Avant DDR Memory PHY Module - Lattice Radiant Software

User Guide

FPGA-IPUG-02195-1.0

November 2022

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults and associated risk the responsibility entirely of the Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
DDR	Double Data Rate
FPGA	Field Programmable Gate Array
LPDDR	Low-Power Double Data Rate
SCL	Self-Calibrating Logic
ABC	Adaptive Bit Calibration

1. Introduction

The Lattice Semiconductor DDR Memory PHY (DDRPHY) Module is an implementation of the DFI version 4.0 specification (© Cadence Design Systems, Inc.) that describes the inter-operation between a DDR memory controller and the physical interface (PHY). The DDR Memory PHY Module supports interfacing to LPDDR4 memories. It implements the complete LPDDR4 training as well as the associated clocking scheme.

1.1. Quick Facts

Table 1.1 presents a summary of DDR Memory PHY Module.

Table 1.1. Quick Facts

IP Requirements	Supported FPGA Family	Lattice Avant
Resource Utilization	Targeted Devices	LAV-AT-500E
	Supported User Interface	APB, DFI
	Resources	See Appendix A. Resource Utilization Section
Design Tool Support	Lattice Implementation	Lattice Radiant™ software 2022.1
	Synthesis	Synopsys® Synplify Pro® for Lattice
	Simulation	For a list of supported simulators, see the Lattice Radiant Software user guide.

1.2. Features

Key features of DDR Memory PHY Module include:

- Supports LPDDR4 (JESD209-4C) and DDR4 (JESD79-4C)
- Frequency Supported: 266, 300, 350, 400, 533, 666, 800, 933 MHz
- Supports 8:1 gearing ratio (1:4 frequency ratio)
- DDR widths of x16, x32, and x64
  - If Side-band ECC is enabled for DDR4, the DDR widths x40 and x72 are also supported.
- Training and initialization support
  - Automatic SDRAM initialization
  - Command Bus Training (CS, CA, CA_VREF) (only LPDDR4)
  - Write Leveling
  - Read Bit Leveling
  - Write Bit Leveling
  - Self Calibrating Logic
- Internal programmable Vref
- Includes PLL for clock generation

1.3. Conventions

1.3.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.3.2. Signal Names

Signal names that end with:

- *_n* are active low
- *_i* are input signals
- *_o* are output signals
- *_io* are bi-directional input/output signals

2. Functional Description

2.1. Overview

The DDRPHY Module instantiates a fully verified, fully validated, hard core PHY from Uniquify, which is an implementation of DFI 4.0 Standard. It MC to PHY implements frequency ratio = 1:2 (4:1 Gearing Ratio), this is converted to frequency Ratio 1:4 (8:1 Gearing Ratio) by adding 2:1, 1:2 Gearing Converter on the DFI side as shown Figure 2.1. The CDC FIFO ensures signal integrity when transferring data to/from FPGA fabric. The DDR-side signals are connected to Lattice high performance IO (HPIO) to allow reuse of the FPGA IO for other function when DDRPHY is not in used. All the logic inside the DDRPHY primitive is hardened, there is also a dedicated PLL and quarter_rate_clk generator for DDRPHY which utilize a dedicated clock routing to enable high frequency operation.

The Training CPU reads instruction from System Memory and execute the initialization and training routines by accessing the CSR of the Uniquify PHY, PLL and DDRPHY Module.

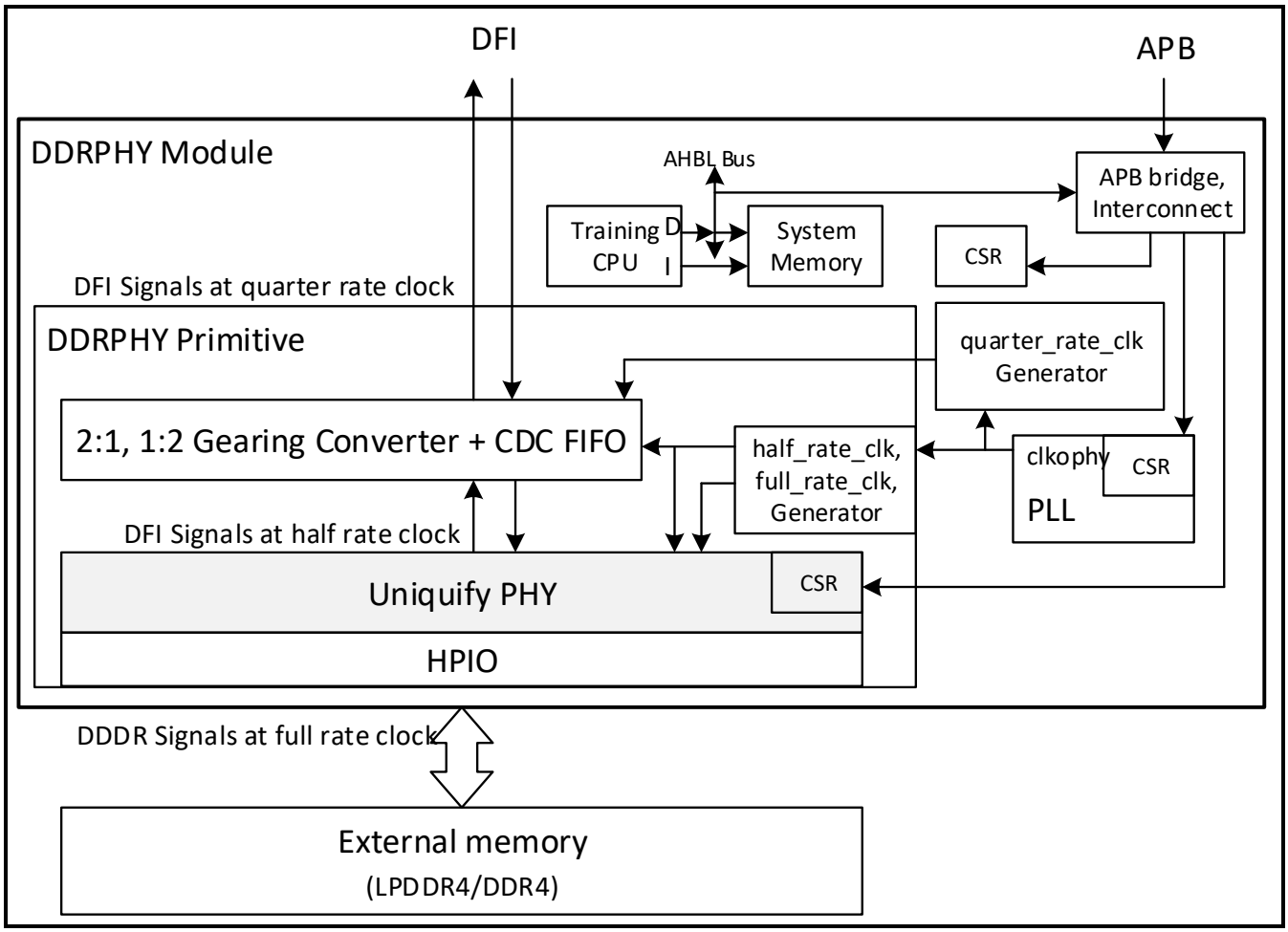

Figure 2.1. DDRPHY Module Block Diagram

2.2. Signal Description

Table 2.1 describes the LPDDR4 Memory PHY Module signals.

Table 2.1. DDR Memory PHY Module Signal Description

Pin Name	Direction	Width (Bits)[1]	Description
Clock and Reset			
pll_refclk_i	In	1	Reference clock input for PLL. The IO type of this signal is the same as the DDR memory signals.
pll_rst_n_i	In	1	Reset pin of the PLL. The user need assert this when the PLL loss lock.
pll_lock_o	Out	1	Indicates PLL lock status. This is an asynchronous signal.
rst_i	In	1	Active HIGH asynchronous reset signal. This signal is synchronized internally to the IP's internal clocks. The minimum pulse width is 2x the period of pclk_i.
pclk_i	In	1	Input clock for the APB interface and the internal RISC-V CPU. It also drives the soft logic that synchronize the other internal clocks. The user should not connect to sclk_o because the sclk_o is stopped during clock frequency change.
preset_n_i	In	1	Reset for the APB interface.
sclk_o	Out	1	Output clock with frequency equal to DDR clock divided by 4.
APB Interface[2]			
apb_psel_i	In	1	APB Select signal.
apb_penable_i	In	1	APB Enable signal.
apb_pwrite_i	In	1	APB Write data signal.
apb_paddr_i	In	APB_ADDR_WIDTH	APB Address signal.
apb_pwdata_i	In	32	APB Write data signal.
apb_pready_o	Out	1	APB Ready signal.
apb_pslverr_o	Out	1	APB Error signal. This signal is tied to 1'b0.
apb_prdata_o	Out	32	APB Read data signal.
DDR PHY Interface (DFI)[3]			
dfi_ctrlupd_req_i	In	1	DFI MC-initiated update request signal.
dfi_ctrlupd_ack_o	Out	1	DFI MC-initiated update acknowledge signal.
dfi_phyupd_req_o	Out	1	DFI PHY-initiated update request.
dfi_phyupd_type_o	Out	2	DFI PHY-initiated update select.
dfi_phyupd_ack_i	In	1	DFI PHY-initiated update acknowledge.
dfi_reset_n_i	In	1	DFI reset signal.
dfi_cke_p0_i dfi_cke_p1_i dfi_cke_p2_i dfi_cke_p3_i	In	CK_WIDTH	DFI clock enable signal.
dfi_cs_p0_i dfi_cs_p1_i dfi_cs_p2_i dfi_cs_p3_i	In	CS_WIDTH	DFI chip select.
dfi_ca_p0_i dfi_ca_p1_i dfi_ca_p2_i dfi_ca_p3_I	In	DFI_CA_WIDTH	DFI address bus.

Pin Name	Direction	Width (Bits)[1]	Description
dfi_wrdata_cs_p0_i dfi_wrdata_cs_p1_i dfi_wrdata_cs_p2_i dfi_wrdata_cs_p3_i	In	CS_WIDTH	DFI write data chip select.
dfi_wrdata_en_p0_i dfi_wrdata_en_p2_i	In	1	DFI Write data and data mask enable.
dfi_wrdata_i	In	BUS_WIDTH*8	DFI Write data.
dfi_wrdata_mask_i	In	BUS_WIDTH	DFI Write data byte mask.
dfi_rddata_cs_p0_i dfi_rddata_cs_p1_i dfi_rddata_cs_p2_i dfi_rddata_cs_p3_i	In	CS_WIDTH	DFI read data chip select.
dfi_rddata_en_p0_i dfi_rddata_en_p2_i	In	1	DFI Read data enable.
dfi_rddata_valid_p0_o dfi_rddata_valid_p2_o	Out	BUS_WIDTH/8	DFI Read data valid indicator. Each bit indicates read data valid for the corresponding data lane.
dfi_rddata_o	Out	BUS_WIDTH*8	DFI Read data bus.
dfi_rddata_dbi_o	Out	BUS_WIDTH	DFI Read data DBI.
dfi_init_complete_o	Out	1	DFI Init Completion
dfi_ba_p0_I dfi_ba_p1_I dfi_ba_p2_I dfi_ba_p3_i	In	2	DFI Bank Address select
dfi_bg_p0_i dfi_bg_p1_I dfi_bg_p2_I dfi_bg_p3_i	In	2	DFI Bank Group select
dfi_ras_n_p0_I dfi_ras_n_p1_I dfi_ras_n_p2_I dfi_ras_n_p3_i	In	1	DFI Row Address Strobe
dfi_cas_n_p0_I dfi_cas_n_p1_I dfi_cas_n_p2_I dfi_cas_n_p3_i	In	1	DFI Column Address Strobe
dfi_we_n_p0_I dfi_we_n_p1_I dfi_we_n_p2_I dfi_we_n_p3_i	In	1	DFI Write Enable
dfi_act_n_p0_I dfi_act_n_p1_I dfi_act_n_p2_I dfi_act_n_p3_i	In	1	DFI Activation
dfi_odt_p0_I dfi_odt_p1_I dfi_odt_p2_I dfi_odt_p3_i	In	1	DFI On-Die Termination port
Other Signals			
turn_off_addr_ctrl_drv_i	In	1	Disables the output drive (set to High-Z) of DDR clock, CS and address/command signals.
irq_o	Out	1	Interrupt request signal.
trn_done_o	Out	1	Indicates the done status of the memory initialization and training.

Pin Name	Direction	Width (Bits)[1]	Description
LPDDR4/DDR4 Interface[4]			
ddr_ck_o	Out	1	DDR Clock output. This becomes differential pair signal at the FPGA IO pad.
ddr_cke_o	Out	1	DDR Clock Enable output.
ddr_cs_o	Out	1	DDR Chip Select output.
ddr_ca_o	Out	CA_WIDTH	DDR Command/Address output.
ddr_reset_n_o	Out	1	DDR Reset output
ddr_dq_io	In/Out	BUS_WIDTH	DDR Data Input/Output.
ddr_dqs_io	In/Out	BUS_WIDTH/8	DDR Data Strobe Input/Output. This signal becomes differential pair at the IO FPGA pad.
ddr_dmi_io	In/Out	BUS_WIDTH/8	DDR Data Mask Inversion Input/Output.
ddr_act_n	Out	1	Activation Command Input (DDR4 only)
ddr_odt	Out	1	On Die Termination (DDR4 only)
ddr_ba	Out	BA_WIDTH	Bank Address (DDR4 only)
ddr_bg	Out	BG_WIDTH	Bank Group (DDR4 only)
ddr_ras_n	Out	1	Row Address Strobe (DDR4 only)
ddr_cas_n	Out	1	Column Address Strobe (DDR4 only)
ddr_we_n	Out	1	Write Enable (DDR4 only)
ddr_alert_n	Out	1	Alert (Driven high) (DDR4 only)

Notes:

1. Please refer to Attribute Summary section for description of the width parameters.
2. Please refer to AMBA 3 APB Protocol v1.0 Specifications for the description of these signals.
3. Please refer to DDR PHY Interface 4.0 Specification for the description of these signals.
4. Please refer to JESD209-4C LPDDR4 and JESD79-4C DDR4 SDRAM Standard for the description of these signals.

2.2.1. DFI Signals Consideration

The DFI4.0 specifies several interface training specific signals. These are not required by the DDRPHY Module, since it operates in independent stand-alone mode, and it fully implements the trainings.

Please note these special connectivity requirements:

1. The dfi_rddata_en_p0_i and dfi_rddata_en_p2_i may be provided as a per byte lane output from controller with all bits having identical timing. The PHY requires only 1 dfi_rddata_en input, so it is suggested to simply connect the LSB (bit 0) of controller dfi_rddata_en to PHY dfi_rddata_en. Same thing also applies to dfi_wrdata_en_p0_i and dfi_wrdata_en_p2_i.
2. The turn_off_addr_ctrl_drv_i is a special PHY input that can be used to dynamically turn off DRAM clock and command / address outputs during self-refresh. It may be used by the MC to automatically shut off these signals during self-refresh to save power. If the controller does not support such feature, then it is recommended to control it using a software programmable register bit to have the ability to turn off these outputs if required. During memory initialization, the DDRPHY module assert this signal internally to turn off these signals during reset and CKE wait periods.

When controlling the signal using the controller, for turn_off_addr_ctrl_drv_i assertion (logic 1) timing after issuing self-refresh entry command on DFI bus, controller needs to follow JEDEC spec for clock shut off after entering self-refresh mode + additional 5 sclk_o margin to ensure DFI command propagation through the PHY is satisfied before the assertion.

Same also applies when de-asserting turn_off_addr_ctrl_drv_i (logic 0) before issuing self-refresh exit command. In this case also, PHY needs additional 5 sclk_o margin to ensure turn_off_addr_ctrl_drv_i is propagated to the IOs and clock output is turned on and stable before issuing self-refresh exit command on the DFI bus.

2.3. Attribute Summary

Table 2.2. Attribute Table

Attribute	Selectable Values	Default	Dependency on Other Attributes
General Tab			
General Group			
Interface Type	LPDDR4, DDR4	LPDDR4	
I/O Buffer Type	LVSTL_I, LVSTL_II	LVSTL_I	If *Interface Type*==LPDDR4
	POD11, POD12	POD11	If *Interface Type*==DDR4
Gearing Ratio	8:1	8:1	Display Information only
Enable Sideband ECC	Checked, Unchecked	Unchecked	If *Interface Type*==DDR4
Enable DBI	Checked, Unchecked	Unchecked	—
Read Latency	Calculated	N/A	*DDR Memory Frequency (MHz)*
Write Latency	Calculated	N/A	*DDR Memory Frequency (MHz)*
DDR Bus Width	16, 32, 64	32	If *Enable Sideband ECC* is Unchecked
	40, 72	40	If *Enable Sideband ECC* is Checked
Enable Training CPU	Checked, Unchecked	Checked	—
Clock Settings Group			
Enable PLL	Checked	Checked	Display Information only
DDR Memory Frequency (MHz)	266 300, 350, 400, 533, 666, 800, 933	800	
System Clock Frequency (MHz)	Calculated = (DDR Memory Frequency) /(Gearing Ratio/2)	N/A	Display Information only
Reference Clock Frequency (MHz)	25, 50, 100	100	If *Interface Type*==LPDDR4
	10 – 800		If *Interface Type*==DDR4
DDR Memory Actual Frequency (MHz)	Calculated	N/A	Display Information only
Clock/Address/Command Group			
Number of Clocks	1, 2	1	Only 1 Rank is currently supported
Number of Chip Selects (Rank)	1, 2	1	Only 1 Rank is currently supported
Address Width	6,14	6	Calculated based on *Interface Type*
Number of Chip ODT	1, 2	1	Number of Chip Selects (Rank) If *Interface Type*==DDR4
BA Width	2	2	If *Interface Type*==DDR4
BG Width	2	2	If *Interface Type*==DDR4

Note: The DDR4 features are plan for the board market release, it is not available for Beta release.

Table 2.3. Attributes Descriptions

Attribute	Description
General Tab	
General Group	
Interface Type	Specifies the SDRAM Memory interface: LPDDR4, DDR4
I/O Buffer Type	Specifies the I/O Standard for the DDR Interface signals and the PLL reference clock, pll_refclk_i.
Gearing Ratio	Ratio of DDR data speed to controller speed. The *Gearing Ratio*=8:1 means there is 8 * DDR_WIDTH amount of data for every sclk_o cycle. Therefore, the bit width of dfi_wrdata_i and dfi_rddata_o is BUS_WIDTH*8.
Enable Sideband ECC	Enable SEC DED ECC side band ECC. This will be enabled for DDR4.
Enable DBI	Data bus inversion
Read Latency	Read latency is delay from launching read command to receiving read data from SDRAM

Attribute	Description
Write Latency	Write latency is delay from launching write command to time write data is provided
DDR Bus Width (BUS_WIDTH)	Data bit width of DDR interface
Enable Training CPU	Enables the internal RISC-V training CPU.
Clock Settings Group	
Enable PLL	Shows that PLL is enabled inside the IP.
DDR Memory Frequency (MHz)	Desired speed of the DDR clock signal, ddr_ck_o.
System Clock Frequency (MHz)	Speed of the DFI clock, sclk_o which should be used by the controller when driving the DFI signals.
Reference Clock Frequency (MHz)	Speed of the PLL reference clock pll_refclk_i.
DDR Memory Actual Frequency (MHz)	Speed of the DDR clock signal, ddr_ck_o based on the calculated PLL settings.
Clock/Address/Command Group	
Number of Clocks (CK_WIDTH)	Number of DDR clocks to be driven to interface to SDRAM
Number of Chip Selects (Rank) (CS_WIDTH)	Specifies the number of Ranks. Also specifies the bit with of ddr_cs_o and ddr_odt_o when available.
Address Width (CA_WIDTH)	Specifies the bit width of the ddr_ca_o signal.
Number of Chip ODT	Specifies the bit width of the ddr_odt_o signal.
BA Width (BA_WIDTH)	Specifies the bit width of the ddr_ba_o signal.
BG Width (BG_WIDTH)	Specifies the bit width of the ddr_bg_o signal.

2.4. Register Description

2.4.1. Overview

Table 2.4 lists the DDRPHY Registers that are available to user. Some registers are reserved for use by the training CPU, and user should not write to this register as it may cause failure in the initialization and training operations.

Table 2.4. Summary of DDRPHY Module Registers

Offset	Register Name	Access Type	Description
0x000 to 0x1FC	For Uniquify PHY registers	—	T.B.D.
0x200	FEATURE_CTRL_REG	RW	Feature Control Register
0x204	RESET_REG	RW	Reset Register
0x208	SETTINGS_REG	RW	Settings Register
0x20C	CLOCK_CTRL_STAT_REG	RW	PHY Clock Control and Status Register
0x210	INT_STATUS_REG	RW1C	Interrupt Status Register
0x214	INT_ENABLE_REG	RW	Interrupt Enable Register
0x218	INT_SET_REG	WO	Interrupt Set Register
0x21C	INIT_CTRL_REG	WO	Reserved for use by the internal CPU
0x220	TRN_OP_REG	RW	Training Operation Register
0x224	STATUS_REG	RW	Training Status Register
0x228	TIMEOUT_REG	WR1C	Timeout Register
0x22C	TIMER_RELOAD_REG	WO	Timer Reload Register
0x230	MRW_CTRL_DFI_REG	WO	DDR Mode Register Write Control
0x234 to 0x23C	Reserved	—	Reserved
0x240	SCRATCH_0_REG	RW	SCRATCH Register 0 for debug

Offset	Register Name	Access Type	Description
0x244	SCRATCH_1_REG	RW	SCRATCH Register 1 for debug
0x234 to 0x3FE	Reserved	—	Reserved
0x400 to 0x46C	For PLL registers	—	T.B.D.
0x470 to 0x7FC	Reserved	—	Reserved

The behavior of registers to write and read access is defined by its access type, which is defined in Table 2.5.

Table 2.5. Access Type Definition

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value	Ignores write access
WO	Returns 0	Updates register value
RW	Returns register value	Updates register value
RW1C	Returns register value	Writing 1'b1 on register bit clears the bit to 1'b0. Writing 1'b0 on register bit is ignored.
RSVD	Returns 0	Ignores write access

2.4.2. Feature Control Register (FEATURE_CTRL_REG)

The FEATURE_CTRL_REG reflects the modes/features that are set through the attributes. These are only set during IP configuration and cannot be modified during run-time. The CPU reads this register to identify the modes of operation.

Table 2.6. Feature Control Register

Field	Name	Access	Width	Reset
[31:21]	reserved	RSVD	11	—
[20]	reset_init_by_phy	RO	1	1
[19]	dq_lane_swizz_en	RO	1	0
[18]	dq_swizz_en	RO	1	0
[17]	ca_swizz_en	RO	1	0
[16]	num_ranks	RO	1	*Number of Ranks*
[15:12]	ddr_width	RO	4	*DDR Bus Width*
[11:8]	ddr_type	RO	4	*DDR Interface Type*
[7:4]	addr_translation	RO	4	0
[3]	gear_ratio	RO	1	*Gearing Ratio*
[2]	pwr_down_en	RO	1	*Enable Power Down*
[1]	dbi_en	RO	1	*Enable DBI*
[0]	ecc_en	RO	1	*Enable ECC*

- reset_init_by_phy
  - Enables the logic and routine that performs drives the ddr_reset_n_o and ddr_cke_o signals during memory initialization as well as the initial mode register writes and ZQ Calibration right after the initialization. This is fixed to 1 for the first release.
    - 0 – The MC will do the memory reset and initialization
    - 1 – The PHY will do the memory reset and initialization
- dq_lane_swizz_en
  - Enables the swapping of DQ byte lanes. This is fixed to 0 for the first release.
    - 0 – DQ byte lane swap is disabled.
    - 1 – DQ byte lane swap is enabled.

- dq_swizz_en
  - Enables the swapping of DQ bit per byte lane. This is fixed to 0 for the first release.
    - 0 – DQ bit swap per byte lane is disabled.
    - 1 – DQ bit swap per byte lane is enabled.
- ca_swizz_en
  - Enables the swapping of command and address bits. This is fixed to 0 for the first release.
    - 0 – Command and address bit swap is disabled.
    - 1 – Command and address bit swap is disabled.
- num_ranks
  - The num_ranks specifies the number of DDR ranks as follows:
    - 0 – Single Rank
    - 1 – Dual Rank
- ddr_width
  - The ddr_width specifies the bit width of the DDR data bus as follows:
    - 0 – *DDR Bus Width* = 8 bits
    - 1 – *DDR Bus Width* = 16 bits
    - 2 – reserved
    - 3 – *DDR Bus Width* = 32 bits
    - 4 – 4'h6: reserved
    - 7 – *DDR Bus Width* = 64 bits
- ddr_type
  - The ddr_type specifies the DDR standard implemented by the Memory Controller IP Core. This is based on the *Interface Type* attribute.
    - 4 – *Interface Type* = DDR4
    - 12 – *Interface Type* = LPDDR4
- addr_translation
  - The address translation specifies the mapping of the address bits of the local memory-mapped address and the memory address in terms of row, column, bank, and rank.
    This is for future enhancement is currently unused.
- gear_ratio
  - The gear_ratio specifies the ratio of clock frequency of DDR clock domain and system clock domain.
    - 0 – 4:1 gearing: ddr_ck_o frequency = 2 x sclk_o frequency
    - 1 – 8:1 gearing: ddr_ck_o frequency = 4 x sclk_o frequency
- pwr_down_en
  - The pwr_down_en enables the power saving mode by putting the memory in self-refresh when there is no traffic for the amount of sclk_o cycles specified in the attribute *No. of SCLK to enter Self-Refresh from no traffic*.
- dbi_en
  - The dbi_en enables the Data Bus Inversion function to reduce the toggling of DDR data signal in the board, thus improving the signal integrity and reduce dynamic power consumption.
- ecc_en
  - The ecc_en enables the side-band ECC function.

2.4.3. Reset Register (RESET_REG)

The RESET_REG controls the reset of the internal CPU, this is on reset state at power-on. The host should un-reset these to start the memory initialization and training. Upon training completion, the internal CPU writes to this register to return the Memory Training Engine to reset state, thus saving power consumption. This register does not reset the CSR.

Table 2.7. Reset Register

Field	Name	Access	Width	Reset
[31:1]	reserved	RSVD	30	—
[0]	cpu_rst_n	RW	1	0

2.4.4. Settings Register (SETTINGS_REG)

The SETTINGS_REG controls the DDR write and read latencies according to the selected attribute.

Table 2.8. Settings Register

Field	Name	Access	Width	Reset
[31:28]	reserved	RSVD	4	—
[27:16]	cmd_freq	RO	12	*DDR Memory Frequency*
[15:8]	read_latency	RW	8	*Read Latency*
[7:0]	write_latency	RW	8	*Write Latency*

- cmd_freq
  - The cmd_freq reflects the *DDR Memory Frequency* attribute value. The Training routine use this value to trim initial delay settings of the DDR signals.
- read_latency
  - The read_latency specifies the number of DDR clock cycles from read command to the first read data.
- write_latency
  - The write_latency specifies the number of DDR clock cycles from write command to the first write data.

2.4.5. Clock Control and Status Register (CLOCK_CTRL_STAT_REG)

The CLOCK_CTRL_STAT_REG is used to disable the PLL.clkophy internal clock, it also shows the PLL lock status.

Table 2.9. Error Log Register

Field	Name	Access	Width	Reset
[31:16] & [3:2]	reserved	RSVD	18	—
[15:4]	pll_ref_clk	RO	12	*Reference Clock Frequency*
[3:2]	reserved	RSVD	30	—
[1]	pll_lock	RO	1	0
[0]	disable_clkophy	WO	1	0

- pll_ref_clk
  - Returns the PLL reference clock frequency in MHz.
- pll_lock
  - The cmd_freq reflects the *DDR Memory Frequency* attribute value. The Training routine use this value to trim initial delay settings of the DDR signals.
- disable_clkophy
  - Set to 1 to disable the PLL output clock going to the DDRPHY and ECLKSYNC hard IPs.

2.4.6. Interrupt Status Register (INT_STATUS_REG)

Table 2.10 contains all the interrupts currently pending in the DDRPHY Module. When an interrupt bit asserts, it remains asserted until it is cleared by the host by writing 1'b1 to the corresponding bit.

The interrupt status bits are independent of the interrupt enable bits. In other words, status bits may indicate pending interrupts, even though those interrupts are disabled in the Interrupt Enable Register. The logic which handles interrupts should mask (bitwise and logic) the contents of INT_STATUS_REG and INT_ENABLE_REG registers to determine the interrupts to service. The irq_o interrupt signal is asserted whenever both an interrupt status bit and the corresponding interrupt enable bits are set.

Table 2.10. Interrupt Status Register

Field	Name	Access	Width	Reset
[1]	trn_err_int	RW1C	1	0
[0]	trn_done_int	RW1C	1	0

- trn_err_int
  - Training Error Interrupt. This Interrupt bit asserts when the Memory Training Engine encounters an error during training. The user should read the status register to determine the specific error.
    - 0 – No interrupt
    - 1 – Interrupt pending
- trn_done_int
  - Training Done Interrupt. This Interrupt bit asserts when initialization and training is completed successfully.
    - 0 – No interrupt
    - 1 – Interrupt pending

2.4.7. Interrupt Enable Register (INT_ENABLE_REG)

Table 2.11 presents the summary of INT_ENABLE_REG.

Table 2.11. Interrupt Enable Register

Field	Name	Access	Width	Reset
[31:5]	reserved	RSVD	27	—
[4]	temp_change_en	RW	1	0
[3]	ue_en	RW	1	0
[2]	ce_en	RW	1	0
[1]	trn_err_en	RW	1	0
[0]	trn_done_en	RW	1	0

- trn_err_en
  - Training Error Interrupt Enable.
    - 0 – Interrupt disabled
    - 1 – Interrupt enabled
- trn_done_en
  - Training Done Interrupt Enable.
    - 0 – Interrupt disabled
    - 1 – Interrupt enabled

2.4.8. Interrupt Set Register (INT_SET_REG)

Table 2.12 shows the summary of INT_SET_REG. Writing 1'b1 to a register bit in this register asserts the corresponding interrupts status bit in INT_STATUS_REG while writing 1'b0 is ignored.

Table 2.12. Interrupt Set Register

Field	Name	Access	Width	Reset
[31:5]	reserved	RSVD	27	—
[4]	temp_change_set	WO	1	0
[3]	ue_set	WO	1	0
[2]	ce_set	WO	1	0
[1]	trn_err_set	WO	1	0
[0]	trn_done_set	WO	1	0

- trn_err_set
  - Training Error Interrupt Set.
    - 0 – No Action
    - 1 – Assert INT_STATUS_REG.trn_err_int
- trn_done_set
  - Training Done Interrupt Set.
    - 0 – No Action
    - 1 – Assert INT_STATUS_REG.trn_done_int

2.4.9. Initialization Control Register (INIT_CTRL_REG)

Table 2.13 shows the summary of INIT_CTRL_REG. This register controls the DDR signals for memory reset and initialization. This is used by the training CPU when performing memory reset and initialization routine. The user should not write to this register.

Table 2.13. Initialization Control Register

Field	Name	Access	Width	Reset
[31:7]	reserved	RSVD	25	—
[6]	ddr_ck_dis	WO	1	0
[5]	zq_lat	WO	1	0
[4]	zq_cal	WO	1	0
[3:2]	dfi_cke	WO	2	0
[1]	turn_off_addr_ctrl	WO	1	1
[0]	ddr_reset_n	WO	1	0

- ddr_ck_dis
  - Set ddr_ck_dis=1 to disable (tie Low) the ddr_ck_o.
- zq_lat
  - Set zq_lat=1 to issue a ZQ Latch command to the DDR I/F. This is only enabled for LPDDR4.
- zq_cal
  - Set zq_cal=1 to issue a ZQ Calibration command to the DDR I/F.
- dfi_cke
  - Controls the ddr_cke_o signal. The bit[3] is only used when *Number of Chip Selects* = 2.
- turn_off_addr_ctrl
  - Set turn_off_addr_ctrl=1 to turn off the DDR clock, address and control signals.
- ddr_reset_n
  - Controls the ddr_reset_n_o signal.

2.4.10. Training Operation Register (TRN_OP_REG)

Table 2.14 shows the summary of TRN_OP_REG. This register controls the memory initialization and training. It is recommended to disable these during simulation if user want to skip the lengthy initialization and training.

Table 2.14. Training Operation Register

Field	Name	Access	Width	Reset
[31:5]	reserved	RSVD	24	—
[7]	mem_verf_training_en	RW	1	0
[6]	mc_vref_training_en	RW	1	0
[5]	ca_vref_training_en	RW	1	0
[4]	write_trn_en	RW	1	1
[3]	read_trn_en	RW	1	1
[2]	write_lvl_en	RO	1	1
[1]	cbt_en	RW	1	1
[0]	init_en	RW	1	1

- mem_verf_training_en
  - The mem_vref_trn_en enables the memory's DQ Vref training. When enabled, the DDRPHY will do write training across multiple memory Vref points and select the optimal Vref value.
    Note that this is an on-going development, not yet enabled in the DDRPHY IP v1.0.0 and will be enabled in the succeeding release.
- mc_verf_training_en
  - The mc_vref_trn_en enables the DDRPHY's DQ Vref training. When enabled, the DDRPHY will do write training across multiple Vref points of the FPGA's IO and select the optimal Vref value.
- Note that this is an on-going development, not yet enabled in the DDRPHY IP v1.0.0 and will be enabled in the succeeding release. mem_verf_training_en
- ca_vref_trn_en
  - The ca_vref_trn_en enables the memory's CA Vref training. When enabled, the DDRPHY will do CA training across multiple CA_Vref points and select the optimal CA_Vref value. This bit is currently used because the lab results show that this feature is not needed for the currently supported speeds.
- write_trn_en
  - The write_trn_en enables the write bit leveling. The write training optimizes the write DQ delay with respect to the write DQS to improves the data valid window for writes. See Write Bit Leveling section for details.
    - 0 – Do not perform write bit leveling.
    - 1 – Perform write bit leveling.
- read_trn_en
  - The read_trn_en enables the read training. The read training tunes the PHY to capture the incoming read DQS burst according to the target read latency. See **Error! Reference source not found.** section for details.
    - 0 – Do not perform read training.
    - 1 – Perform read training.
- write_lvl_en
  - The write_lvl_en enables the write leveling during initialization and training. The write leveling compensates for CK-DQS timing skews. See Write Leveling section for details. Write Leveling is always performed during training.
- cbt_freq_change_en
  - The cbt_en enables the command bus during initialization and training. The command bus training performs CA_VREF programming and aligns the CS/CA and CK for high frequency operation. See Command Bus Training for LPDDR4 section for details.
    - 0 – The clock frequency change is not performed during CBT.
    - 1 – The clock frequency change is performed during CBT.

- init_en
  - The init_en provides option to shorten the initialization for simulation purposes.
    - 0 – Initialization is greatly reduced. For example, reset time and CKE low time is greatly shortened.
    - 1 – Initialization is done according to the corresponding DDR standard.

2.4.11. Training Status Register (TRN_STATUS_REG)

Table 2.15 shows the summary of STATUS_REG. This register is written by the internal CPU to communicate the status to the System CPU.

Table 2.15. Training Status Register

Field	Name	Access	Width	Reset
[31:14]	reserved	RSVD	18	—
[13:12]	error_on_rank	RW	2	0
[11]	write_trn_err	RW	1	0
[10]	read_trn_err	RW	1	0
[9]	write_lvl_err	RW	1	0
[8]	cbt_err	RW	1	0
[7:6]	reserved	RSVD	2	—
[5]	scl_done	RW	1	0
[4]	write_trn_done	RW	1	0
[3]	read_trn_done	RW	1	0
[2]	write_lvl_done	RW	1	0
[1]	cbt_done	RW	1	0
[0]	init_done	RW	1	0

- error_on_rank
  - The error_on_rank specifies the rank where the training error encountered.
    - 2'bx1 – Training error occurred on rank 0
    - 2'b1x – Training error occurred on rank 1
- write_trn_err
  - The write_trn_err asserts when there is a failure in write training.
- read_trn_err
  - The read_trn_err asserts when there is a failure in read training.
- write_lvl_err
  - The write_lvl_err asserts when there is a failure in write leveling.
- cbt_err
  - The cbt_err asserts when there is a failure in command bus training.
- scl_done
  - The scl_done asserts when self-calibrating logic is completed successfully.
- write_trn_done
  - The write_trn_done asserts when write training is completed successfully.
- read_trn_done
  - The read_trn_done asserts when read training is completed successfully.
- write_lvl_done
  - The write_lvl_done asserts when write leveling is completed successfully.

- cbt_done
  - The cbt_done asserts when command bus training is completed successfully.
- init_done
  - The init_done asserts when the PHY initialization is done.

2.4.12. Scratch Registers (SCRATCH_0_REG and SCRATCH_1_REG)

These are 32-bit R/W registers that are used by the training CPU to write debug information which can be captured using the reveal tool.

2.5. Initialization

The DDRPHY Module implements a Training CPU that executes the initialization and training routines that should be for all memory types. It implements all the supported training features for each DRAM type, such as command bus training (CBT) or CS/CA training, VREF CA training, VREF DQ training, write leveling, read side bit-leveling, write side bit-leveling and Self-Calibrating Logic (SCL) (encompassing DQS gate training and read leveling). It thus automatically configures all VREF settings and write and read timing parameters in the PHY.

The Setup and hold budgets of each individual DQ with respect to DQS are dynamically optimized during write and read bit leveling.

The Setup and hold budgets of Address/Command signals with respect to clock are optimized during command bus training (applicable for LPDDR3 and LPDDR4) which is typically run at initialization. For DDR3 and DDR4, the PHY provides software register control to manually adjust the skew between clock and Address/Command signals as necessary across corners. The timing of per-rank signals can also be controlled by register setting of the PHY.

To Start the memory reset and initialization, write 1 to RESET_REG. Then poll TRN_STATUS_REG until it indicates that all training is done. The DDRPHY Module is ready for DFI access in this case.

2.6. Training

2.6.1. VREF_DQ/CA Training

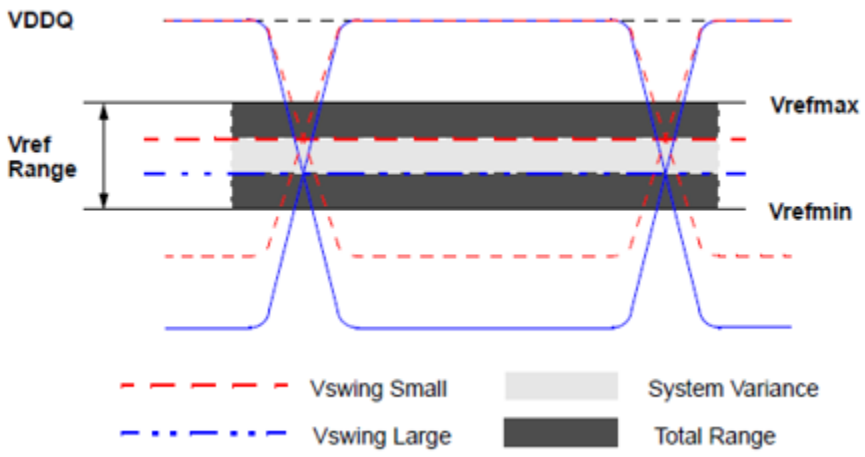

Figure 2.2. Vref operating range(Vrefmin, Vrefmax)

For DDR4, LPDDR4, the driver and termination together determine common mode voltage on receiving side. The common mode will drift with different die to die combination. VREF Training is required to center VREF to vertical data window. The DDRPHY training routine performs VREF_CA training during command bus training and VREF DQ training during Read and Write Training.

2.6.2. Command Bus Training for LPDDR4

For LPDDR4, the DDRPHY will perform initialization with 266MHz boot frequency, it will change the clock frequency to the selected *DDR Memory Frequency* during command bus training. The DDRPHY Module training routine performs the following as described in the JESD209-4B LPDDR4 Standard:

- Memory VREF_CA setting
- CS Training – aligns the rising edge of CK to the middle of CS pulse. The algorithm uses a half-cycle CS signal.
- CA Training – aligns the rising edge of CK to the middle of CA data valid window.

The DDRPHY routine uses the CA feedback on the memory DQ[13:8] for each LPDDR4 memory channel. In case of dual rank, the rank 0 will be trained first, thus, this should be the terminating rank.

The command bus training is only performed after the reset.

2.6.3. Write Leveling

From DDR3 and up, the DIMM fly-by topology requires write leveling to compensate for the CK-to-DQS timing skew. The fly-by topology allows different memory chips to receive write command at different times. An example of DDR3 DIMM fly-by topology is shown in Figure 2.3. The Memory expects a pre-defined number of cycles from the receipt of the command to the arrival of the write data in the memory device pins. The DDRPHY Module adjusts DQS/DQ/DM delay so that write data arrive at memory device pin at different times to match the CLK/CA bus Fly-By delay. The DDRPHY Module performs write leveling according to the each DDRx/LPDDRx standard.

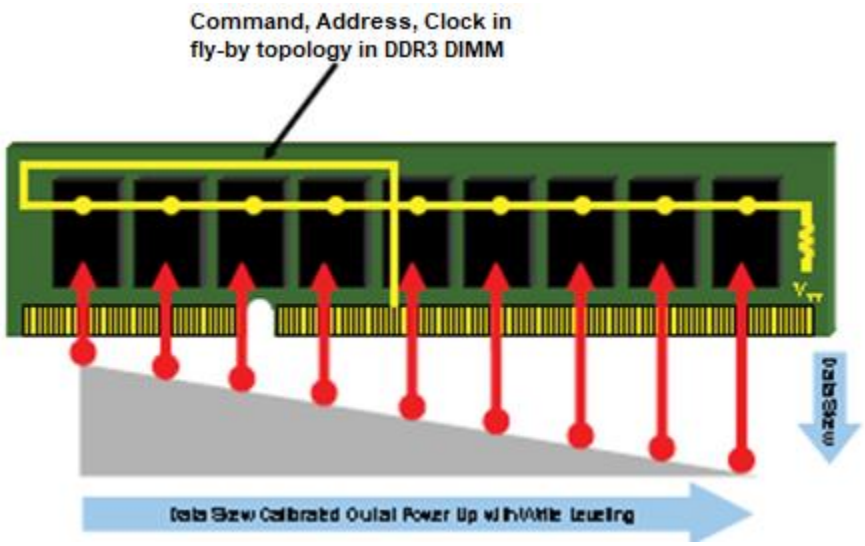

Figure 2.3. DDR3 DIMM Fly-by Topology

2.6.4. Read Bit Leveling

The Uniquify PHY submodule implements the Adaptive Bit Calibration (ABC) to ensure that the rising edge of input DQS is correctly sampling the respective input DQ, the ABC aligns the rising edge to the center of the DQ data. This is done to compensate for the unpredictable amount of delay on DQ that results in an undesirable skew between the input DQ and DQS signals. The algorithm is implemented by reading a continuous 1-0-1-0 pattern from the DRAM on DQ and DQS, where DQS is delayed by 90 degrees. Therefore, ideally the rising edge of DQS is always sampling a 0 on DQ.

The ABC operates by sampling input DQ at the rising edge of input DQS and delaying and advancing the internal DQ with respect to DQS to measure the distance to each edge of DQ. These distances, referred to as valid windows, are used to determine how far, or close, the rising edge of DQS is from the center of DQ and how much DQ needs to be delayed or advanced to center it. An example below demonstrates a scenario where DQ is being sampled too early, and bit-leveling therefore determined that a delay of (TA- TB)/2 is to be added to DQ to align with rising edge of DQS as shown in Figure 2.4.

By determining the size of the windows, ABC determines if DQ is being sampled too early or too late, and therefore adjusts the delay on DQ to sample it correctly. This is determined by comparing the size of the windows and setting the trim delay to the amount needed to alight the rising edge of DQS as close to the center of the '0' on DQ as possible.

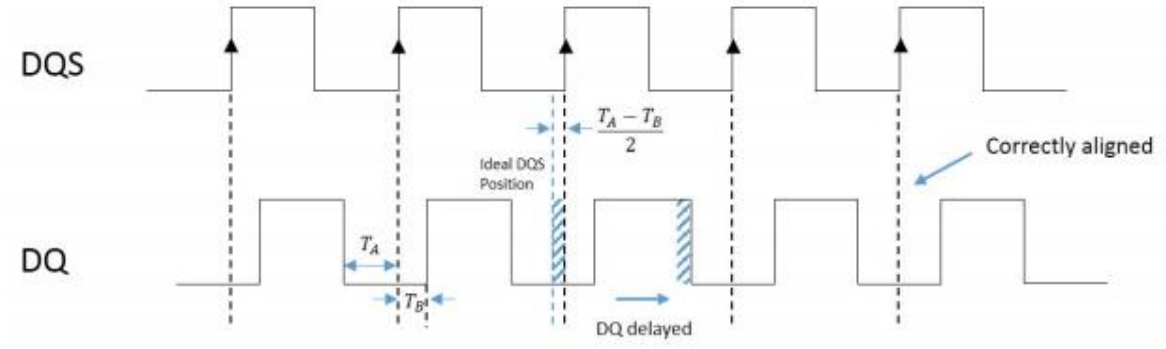

Figure 2.4. Read Bit Leveling

2.6.5. Write Bit Leveling

During write Bit Leveling adjusts the write DQ delay such that memory device DQS edges are center aligned to the write DQ data valid window. To achieve this, the DDRPHY Module writes consecutive data to the memory and reads it back and compares it to the written data. In case of LPDDR4, this is done using the MPC Write FIFO and MPC Read FIFO commands. For each iteration of write and read-back, the ABC logic adjusts write DQ delay finding the minimum delay (TA) and maximum delay (TB) that will give a compare pass on the read data. The mentioned TA and TB is shown Figure 2.4, the average of these is used as the final write DQ delay.

2.6.6. Self-Calibrating Logic

The self-calibrating logic(SCL) are special features meant to eliminate DDR timing problems that may be faced by the user. Read data capture timing and write alignment timing are setup automatically within the DDRPHY at initialization using SCL.

During Read Gate Leveling the DDRPHY delays the DQS gate until the first DQS rising edge that samples the value DQ to be a 1 instead of a 0 as shown in Figure 2.5. This requires that the DQS gate position must be somewhere in the 1 cycle preamble region of DQS even before training starts during initialization.

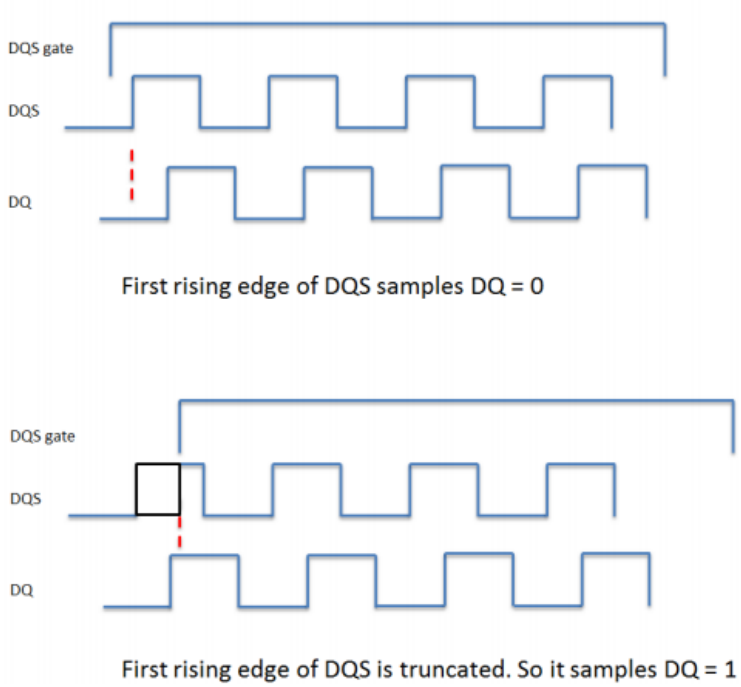

Figure 2.5. Read Gate Leveling of SCL

The initial position of the DQS gate is based on the *Read Latency* attribute in the IP configuration GUI. The SCL makes read capture adjustments without using the DDR4/LPDDR4 multi-purpose register, it first performs a write and then read it continuously while doing the internal delay adjustments. The circuit used to implement the read capture logic is shown in Figure 2.6. The data capture_clk is the variable delay clock which the SCL will tune so that there is optimal setup and hold margins for clocking the data from the input DRAM strobe domain to the PHY clock domain (represented by clock clk).

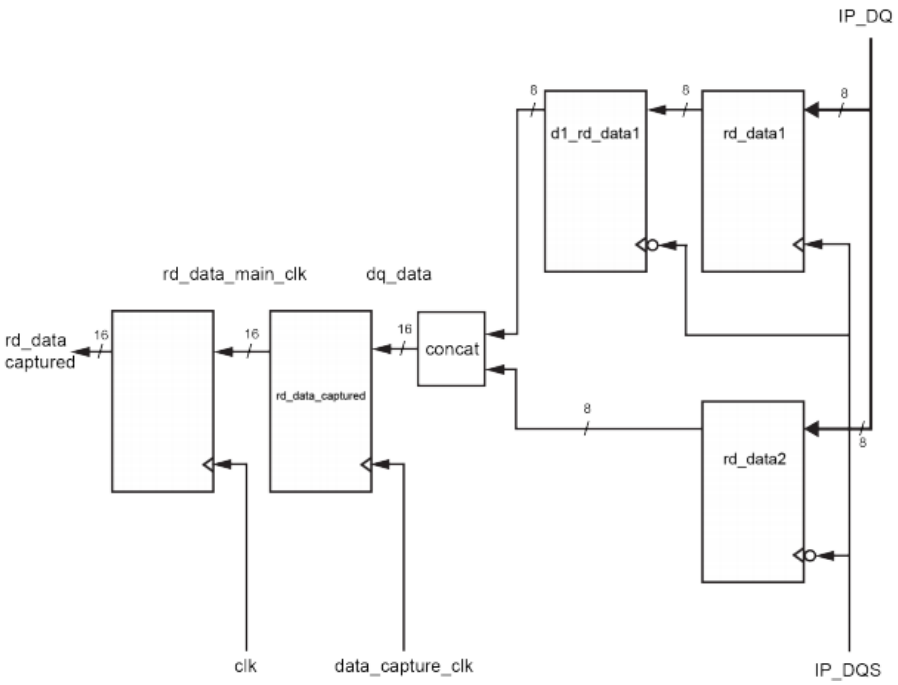

Figure 2.6. Read Gate Leveling Logic

During SCL routine, the DDRPHY will continuously look for the location of the falling edge of input DQS. Just after the input DQS falling edge, the valid data will be available for clocking into the rd_data_captured register so that it can be held for clocking into the main PHY clock domain. The SCL will find the center between the rising edge of clk and the falling edge of the next input DQS strobe. This is shown by points A and B in Figure 2.7. Whichever point gives the largest setup and hold margins (point B in the below example) will be set as the active edge location of data_capture_clk.

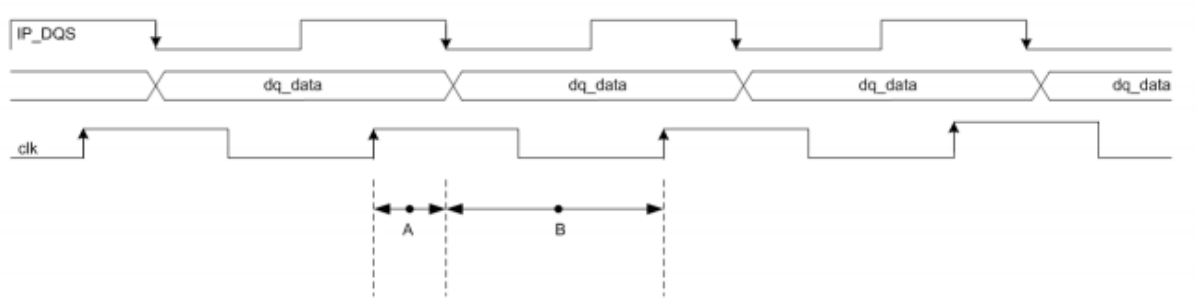

Figure 2.7. Read Gate Leveling Tuning of SCL

2.6.6.1. SCL Latency Calibration

SCL will not only determine the best clock phase to capture the incoming data, but it will also adjust the timing so that the proper latency is achieved. The captured data (rd_data_captured in Figure 2.6) is passed into the circuit shown in Figure 2.8. SCL will incrementally select the output of the mux until the proper data is received. The value of the mux select (main_clk_delta_dly) is then held for subsequent read operations.

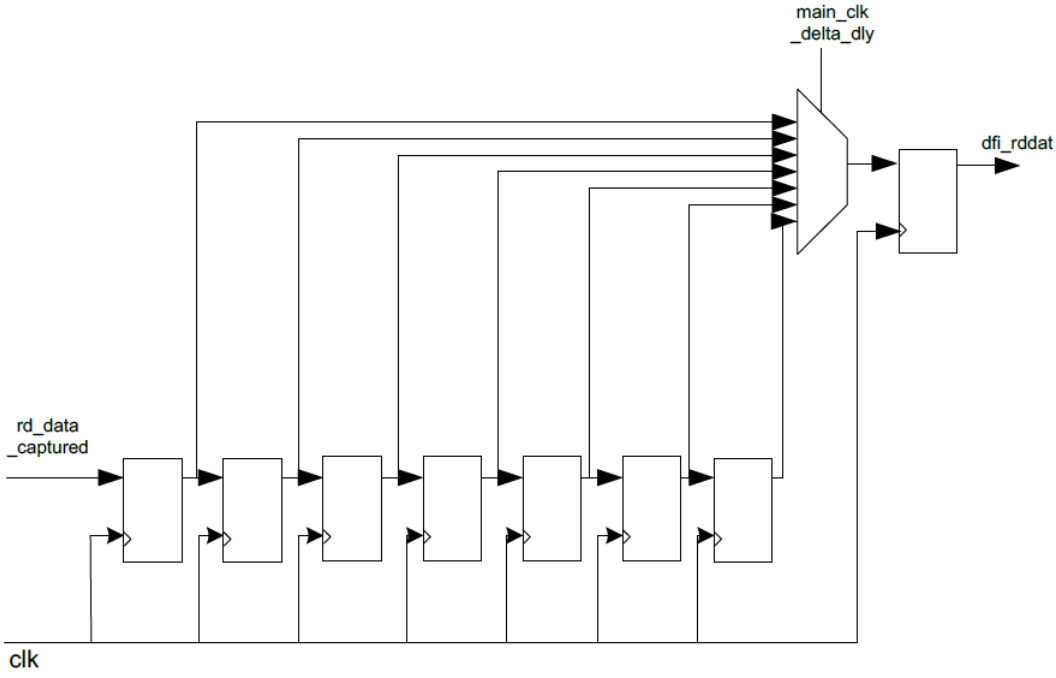

Figure 2.8. Latency Calibration of SCL

3. IP Generation, Simulation, and Validation

This section provides information on how to generate the IP Core using the Lattice Radiant software and how to run simulation and synthesis. For more details on the Lattice Radiant software, refer to the Lattice Radiant software user guide.

3.1. Generating the IP

Lattice Radiant Software allows user to generate and customize modules and IPs and integrate them into the device architecture.

To generate the DDRPHY Module in Lattice Radiant Software:

1. In the **Module/IP Block Wizard** create a new Lattice Radiant Software project for the DDRPHY module.
2. In the **IP Catalog** tab, double-click on **DDRPHY** under **Module, Architecture_Modules, I/O** category. The **Module/IP Block Wizard** opens as shown in Figure 3.1. Enter values in the **Component name** and the **Create in** fields and click **Next**.

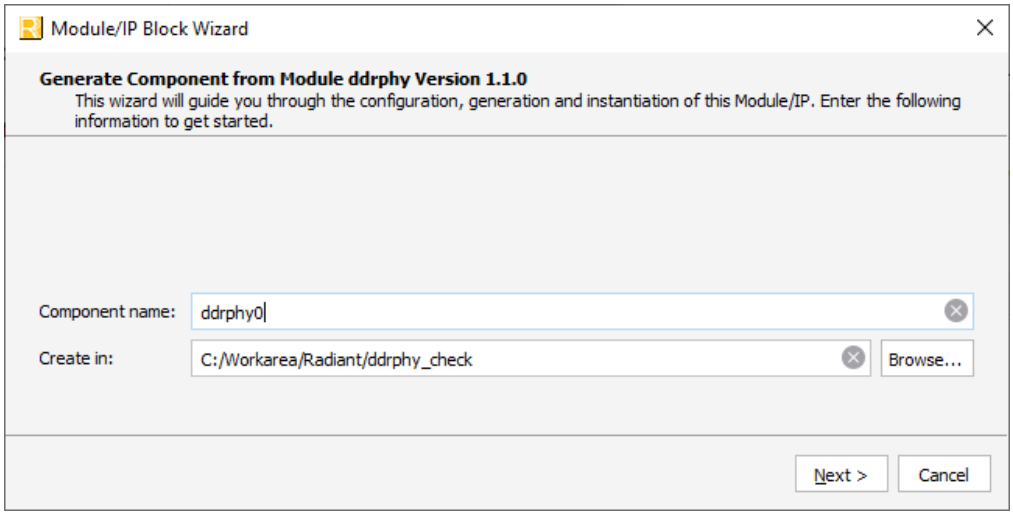

Figure 3.1. Module/IP Block Wizard

3. In the dialog box of the **Module/IP Block Wizard** window, configure DDRPHY module according to custom specifications using drop-down menus and check boxes. As a sample configuration, see Figure 3.2. For configuration options, see Table 2.2.

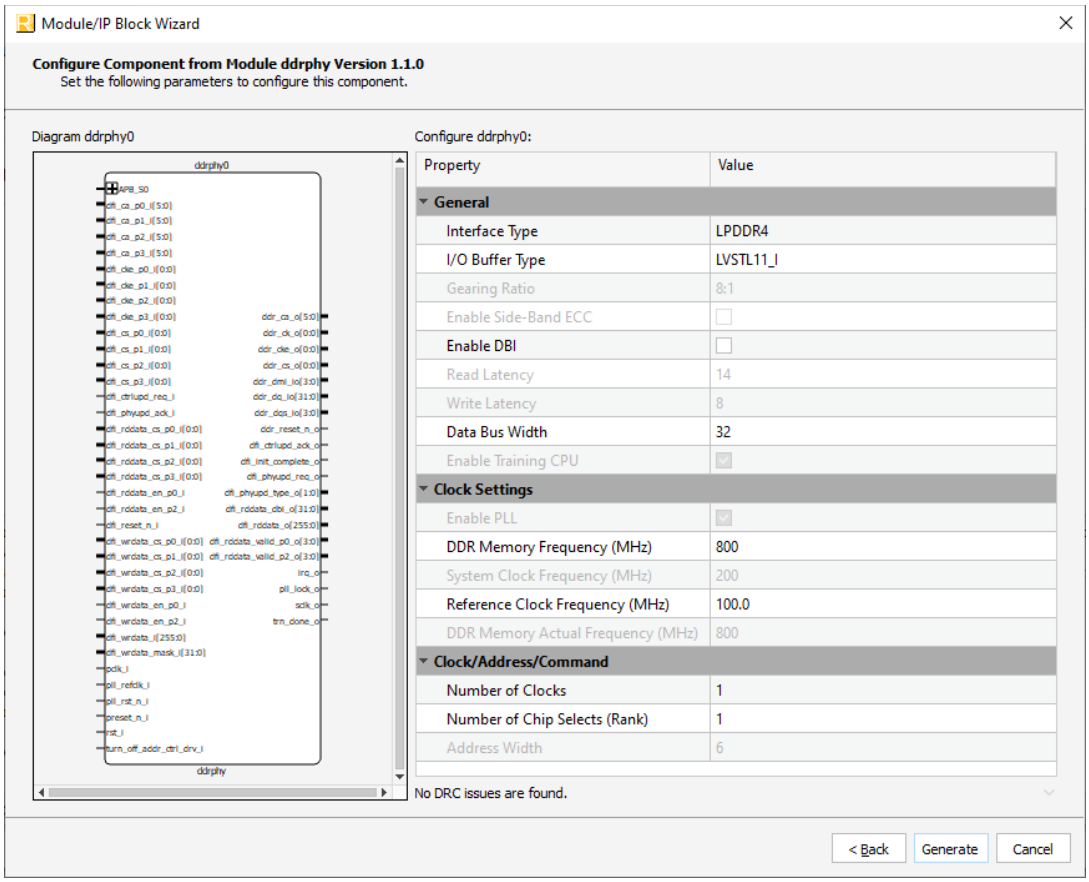

Figure 3.2. Configure Block of DDR Memory Module

4. Click **Generate**. The **Check Generating Result** dialog box opens, showing design block messages and results as shown in Figure 3.3.

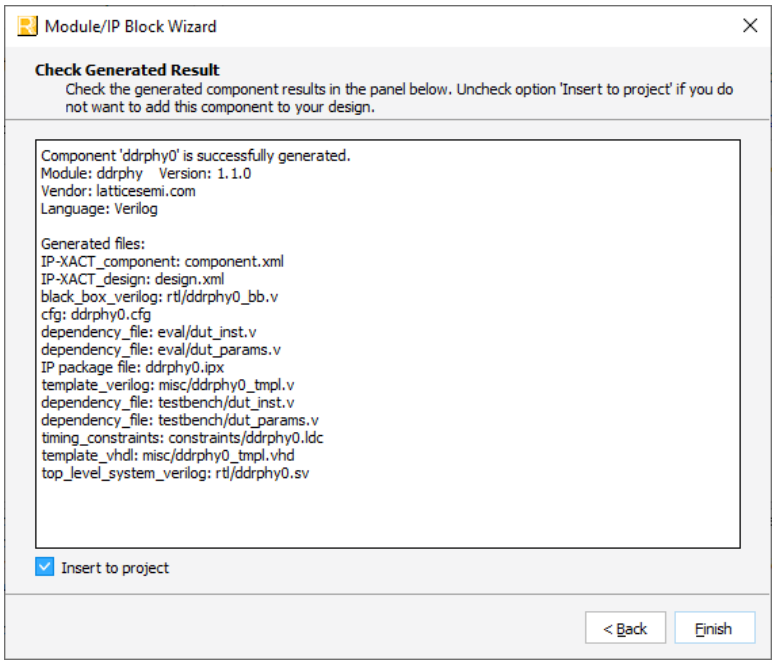

Figure 3.3. Check Generating Result

5. Click **Finish** to generate the Verilog file. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields shown in Figure 3.1.

The generated LPDDR4 Memory Module package includes the black box (<Component name>_bb.v) and instance templates (<Component name>_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Component name>.v) that can be used as an instantiation template for the IP core is also provided. The user may also use this top-level reference as the starting template for the top-level for their complete design. The generated files are listed in Table 3.1.

Table 3.1. Generated File List

Attribute	Description
<Component name>.ipx	This file contains the information on the files associated to the generated IP.
<Component name>.cfg	This file contains the parameter values used in IP configuration.
component.xml	Contains the ipxact:component information of the IP.
design.xml	Documents the configuration parameters of the IP in IP-XACT 2014 format.
rtl/<Component name>.v	This file provides an example RTL top file that instantiates the IP core.
rtl/<Component name>_bb.v	This file provides the synthesis black box.
misc/<Component name>_tmpl.v misc /<Component name>_tmpl.vhd	These files provide instance templates for the IP core.
eval/eval_top.v	Top level RTL files that may be used for running Lattice Radiant software flow check (synthesis to export) on the generated IP. Without this, the Radiant software map process fails due to not enough I/O. This is mainly used for checking resource utilization and fmax for the selected IP configuration, this is not for implementation.
eval/dut_inst.v	A sample instantiation of the generated IP. This is included by the eval_top.v.
eval/dut_params.v	Lists the equivalent localparams of the user settings. This is included by the eval_top.v.
eval/lscc_dummy_model_lfsr.v	A simple linear-feedback shift register.
eval/lscc_apb_master_dummy.v	An APB master made of LFSR. This has no real APB function; it is only used for STA check.
eval/lscc_dfi_master_model_dummy.v	A DFI master made of LFSR. This has no real DFI function; it is only used for STA check.
eval/clock_constraint.sdc	A sample constraint on the input clocks. The user don't need this in design.
eval/constraint.pdc	Constraints for the IP, some of the constraints in this IP should be copied to the Radiant project's constraint file. Please see comments in this constraint for guidance.
eval/ select_protocol.py	Script that is executed by Radiant during IP generation.

3.2. Running Functional Simulation

Below are the steps for running simulation.

1. Add the top-level hardware evaluation design file, eval_top.sv in the project, click **File** Tab, then select **Add** in the drop down menu, and then click **Existing File**. Then select *<Component name>/eval/eval_top.sv* and click Add.
2. Add the top-level testbench file, tb_top.v in the project as a simulation file, click **File** Tab, then select **Add** in the drop down menu, and then click **Existing Simulation File**. Then select *<Component name>/testbench/tb_top.sv* and click Add.
3. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in Figure 3.4.

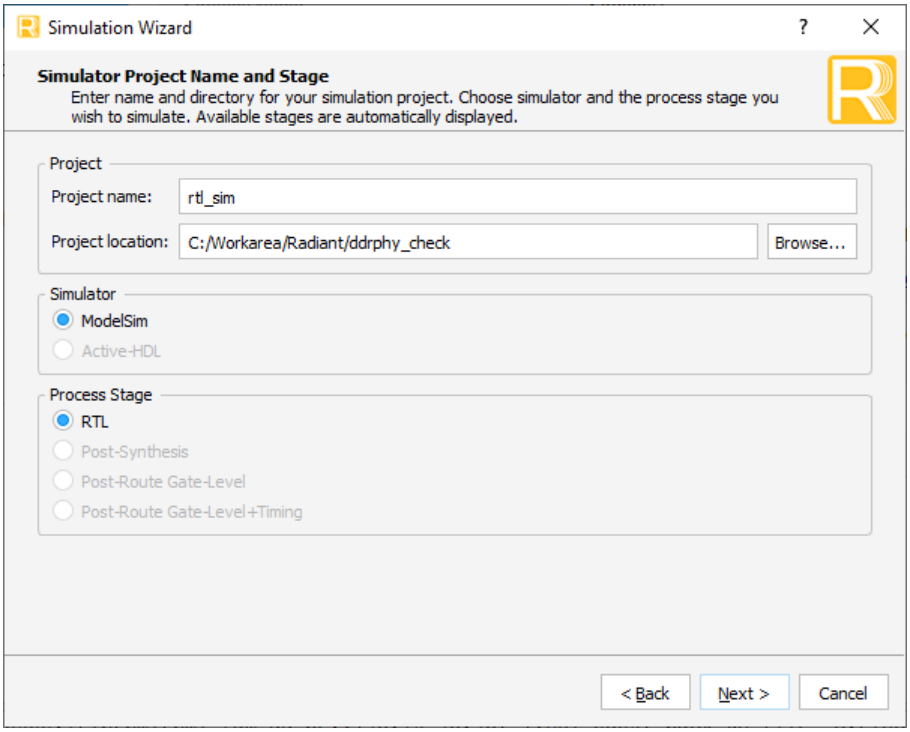

Figure 3.4. Simulation Wizard

4. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 3.5. Notice that the **Source Files** area automatically add the files in the testbench folder.

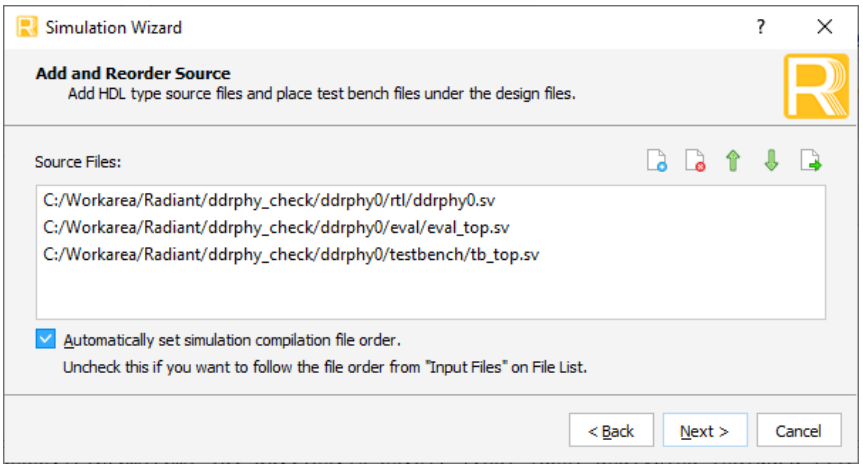

Figure 3.5. Adding and Reordering Source

5. Click **Next**. The **Summary** window is shown. Click **Finish** to run the simulation. The simulation initially runs for 1 µs. The user should execute the following tcl command in the Modelsim tcl window to continue the simulation until the end: run –all. The user might also want to configure the waveform such logging all/some signals before continuing the simulation.

   Notes:

   - It is necessary to follow the procedure above until it is fully automated in the Lattice Radiant software suite.
   - The following Error messages are expected in the log messages due to shortening of reset and CKE initialization, please ignore them.

```
# tb_top.LP4MEM_00.mem_x16_00.ins_1ch.<protected> 26083125000 : ###
lpddr4_debug RESET_n high input
# tb_top.LP4MEM_01.mem_x16_01.ins_1ch.<protected> 26083125000 : ###
lpddr4_debug RESET_n high input
# Error: tb_top.LP4MEM_01.mem_x16_01.ins_1ch.<protected>.<protected>
26083125000  tINIT1 Error.
# Error: tb_top.LP4MEM_00.mem_x16_00.ins_1ch.<protected>.<protected>
26083125000  tINIT1 Error.
# tb_top.LP4MEM_00.mem_x16_00.ins_1ch.<protected> 32804075000 : ###
lpddr4_debug CKE high input
# tb_top.LP4MEM_01.mem_x16_01.ins_1ch.<protected> 32804075000 : ###
lpddr4_debug CKE high input
# Error: tb_top.LP4MEM_01.mem_x16_01.ins_1ch.<protected>.<protected>
32804075000  tINIT3 Error.
# Error: tb_top.LP4MEM_00.mem_x16_00.ins_1ch.<protected>.<protected>
32804075000  tINIT3 Error.
```

3.3. Constraining the IP

The constraint files <ip_instance path>/eval/clock_constraint.sdc and <ip_instance path>/eval/constraint.sdc described in Table 3.1 is generated based on the IP configuration selected by the user. The content of this file should be included to the top-level design constraint files. Please refer to the in-line comments in these constraint files for more details on how to constrain the IP.

Appendix A. Resource Utilization

Table A.1 shows the configuration and resource utilization for LAV-AT-500E-2LFG1156C using Synplify Pro of Lattice Radiant software 2022.1.

Table A.1. Resource Utilization

Configuration	sclk_o Fmax (MHz)	Registers	LUTs	EBR	DSP
INTERFACE_TYPE=LPDDR4, BUS_WIDTH=16, Others=default	270.929	1118	1798	8	0
Default	238.607	1124	1793	8	0
INTERFACE_TYPE=LPDDR4, BUS_WIDTH=64, Others=default	244.141	1124	1801	8	0
INTERFACE_TYPE=DDR4, BUS_WIDTH=16, Others=default	274.424	1134	1804	8	0
INTERFACE_TYPE=DDR4, BUS_WIDTH=32, Others=default	287.026	1134	1802	8	0
INTERFACE_TYPE=DDR4, BUS_WIDTH=64, Others=default	268.025	1134	1809	8	0
INTERFACE_TYPE=DDR4, BUS_WIDTH=40, SB_ECC_EN is Checked, Others=default	286.451	1134	1786	8	0
INTERFACE_TYPE=DDR4, BUS_WIDTH=72, SB_ECC_EN is Checked, Others=default	285.878	1134	1779	8	0

References

- https://www.latticesemi.com/en/Products/FPGAandCPLD/Avant
- Lattice Radiant Software User Guide
- http://www.jedec.org/

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

Revision History

Revision 1.0, November 2022

Section	Change Summary
Introduction	• Updated device name, software version and resource utilization in Table 1.1. • Updated Features section to add DDR4 Support and Self-Calibrating Logic, and updated the supported frequencies and DDR data widths.
Functional Description	• Updated Table 2.1 to add DDR4-specific signals • Updated Table 2.2 and Table 2.3 to add DDR4-related attributes. • Updated ddr_type and addr_translation fields of Feature Control Register section. • Added pll_ref_clk field in Clock Control and Status Register section. • Added ddr_ck_dis and updated dfi_cke in Initialization Control Register section. • Added mem_verf_training_en and mc_vref_training_en in Training Operation Register section. • Added Scratch Registers in Table 2.4 and added section 2.4.12 for these registers. • Removed Read Gate Leveling section and added Self-Calibrating Logic section.
IP Generation, Simulation, and Validation	• Updated section name. • Updated Figure 3.1, Figure 3.2, and Figure 3.3 in Generating the IP section. • Updated Running Functional Simulation section. • Added Constraining the IP section.
Appendix A. Resource Utilization	Added Resource Utilization section.
References	Added web page references of Avant and JEDEC.

Revision 0.8, May 2022

Section	Change Summary
All	Preliminary Release

www.latticesemi.com