank=1 pstates=1 dfi_mode=3 hard_macro=<macro_type> -wdbi0
```

- LPDDR4, rdbi

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1 freq0=266 freq_ratio0=1
rank=1 pstates=1 dfi_mode=1 -pll_bypass0 hard_macro=<macro_type> -rdbi
```

- LPDDR5, full training test

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=0 seqCtrl=0x17df
freq0=800 freq_ratio0=2 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type>
```

- LPDDR5, skip training test

```
% runtc cfg=lp54cs2dq18ch1 tc=demo_basic dram=lpddr5 skip_train=1 freq0=67 freq_ratio0=2
rank=1 pstates=1 dfi_mode=1 -pll_bypass0 hard_macro=<macro_type> -svt_dfi
```

- LPDDR5, quickboot test

```
%runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=0 seqCtrl=0x17df freq0=800
freq_ratio0=2 rank=1 pstates=1 dfi_mode=1  hard_macro=<macro_type> -quickboot
```

- LPDDR5, jtag

```
% runtc cfg=lp54cs1dq18ch1 tc=demo_basic dram=lpddr5 skip_train=1 freq0=344 freq_ratio0=2
rank=1 pstates=1 dfi_mode=1 -jtag hard_macro=<macro_type> -svt_dfi
```

- LPDDR5, pll_bypass

```
% runtc cfg=lp54cs2dq18ch1 tc=demo_basic dram=lpddr5 skip_train=1 freq0=67 freq_ratio0=2
rank=1 pstates=1 dfi_mode=1 -pll_bypass0 hard_macro=<macro_type> -svt_dfi
```

- LPDDR5, dm

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1 freq0=800 freq_ratio0=1
rank=2 pstates=1 dfi_mode=1 hard_macro=<macro_type> -dm0
```

- LPDDR5, wdbi

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1 freq0=800 freq_ratio0=1
rank=2 pstates=1 dfi_mode=1 hard_macro=<macro_type> -wdbi0
```

- LPDDR5, rdbi

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1 freq0=800
freq_ratio0=1 rank=2 pstates=1 dfi_mode=1 hard_macro=<macro_type> -rdbi0
```

2.6.2.2 Low Power test examples

Refer to table “PHY Low Power State Modes” in DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) Databook.

- LPDDR4, dfi_lp_data

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=2133 freq_ratio0=1
rank=1 pstates=1 dfi_mode=1 lp_mode=1 hard_macro=<macro_type>
```

- LPDDR4, dfi_lp_ctl

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=1000 freq_ratio0=2
rank=1 pstates=1 dfi_mode=1 lp_mode=2 hard_macro=<macro_type>
```

- LPDDR4, dfi_lp_ctl + DRAMCLKSTOP

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=1600 freq_ratio0=1
rank=1 pstates=1 dfi_mode=1 lp_mode=3 hard_macro=<macro_type>
```

- LPDDR4, LP2 +DFICLKSTOP

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_lp lp_mode=4 dram=lpddr4 skip_train=1 freq0=1000
freq_ratio0=1 rank=1 pstates=2 dfi_mode=1 hard_macro=<macro_type>
```

- LPDDR4, IO Retention

```
%runtc cfg=lp54cs2dq18ch2 tc=demo_lp lp_mode=6 dram=lpddr4 skip_train=1 freq0=2133
freq_ratio0=1 rank=1 pstates=1 dfi_mode=3 hard_macro=<macro_type>
```

- LPDDR4, UPF power aware simulation

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_lp lp_mode=6 dram=lpddr4 skip_train=1 freq0=1866
freq_ratio0=1 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type> -upf -xprop -inject_x
```

- LPDDR5, dfi_lp_data

```
% runtc cfg=lp54cs1dq16ch1 tc=demo_lp lp_mode=1 dram=lpddr5 skip_train=1 freq0=344
freq_ratio0=1 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type>
```

- LPDDR5, dfi_lp_ctl

```
% runtc cfg=lp54cs1dq16ch1 tc=demo_lp lp_mode=2 dram=lpddr5 skip_train=1 freq0=344
freq_ratio0=2 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type>
```

- LPDDR5, dfi_lp_ctl + DRAMCLKSTOP

```
% runtc cfg=lp54cs1dq16ch1 tc=demo_lp lp_mode=3 dram=lpddr5 skip_train=1 freq0=344
freq_ratio0=2 rank=1 pstates=1 dfi_mode=1 hard_macro=<macro_type>
```

- LPDDR5, LP2 +DFICLKSTOP

```
% runtc cfg=lp54cs1dq16ch2 tc=demo_lp lp_mode=4 dram=lpddr5 skip_train=1 freq0=800
freq_ratio0=1 freq1=344 freq_ratio1=2 rank=1 pstates=2 dfi_mode=3 hard_macro=<macro_type>
```

- LPDDR5, IO Retention

```
%runtc cfg=lp54cs2dq18ch2 tc=demo_lp lp_mode=6 dram=lpddr5 skip_train=1 freq0=800
freq_ratio0=2 rank=1 pstates=1 dfi_mode=3 hard_macro=<macro_type>
```

- LPDDR5, UPF power aware simulation

```
% runtc cfg=lp54cs1dq16ch1 tc=demo_lp lp_mode=6 dram=lpddr5 skip_train=1 freq0=7
freq_ratio0=2 rank=1 pstates=1 dfi_mode=1 -pll_bypass0 hard_macro=<macro_type> -upf -
xprop -inject_x
```

2.6.2.3 ATE Firmware test examples

- LPDDR5

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> ate=revision_check    freq_ratio0=2

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> ate=impedance_cal     freq_ratio0=2

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> ate=pll_lock          freq_ratio0=2

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> ate=lcdl_linearity    freq_ratio0=2

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> ate=ac_loopback       freq_ratio0=2

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> ate=data_1d_loopback  freq_ratio0=2

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> ate=data_1d_loopback freq_ratio0=2 -
ate_cust_mb_cfg_file=$CTB_HOME/testbench/inc/ate/ate_mb_cust_setting.sv -
ate_cust_csr_cfg_file=$CTB_HOME/testbench/inc/ate/ate_csr_cust_setting.sv

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> freq_ratio0=2 ate=rxreplica

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> freq_ratio0=2 ate=dca

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> freq_ratio0=2 ate=burn_in

% runtc cfg=lp54cs2dq18ch2 tc=demo_ate  ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> freq_ratio0=2 ate=data_2d_loopback

% runtc cfg=lp54cs1dq18ch2 tc=demo_ate ate_clockingmode=lpddr5 freq0=800
hard_macro=<macro_type> freq_ratio0=2 ate=MASIS
```

- LPDDR4

```
% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> ate=revision_check    freq_ratio0=1

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> ate=impedance_cal     freq_ratio0=1

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> ate=pll_lock          freq_ratio0=1

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> ate=lcdl_linearity    freq_ratio0=1
```

```
% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> ate=ac_loopback       freq_ratio0=1

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> ate=data_1d_loopback  freq_ratio0=1

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> freq_ratio0=1 ate=data_1d_loopback
ate_cust_mb_cfg_file=$CTB_HOME/testbench/inc/ate/ate_mb_cust_setting.sv
ate_cust_csr_cfg_file=$CTB_HOME/testbench/inc/ate/ate_csr_cust_setting.sv

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> freq_ratio0=1 ate=rxreplica

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> freq_ratio0=1 ate=burn_in

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate   ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> freq_ratio0=1 ate=data_2d_loopback

% runtc cfg=lp4cs1dq18ch2 tc=demo_ate ate_clockingmode=lpddr4 freq0=2133
hard_macro=<macro_type> freq_ratio0=1 ate=MASIS
```

2.6.2.4 Test case result viewing

The test results (log & waveform dump) reside in the simulation directory.

Open waveform:

Executing verdi.run script to open the *.fsdb file.

Executing "`dve -full64 -vpd *.vpd`" to open the *.vpd file.

2.6.3 Runtc Command Options

The test commands explained in "Test Case Description" on page 23 are used to run tests with runtc command options, such as, ACX4/DBYTE number, dram type and train type, etc.

To run tests with different hardware configurations and software configurations, following options can be followed for different PHY features and different configuration scenarios.

Table 2-3 Configurable Test Options

Command option	Description	Note
cfg=<*>	The combination of different RANK/DBYTE/CHANNEL number	Hardware configuration name, such as: cfg= lp54cs2dq18ch2 .
hard_macro=<*>	Specify hard macro type: A or B	For example, if the user wants to specify hard-macro B add option in runtc command: hard_macro=B Default B.
dram=<*>	Specify dram type: lpddr4/lpddr5	For example, if the user wants to specify LPDDR5, add option in runtc command: dram=lpddr5 Default lpddr4
skip_train=<*>	Specify Training type: 0-training mode 1-skip training 2-devinit	For example, if the user wants to specify running with skiping training, add option in runtc command: skip_train=1 Default 1
seqCtrl=<*>	Specify Training steps to run, See the firmware/Latest/ *ddr*/ mnPmuSramMsgBlock_*.h	Optional Controls the training steps to be run. Each bit corresponds to a training step.
-quickboot	Enable quickboot	Optional Must set with skip_train=0 and seqCtrl=<*>
freq0=<*> freq1=<*>	Specify frequency freq0 is SDRAM device frequency of Pstate0 freq1 is SDRAM device frequency of Pstate1	For example, if the user wants to specify SDRAM device frequency to 1600 MHz for PState 0, add option in runtc command: freq0 = 1600 Default 800
freq_ratio0=<*> freq_ratio1=<*>	Specify CK:WCK for LPDDR5 Specify DFI frequency ratio for LPDDR4/LPDDR4X freq_ratio0 is SDRAM device frequency ratio of Pstate0 freq_ratio1 is SDRAM device frequency ratio of Pstate1 1 – 1:2 2 – 1:4	For example, if the user wants to specify frequency ratio to 1:4 for PState 0, add option in runtc command: freq_ratio0=2 Default: 1

Table 2-3 Configurable Test Options (Continued)

Command option	Description	Note
pstates=<*>	Specify Total number of PStates: 1|2	For example, if the user wants to specify PStates number to 2, add option in runtc command: pstates=2; Default: 1.
rank=<*>	Specify rank number：1|2|4	For example, if the user wants to specify dfi0 rank number to 2, add option in runtc command: rank = 2; Default: 1.
dfi_mode=<*>	Specify DFI mode. dfi_mode: 1|3	For example, if the user wants to specify dfi mode to 1, the user should add option in runtc command: dfi_mode = 1; default: 3.
lp_mode=<*>	Specify low power mode: 0 - no low power 1 - dfi_lp_data 2 - dfi_lp_ctl 3 - dfi_lp_ctl + DRAMCLKSTOP 4 - frequency change 5 - LP3 6 - IO Retention	Only support demo_lp cases For example, if the user wants to run the dfi_lp_data , the user can run the demo_lp test with the option: lp_mode=1
-dm0 -dm1	Specify data mask: dm0 is for pstate0, dm1 is for pstate1.	For example, if the user wants to enable data mask for Pstate0, the user can add the option in runtc command: -dm0 If the user wants to enable data mask for Pstate1, the user can add the option in runtc command: -dm1
-rdbi0 -rdbi1	Specify read dbi: rdbi0 is for pstate0, rdbi1 is for pstate1.	For example, if the user wants to enable read dbi for Pstate0, the user can add the option in runtc command: -rdbi0 If the user wants to enable read dbi for Pstate1, the user can add the option in runtc command: -rdbi1

Table 2-3 Configurable Test Options (Continued)

Command option	Description	Note
-wdbi0 -wdbi1	Specify write dbi: wdbi0 is for pstate0, wdbi1 is for pstate1.	For example, if the user wants to enable write dbi for Pstate0, the user can add the option in runtc command: -wdbi0 If the user wants to enable write dbi for Pstate1, the user can add the option in runtc command: -wdbi1
-default_c	Use default dwc_ddrphy_phyinit_setDefault.c, no override	For example, if the user wants to use default dwc_ddrphy_phyinit_setDefault.c, the user can add the option in runtc command: -default_c
override_c=<*>	Use user-specified new override_c file. ■ All must-have command options listed in this table must be specified with the same value for configurations specified in dwc_ddrphy_phyinit_userCustom_overrideUserInput.c. For example, if DRAM type is specified in dwc_ddrphy_phyinit_userCustom_overrideUserInput.c as follows: userInputBasic.DramType= LPDDR4 The runtc command option dram=<*> should be specified as same value of dram=lpddr4. ■ The following options specified in runtc command line will be ignored and get from user-specified new override_c file. - freq0=<*> - freq1=<*> - freq_ratio0=<*> - freq_ratio1=<*> - pstates=<*> - rank=<*> - dfi_mode=<*> - pll_bypass0 - pll_bypass1 ■ The path of your new override_c file cannot be the same directory of the simulation directory (ctb/Latest/sim or another directory you chose to run tests).	For example, if the user wants to use an existed override_c file, the user can add the option in runtc command: override_c=<path of your new dwc_ddrphy_phyinit_userCustom_overrideUserInput.c >

Table 2-3 Configurable Test Options (Continued)

Command option	Description	Note
-pll_bypass0	Specify Pclk coming from bypass_clk for Pstate0	For example, if the user wants to enable pll_bypass for Pstate0, the user can add the option in runtc command: -pll_bypass0
-pll_bypass1	Specify Pclk coming from bypass_clk for Pstate1	For example, if the user wants to enable pll_bypass for Pstate1, the user can add the option in runtc command: -pll_bypass1
-jtag	If specified, access register by TDR interface	For example, if the user wants to use TDR interface to access registers, the user can add the option in runtc command: -jtag
-upf	If specified, UPF power aware simulation will be enabled. This option will be only valid for demo_lp test with option lp_mode=6(IO retention test)	For example, if the user wants to enable UPF power aware simulation, the user can add the option in runtc command: -upf
-xprop	If specified, X state will be injected for all input ports during power off state. This option will be only valid for demo_lp test with option lp_mode=6(IO retention test)	For example, if the user wants to enable X injection for all input ports during power off state, the user can add the option in runtc command: -xprop
-inject_x	Enable X-injection for all input ports during power down.	Optional For example, if the user wants to X-injection for all input ports during power down, the user can add the option: -inject_x At the same time, this option only be added when -upf is specified.
prefix=<*>	Specify the prefix if the RTL created by coreConsultant with prefix.	For example, if the user creates the RTL by coreConsultant with prefix "p"(Refer to section 3.2 of documentation "DesignWare Cores LPDDR5/4/4X PHY coreKit User Guide" for more details of prefix configuration), the user needs add the option in runtc command. prefix=p Note: Each simulation must run in a cleaned simulation directory.

Table 2-3 Configurable Test Options (Continued)

Command option	Description	Note
dump=<*>	Specify waveform dump format VPD or FSDB or NONE.	For example, if the user wants to dump waveform with FSDB format, the user can add the option in runtc command: dump=FSDB Default: NONE
debug=<*>	Specify verbosity level 0 - UVM_NONE(default), disable MemVIP trace; 1- UVM_MEDIUM, enable MemVIP trace; 2- UVM_FULL.	For example, if the user wants to specify verbosity as 1, the user can add the option in runtc command: debug=1 Default: 0
-clean	Cleans out simulation directory	For example, if the user wants to clean out simulation directory, the user can add the option in runtc command: -clean
-fw	Specify Run preparation step only, creating apb configuration tasks using firmware code	For example, if the user wants to run preparation step only, the user can add the option in runtc command: -fw
-sim	Specify run simulation step only	For example, if the user wants to run simulation step only, the user can add the option in runtc command: -sim
ate=<*>	Specify ATE test mode: ■ revision_check ■ impedance_cal ■ pll_lock ■ lcdl_linearity ■ ac_loopback ■ data_1d_loopback ■ data_2d_loopback ■ burn_in ■ rxreplica ■ dca ■ MASIS	Must-have for demo_ate cases Only support demo_ate cases If ate=pll_lock is specified, ATE pll lock mode will be sequentially tested. If ate = dca is specified, only lpddr5 test cases are supported. If ate=MASIS is specified, CTB will generate ATE test scenarios except burn_in and 2d loopback. MASIS burn-in and 2d loopback will be generated separately by setting ate=burn_in or ate=data_2d_loopback.
burn_in_time=<*>	Set extra wait time for ATE burn-in case. Default 0.	ATE burn-in case only.

Table 2-3 Configurable Test Options (Continued)

Command option	Description	Note
ate_cust_csr_cfg_file =<file_path>[a]	Reconfigure ATE CSR with customer's setting file. Example of setting file: $CTB_HOME/testbench/inc/ate/ate_csr_cust_setting.sv	Optional ATE test only
ate_cust_mb_cfg_file =<file_path>[b]	Reconfigure message block with customer's setting file. Example of setting file: $CTB_HOME/testbench/inc/ate/ate_mb_cust_setting.sv	Optional ATE test only
ate_clockingmode	Specify lpddr4 or lpddr5 to message block	Must-have for demo_ate cases Only support demo_ate cases If test for LPDDR4, set to ate_clockingmode=lpddr4 If test for LPDDR5, set to ate_clockingmode=lpddr5
dmem_rd=<*>	Dump and print DMEM data at the end of simulation. 0 - Disable(default) 1 - Enable	MASIS test only. debug=1 should set.
dmem_start_address=<*> dmem_end_address=<*>	Set start and end address for DMEM read. Default 0x58000 to 0x58fff.	MASIS tests only.
masis_dmem_rd=<*>	Dump and print DMEM message behind each ATE MASIS scenarios, otherwise print it only once at the end of the simulation log file. 0 - Disable (default) 1 - Enable	ATE MASIS test only. Must use combined with dmem_rd=1 and debug=1.
-mem_dump	Dump and print IMEM/DMEM data when using backdoor load and frontdoor APB read.	If user want to check the data in IMEM/DMEM after loading and before test, add the option in runtc command: -mem_dump The detailed print info will be saved in IMEM_section4.log and DMEM_section4.log in current simulation directory.
prt_off=<*>	Add MASIS print info in simulation log for burn-in and 2d loopback. 0 - Print 1 - Not print(default)	ATE burn-in and 2d loopback MASIS test only.

Table 2-3 Configurable Test Options (Continued)

Command option	Description	Note
ate_tests_to_run=<*>	Specify scenarios are performed in ATE MASIS test.	MASIS test only. Each filed of ate_tests_to_run matches with the ATE MsgBlock setting "TestsToRun". For more information, please refer to "DesignWare Cores LPDDR54 PHY Firmware ATE Application Note". For example, if user wants to perform PLL Lock and Impedance Calibration test, set ate_tests_to_run=0x6.

a. The detail of how to use “ate_cust_csr_cfg_file=<file_path>” is shown as following:
- Without this option: Recommended CSR config will be applied for ATE test;
- With this option: The CSR setting in “< file_path>” and the others recommended setting will be applied for ATE test. Customer can input CSR ‘s setting in this file, the format must be same as reference file “ate_csr_cust_setting.sv”. Example: refer to “Test case Description”

b. The detail of how to use “ate_cust_mb_cfg_file =<file_path>” is shown as following:
- Without this option: Recommended ATE message block CSR setting is applied for ATE test. The recommended value are set in file “ate_mb_default_setting.sv”;
- With this option: The ATE message block CSR setting in “<file_path>” and the others recommended setting in “ate_mb_default_setting.sv” are applied for ATE test. Customer can input ATE message blockCSR setting in “<file_path>”,the input file’s format must be same as reference file “ate_mb_cust_setting.sv”. Example: refer to “Test case Description”

2.7 Verification Components and Testbench

2.7.1 Customer Testbench Directory Structure

The CTB directory structure is described below:

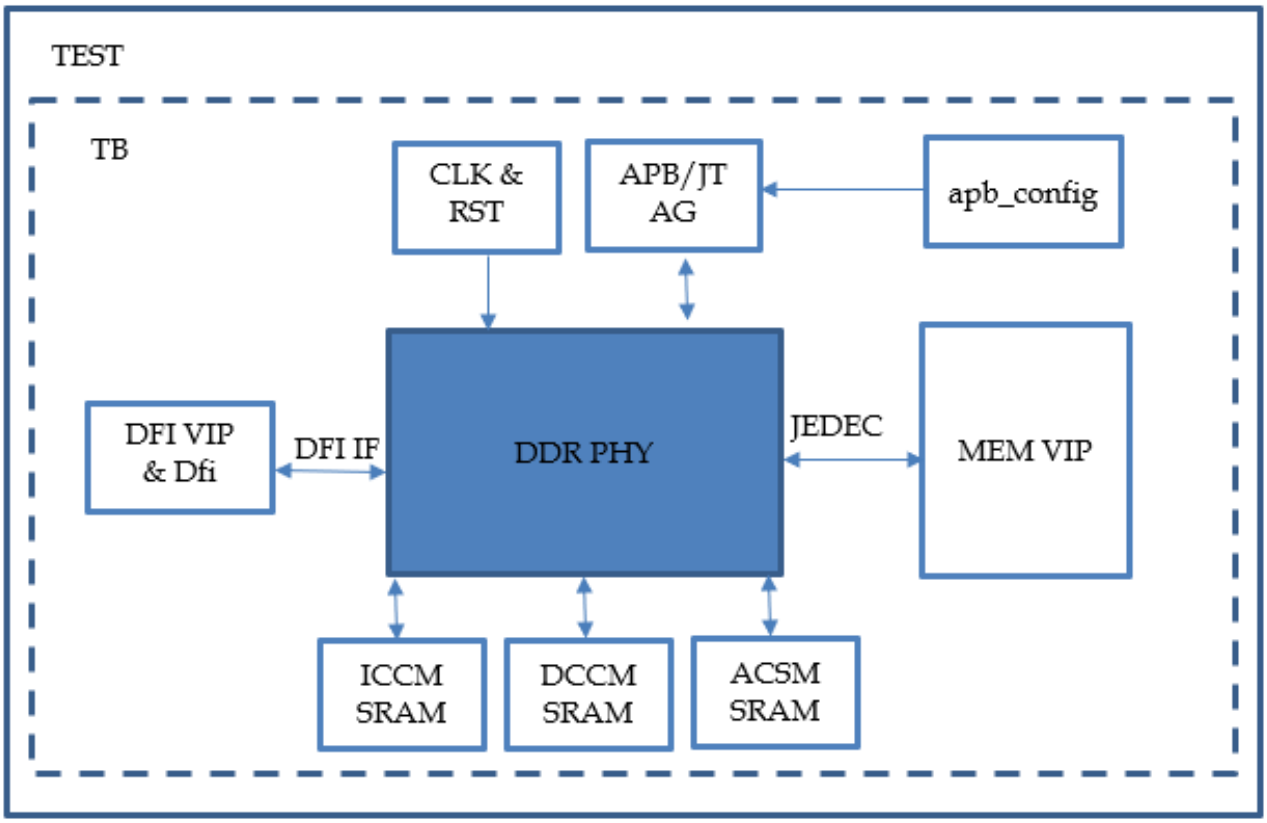
- sim: Simulation directory.
  - runtc: simulation run scripts.
  - run_basic_vcs: automatically run typical simulations.
- testbench/vc: Verification Components directory. Contains sub-folders for each verification component:
  - cfg: Testbench configuration component. Hold information about the testbench software configuration.
  - apb: APB driver. Generates APB read/write transactions to for the DUT software configuration.
  - jtag: JTAG driver. Generates JTAG read/write transactions to for the DUT software configuration.
  - clk_rst: Clock and reset generation logic.
  - lpddr5: LPDDR5 System Verilog wrapper for the Synopsys DDR UVC.
  - lpddr4: LPDDR4 System Verilog wrapper for the Synopsys DDR UVC.
  - dfi: DFI VIP and Simplified DFI driver with LPDDR4/LPDDR5 mode support.
- testbench/tc: Test case directory. All tests in the sub-folder named demo.
- testbench/tb: Testbench top-level directory. Contains the testbench top-level module.
- testbench/rtl: Contains the models of SRAMs (the instruction and data synchronous SRAMS, ACSM SRAM).
- testbench/inc: Include directory. Contains include files for the tests and run scripts.

2.7.2 Customer Testbench Implementation

The Customer Testbench is designed using methodology-independent System Verilog language. The UVM-based components such as Synopsys MEM VIP is wrapped with the System Verilog wrappers to enable their instantiation in the module-based Customer Testbench environment.

The following figure outlines the Customer Testbench architecture.

Figure 2-1 Customer Testbench Architecture



The test case is compiled as a top-level module. The testbench top (TB) is instantiated in the test case. TB contains an instances of the Design-under-test ((DDR PHY) as well as all required verification components. The control for the verification components is done from the test level.

The apb_config code is generated from the PHYINIT C code prior to running the testbench.

The DFI driver generate the DFI protocol - compatible transactions on the DUT (Design-Under-Test) DFI interface as well as monitor the DFI read data reception.

Both ICCM and DCCM SRAM instances are required to store the DDR controller instruction and data code.

ACSM SRAM is used to store playback command for HWT.

The LPDDR4/5 component encapsulates the Synopsys LPDDR4/5 SDRAM memory VIP model.

2.7.3 DFI Driver

The Customer Testbench implement DFI driver uses Synopsys DFI Verification IP. In order to implemented a basic functioning SV testbench using UVM DFI Verification IP, CTB create an wrapper named svt_dfi_uvmvlog.sv for DFI VIP and this wrapper are connected to DUT in top.v file. Some basic task also created in svt_dfi_uvmvlog.sv to provide DFI controls towards the DUT.

The DFI control interface provides all required DFI controls towards the DUT. The following table shows the tasks that are implemented and can be called in tc directly.

Table 2-4 DFI Control Interface Tasks

Task	Arguments	Description
device_init	Input bit freq_change Input bit Pstate	Realize device initialization by sends series MRS command with specified address and data through the DFI interface. freq_change value stand for whether the device initial is happened during frequency change. Pstate value stand for the current work phase of device. Example: device_init(0,0);
activate	Input bit [3:0] rank Input bit [1:0] bank_group Input bit [2:0] bank_address Input bit [16:0] row	Activates the row in the specified bank and rank. Example: activate(4'b1110,2'b0, 3'b0, 17'h70);
wrs	Input bit [3:0] rank Input bit [1:0] bank_group Input bit [2:0] bank_address Input bit [16:0] column Input reg [8*`DWC_DDRPHY_NUM_DBYTES-1:0] data[32] reg [`DWC_DDRPHY_NUM_DBYTES-1:0] data_dm[32]	Write BL16, on the Fly command. By default, the data[32] and data_dm[32] are written with fixed pattern and can be overridden to another pattern at any time. Example: wrs(4'b1110, 2'b0, 3'b0, 10'h70);
rds	Input bit [3:0] rank Input bit [1:0] bank_group Input bit [2:0] bank_address Input bit [16:0] column	Read BL16, on the Fly command. The read data is received at the DFI interface and compared against the write pattern. Example: rds(4'b1110, 2'b0, 3'b0, 10'h70);
pm_fc_wrapper	Input bit [4:0] next_freq	Change to the specified frequency
dfi_lp	Input int hold_time	Entering the DFI LPCTRL state or DFI LPCTRL + DRAMCLKSTOP state or the DFI LPDATA state The selector is the command option “lp_mode=” Example: lp_mode=1 Entering the DFI LPDATA state

Note The data checking at the DFI interface assumes that write to the specified address is followed with the read from the same address. Then, read data is compared against the written one. It is possible to override the write pattern to another value between the commands, at any simulation time.

Example:

```
wrs(4'b1110, 2'b0, 3'b0, 10'h70);
rds(4'b1110, 2'b0, 3'b0, 10'h70);
```

2.7.4 APB Driver

The APB Driver generates write and read command at the APB interface. The following table shows the tasks that are implemented.

Table 2-5 APB Driver Tasks

Task	Arguments	Description
write	input [31:0] addr input [15:0] data	Generates APB write command with the specified address and data. Example: top.apb.write(32'h0000_0010, 16'haabb);
read	input [31:0] addr output [15:0] data	Generates APB read command with the specified address. Returns back the data read.

Note The firmware generator script writes down the "csr_defines.v" file to the test directory. This file contains all the register name – to – address mapping macros for the specified hardware configuration. Each register name consists of 2 parts:

```
<address_block_name>__<register_name>, for example:
DWC_DDRPHYA_DBYTE5__RxFifoInfo
```

It is convenient to use register name macros instead of addresses as the addresses to the APB write and read commands.

2.7.5 JTAG driver

The JTAG Driver generates write and read command at the JTAG interface. The following table shows the tasks that are implemented..

Table 2-6 JTAG Driver Tasks

Task	Arguments	Description
write	input [31:0] addr input [15:0] data	Generates JTAG write command with the specified address and data. Example: 1, add “-jtag” option in runtc command; 2, top.jtag.write(32’h0000_0010, 16’haabb);
read	input [31:0] addr output [15:0] data	Generates JTAG read command with the specified address. Returns back the data read.

Note The firmware generator script writes down the “csr_defines.v” file to the test directory. This file contains all the register name – to – address mapping macros for the specified hardware configuration. Each register name consists of 2 parts:

```
<address_block_name>__<register_name>, for example:
DWC_DDRPHYA_DBYTE5__RxFifoInfo
```

It is convenient to use register name macros instead of addresses as the addresses to the APB write and read commands.

2.7.6 LPDDR4/5 DIMM BFM

The Customer Testbench uses Synopsys LPDDR4/5 UVC to model the DIMM.

See the VC Verification IP LPDDR4/5 Memory UVM User Guide for details.

2.8 Functional GateSim on CTB

This chapter provides an overview of the step-by-step process to simulate functional GateSim with or without SDF back annotation on synopsys' customer testbench. The functional GateSim targets toward demonstrating functional gate level simulation on the Customer Testbench (CTB) and providing a reference when customers need to use DDRPHY to set up customers' own gate level simulation environment.

Note The presence of this "Functional GateSim on CTB" section in this documentation and the gatesim_script directory in a PHY release package are not an indication all the necessary views are present or available in the PHY release package to support functional gate level simulations. Depending on the maturity of the release, this capability may be supported (with limitations). It is recommended to run a simulation with the -compile option (see "GateSim Command Options" on page 40 for details) to determine if all the views are present in the release.

The functional GateSim on the Customer Testbench (CTB) uses gate level netlists of the hardened IP components (DIFF, SE, SEC and Master), RTL simulation models of the analog custom circuits, and verilog design of the soft IP--DDRPHY utility block(PUB) and AC/Dbyte wrapper.

Note Not all the features and configurations on the Customer Testbench(CTB) are supported by this version of functional GateSim. For more information about this, see section "Unsupported Features" on page 44.

2.8.1 GateSim Command Options

Table 2-7 GateSim Command Options

Task	Arguments	Description
-gatesim	If specified, gate level simulation will be performed, otherwise RTL simulation will be performed.	Mandatory for GateSim.
-sdf	If specified, gate level simulation with SDF back annotation to HardMacro netlist will be performed, otherwise gate level simulation without SDF back annotation will be performed, when -gatesim option specified.	Optional GateSim option.

Table 2-7 GateSim Command Options (Continued)

Task	Arguments	Description
cfg=<*>	The combination of specific number of AC blocks and specific number of Dbyte blocks.	Mandatory command option. For GateSim, the cfg option must be specified to the following values: For LPDDR4 tests, cfg= lp54cs2dq18ch2 For LPDDR5 tests, cfg= lp54cs2dq18ch2 Note: Functional gate level simulation support lp54cs2dq18ch2 PHY configuration only. If the generated PHY configuration is not lp54cs2dq18ch2, user must re-configure the PHY using coreConsultant to generate the required configuration for functional gate level simulation. User can find the generated PHY configuration from the filename: <workspace>/src/dwc_ddrphy_<configuration_name>_VDEFINES.v
ac_ew_use=<*>	Specify the orientation for each AC blocks specified by cfg=<*> option.	Mandatory for GateSim with or without SDF back annotation. The specified value of ac_ew_use should be hexadecimal to cover all the AC blocks specified by cfg=<*> option. Use binary to indicate the orientation of each AC block. 1: EW (East-West sides of an SoC); 0: NS (North-South sides of an SoC). undefined: indicating no orientation (by default) For example, if the cfg option has been set to lp54cs2dq18ch2, it means the number of AC blocks is configured to 2, it determined by ch*(the channel number) 1.) If all AC blocks are placed to NS orientation, then set ac_ew_use=0 2.) If all AC blocks are placed to EW orientation, then set ac_ew_use=3

Table 2-7 GateSim Command Options (Continued)

Task	Arguments	Description
dbyte_ew_use=<*>	Specify the orientation for each DBYTE blocks specified by cfg=<*> option.	Mandatory for GateSim with or without SDF back annotation. The specified value of dbyte_ew_use should be hexadecimal to cover all the DBYTE blocks specified by cfg=<*> option. Use binary to indicate the orientation of each DBYTE block. 1: EW(East-West sides of an SoC); 0: NS (North-South sides of an SoC). undefined: indicating no orientation (by default) For example, if the cfg option has been set to lp54cs2dq18ch2, it means the number of DBYTE blocks is configured to 4, it determined by (dq*/8)*(ch*) If DBYTE0 ~ DBYTE1 blocks are placed to NS orientation, other DBYTE blocks(DBYTE2~DBYTE3) are placed to EW orientation, then set dbyte_ew_use=c
master_ew_use=<*>	Specify the orientation for master block	Mandatory for GateSim with or without SDF when master has the orientation. 1: EW(East-West sides of an SoC); 0: NS(North-South sides of an SoC); undefined: indicate no orientation (by default)
corner=<*>	Specify the process corner of SDF file.	Only for GateSim with SDF back annotation. If this option is not specified, ff corner is used by default, otherwise, user can specify it to other process corners, such as, <ss>;<sspg>; <ff>;<ffpg>; <tt>;<ttpg> If users want to specify the pvt condition in the pvt command option, then this associated process corner option must be specified correctly.
pvt=<*>	Specify the pvt condition of SDF file.	Only for GateSim with SDF back annotation. If the option is not specified, embedded script will automatically capture the first sdf file among all the pvt conditions associated with specified process corner. Or users can specify pvt condition associating with process corner option. This pvt condition can be extracted from sdf file name and it is next to the process corner. For example, the sdf file is dwc_ddrphymaster_top_ss0p675v125c_rcworst_CCworst.sdf The process corner specified by user is ss, then the pvt should be specified as 0p675v125c_rcworst_CCworst.

Table 2-7 GateSim Command Options (Continued)

Task	Arguments	Description
mtm_spec=<*>	Specify using the maximum /minimum/typical delays in min:typ:max delay triplets in the SDF file.	Only for GateSim with SDF back annotation. If this option is not specified, min delay will be used by default, otherwise, user can specify it to other delay groups, such as max, min, typical
-gzip_sdf	Specify using gzip sdf. Note: Only for Gatesim with SDF back annotation.	Optional. Gatesim option. If this option is not specified, no gzip sdf will be used by default.
-compile	If specified, will only do compilation.	Optional. This option can be used when users want to check the availability of GateSim related deliverables. All above GateSim options(such as, -gatesim, -sdf, cfg=<*>, ac_ew_use=<*>, dbyte_ew_use=<*>, master_ew_use=<*>, corner=<*>, pvt=<*>, etc.) must be specified as well when this option is specified.

2.8.2 GateSim Features

2.8.2.1 Support Features

Functional GateSim supports the following features on this version of Customer Testbench:

- PHY initialization and control
- Supports skip-training mode
- DFI low power mode
- Different PState(0,1) for frequency change feature (disable FSP)
- DFI frequency change
- DFI frequency change with pll bypass
- IO retention
- PLL bypass mode
- DM/DBI support
- LPDDR4/LPDDR4X/LPDDR5
- 2 Channel
  - 18-bit data path widths per channel
  - Total up to 36 bits data path using two channels

- Different memory rank number support
  - Supports 1, 2 memory ranks for LPDDR4.
  - Supports 1, 2 memory ranks for LPDDR4X.
  - Supports 1, 2 memory ranks for LPDDR5.
- Fixed SDRAM device type
  - X16 SDRAM device type for LPDDR4.
  - X16 SDRAM device type for LPDDR4X.
  - X16 SDRAM device type for LPDDR5.
- Frequency ratio
  - CK:WCK = 1:2, CK:WCK = 1:4, DFI:CK=1:1 for LPDDR5
  - DFI 1:2 and 1:4 mode support for LPDDR4
- Synopsys DFI/DDR memory VIP support
- APB interface to configuration registers
- ATE FW
- support LPDDR4 training FW
- support LPDDR5 training FW

2.8.2.2 Unsupported Features

All the CTB unsupported features listed on the section “Unsupported Features” on page 17 are also not supported by functional GateSim.

2.8.2.3 Testcase list

Currently, functional GateSim on this version of Customer Testbench (CTB) supports the following testcases:

- demo_basic test
  - LPDDR4, Skip train
  - LPDDR4, training mode
  - LPDDR5, training mode
  - LPDDR5, Skip train
  - LPDDR4X, Skip train
  - Pll bypass
  - DM/DBI
- demo_lp test
  - DFI low power data
  - DFI low power control

- DFI frequency change
- DFI frequency change with pll bypass
- IO retention

- demo_ate test
  - revision_check
  - impedance_cal
  - pll_lock
  - lcdl_linearity
  - ac_loopback
  - data_1d_loopback

2.8.3 Run GateSim on the Customer Testbench

Note If “Functional Gatesim” activity is available in the GUI, it is recommended to run Functional Gatesim in the coreConsultant GUI. See section “Functional Gatesim” in the “DesignWare Cores LPDDR5/4/4X PHY coreKit User Guide” for details. Then this section can be skipped. Otherwise, if user expect to run CTB in <WORKSPACE>/ctb or synopsys/<ip_name>/Latest/ctb/Latest/sim or another writable directory as the simulation directory without CoreTool, out of coreConsultant GUI, follow this section to execute simulation.

To run GateSim on the Customer Testbench (CTB), follow the STEPS to set up simulation environment and run the tests:

STEP 1: Install the Customer Testbench(CTB). See section “Installation” on page 17

STEP 2: Update the bootenv script

1. Open the file of bootenv.
   a. If the PHY release package includes an installed_coreKit component, open the file of <WORKSPACE>/ctb/Latest/sim/bootenv
   b. If the PHY release package does not include an installed_coreKit component, which includes pub and macro components, open the file of ctb/Latest/sim/bootenv
2. Find the line of “set sdf_path <metal stack>” and update the metal stack name.

Hint The user can find the metal stack in $CTB_HOME/../../<master_ns|master_ew|master>/Latest directory.
It is the name between timing and sdf.
For example, 6M_1X_h_1Xa_v_1Ya_h_2Y_vh

Note:

For PHY release package that do not use an installed_coreKit, the environment variable must be changed in bootenv: `setenv CORETOOLS 0.`

3. Save the bootenv file.
4. % source bootenv

STEP 3: Go to simulation directory

1. For PHY release package that do not use an installed_coreKit component, which includes pub+macro components:

   The user can enter into CTB sub-directory(synopsys/<ip_name>/Latest/ctb/Latest/sim) or another writable directory as the simulation directory.

2. For PHY release package that includes an installed_coreKit component, the user must enter the simulation directory in the workspace created by coreConsultant with Installation Procedure Step1.(<workspace>/ctb/Latest/sim).

STEP 4: Run GateSim with or without SDF for different tests listed on section "Testcase list" on page 44

1. Running GateSim without SDF:

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=< Highest Frequency >
freq_ratio0=1 rank=2 pstates=1 dfi_mode=3  hard_macro=<*> -gatesim
ac_ew_use=<*>
dbyte_ew_use=<*> master_ew_use=<*>
```

Hint

1. hard_macro can be A/B, user must set it correctly according to the *PHY Databook*
2. cfg=<*>, refer to AC/DBYTE configuration in section"GateSim Command Options" on page 40
3. tc=<*>, refer to section "Testcase list" on page 44
4. -gatesim is mandatory. Refer to table "GateSim Command Options" on page 40
5. ac_ew_use=<*>, refer to table in section "GateSim Command Options" on page 40
6. dbyte_ew_use=<*>, refer to table in section "GateSim Command Options" on page 40
7. master_ew_use=<*>, refer to table in section "GateSim Command Options" on page 40
8. freq0=< Highest Frequency >, refer to PHY data book for the supported highest frequency

1. Running GateSim with SDF:

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=< Highest Frequency > freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim -sdf corner=<*> pvt=<*> ac_ew_use=<*>
dbyte_ew_use=<*> mtm_spec=<*> <-gzip_sdf>
```

Hint

1. hard_macro can be A/B, user must set it correctly according to the PHY Databook
2. cfg=<*>, refer to AC/DBYTE configuration in section "GateSim Command Options" on page 40
3. tc=<*>, refer to section "Testcase list" on page 44
4. -gatesim is mandatory. refer to Table 2-6 in section "GateSim Command Options" on page 40
5. -sdf is mandatory. refer to table in section "GateSim Command Options" on page 40
6. ac_ew_use=<*>, refer to table in section "GateSim Command Options" on page 40
7. dbyte_ew_use=<*>, refer to table in section "GateSim Command Options" on page 40
8. master_ew_use=<*>, refer to table in section "GateSim Command Options" on page 40
9. corner=<*>, refer to table in section "GateSim Command Options" on page 40
10. pvt=<*>, refer to table in section "GateSim Command Options" on page 40
11. mtm_spec=<*>, refer to table in section "GateSim Command Options" on page 40
12. freq0=< Highest Frequency >, refer to PHY data book for the supported highest frequency
13. -gzip_sdf,refer to Table 2-7 on page 40.

2.8.3.1 GateSim Run Command without SDF back annotation

The following command list is for tests supported in the section "Testcase list" on page 44 :

- demo_basic test

```
//LPDDR4, skip train

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency > freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//LPDDR4, training mode

% runtc tc=demo_basic cfg=lp54cs2dq18ch2 dram=lpddr4 skip_train=0 seqCtrl=121f
freq0=<Highest Frequency > freq_ratio0=1 rank=1 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//LPDDR4X, skip train

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 -lp4x skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>
```

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 -lp4x skip_train=1
freq0=<Highest Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//LPDDR5, skip train

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//LPDDR5, training mode

% runtc tc=demo_basic cfg=lp54cs2dq18ch2 dram=lpddr5 skip_train=0 seqCtrl=121f
freq0=<Highest Frequency> freq_ratio0=2 rank=1 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>
```

- demo_lp test

```
//dfi lp data mode

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=1 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=1 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=1 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=1 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//dfi lp data ctrl

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=2 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=2 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=2 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>
```

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=2 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//dfi lp data+ctrl mode

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=3 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=3 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=3 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=3 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//dfi frequency change

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=2 dfi_mode=3 lp_mode=4 -fsp_disable
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=2 dfi_mode=3 lp_mode=4 -fsp_disable
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 freq1=344 freq_ratio1=2 rank=2 pstates=2 dfi_mode=3
lp_mode=4  -fsp_disable hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*>
master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 freq1=344 freq_ratio1=2 rank=2 pstates=2 dfi_mode=3
lp_mode=4  -fsp_disable hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*>
master_ew_use=<*>

//dfi frequency change with pll bypass

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 freq1=25 rank=2 pstates=2 dfi_mode=3 lp_mode=4 -
fsp_disable -pll_bypass1 hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*>
master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 freq1=25 rank=2 pstates=2 dfi_mode=3 lp_mode=4 -
fsp_disable -pll_bypass1 hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*>
master_ew_use=<*>
```

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 freq1=7 freq_ratio1=2 rank=2 pstates=2 dfi_mode=3
lp_mode=4 -fsp_disable -pll_bypass1 hard_macro=<*> -gatesim ac_ew_use=<*>
dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 freq1=7 freq_ratio1=2 rank=2 pstates=2 dfi_mode=3
lp_mode=4 -fsp_disable -pll_bypass1 hard_macro=<*> -gatesim ac_ew_use=<*>
dbyte_ew_use=<*> master_ew_use=<*>

//IO retention

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=6 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=6 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=6 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=6 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//DM/DBI

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -wdbi0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -rdbi0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -dm0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -wdbi0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -dm0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -rdbi0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

//PLL Bypass
```

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333
freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 -pll_bypass0 -gatesim ac_ew_use=<*>
dbyte_ew_use=<*> master_ew_use=<*>
```

- demo ate

```
% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=revision_check
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=impedance_cal
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=pll_lock
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=lcdl_linearity
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=ac_loopback
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=data_1d_loopback
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

%runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=dca freq_ratio0=2 -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

%runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=rxreplica
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=revision_check
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=impedance_cal
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=pll_lock
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=lcdl_linearity
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=ac_loopback
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=data_1d_loopback
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>

%runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=rxreplica
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>
```

2.8.3.2 GateSim Run Command with SDF back annotation

The following command list is for tests supported in the section "Testcase list" on page 44:

- demo_basic test

```
//LPDDR4, training mode

% runtc tc=demo_basic cfg=lp54cs2dq18ch2 dram=lpddr4 skip_train=0 seqCtrl=121f
freq0=<Highest Frequency > freq_ratio0=1 rank=1 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

//LPDDR4, skip train

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

//LPDDR4X, skip train

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 -lp4x skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 -lp4x skip_train=1
freq0=<Highest Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

//LPDDR5, training mode

% runtc tc=demo_basic cfg=lp54cs2dq18ch2 dram=lpddr5 skip_train=0 seqCtrl=121f
freq0=<Highest Frequency> freq_ratio0=2 rank=1 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

//LPDDR5, skip train

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>
```

- demo_lp test

```
//dfi lp data mode

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=1 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=1 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=1 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=1 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

//dfi lp data ctrl

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=2 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=2 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=2 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=2 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

//dfi lp data+ctrl mode

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=3 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=3 hard_macro=<*> -
```

```
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=3 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=3 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

//dfi frequency change

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=2 dfi_mode=3 lp_mode=4 -fsp_disable
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=2 dfi_mode=3 lp_mode=4 -fsp_disable
hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 freq1=344 freq_ratio1=2 rank=2 pstates=2 dfi_mode=3
lp_mode=4  -fsp_disable hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*>
master_ew_use=<*> -sdf corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 freq1=344 freq_ratio1=2 rank=2 pstates=2 dfi_mode=3
lp_mode=4  -fsp_disable hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*>
master_ew_use=<*> -sdf corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

//dfi frequency change with pll bypass

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 freq1=25 rank=2 pstates=2 dfi_mode=3 lp_mode=4 -
fsp_disable -pll_bypass1 hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*>
master_ew_use=<*> -sdf corner=<*> pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 freq1=25 rank=2 pstates=2 dfi_mode=3 lp_mode=4 -
fsp_disable -pll_bypass1 hard_macro=<*> -gatesim ac_ew_use=<*> dbyte_ew_use=<*>
master_ew_use=<*> -sdf corner=<*> pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 freq1=7 freq_ratio1=2 rank=2 pstates=2 dfi_mode=3
lp_mode=4 -fsp_disable -pll_bypass1 hard_macro=<*> -gatesim ac_ew_use=<*>
dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*> pvt=<*>  mtm_spec=<*> <-
gzip_sdf>
```

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 freq1=7 freq_ratio1=2 rank=2 pstates=2 dfi_mode=3
lp_mode=4 -fsp_disable -pll_bypass1 hard_macro=<*> -gatesim ac_ew_use=<*>
dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*> pvt=<*>  mtm_spec=<*> <-
gzip_sdf>

//IO retention

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=6 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr4 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=6 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 lp_mode=6 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_lp dram=lpddr5 skip_train=1 freq0=<Highest
Frequency> freq_ratio0=2 rank=2 pstates=1 dfi_mode=3 lp_mode=6 hard_macro=<*> -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*>  mtm_spec=<*> <-gzip_sdf>

//DM/DBI

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -wdbi0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>
-sdf corner=<*> pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -rdbi0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>
-sdf corner=<*> pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -dm0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -
sdf corner=<*> pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -wdbi0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>
-sdf corner=<*> pvt=<*>  mtm_spec=<*> <-gzip_sdf>

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -dm0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -
sdf corner=<*> pvt=<*>  mtm_spec=<*> <-gzip_sdf>
```

```
% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr5 skip_train=1
freq0=<Highest Frequency> freq_ratio0=1 rank=2 pstates=1 dfi_mode=3
hard_macro=<*> -rdbi0 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>
-sdf corner=<*> pvt=<*>  mtm_spec=<*> <-gzip_sdf>

//PLL Bypass

% runtc cfg=lp54cs2dq18ch2 tc=demo_basic dram=lpddr4 skip_train=1 freq0=333
freq_ratio0=1 rank=2 pstates=1 dfi_mode=3 -pll_bypass0 -gatesim ac_ew_use=<*>
dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*> pvt=<*>  mtm_spec=<*> <-
gzip_sdf>
```

- demo ate

```
% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=revision_check
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=impedance_cal
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=pll_lock
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=lcdl_linearity
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=ac_loopback
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=data_1d_loopback
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

%runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=dca freq_ratio0=2 -
gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf corner=<*>
pvt=<*> mtm_spec=<*> <-gzip_sdf>

%runtc tc=demo_ate  ate_clockingmode=lpddr5 freq0=800  ate=rxreplica
freq_ratio0=2 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=revision_check
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*>  -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=impedance_cal
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=pll_lock
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=lcdl_linearity
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=ac_loopback
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>
```

```
% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=data_1d_loopback
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>

% runtc tc=demo_ate  ate_clockingmode=lpddr4 freq0=2133 ate=rxreplica
freq_ratio0=1 -gatesim ac_ew_use=<*> dbyte_ew_use=<*> master_ew_use=<*> -sdf
corner=<*> pvt=<*> mtm_spec=<*> <-gzip_sdf>
```

2.8.4 Check GateSim result

> **Note** If user run Functional Gatesim in the coreConsultant GUI, the test results (logs or waveform dump files) will be in <WORKSPACE>/funcGateSim/<test_*>/. See section “Functional Gatesim” in the “DesignWare Cores LPDDR5/4/4X PHY coreKit User Guide” for details.

Table 2-8 describes the test results (logs or waveform dump files) in the <simulation directory>

For PHY release package that includes an installed_coreKit component, the simulation directory is <WORKSPACE>/ctb/Latest/sim

For PHY release package that does not include an installed_coreKit component, which includes pub and macro components, the simulation directory is ctb/Latest/sim/ or another writable simulation directory user choose to run tests.

Table 2-8 Test Results

File Name	Description
compile.log	Indicate the compile result.
simv.log	Indicate the final result of the simulation. If the simulation passes, there will be a “test passed” indicator, otherwise, there will be a “test failed” indicator at the end of log to inform users the final result of the simulation.
ac<n>_<diff_ck|se<n>|sec<n>>_sdf.log	SDF Gatesim only. Indicate the SDF annotation result of each sub-module in AC block.
dbyte<n>_<diff_DQS|diff_WCK|se<n>>_sdf.log	SDF Gatesim only. Indicate the SDF annotation result of each sub-module in Dbyte block.
master_sdf.log	SDF Gatesim only. Indicate the SDF annotation result of the Master block.
test.fsdb	If users specify the dump option to FSDB, the waveform dump will be test.fsdb
test.vpd	If users specify the dump option to VPD, the waveform dump will be test.vpd

> **Hint** GateSim test result should check ac0_<*>_sdf.log, dat0_<*>_sdf.log, master_sdf.log first to make sure there is no SDF back anotation errors, then check simv.log for the simulation pass/fail result at the bottom of the log. If simv.log is empty, it may indicate there are compile errors in the compile.log .

2.8.5 Troubleshooting and Support

This section provides troubleshooting tips if users encounter problems when setting up your own environment. If a problem that users cannot resolve is not covered by this section, open a Customer Support case.

2.8.5.1 SDF Warnings and Errors

Q1: SDF Warning-[SDFCOM_INF] IOPATH not found

Q2: SDF Warning-[SDFCOM_TANE] TIMINGCHECK Annotation Not Enabled

A: Make sure that these SDF warnings only exist for custom analog circuits (such as, IO, PLL, LCDL, POR, etc.) Those warnings can be waived because the custom analog circuits use RTL simulation model in GateSim and should NOT be annotated by SDF delay.

Q3: SDF Warning- [SDFCOM_SWC] Simple Wire Connection

Q4: SDF Warning- [SDFCOM_IWSBA] INTERCONNECT will still be annotated

A: These two type of warnings can be waived.

2.8.5.2 GateSim Setup Reference

Following are GateSim setup files on the CTB, which might be helpful as a reference when user set up GateSim environment on their own SoC level GateSim:

- **File list**

  Location: <simulation directory>(<WORKSPACE>/ctb/Latest/sim or synopsys/<ip_name>/Latest/ctb/Latest/sim/ or another writable simulation directory user choose to run tests)

  The file list is generated automatically by running GateSim test in <simulation directory or another writable simulation directory user choose to run tests > (See "Run GateSim on the Customer Testbench" on page 45).

- **VCS Command Options**

  Location: <WORKSPACE>/ctb/Latest/sim/runtc or synopsys/<ip_name>/Latest/ctb/Latest/sim/runtc

  Roll down to "#deal with cmd_opts for gatesim" part in runtc file to get the reference command options for GateSim.

- **SDF Back Annotation**

  Location: <WORKSPACE>/ctb/Latest/testbench/tb/top.sv or synopsys/<ip_name>/Latest/ctb/Latest/testbench/tb/top.sv

  Search macro "DWC_DDRPHY_GATESIM_SDF" in top.sv to get the sdf annotation settings.

  For SDF files used, search "#SDF Back Annotation files select for SDF GateSim" in runtc

- **Timing Disable List for SDF GateSim**

  Location: <WORKSPACE>/ctb/Latest/testbench/inc/gatesim_timing_disable_list.sv or synopsys/<ip_name>/Latest/ctb/Latest/testbench/inc/gatesim_timing_disable_list.sv

3 Behavioral Verification

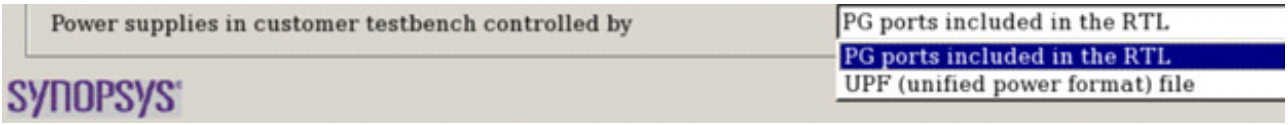
This section provides a list of steps to instantiate and verify the DDRPHY is connected properly: power up, clock creation, reset, and initialization. Additionally, there are sections on how to verify specific areas of the DDRPHY after instantiation and connection; firmware, PHYINIT and memory models. The last sections provide examples of how to verify different interfaces of the DDRPHY: Firmware, DFI interfaces, DFI modes, etc.

The following chapters are included in this section:

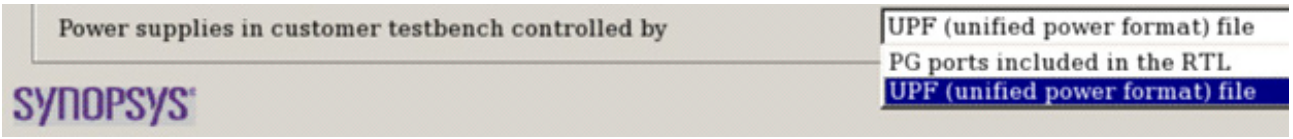
3.1 Power Up

Depending on customer configuration regarding PG (power/ground) pins, power pins will need to be driven to the DDRPHY.

If DDRPHY is configured using DWC_DDRPHY_PG_PINS, user must drive power pins VAA/VDD/VDDQ to 1'b1 and ground pin VSS to 1'b0. In the coreConsultant GUI, set the following:



If DDRPHY does *not* have DWC_DDRPHY_PG_PINS defined, customer uses UPF file/flow to drive power/ground. In the coreConsultant GUI, set the following:



Refer to "(A) Bring up VDD, VDDQ, and VDD2" section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook for more information on driving power/ground pins.

Refer to "DWC_DDRPHY_PG_PINS"section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook for more information on power/ground pins.

3.2 Clock Creation

There are four main DDRPHY clocks to generate; DfiClk, APBCLK, PllRefClk, PllBypClk (if PLL Bypass enabled), and TDRCLK (if JTAG interface is enabled).

Refer to "Clocks" section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook for more information regarding speed restrictions/dependencies between all clocks.

Clock generation example:

```
initial begin
  clk = 1'b0;
  half_period = ClockPeriod/2;
  forever begin
    (#half_period) clk <= ~clk;
  end
end
```

Note Confirm that SoC hardware and/or testbench generates the synchronous clocks DfiClk and DfiCtlClk from a single root clock source.

3.3 Reset

Refer to “Resets” section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook for description on how to drive ColdReset to start simulation.

3.4 SRAM

The ARCv2 Microcontroller uses 2 SRAM’s (ICCM and DCCM) for storage of program data and instructions. These 2 SRAM’s must be instantiated and connected to the SRAM ports at the RTL top level dwc_ddrphy_top.v. The DCCM and ICCM SRAM’s must be connected to the separate interfaces. Figure “Microcontroller Subsystem, including external SRAMs” in the PUB databook shows an example implementation of the DCCM. Table “ICCM and DCCM Interface Signals” in the PUB databook lists out the ICCM and DCCM interface signals to connect the SRAM’s

To verify these 2 SRAM’s are properly instanced, the user can create a WR/RD test case that will verify that the SRAM’s can be written to (verify WR path) and read from (verify RD path).

The ICCM and DCCM SRAM’s are mapped to DDRPHY CSR address spaces. Using these 2 addresses, a WR/RD CSR test case can verify SRAM connection.

Test sequence to verify ICCM and DCCM SRAM connectivity:

```
// Power up and initialize
power_up sequence, skip training

// Test vars
ICCM_csr_addr = 'h5_0000
DCCM_csr_addr = 'h5_8000
wr_csr_data[3];
rd_csr_data[3];

// Init test vars
wr_csr_data[0] = $random;
wr_csr_data[1] = $random;
wr_csr_data[2] = $random;

// WR to ICCM SRAM
APB_WR(ICCM_csr_addr,   wr_csr_data[0]);
APB_WR(ICCM_csr_addr+1, wr_csr_data[1]);
APB_WR(ICCM_csr_addr+2, wr_csr_data[2]);

// RD from ICCM SRAM
APB_RD(ICCM_csr_addr, rd_csr_data[0]);
APB_RD(ICCM_csr_addr+1, rd_csr_data[1]);
APB_RD(ICCM_csr_addr+2, rd_csr_data[2]);

// Compare ICCM WR/RD data
if (rd_csr_data[0] !== wr_csr_data[0])
  error();

if (rd_csr_data[1] !== wr_csr_data[1])
  error();
```

```
if (rd_csr_data[2] !== wr_csr_data[2])
  error();


// WR to DCCM SRAM
APB_WR(DCCM_csr_addr,   wr_csr_data[0]);
APB_WR(DCCM_csr_addr+1, wr_csr_data[1]);
APB_WR(DCCM_csr_addr+2, wr_csr_data[2]);

// RD from DCCM SRAM
APB_RD(DCCM_csr_addr, rd_csr_data[0]);
APB_RD(DCCM_csr_addr+1, rd_csr_data[1]);
APB_RD(DCCM_csr_addr+2, rd_csr_data[2]);

// Compare DCCM WR/RD data
if (rd_csr_data[0] !== wr_csr_data[0])
  error();

if (rd_csr_data[1] !== wr_csr_data[1])
  error();

if (rd_csr_data[2] !== wr_csr_data[2])
  error();
```

Refer to “ARCv2 Microcontroller” section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook for a description on ICCM and DCCM register accesses.

3.5 PHYINIT

Refer to “PHY Initialization Sequence” and “DFI Initialization” section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook for details on the PHY initialization. After PHY initialization, DRAM is in self refresh mode. For memory to take the control, assert dfi_init_start and wait for dfi_init_complete to be asserted back from the DDRPHY and issue self refresh exit. The DRAM states after PHY initialization for each DRAM modes are described in DesignWare Cores LPDDR5/4/4X PHY Training Firmware Application Note.

3.6 Memory Model

DDRPHY requires external memory models to send traffic to/from. Memory models can be obtained from Synopsys VIP or downloaded from vendors. The model’s port pins (e.g. ck, ck_n, cke, dq, dqs, dqs_n, etc.) must be connected to the bumps of the DDRPHY following the bump map “PHY Pin List” section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook.

To verify connectivity of the DDRPHY and memory model, refer to section “FirmWare (FW) Testing” on page 65 and Mission Mode Testing.

3.7 Mission Mode Testing

Mission mode testing will verify many of the pin connections. When the DDRPHY is reset, initialized, DRAM initialized and waiting in mission mode, it is ready to send WR/RD transactions to the memory. Sending DRAM traffic through the DDRPHY will verify DFI connections between MCTL/DDRPHY and BP_* connections between the DDRPHY/DRAM. The major buses of interest that will be exercised:

- dfi_reset_n
- dfi[0,1]_address_P{0..3}
- dfi[0,1]_cke_P{0..3}
- dfi[0,1]_cs_ P{0..3}
- dfi_rddata_W{0..3}
- dfi_rddata_cs_n_P{0..3}
- dfi_rddata_dbi_W{0..3}
- dfi_rddata_en_P{0..3}
- dfi_rddata_valid_W{0..3}
- dfi_wrdata_P{0..3}
- dfi_wrdata_cs_n_P{0..3}
- dfi_wrdata_en_P{0..3}
- dfi_wrdata_mask_P{0..3}
- dfi[0,1]_odt_P{0..3}

To drive mission mode traffic, refer to the DFI specification to understand how to drive transactions to the DRAM. Refer to DFI specification sections 4.4 “Write Transactions”, 4.5 “Read Transactions”, 4.6 “Update” and other sections in chapter 4 to understand the protocol to drive transactions to the DRAM.

3.8 FirmWare (FW) Testing

Note This section is under construction.

3.9 Connection Verification

3.9.1 DFI Address/Command and Data Buses

Mission mode transactions will verify DFI address/command/data buses (refer to the section “Mission Mode Testing” on page 64).

3.9.2 DFI Sideband Buses

- DFI controller update req/ack
  - Verify sideband bus dfi[0,1]_ctrlupd_req/dfi[0,1]_ctrlupd_ack.
  - In mission mode, drive dfi0_ctrlupd_req and wait for dfi0_ctrlupd_ack.
- DFI low power req/ack

- Verify sideband bus dfi[0,1]_lp_ctrl_req/dfi[0,1]_lp_ctrl_ack and dfi[0,1]_lp_data_req/dfi[0,1]_lp_data_ack.
- In mission mode, drive dfi0_lp_ctrl_req and wait for dfi0_lp_ctrl_ack.
- In mission mode, drive dfi0_lp_data_req and wait for dfi0_lp_data_ack.
- This will verify that that the DDRPHY will assert the proper acknowledge signal for DFI low power request.

- DFI PHY update req/ack
  - PHY update request is generated by the DDRPHY for delay element calibration.
  - To create a scenario that will drive PHY update request, force an internal RTL variable. Note that this is a workaround to verify PHY update req/ack connectivity and is not a replacement for PHY update request generation.

```
// Power up and initialize
power_up sequence, skip training

// Force phyupd_cnt to 1024 to generate a dfi_phyupd_req
force `TOP.phywrap.pac4a.u_DWC_ddrphy_pub.dfi2acx.phyupd_cnt = 1024;

// Wait for dfi_phyupd_req
wait (`TOP.phywrap.pac4a.dfi0_phyupd_req == 1'b1);
```

- DFI Master req/ack
  - PHY master request is generated by the DDRPHY for DQS drift compensation.
  - To create a scenario that will drive PHY master request, force an internal RTL variable. Note that this is a workaround to verify PHY master req/ack connectivity and is not a replacement for PHY master request generation.

```
// Power up and initialize
power_up sequence, skip training

// Force phyupd_cnt to 1024 to generate a dfi_phyupd_req
force
`TOP.phywrap.pac4a.u_DWC_ddrphy_pub.dfi2acx.dwc_ddrphy_phymasint0.ttr_ctr[27:0
] = 1024;

// Wait for dfi_phyupd_req
wait (`TOP.phywrap.pac4a.dfi0_phymstr_req == 1'b1);
```

- DFI error bus
  - Utilizing the DFI PHY update request scenario, by not driving DFI PHY update ack back to the DDRPHY, the DFI error bus will be driven.

```
// Power up and initialize
power_up sequence, skip training

// Force phyupd_cnt to 1024 to generate a dfi_phyupd_req
force `TOP.phywrap.pac4a.u_DWC_ddrphy_pub.dfi2acx.phyupd_cnt = 1024;

// Wait for dfi_phyupd_req
wait (`TOP.phywrap.pac4a.dfi0_phyupd_req == 1'b1);
```

```
// Memory controller doesn't drive dfi0_phyupd_ack back to DDRPHY IP

// Wait for dfi_error/dfi_error_info bus
wait (`TOP. phywrap.pac4a.dfi0_error == 1'b1 && `TOP.
phywrap.pac4a.dfi0_error_infto == 4'b0000);
```

3.9.3 Frequency Change

Refer to "Active/Mission Mode PHY Power States (PS0 and PS1)" section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook to verify connection of the power management (i.e. frequency change) signals.

3.9.4 DFI Frequency Ratio

To run the DDRPHY at different frequency ratios (1:2, 1:4), program the following CSR and drive the following DFI inputs to specify the ratio between DfiClk and the memory controller (called DfiCtlClk inside the DDRPHY) and the platform MEMCLK:

- DfiFreqRatio[DfiFreqRatio]

  See "DfiFreqRatio_pX" section of DesignWare Cores LPDDR5/4/4X PHY Utility Block (PUB) DataBook for the CSR register definition

- dfi[0,1]_freq_ratio[1:0]:
  - 'b01 = 1:2
  - 'b10 = 1:4

4 GateSim

This chapter is a guide to locating and identifying the files necessary for integrating functional gate level simulations (GLS) into a customer simulation environment and some additional setups in the customer' own testbench. In a release package, gate level netlists are used to represent the digital sections of the hard-macros and behavioral models are provided for the custom analog circuits.

4.1 Running GateSim

1. Locate the files needed and compose a file list
2. Additional test-bench setups for running GLS

   Set RTL defines

   Add additional VCS compile options

   Create timing disable list (only for SDF GateSim)

   Include design file list, i.e. design_flist.f

   Include testbench setups, file list, i.e.tb_flist.f
3. Run VCS

Note

See "VCS Compile Options" on page 73 for details
See "Create Timing Disable List for SDF GateSim" on page 74 for details
See "Additional Recommendations" on page 79 for details
See "Example of Running VCS" on page 74 for details

4.2 Files Location

The following files need to be added to design file list – design_flist.f to run GateSim.

Note The "design_flist.f "should be put into the VCS compile options.

4.2.1 Hard-macro Digital Netlist

```
synopsys/<path_for_master>/Latest/gate_level_netlist/<metal_stack>/dwc_ddrphymaster_top
<_ns|_ew>_<pg|lvs>.v
synopsys/<path_for_diff>/Latest/gate_level_netlist/<metal_stack>/dwc_ddrphydiff_top<_ns
|_ew>_<pg|lvs>.v
synopsys/<path_for_sec>/Latest/gate_level_netlist/<metal_stack>/dwc_ddrphysec_top<_ns|_
ew>_<pg|lvs>.v
synopsys/<path_for_se>/Latest/gate_level_netlist/<metal_stack>/dwc_ddrphyse_top<_ns|_ew
>_<pg|lvs>.v
```

```
<path_for_master> = <product_name>/Latest/<master_ns|master_ew|master>
<path_for_diff> =  <product_name>/Latest/<diff_ns|diff_ew|diff>
<path_for_sec> = <product_name>/Latest/<sec_ns|sec_ew|sec>
<path_for_se> = <product_name>/Latest/<se_ns|se_ew|se>
```

4.2.2 Standard Cell Library Files

```
synopsys/<path_for_diff>/Latest/include/*.mv
synopsys/<path_for_diff>/Latest/include/std_primitives.v
```

Note Whether std_primitives.v is needed depends on the standard cell library file provided by standard cell library team.

4.2.3 Custom Analog Circuit Behavioral Models

```
synopsys/<path_for_master>/Latest/behavior/*.v
synopsys/<path_for_diff>/Latest/behavior/*.v
synopsys/<path_for_sec>/Latest/behavior/*.v
synopsys/<path_for_se>/Latest/behavior/*.v
```

For details about which files should be added into design file list-design_flist.f to run GateSim, refer to “Example File List (design_flist.f)”.

Additionally, some analog circuit behavioral model files with “ew|ns” suffix can be found in the following four locations:

```
synopsys/<path_for_master>/Latest/include/*.v
synopsys/<path_for_diff>/Latest/include/*.v
synopsys/<path_for_sec>/Latest/include/*.v
synopsys/<path_for_se>/Latest/include/*.v
```

4.2.4 PUB Netlist

PUB netlist is generated by customers and should be embedded in the PHY top level netlist (dwc_ddrphy_top.v).

4.2.5 SDF Files

```
synopsys/<path_for_master>/Latest/timing/<metal_stack>/sdf/dwc_ddrphymaster_top<_ew|_ns
>_<pvt_corner>.sdf
synopsys/<path_for_diff>/Latest/timing/<metal_stack>/sdf/dwc_ddrphydiff_top<_ns|_ew>
_<pvt_corner>.sdf
synopsys/<path_for_sec>/Latest/timing/<metal_stack>/sdf/dwc_ddrphysec_top<_ns|_ew>
_<pvt_corner>.sdf
synopsys/<path_for_se>/Latest/timing/<metal_stack>/sdf/dwc_ddrphyse_top<_ns|_ew>
_<pvt_corner>.sdf
```

Note SDF files are not needed to be added into design file list.

4.3 Additional Testbench Setups

The following sections are additional testbench setups to run GateSim

4.3.1 SDF Back Annotation

Add the following lines to the customer's top level testbench file for SDF back annotation.

- SDF back annotation for SS corner

```
// annotate SDF delay to MASTER
$sdf_annotate("<master_sdf_path>/dwc_ddrphymaster_top<_ew|_ns>_<pvt_corner>.sdf,
top.dut.u_DWC_DDRPHYMASTER,, "master_sdf.log", "MAXIMUM");
// annotate SDF delay to AC Wrapper
$sdf_annotate("<diff_sdf_path>/dwc_ddrphydiff_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.DIFF_CK ,, "ac*_diff_ck_sdf.log","MAXIMUM");
$sdf_annotate("<sec_sdf_path>/dwc_ddrphysec_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SEC_0 ,, "ac*_sec0_sdf.log", "MAXIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_0 ,, "ac*_se0_sdf.log", "MAXIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_1 ,, "ac*_se1_sdf.log", "MAXIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_2 ,, "ac*_se2_sdf.log", "MAXIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_3 ,, "ac*_se3_sdf.log", "MAXIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_4 ,, "ac*_se4_sdf.log", "MAXIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_5 ,, "ac*_se5_sdf.log", "MAXIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_6 ,, "ac*_se6_sdf.log", "MAXIMUM");
//If 2 Rank System, the following should be added
$sdf_annotate("<sec_sdf_path>/dwc_ddrphysec_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SEC_1 ,, "ac*_sec1_sdf.log", "MAXIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_7 ,, "ac*_se7_sdf.log", "MAXIMUM");
```

```
//If Dual Channel … (iterate from “0” to “1”), if Single channel only “0” for AC WRAPPER
// annotate SDF delay to DBYTE
$sdf_annotate(“<diff_sdf_path>/dwc_ddrphydiff_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.DIFF_DQS ,, "dbyte*_diff_DQS_sdf.log",“MAXIMUM”);
$sdf_annotate(“<diff_sdf_path>/dwc_ddrphydiff_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.DIFF_WCK ,, "dbyte*_diff_WCK_sdf.log", “MAXIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_0 ,, "dbyte*_se0_sdf.log", “MAXIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_1 ,, "dbyte*_se1_sdf.log", “MAXIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_2 ,, "dbyte*_se2_sdf.log", “MAXIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_3 ,, "dbyte*_se3_sdf.log", “MAXIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_4 ,, "dbyte*_se4_sdf.log", “MAXIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_5 ,, "dbyte*_se5_sdf.log", “MAXIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_6 ,, "dbyte*_se6_sdf.log", “MAXIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_7 ,, "dbyte*_se7_sdf.log", “MAXIMUM”);
… (iterate from “0” to “* minus 1”). * is actual Dbyte number.
//If DMI enabled,the following should be added
$sdf_annotate(“ <se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_DBYTE_WRAPPER_*.SE_8 ,, "dbyte*_se8_sdf.log", “ MAXIMUM”);
```

- SDF back annotation for FF corner

```
// annotate SDF delay to MASTER
$sdf_annotate(“<master_sdf_path>/dwc_ddrphymaster_top<_ns|ew>_<pvt_corner>.sdf,
top.dut.u_DWC_DDRPHYMASTER,, "master_sdf.log", “MINIMUM”);
// annotate SDF delay to AC Wrapper
$sdf_annotate(“<diff_sdf_path>/dwc_ddrphydiff_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.DIFF_CK ,, "ac*_diff_ck_sdf.log",“MINIMUM”);
$sdf_annotate(“<sec_sdf_path>/dwc_ddrphysec_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SEC_0 ,, "ac*_sec0_sdf.log", “MINIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SE_0 ,, "ac*_se0_sdf.log", “MINIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SE_1 ,, "ac*_se1_sdf.log", “MINIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SE_2 ,, "ac*_se2_sdf.log", “MINIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SE_3 ,, "ac*_se3_sdf.log", “MINIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SE_4 ,, "ac*_se4_sdf.log", “MINIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SE_5 ,, "ac*_se5_sdf.log", “MINIMUM”);
$sdf_annotate(“<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SE_6 ,, "ac*_se6_sdf.log", “MINIMUM”);
//If 2 Rannk System, the following should be added
$sdf_annotate(“<sec_sdf_path>/dwc_ddrphysec_top<_ns|_ew> _<pvt_corner>.sdf”,
top.dut.u_AC_WRAPPER_*.SEC_1 ,, "ac*_sec1_sdf.log", “MINIMUM”);
```

```
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_AC_WRAPPER_*.SE_7 ,, "ac*_se7_sdf.log", "MINIMUM");
//If Dual Channel … (iterate from "0" to "1"), if Single channel only "0" for AC WRAPPER
// annotate SDF delay to DBYTE
$sdf_annotate("<diff_sdf_path>/dwc_ddrphydiff_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.DIFF_DQS ,, "dbyte*_diff_DQS_sdf.log","MINIMUM");
$sdf_annotate("<diff_sdf_path>/dwc_ddrphydiff_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.DIFF_WCK ,, "dbyte*_diff_WCK_sdf.log", "MINIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_0 ,, "dbyte*_se0_sdf.log", "MINIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_1 ,, "dbyte*_se1_sdf.log", "MINIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_2 ,, "dbyte*_se2_sdf.log", "MINIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_3 ,, "dbyte*_se3_sdf.log", "MINIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_4 ,, "dbyte*_se4_sdf.log", "MINIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_5 ,, "dbyte*_se5_sdf.log", "MINIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_6 ,, "dbyte*_se6_sdf.log", "MINIMUM");
$sdf_annotate("<se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_7 ,, "dbyte*_se7_sdf.log", "MINIMUM");
    … (iterate from "0" to "* minus 1"). * is actual Dbyte number.
//If DMI enabled,the following should be added
$sdf_annotate(" <se_sdf_path>/dwc_ddrphyse_top<_ns|_ew> _<pvt_corner>.sdf",
top.dut.u_DBYTE_WRAPPER_*.SE_8 ,, "dbyte*_se8_sdf.log", " MINIMUM");
```

4.3.2 VCS Compile Options

- VCS options for SDF GateSim

1. Set RTL defines:

   +define+DWC_DDRPHY_PG_PINS

   +define+DWC_DDRPHY_TOP_PG_PINS

2. Add/Remove some options for SDF GateSim

```
...
   ... (project settings/license..etc)
   ....
  #+nospecify \ (this option should be excluded when running SDF GateSim)
  #+notimingchecks \ (this option should be excluded when running SDF GateSim)
  +maxdelays \ (for ss corner)
  +mindelays \ (for ff corner)
  +transport_path_delays \
  +pulse_e/0 +pulse_r/0 \
  +transport_int_delays\
  +pulse_int_e/0 +pulse_int_r/0 \
  +neg_tchk -negdelay \
  $*"
```

- VCS options for GateSim without SDF

1. 1. Set RTL defines:

```
+define+DWC_DDRPHY_PG_PINS
+define+DWC_DDRPHY_TOP_PG_PINS
```

2. Add/Remove some options for GateSim without SDF

```
 ...
 ... (project settings/license..etc)
 ....
+nospecify \ (this option should be excluded when running SDF GateSim)
+notimingchecks \ (this option should be excluded when running SDF GateSim)
```

Note

When running SDF GateSim and GateSim without SDF, some custom analog circuit behavioral models, such as PLL, can work only by driving power ports.

Pick up the netlists with power ports and drive the power ports through testbench.

The define macro “DWC_DDRPHY_TOP_PG_PINS” and “DWC_DDRPHY _PG_PINS” is aimed to add power ports to dwc_ddrphy_top/dwc_ddrphy_ac_wrapper/dwc_ddrphy_dbyte_wrapper when doing GateSim with RTL design of dwc_ddrphy_top/dwc_ddrphy_ac_wrapper/dwc_ddrphy_dbyte_wrapper and pass the power pins to sub-modules.

4.3.3 Example of Running VCS

`$cmd_vcs` is a wrapper of vcs to include all necessary license key, vcs compile option, etc.

To compile models for running functional tests for GLS, example:

```
$cmd_vcs -f design_flist.f -f tb_flist.f
```

4.3.4 Create Timing Disable List for SDF GateSim

If customer encounter timing issues inside PHY during SOC level SDF GateSim and require a timing disable list, here is the instruction about how to create it:

1. Find out all of the synchronizers

   a. Find out all of the synchronizers in PHY hard-macros. The list shown as follows:

```
//synchronizers in Master
u_DWC_DDRPHYMASTER_top/zSync6_DLY500PS_PllLock
u_DWC_DDRPHYMASTER_top/zSync6_DLY156PS_PtrInit_sync
u_DWC_DDRPHYMASTER_top/zSync6_DLY3000PS_csrPORMemReset
u_DWC_DDRPHYMASTER_top/zSync6_DLY500PS_ZCalCompOut_cdc
//synchronizers in AC wrapper
u_AC_WRAPPER_*/DIFF_CK/zSync6_DLY500PS_cal_clk_en_sync
u_AC_WRAPPER_*/DIFF_CK/zSync6_DLY500PS_repl_cal_clk_en_sync
u_AC_WRAPPER_*/DIFF_CK/RingOsc_lcdl/zSync6_DLY500PS_RO_run_sync
u_AC_WRAPPER_*/SEC_0/zSync6_DLY500PS_cal_clk_en_sync
u_AC_WRAPPER_*/SEC_0/RingOsc_lcdl/zSync6_DLY500PS_RO_run_sync
u_AC_WRAPPER_*/SE_*/zSync6_DLY500PS_cal_clk_en_sync
```

```
u_AC_WRAPPER_*/SE_*/RingOsc_lcdl/zSync6_DLY500PS_RO_run_sync
… (iterate from “0” to “6” for SE)
//SEC1, SE7 if RANK system is 2
u_AC_WRAPPER_*/SEC_1/zSync6_DLY500PS_cal_clk_en_sync
u_AC_WRAPPER_*/SEC_1/RingOsc_lcdl/zSync6_DLY500PS_RO_run_sync
u_AC_WRAPPER_*/SE_7/zSync6_DLY500PS_cal_clk_en_sync
u_AC_WRAPPER_*/SE_7/RingOsc_lcdl/zSync6_DLY500PS_RO_run_sync
… (iterate from “0” to “1” for AC WRAPPER  if Dual Channel, only “0” for AC WRAPPER  if
Single channel)
//synchronizers in Dbyte wrapper
u_DBYTE_WRAPPER_*/DIFF_DQS/zSync6_DLY500PS_cal_clk_en_sync
u_DBYTE_WRAPPER_*/DIFF_DQS/zSync6_DLY500PS_repl_cal_clk_en_sync
u_DBYTE_WRAPPER_*/DIFF_DQS/RingOsc_lcdl/zSync6_DLY500PS_RO_run_sync
u_DBYTE_WRAPPER_*/DIFF_WCK/zSync6_DLY500PS_cal_clk_en_sync
u_DBYTE_WRAPPER_*/DIFF_WCK/zSync6_DLY500PS_repl_cal_clk_en_sync
u_DBYTE_WRAPPER_*/DIFF_WCK/RingOsc_lcdl/zSync6_DLY500PS_RO_run_sync
u_DBYTE_WRAPPER_*/SE_*/zSync6_DLY500PS_cal_clk_en_sync
u_DBYTE_WRAPPER_*/SE_*/RingOsc_lcdl/zSync6_DLY500PS_RO_run_sync
… (iterate from "0" to * for SE, if DMI not enabled, * is 7, or * is 8)
… (iterate from “0” to “Actual Dbyte Number - 1” for DBYTE WRAPPER)
```

b. Find out all of the synchronizers in PUB. Following is an example, double check by users

The list can be formed by grep “zSync”in the tcl files in the release package, every line containing “zSync” should be extracted out: “synopsys/<product_name>/Latest/macro/Latest/constraints/” (Note: If the PHY release package includes an installed_coreKit component, the path should be synopsys/<product_name>/Latest/example/Latest/constraints/)

Here is an example of a few lines of the list:

```
set my_zSync5_list [list
u_DWC_ddrphy_pub/MRTUB/dwc_ddrphy_apb2cfg/dwc_ddrphy_apbfifo/u_fifo/zSync5_DLY500PS_write
_gc_dfi/DataIn[7] \

u_DWC_ddrphy_pub/MRTUB/dwc_ddrphy_apb2cfg/dwc_ddrphy_apbfifo/u_fifo/zSync5_DLY500PS_write
_gc_dfi/DataIn[6] \

u_DWC_ddrphy_pub/MRTUB/dwc_ddrphy_apb2cfg/dwc_ddrphy_apbfifo/u_fifo/zSync5_DLY500PS_write
_gc_dfi/DataIn[5] \

u_DWC_ddrphy_pub/MRTUB/dwc_ddrphy_apb2cfg/dwc_ddrphy_apbfifo/u_fifo/zSync5_DLY500PS_write
_gc_dfi/DataIn[4] \

u_DWC_ddrphy_pub/MRTUB/dwc_ddrphy_apb2cfg/dwc_ddrphy_apbfifo/u_fifo/zSync5_DLY500PS_write
_gc_dfi/DataIn[3] \

u_DWC_ddrphy_pub/MRTUB/dwc_ddrphy_apb2cfg/dwc_ddrphy_apbfifo/u_fifo/zSync5_DLY500PS_write
_gc_dfi/DataIn[2] \

u_DWC_ddrphy_pub/MRTUB/dwc_ddrphy_apb2cfg/dwc_ddrphy_apbfifo/u_fifo/zSync5_DLY500PS_write
_gc_dfi/DataIn[1] \

u_DWC_ddrphy_pub/MRTUB/dwc_ddrphy_apb2cfg/dwc_ddrphy_apbfifo/u_fifo/zSync5_DLY500PS_write
_gc_dfi/DataIn[0] ]
```

For all the synchronizers listed in upper steps, find out the name of the first stage flip-flop in each of them, and then add them to the following always block.

```
initial begin
```

```
`ifdef DWC_DDRPHY_GATESIM_SDF
// disable timing checks of some flops during customer's expected simulation duration.
->force_notifiers;
`endif
end
always@(force_notifiers)
begin
force <customer_path>.u_DWC_DDRPHYMASTER_top.<zSync6_DLY500PS_PllLock_Stage1>.<NOTIFIER>
= 1'b0;
$disable_warnings("timing",
<customer_path>.u_DWC_DDRPHYMASTER_top.<zSync6_DLY500PS_PllLock
_Stage1>);
… (do the same thing for the rest of the flops on the list)
end
```

2. Some flops should only be disabled during training. Since training algorithm will sweep the clock to sample the data in order to find an optimal location, and timing violation happens when sweeping into the setup/hold timing window on certain flops.

   a. The notifier of the following flops can be disabled during max read latency training stage. Customer can add the testbench code:

```
 initial begin
   wait(<customer_path>.u_DWC_ddrphy_pub.csrResetToMicro===1'b1);
   forever begin
     @(negedge <customer_path>.u_DWC_ddrphy_pub.csrStallToMicro);
     fork
       begin
         wait(<customer_path>.u_DWC_ddrphy_pub.dx_*.csrDFIMRL[4:0]==0);
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_0.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b0.viol_0=1'b0;
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_0.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b0.viol_0=1'b0;
             ...(iterate from RxDatLn0 bit_0 to bit_7)
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_0.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b7.viol_0=1'b0;
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_0.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b7.viol_0=1'b0;
         ...(iterate from SE_0 to SE_7)
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_7.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b0.viol_0=1'b0;
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_7.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b0.viol_0=1'b0;
             ...(iterate from RxDatLn0 bit_0 to bit_7)
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_7.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b7.viol_0=1'b0;
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_7.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b7.viol_0=1'b0;

         //If DMI enabled, the following should be added
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_8.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b0.viol_0=1'b0;
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_8.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b0.viol_0=1'b0;
             ...(iterate from RxDatLn0 bit_0 to bit_7)
         force
<customer_path>.u_DBYTE_WRAPPER_*.SE_8.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b7.viol_0=1'b0;
```

```
            force
<customer_path>.u_DBYTE_WRAPPER_*.SE_8.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b7.viol_0=1'b0;
          end
         begin
          ...(iterate from Dbyte "0" to "* minus 1"). * is Dbyte number
         end
        join
        @(posedge <customer_path>.u_DWC_ddrphy_pub.csrStallToMicro);
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_0.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b0.viol_0;
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_0.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b0.viol_0;
             ...(iterate from RxDatLn0 bit_0 to bit_7)
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_0.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b7.viol_0;
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_0.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b7.viol_0;

        ...(iterate from SE_0 to SE_7)
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_7.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b0.viol_0;
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_7.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b0.viol_0;
             ...(iterate from RxDatLn0 bit_0 to bit_7)
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_7.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b7.viol_0;
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_7.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b7.viol_0;

        //If DMI enabled, the following should be added
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_8.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b0.viol_0;
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_8.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b0.viol_0;
             ...(iterate from RxDatLn0 bit_0 to bit_7)
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_8.RXDATAFIFO.DFE0_RXDATA.I_RxDatLn0_b7.viol_0;
        release
<customer_path>.u_DBYTE_WRAPPER_*.SE_8.RXDATAFIFO.DFE1_RXDATA.I_RxDatLn0_b7.viol_0;

        ...(iterate from Dbyte "0" to "* minus 1"). * is Dbyte number
      end
    end
```

Note Above is an example when the flops' name in netlist is I_RxDatLn0_b*, actually the flops' name may be I_RxDatLn0_b* or RxDatLn0_reg_*_ or some other names (the asterisk represents the bit number). Customers could find out the correct name of the flops in netlist corresponding to the RTL path: <customer_path>.u_DBYTE_WRAPPER_*.SE_*.RXDATAFIFO.DFE*_RXDATA.RxDatLn0[7:0].

Note This section will keep on updating.

3. Some flops should only be disabled during ATE FW stage:

   a. When doing AC loopback, the notifier of following Flip-Flops can be disabled. Customer can add the following testbench code:

```
initial
begin
force <customer_path>.u_AC_WRAPPER_*.SEC_0.RxDataPD_RxStrb_reg.viol_0 = 1'b0;
force <customer_path>.u_AC_WRAPPER_*.DIFF_CK.RxCoreSampRxEn_c_reg.viol_0 =
1'b0;
force <customer_path>.u_AC_WRAPPER_*.DIFF_CK.RxCoreSampRxEn_t_reg.viol_0 =
1'b0;
//If 2 Rank System, the following should be added
force <customer_path>.u_AC_WRAPPER_*.SEC_1.RxDataPD_RxStrb_reg.viol_0 = 1'b0;

… (iterate from "0" to "1" for AC WRAPPER if Dual Channel, only "0" for AC
WRAPPER if Single channel)
end
```

   b. When doing Data loopback, the notifier of following Flip-Flops can be disabled. Customer can add the following testbench code:

```
initial
begin
force <customer_path>.u_DBYTE_WRAPPER_*.DIFF_DQS.RxCoreSampRxEn_c_reg.viol_0 =
1'b0;
force <customer_path>.u_DBYTE_WRAPPER_*.DIFF_DQS.RxCoreSampRxEn_t_reg.viol_0 =
1'b0;
//If LPDDR5 Enabled, the following lines should be added
force
<customer_path>.u_DBYTE_WRAPPER_*.DIFF_WCK.RxCoreSampRxEn_c_reg.viol_0 = 1'b0;
force <customer_path>.u_DBYTE_WRAPPER_*.DIFF_WCK.RxCoreSampRxEn_t_reg.viol_0 =
1'b0;

… (iterate from "0" to "* minus 1"). * is actual Dbyte number.
end
```

4. Notifications:

   a. "<customer_path>" should be replaced with the actual instance path accordingly.

   b. Each "zSync6/zSync5/zSync4/zSync3"cell contains 6/5/4/3 flip-flops, and only the first FF should be timing disabled.The "<*_Stage1>" above is just an example for illustration, customer needs to replace with the actual name.

   c. If there is an asterisk "*" or <> containing in the cell's name, expand it to the full iteration (maybe reported by PT), and add every instance/flop to the always block.

   d. The word "<NOTIFIER>" may need to be changed to some other signal name, which is subject to the actual notifier port name of the DFF module chosen by customer.

   e. For Hard-macro synchronizers, if there is no specific notifier for the first stage flop, then disable the whole synchronizer flops. Hard-macro netlist is fully verified internally, and the risk is low for disabling timing check on all the stages of CDC flops.

   f. All of the code above should be added to customer's top level testbench in order to run SDF Gatesim.

   g. After creating the timing disable list, all other timing violations reported are "don't care" as long as they do not cause the simulation to fail and STA check is cleaned.

4.3.5 Additional Recommendations

1. When running Functional SDF GateSim and Functional GateSim without SDF, "+define+VIRL_functiononly" is NOT required and NOT recommended. However, there is a chance that other library teams recommend adding it for some reason. Thus, it can only be added when running Functional GateSim without SDF and if it is added, then

   `"+define+SNPS_COMBO_DELAY=0 +define+SNPS_CKGTLT_DELAY=0 +define+SNPS_SEQ_DELAY=0`

   `+define+SNPS_LAT_DELAY=0"` must be set together with it. Also, do not put any extra delay settings in vcs command line or in standard cell Verilog file.

2. When running SDF GateSim and GateSim without SDF, do not add additional "#" delay inside the RTL behavioral models for any analog circuits. (The list of analog circuits could be found in the following page).

3. When running SDF GateSim, all of the analog circuits inside the hard IPs (DIFF, SEC, SE, MATSER) should NOT be SDF back annotated. They should only use the RTL behavioral model. In order to achieve it:

   a. Check the LCDL module name (dwc_ddrphy_lcdl | dwc_ddrphy_lcdl_ew | dwc_ddrphy_lcdl_ns) both in netlist and SDF file, and make sure

      - There are no CELLs of LCDL in the SDF file.
      - If there are CELLs of LCDL in the SDF file, make sure LCDL module name (`dwc_ddrphy_lcdl | dwc_ddrphy_lcdl_ew | dwc_ddrphy_lcdl_ns`) both in netlist and SDF file are NOT identical. If they are the same in both files, either of them should be changed to a different name.

   The intention of doing this is to make sure the LCDL module will NOT be annotated by SDF delay.

   Besides, it is only necessary to do this check for LCDL.

   b. Waive and ignore all the SDF warnings generated by the analog circuits. Those warnings can be ignored because the analog parts should NOT be annotated by SDF delay.

      Some examples are shown below:

      Warning-[SDFCOM_PONF] Port not found

      …

      module: dwc_ddrphy_pll, instance:

      …

      SDF Error: Input port clk_intb specified in SDF file using IOPATH does not exist. If this warning is not expected, check if bit select is escaped with a backslash in the SDF.

      Warning-[SDFCOM_IANE] IOPATH Annotation Not Enabled,

      …

      module: dwc_ddrphy_por

      …

      SDF Warning: Cannot find matching IOPATH in the corresponding instance And module.

      Make sure that specify block exists and IOPATH annotation is Not disabled in configuration file.

Warning-[SDFCOM_II] Mismatched Instance CELLTYPE

SDF Error: Instance dwc_ddrphy_cmpmeas.txzqint is of Type

"dwc_ddrphy_txrxdq", expecting "dwc_ddrphy_txrxdq_ns". Ignoring all delays.

4. The full list of analog circuits can be found below:

```
synopsys/<path_for_master>/Latest/behavior/*.v
synopsys/<path_for_diff>/Latest/behavior/*.v
synopsys/<path_for_sec>/Latest/behavior/*.v
synopsys/<path_for_se>/Latest/behavior/*.v
```

5. When running SDF GateSim and GateSim without SDF, there are some non-resettable registers in PUB netlist, which are expected to not creating problem in real silicon and can work normally in RTL sim, but may generate X in netlist GateSim.

   Non-resettable registers in PUB netlist need to be initialized by the following settings (this is just an example, customers may need to change these options if different VCS version is used):

   a. Set ‘setenv VCS_PRINT_INITREG_INITIALIZATION 1':
   it enables printing of all initialized registers and their initialized values to a file named “vcs_initreg_random_value.txt”.

   b. Create a configuration file named “ddrphy_initreg_config”, and add the following setting in the file: modtree dwc_ddrphy_top 0 0

      Note: The module name of “dwc_ddrphy_top” should be consistent with the top module name of DDRPHY in the design of SOC.

   c. Add the following switch to the VCS compile options:

      +vcs+initreg+config+{$PATH}/ddrphy_initreg_config

   d. Add following switch to the VCS runtime options (running simv):

      +vcs+initreg+config+{$PATH}/ddrphy_initreg_config

      Note:

      1. The {$PATH} is the directory where the configuration file of “ddrphy_initreg_config” generated in step b is located.

      2. The steps (b,c,d) are aimed to enable the initialization of registers just in the DDRPHY instead of the whole design of SOC.

   e. When step (a,b,c,d) are finished and the simulation is actually started, check if there is a text named “vcs_initreg_random_value.txt”(it contains a bunch of registers together with their initialized value for the selective parts of design) automatically generated in the vcs simulation directory. If not, it might be VCS tool version issue, please check with VCS team or user guide.

6. If customer wants to run GateSim for mission mode only, customer needs to set the `PHY_tDQSDQ` to 'd111 additionally by programming the PHYINIT UserInputSim

   `PHY_tDQS2DQ`, as well as set the RxClkTrackEn[pstate] to 0 by programming the PHYINIT UserInputAdvanced RxClkTrackEn[pstate], given that path delay from DQS to DQ in the PHY has been removed within the gate level netlist.

The setting 'd111 of PHY_tDQSDQ is a typical value, it may be tuned in SDF gate level simulation.

4.3.6 Example of File List (design_file.f)

```
// top level RTL file
//(refer to Exception #10,11)
//synopsys/<product_name>/Latest/macro/Latest/rtl/dwc_ddrphy_top.v
//synopsys/<product_name>/Latest/macro/Latest/rtl/dwc_ddrphy_ac_wrapper.v
//synopsys/<product_name>/Latest/macro/Latest/rtl/dwc_ddrphy_dbyte_wrapper.v

//SRAM file
//customer need to add their own SRAM files

//Standard cell library files
synopsys/<product_name>/Latest/master/Latest/include/*.mv (refer to Exception #1)
//customer need to add the standard cell library files for PHY top netlist or SRAM

//Netlist needed by GateSim
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/gate_level_netlist/
<metal_stack>/dwc_ddrphymaster_top<_ns|_ew>_<pg|lvs>.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/gate_level_netlist/<metal_
stack>/
dwc_ddrphydiff_top<_ns|_ew>_<pg|lvs>.v
synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/gate_level_netlist/<metal_sta
ck>/
dwc_ddrphysec_top<_ns|_ew>_<pg|lvs>.v
synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/gate_level_netlist/<metal_stack>
/
dwc_ddrphyse_top<_ns|_ew>_<pg|lvs>.v

//"DIFF" files
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_cmdfifo.v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_dftclkmux
.v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphydiff_top.v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_dlytest.v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_dqssamp_f
ifo.v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_lcdl_wrap
per.v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_mtestmux.
v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_rsm_fifo.
v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_txfifo.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_cdc_syn_std
lib.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/rtl/dwc_ddrphy_rtl_syn_std
lib.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_diff_i
o.v
-v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/include/dwc_ddrphy_diff_io
_ns.v
```

```
-v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/include/dwc_ddrphy_diff_io
_ew.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_diff_r
x_base.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_diff_r
x_core.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_diff_r
x.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_drvx.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_lcdl.v
//(refer to Exception #2)
-v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/include/dwc_ddrphy_lcdl_ns
.v
-v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/include/dwc_ddrphy_lcdl_ew
.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_pclk_r
x.v
//(refer to Exception #3)
//-v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_pclk_r
x_ns.v
//-v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_pclk_r
x_ew.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_phase_
detect.v
//synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_rxre
plica.v
//(refer to Exception #4)
-v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/include/dwc_ddrphy_rxrepli
ca_ns.v
-v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/include/dwc_ddrphy_rxrepli
ca_ew.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_rxrepl
ica_gated_se2diff.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_rxrepl
ica_phase_detector.v
//(refer to Exception #5)
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_txbe.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_txfedi
ff.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_txfe.v
synopsys/<product_name>/Latest/<diff_ns|diff_ew|diff>/Latest/behavior/dwc_ddrphy_vrefto
p.v
//"SEC" files
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphy_cdc_syn_stdl
ib.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphy_cmdfifo.v
```

```
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphy_dftclkmux.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphy_dlytest.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphy_lcdl_wrapper
.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphy_mtestmux.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphy_rtl_syn_stdl
ib.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphysec_top.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/rtl/dwc_ddrphy_txfifo.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_cdc_syn
_stdlib.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_rtl_syn
_stdlib.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_drvx.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_lcdl.v
//synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_pclk_rx
.v
synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_sec_io.v
-v
synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/include/dwc_ddrphy_sec_io_ew.
v
-v
synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/include/dwc_ddrphy_sec_io_ns.
v
synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_sec_rx.v
synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_txbesec.v
synopsys/<product_name>/Latest/<sec_ns|sec_ew|sec>/Latest/behavior/dwc_ddrphy_txfesec.v
//"SE" files
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_cdc_syn_stdlib.
v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_cmdfifo.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_data_fifo_rx.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_dftclkmux.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_dlytest.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_lcdl_rxclk_wrap
per.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_lcdl_wrapper.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_mtestmux.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_rtl_syn_stdlib.
v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_rxdatfifo.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphyse_top.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/rtl/dwc_ddrphy_txfifo.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_cdc_syn_st
dlib.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_cdc_syn_st
dlib.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_drvx.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_lcdl.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_pclk_rx.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_vreftop.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_txbe.v
//synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_txfe.v
```

```
synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_se_io.v
-v synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/include/dwc_ddrphy_se_io_ns.v
-v synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/include/dwc_ddrphy_se_io_ew.v
synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_se_rx_base.v
synopsys/<product_name>/Latest/<se_ns|se_ew|se>/Latest/behavior/dwc_ddrphy_se_rx.v

//"MASTER" files
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/rtl/dwc_ddrphymast
er_top.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/rtl/dwc_ddrphy_cmd
fifo.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/rtl/dwc_ddrphy_mas
ter_pclk_mux.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/rtl/dwc_ddrphy_and
2_asst_clk.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/rtl/dwc_ddrphy_mte
stmux.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/rtl/dwc_ddrphy_rtl
_syn_stdlib.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/rtl/dwc_ddrphy_cdc
_syn_stdlib.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/rtl/dwc_ddrphy_and
2_asst_clk.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
pclk_master.v
//-v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/include/dwc_ddrphy_p
clk_master_ns.v(refer to Exception #6)
//-v
synopsys/<product_name>/Latest/master/Latest/include/dwc_ddrphy_pclk_master_ew.v(refer
to Exception #6)
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
por.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
scdlib_sdffr.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
techrevision.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
vrefdacref.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana_calmux.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana_cmpr.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana_cmpr_bias.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana_cmpr_pasa.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana_dac.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana_foldedcascode.v
```

```
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana_rcfilt.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrphy_
zcalana_sampler.v
synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_lpddr5p
hy_pll_ns.v(refer to Exception #7,8)
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_vreftop.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_drvx.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_sec_io.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_sec_rx.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_txbesec.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_txbe.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_txfesec.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_txfe.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_se_io.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_se_rx_base.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_se_rx.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_rtl_syn_stdlib.v
//synopsys/<product_name>/Latest/<master_ns|master_ew|master>/Latest/behavior/dwc_ddrph
y_cdc_syn_stdlib.v
```

4.4 Exceptions

In this section we present some specific scenarios and exceptions.

1. For some release, it may need std_primitives.v additionally, add synopsys/<product_name>/Latest/diff<_ew|_ns>/Latest/include/std_primitives.v to the design_file.f as well.
2. For some release, it may need dwc_ddrphy_lcdl_<ew|ns>. In this case, the files will be placed into synopsys/<product_name>/Latest/diff<_ew|_ns>/Latest/include, add these files to the design_file.f as well.
3. For some technology, it may need dwc_ddrphy_pclk_rx_<ns|ew>.v. In this case, the files will be placed into synopsys/<product_name>/Latest/diff<_ew|_ns>/Latest/include, add these files to the design_file.f as well.
4. For some release, it may need dwc_ddrphy_rxreplica_<ew|ns>.v. In this case, the files will be Placed into synopsys/<product_name>/Latest/diff<_ew|_ns>/Latest/include, add these files to the design_file.f as well.
5. The two files dwc_ddrphy_rxreplica_deglitch.v and dwc_ddrphy_se2diff.v are deleted, whereas dwc_ddrphy_rxreplica_phase_detector.v and dwc_ddrphy_rxreplica_gated_se2diff.v are added in all hard-macro B releases starting from rev0_74a. For a project version corresponding to the hard-macro type contact Synopsys.
6. For some technology, it may need dwc_ddrphy_pclk_master_<ns|ew>.v. In this case, the files will be placed into synopsys/<product_name>/Latest/master<_ew|_ns>/Latest/include, add these files to the design_file.f as well.
7. For PLL core, use dwc_lpddr5phy_pll_ns.v
8. It is recommended to put “dwc_lpddr5phy_pll_ns.v” to the end of the filelist in case of potential VCS tool issue about timescale.
9. The modules in the commented out files have already been included in netlist DIFF/SEC/SE/MASTER) or lost been in other behavior directories.
10. The top level RTL file `"dwc_ddrphy_top.v"` and the wrapper files `"dwc_ddrphy_ac_wrapper.v"` and `"dwc_ddrphy_dbyte_wrapper.v"`are also commented out and should use netlist version instead of the version generated by the customer. If the customer use the RTL wrapper, update the name of instantiation for DIFF/SEC/SE with orientation respectively.
11. In some release packages associate with CoreTool flow, there is no macro directory any more. User can find dwc_ddrphy_top.v,dwc_ddrphy_ac_wrapper.v,dwc_ddrphy_dbyte_wrapper.v in <WORKSPACE>/src .

4.5 Troubleshooting and Support

This section provides troubleshooting tips to help customer with some common issues.

Q1: When running GateSim, it happens that simulation runs a few days and cannot finish.

A: The following steps are helpful in order to do a quick check:

1. Remove the dump options and see if it can finish;
2. Only dump the first part of the waveform (for example, kill the sim after a few hours' run), and check the following signals:

   - Check if VDD/VDDQ/VAA are successfully driven to 1 in the waveform. If they are HiZ, that would make analog macros not working.

   - Check if any ICCM/DCCM signals become X an cannot recover.
3. Add vcs option +vcs+loopreport

   - Check loop-info-verbose-*.log to investigate whether there are zero-loop condition in environment/VIPs
4. Check whether the job is killed
5. Check whether exactly following the GateSim Application Note

Q2: **GateSim takes days to finish, how to reduce runtime?**

A: The following steps are helpful in order to reduce runtime

1. Remove the dump option
2. If dump is needed, just dump a specific hierarchy or dump out a particular period of time (`$time_finish – $time_start`).
3. Adjust "HdtCtrl" for different verbosity of streaming message.
4. Remove –debug all, -debug_pp
5. Use save&restart checkpoint VCS settings

Q3: The dump file is too large to upload, how to split it into separate files?

A:

1. vcsplit –full64 -o split.vpd -scope tb.dut.phy_top -min {<overall_simulation_time> – 200000 ns} -max <overall_simulation_time> dump.vpd
2. fsdbextract verilog.fsdb -bt time -et time -o result.fsdb. Also, the user can use the following command to get more guidance: fsdbextract -h

Q4: What to do if these warnings are presented?

SDF Warning- [SDFCOM_SWC] Simple Wire Connection

SDF Warning- [SDFCOM_IWSBA] INTERCONNECT will still be annotated

A: These two type of warnings can be waived.

Q5: Whether customer can disable the whole synchronizer cell in this way:

In a config file, write as below:

instance `{*/u_DWC_DDRPHYMASTER_top/zSync6_DLY500PS_PllLock} {noIopath, noSpecify, noTiming };`

Then during run vcs, include the config file using `"+optconfigfile+"` like below:

```
% vcs verilog_files +optconfigfile+configuration_filename
```

A: It is not recommend. However, if customer insist doing this way. For synchronizers in hard-macro only (Master/SEC/SE/DIFF), customer can do it just with "`noTiming`".