

# External Memory Interface Handbook Volume 2: Design Guidelines

Updated for Intel® Quartus® Prime Design Suite: **17.0**



[Subscribe](#)



[Send Feedback](#)

**EMI\_DG | 2017.05.08**  
Latest document on the web: [PDF](#) | [HTML](#)



## Contents

---

<b>1. Planning Pin and FPGA Resources.....</b>	<b>9</b>
1.1. Interface Pins.....	9
1.1.1. Estimating Pin Requirements.....	12
1.1.2. DDR, DDR2, DDR3, and DDR4 SDRAM Clock Signals.....	13
1.1.3. DDR, DDR2, DDR3, and DDR4 SDRAM Command and Address Signals.....	13
1.1.4. DDR, DDR2, DDR3, and DDR4 SDRAM Data, Data Strobes, DM/DBI, and Optional ECC Signals.....	14
1.1.5. DDR, DDR2, DDR3, and DDR4 SDRAM DIMM Options.....	15
1.1.6. QDR II, QDR II+, and QDR II+ Xtreme SRAM Clock Signals.....	18
1.1.7. QDR II, QDR II+ and QDR II+ Xtreme SRAM Command Signals.....	19
1.1.8. QDR II, QDR II+ and QDR II+ Xtreme SRAM Address Signals.....	19
1.1.9. QDR II, QDR II+ and QDR II+ Xtreme SRAM Data, BWS, and QVLD Signals....	20
1.1.10. QDR IV SRAM Clock Signals.....	20
1.1.11. QDR IV SRAM Commands and Addresses, AP, and AINV Signals.....	21
1.1.12. QDR IV SRAM Data, DINV, and QVLD Signals.....	22
1.1.13. RLDRAM II and RLDRAM 3 Clock Signals.....	23
1.1.14. RLDRAM II and RLDRAM 3 Commands and Addresses.....	24
1.1.15. RLDRAM II and RLDRAM 3 Data, DM and QVLD Signals.....	24
1.1.16. LPDDR2 and LPDDR3 Clock Signal.....	25
1.1.17. LPDDR2 and LPDDR3 Command and Address Signal.....	26
1.1.18. LPDDR2 and LPDDR3 Data, Data Strobe, and DM Signals.....	26
1.1.19. Maximum Number of Interfaces.....	26
1.1.20. OCT Support .....	37
1.2. Guidelines for Intel Arria® 10 External Memory Interface IP.....	38
1.2.1. General Pin-Out Guidelines for Arria 10 EMIF IP.....	38
1.2.2. Resource Sharing Guidelines for Arria 10 EMIF IP.....	43
1.3. Guidelines for Intel Stratix® 10 External Memory Interface IP.....	45
1.3.1. General Pin-Out Guidelines for Stratix 10 EMIF IP.....	45
1.3.2. Resource Sharing Guidelines for Stratix 10 EMIF IP.....	50
1.4. Guidelines for UniPHY-based External Memory Interface IP.....	51
1.4.1. General Pin-out Guidelines for UniPHY-based External Memory Interface IP.....	51
1.4.2. Pin-out Rule Exceptions for ×36 Emulated QDR II and QDR II+ SRAM Interfaces in Arria II, Stratix III and Stratix IV Devices.....	53
1.4.3. Pin-out Rule Exceptions for RLDRAM II and RLDRAM 3 Interfaces.....	58
1.4.4. Pin-out Rule Exceptions for QDR II and QDR II+ SRAM Burst-length-of-two Interfaces.....	60
1.4.5. Pin Connection Guidelines Tables.....	61
1.4.6. PLLs and Clock Networks.....	71
1.5. Using PLL Guidelines.....	75
1.6. PLL Cascading.....	76
1.7. DLL.....	76
1.8. Other FPGA Resources.....	78
1.9. Document Revision History.....	78
<b>2. DDR2, DDR3, and DDR4 SDRAM Board Design Guidelines.....</b>	<b>81</b>
2.1. Leveling and Dynamic Termination.....	82
2.1.1. Read and Write Leveling.....	82
2.1.2. Dynamic ODT.....	84



2.1.3. Dynamic On-Chip Termination.....	84
2.1.4. Dynamic On-Chip Termination in Stratix III and Stratix IV Devices.....	85
2.1.5. Dynamic OCT in Stratix V Devices.....	87
2.1.6. Dynamic On-Chip Termination (OCT) in Arria 10 and Stratix 10 Devices.....	87
2.2. DDR2 Terminations and Guidelines.....	87
2.2.1. Termination for DDR2 SDRAM.....	87
2.2.2. DDR2 Design Layout Guidelines.....	93
2.2.3. General Layout Guidelines.....	93
2.2.4. Layout Guidelines for DDR2 SDRAM Interface.....	94
2.3. DDR3 Terminations in Arria V, Cyclone V, Stratix III, Stratix IV, and Stratix V.....	97
2.3.1. Terminations for Single-Rank DDR3 SDRAM Unbuffered DIMM.....	98
2.3.2. Terminations for Multi-Rank DDR3 SDRAM Unbuffered DIMM.....	99
2.3.3. Terminations for DDR3 SDRAM Registered DIMM.....	100
2.3.4. Terminations for DDR3 SDRAM Load-Reduced DIMM.....	100
2.3.5. Terminations for DDR3 SDRAM Components With Leveling.....	101
2.4. DDR3 and DDR4 on Arria 10 and Stratix 10 Devices.....	101
2.4.1. Dynamic On-Chip Termination (OCT) in Arria 10 and Stratix 10 Devices.....	101
2.4.2. Dynamic On-Die Termination (ODT) in DDR4.....	102
2.4.3. Choosing Terminations on Arria 10 Devices.....	102
2.4.4. On-Chip Termination Recommendations for DDR3 and DDR4 on Arria 10 Devices.....	102
2.5. Layout Approach.....	103
2.6. Channel Signal Integrity Measurement.....	104
2.6.1. Importance of Accurate Channel Signal Integrity Information.....	104
2.6.2. Understanding Channel Signal Integrity Measurement.....	104
2.6.3. How to Enter Calculated Channel Signal Integrity Values.....	105
2.6.4. Guidelines for Calculating DDR3 Channel Signal Integrity.....	106
2.6.5. Guidelines for Calculating DDR4 Channel Signal Integrity.....	108
2.7. Design Layout Guidelines.....	111
2.7.1. General Layout Guidelines.....	112
2.7.2. Layout Guidelines for DDR3 and DDR4 SDRAM Interfaces.....	113
2.7.3. Length Matching Rules.....	116
2.7.4. Spacing Guidelines.....	117
2.7.5. Layout Guidelines for DDR3 and DDR4 SDRAM Wide Interface (>72 bits).....	118
2.8. Package Deskew.....	121
2.8.1. Package Deskew Recommendation for Stratix V Devices.....	121
2.8.2. DQ/DQS/DM Deskew.....	122
2.8.3. Address and Command Deskew.....	122
2.8.4. Package Deskew Recommendations for Arria 10 and Stratix 10 Devices.....	122
2.8.5. Deskew Example.....	123
2.8.6. Package Migration.....	124
2.8.7. Package Deskew for RLDRAM II and RLDRAM 3.....	125
2.9. Document Revision History.....	125
<b>3. Dual-DIMM DDR2 and DDR3 SDRAM Board Design Guidelines.....</b>	<b>128</b>
3.1. General Layout Guidelines.....	128
3.2. Dual-Slot Unbuffered DDR2 SDRAM.....	129
3.2.1. Overview of ODT Control.....	130
3.2.2. DIMM Configuration.....	131
3.2.3. Dual-DIMM Memory Interface with Slot 1 Populated.....	132
3.2.4. Dual-DIMM with Slot 2 Populated.....	133



3.2.5. Dual-DIMM Memory Interface with Both Slot 1 and Slot 2 Populated.....	134
3.2.6. Dual-DIMM DDR2 Clock, Address, and Command Termination and Topology....	137
3.2.7. Control Group Signals.....	138
3.2.8. Clock Group Signals.....	138
3.3. Dual-Slot Unbuffered DDR3 SDRAM.....	138
3.3.1. Comparison of DDR3 and DDR2 DQ and DQS ODT Features and Topology.....	139
3.3.2. Dual-DIMM DDR3 Clock, Address, and Command Termination and Topology...	139
3.3.3. FPGA OCT Features.....	140
3.4. Document Revision History.....	141
<b>4. LPDDR2 and LPDDR3 SDRAM Board Design Guidelines.....</b>	<b>142</b>
4.1. LPDDR2 Guidance.....	142
4.1.1. LPDDR2 SDRAM Configurations.....	142
4.1.2. OCT Signal Terminations for Arria V and Cyclone V Devices.....	145
4.1.3. General Layout Guidelines.....	148
4.1.4. LPDDR2 Layout Guidelines.....	148
4.2. LPDDR3 Guidance.....	150
4.2.1. Signal Integrity, Board Skew, and Board Setting Parameters.....	151
4.2.2. LPDDR3 Layout Guidelines.....	151
4.2.3. Package Deskew.....	151
4.3. Document Revision History.....	155
<b>5. RLDRAM II and RLDRAM 3 Board Design Guidelines.....</b>	<b>156</b>
5.1. RLDRAM II Configurations.....	157
5.2. RLDRAM 3 Configurations.....	158
5.3. Signal Terminations.....	160
5.3.1. Input to the FPGA from the RLDRAM Components.....	162
5.3.2. Outputs from the FPGA to the RLDRAM II and RLDRAM 3 Components.....	162
5.3.3. RLDRAM II Termination Schemes.....	163
5.3.4. RLDRAM 3 Termination Schemes.....	163
5.4. PCB Layout Guidelines.....	164
5.5. General Layout Guidelines.....	165
5.6. RLDRAM II and RLDRAM 3 Layout Guidelines.....	165
5.7. Layout Approach.....	167
5.7.1. Arria V and Stratix V Board Setting Parameters.....	168
5.7.2. Arria 10 Board Setting Parameters.....	168
5.8. Package Deskew for RLDRAM II and RLDRAM 3.....	168
5.9. Document Revision History.....	169
<b>6. QDR II and QDR-IV SRAM Board Design Guidelines.....</b>	<b>170</b>
6.1. QDR II SRAM Configurations.....	171
6.2. Signal Terminations.....	172
6.2.1. Output from the FPGA to the QDR II SRAM Component.....	174
6.2.2. Input to the FPGA from the QDR II SRAM Component.....	174
6.2.3. Termination Schemes.....	174
6.3. General Layout Guidelines.....	176
6.4. QDR II Layout Guidelines.....	177
6.5. QDR II SRAM Layout Approach.....	178
6.6. Package Deskew for QDR II and QDR-IV.....	180
6.7. QDR-IV Layout Approach.....	180
6.8. QDR-IV Layout Guidelines.....	181
6.9. Document Revision History.....	182



<b>7. Implementing and Parameterizing Memory IP.....</b>	<b>184</b>
7.1. Installing and Licensing IP Cores.....	184
7.2. Design Flow.....	185
7.2.1. IP Catalog Design Flow.....	186
7.2.2. Qsys System Integration Tool Design Flow.....	193
7.3. UniPHY-Based External Memory Interface IP.....	195
7.3.1. Qsys Interfaces.....	196
7.3.2. Generated Files for Memory Controllers with the UniPHY IP.....	215
7.3.3. Parameterizing Memory Controllers.....	218
7.3.4. Board Settings .....	232
7.3.5. Controller Settings for UniPHY IP.....	243
7.3.6. Diagnostics for UniPHY IP.....	247
7.4. Intel Arria 10 External Memory Interface IP.....	248
7.4.1. Qsys Interfaces.....	248
7.4.2. Generated Files for Arria 10 External Memory Interface IP.....	260
7.4.3. Arria 10 EMIF IP DDR4 Parameters.....	264
7.4.4. Arria 10 EMIF IP DDR3 Parameters.....	286
7.4.5. Arria 10 EMIF IP LPDDR3 Parameters.....	305
7.4.6. Arria 10 EMIF IP QDR-IV Parameters.....	320
7.4.7. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters.....	332
7.4.8. Arria 10 EMIF IP RLDRAM 3 Parameters.....	343
7.4.9. Equations for Arria 10 EMIF IP Board Skew Parameters.....	355
7.5. Intel Stratix 10 External Memory Interface IP.....	361
7.5.1. Qsys Interfaces.....	361
7.5.2. Generated Files for Stratix 10 External Memory Interface IP.....	372
7.5.3. Stratix 10 EMIF IP DDR4 Parameters.....	377
7.5.4. Stratix 10 EMIF IP DDR3 Parameters.....	397
7.5.5. Stratix 10 EMIF IP LPDDR3 Parameters.....	413
7.5.6. Stratix 10 EMIF IP QDR-IV Parameters.....	428
7.5.7. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters.....	440
7.5.8. Stratix 10 EMIF IP RLDRAM 3 Parameters.....	451
7.6. Document Revision History.....	463
<b>8. Simulating Memory IP.....</b>	<b>468</b>
8.1. Simulation Options.....	468
8.2. Simulation Walkthrough with UniPHY IP.....	470
8.2.1. Simulation Scripts.....	471
8.2.2. Preparing the Vendor Memory Model.....	471
8.2.3. Functional Simulation with Verilog HDL.....	474
8.2.4. Functional Simulation with VHDL.....	474
8.2.5. Simulating the Example Design.....	475
8.2.6. UniPHY Abstract PHY Simulation.....	476
8.2.7. PHY-Only Simulation.....	478
8.2.8. Post-fit Functional Simulation.....	479
8.2.9. Simulation Issues.....	481
8.3. Simulation Walkthrough with Arria 10 EMIF IP.....	483
8.3.1. Skip Calibration Versus Full Calibration.....	483
8.3.2. Arria 10 Abstract PHY Simulation.....	484
8.3.3. Simulation Scripts.....	485
8.3.4. Functional Simulation with Verilog HDL.....	485



8.3.5. Functional Simulation with VHDL.....	486
8.3.6. Simulating the Example Design.....	487
8.4. Simulation Walkthrough with Stratix 10 EMIF IP.....	488
8.4.1. Skip Calibration Versus Full Calibration.....	489
8.4.2. Simulation Scripts.....	490
8.4.3. Functional Simulation with Verilog HDL.....	490
8.4.4. Functional Simulation with VHDL.....	491
8.4.5. Simulating the Example Design.....	491
8.5. Document Revision History.....	493
<b>9. Analyzing Timing of Memory IP.....</b>	<b>494</b>
9.1. Memory Interface Timing Components.....	495
9.1.1. Source-Synchronous Paths.....	495
9.1.2. Calibrated Paths.....	495
9.1.3. Internal FPGA Timing Paths.....	496
9.1.4. Other FPGA Timing Parameters.....	496
9.2. FPGA Timing Paths.....	497
9.2.1. Arria II Device PHY Timing Paths.....	497
9.2.2. Stratix III and Stratix IV PHY Timing Paths.....	499
9.2.3. Arria V, Arria V GZ, Arria 10, Cyclone V, and Stratix V Timing paths.....	501
9.3. Timing Constraint and Report Files for UniPHY IP.....	503
9.4. Timing Constraint and Report Files for Arria 10 EMIF IP.....	505
9.5. Timing Constraint and Report Files for Stratix 10 EMIF IP.....	506
9.6. Timing Analysis Description .....	508
9.6.1. UniPHY IP Timing Analysis.....	509
9.6.2. Timing Analysis Description for Arria 10 EMIF IP.....	517
9.6.3. Timing Analysis Description for Stratix 10 EMIF IP.....	521
9.7. Timing Report DDR.....	524
9.8. Report SDC.....	527
9.9. Calibration Effect in Timing Analysis.....	528
9.9.1. Calibration Emulation for Calibrated Path.....	528
9.9.2. Calibration Error or Quantization Error.....	528
9.9.3. Calibration Uncertainties.....	529
9.9.4. Memory Calibration.....	529
9.10. Timing Model Assumptions and Design Rules.....	529
9.10.1. Memory Clock Output Assumptions.....	530
9.10.2. Write Data Assumptions.....	531
9.10.3. Read Data Assumptions.....	533
9.10.4. DLL Assumptions.....	534
9.10.5. PLL and Clock Network Assumptions for Stratix III Devices.....	534
9.11. Common Timing Closure Issues.....	535
9.11.1. Missing Timing Margin Report.....	535
9.11.2. Incomplete Timing Margin Report.....	535
9.11.3. Read Capture Timing.....	536
9.11.4. Write Timing.....	536
9.11.5. Address and Command Timing.....	536
9.11.6. PHY Reset Recovery and Removal.....	537
9.11.7. Clock-to-Strobe (for DDR and DDR2 SDRAM Only).....	537
9.11.8. Read Resynchronization and Write Leveling Timing (for SDRAM Only).....	537
9.12. Optimizing Timing.....	538



9.13. Timing Derivation Methodology for Multiple Chip Select DDR2 and DDR3 SDRAM Designs.....	539
9.13.1. Multiple Chip Select Configuration Effects.....	540
9.13.2. Timing Derivation using the Board Settings.....	542
9.14. Early I/O Timing Estimation for Arria 10 EMIF IP.....	545
9.14.1. Performing Early I/O Timing Analysis for Arria 10 EMIF IP.....	546
9.15. Early I/O Timing Estimation for Stratix 10 EMIF IP.....	547
9.15.1. Performing Early I/O Timing Analysis for Stratix 10 EMIF IP.....	547
9.16. Performing I/O Timing Analysis.....	548
9.16.1. Performing I/O Timing Analysis with Third-Party Simulation Tools.....	549
9.16.2. Performing Advanced I/O Timing Analysis with Board Trace Delay Model.....	549
9.17. Document Revision History.....	550
<b>10. Debugging Memory IP.....</b>	<b>551</b>
10.1. Resource and Planning Issues.....	551
10.1.1. Dedicated IOE DQS Group Resources and Pins.....	551
10.1.2. Dedicated DLL Resources.....	552
10.1.3. Specific PLL Resources.....	553
10.1.4. Specific Global, Regional and Dual-Regional Clock Net Resources.....	553
10.1.5. Planning Your Design.....	553
10.1.6. Optimizing Design Utilization.....	554
10.2. Interface Configuration Performance Issues.....	554
10.2.1. Interface Configuration Bottleneck and Efficiency Issues.....	555
10.3. Functional Issue Evaluation.....	556
10.3.1. Correct Combination of the Quartus Prime Software and ModelSim - Intel FPGA Edition Device Models.....	556
10.3.2. Intel IP Memory Model.....	557
10.3.3. Vendor Memory Model.....	557
10.3.4. Insufficient Memory in Your PC.....	557
10.3.5. Transcript Window Messages.....	558
10.3.6. Passing Simulation.....	559
10.3.7. Modifying the Example Driver to Replicate the Failure.....	559
10.4. Timing Issue Characteristics.....	560
10.4.1. Evaluating FPGA Timing Issues.....	561
10.4.2. Evaluating External Memory Interface Timing Issues.....	562
10.5. Verifying Memory IP Using the Signal Tap II Logic Analyzer.....	562
10.5.1. Signals to Monitor with the Signal Tap II Logic Analyzer.....	564
10.6. Hardware Debugging Guidelines.....	565
10.6.1. Create a Simplified Design that Demonstrates the Same Issue.....	565
10.6.2. Measure Power Distribution Network.....	565
10.6.3. Measure Signal Integrity and Setup and Hold Margin.....	565
10.6.4. Vary Voltage.....	566
10.6.5. Use Freezer Spray and Heat Gun.....	566
10.6.6. Operate at a Lower Speed.....	566
10.6.7. Determine Whether the Issue Exists in Previous Versions of Software.....	566
10.6.8. Determine Whether the Issue Exists in the Current Version of Software.....	566
10.6.9. Try A Different PCB.....	567
10.6.10. Try Other Configurations.....	567
10.6.11. Debugging Checklist.....	568
10.7. Categorizing Hardware Issues.....	568
10.7.1. Signal Integrity Issues.....	569



10.7.2. Hardware and Calibration Issues.....	571
10.8. EMIF Debug Toolkit Overview.....	575
10.9. Document Revision History.....	575
<b>11. Optimizing the Controller.....</b>	<b>577</b>
11.1. Factors Affecting Efficiency.....	577
11.1.1. Interface Standard.....	578
11.1.2. Bank Management Efficiency.....	578
11.1.3. Data Transfer.....	580
11.2. Ways to Improve Efficiency.....	581
11.2.1. DDR2 SDRAM Controller.....	582
11.2.2. Auto-Precharge Commands.....	582
11.2.3. Additive Latency.....	584
11.2.4. Bank Interleaving.....	585
11.2.5. Command Queue Look-Ahead Depth.....	587
11.2.6. Additive Latency and Bank Interleaving.....	588
11.2.7. User-Controlled Refresh.....	589
11.2.8. Frequency of Operation.....	590
11.2.9. Burst Length.....	591
11.2.10. Series of Reads or Writes.....	591
11.2.11. Data Reordering.....	591
11.2.12. Starvation Control.....	592
11.2.13. Command Reordering.....	593
11.2.14. Bandwidth.....	594
11.2.15. Efficiency Monitor.....	595
11.3. Document Revision History.....	596
<b>12. PHY Considerations.....</b>	<b>598</b>
12.1. Core Logic and User Interface Data Rate.....	598
12.2. Hard and Soft Memory PHY.....	599
12.3. Sequencer.....	599
12.4. PLL, DLL and OCT Resource Sharing.....	600
12.5. Pin Placement Consideration.....	601
12.6. Document Revision History.....	602
<b>13. Power Estimation Methods for External Memory Interfaces.....</b>	<b>603</b>
13.1. Performing Vector-Based Power Analysis with the Power Analyzer.....	604
13.2. Document Revision History.....	604

## 1. Planning Pin and FPGA Resources

---

This information is for board designers who must determine FPGA pin usage, to create board layouts. The board design process sometimes occurs concurrently with the RTL design process.

Use this document with the External Memory Interfaces chapter of the relevant device family handbook.

Typically, all external memory interfaces require the following FPGA resources:

- Interface pins
- PLL and clock network
- DLL
- Other FPGA resources—for example, core fabric logic, and on-chip termination (OCT) calibration blocks

After you know the requirements for your external memory interface, you can start planning your system. The I/O pins and internal memory cannot be shared for other applications or external memory interfaces. However, if you do not have enough PLLs, DLLs, or clock networks for your application, you may share these resources among multiple external memory interfaces or modules in your system.

Ideally, any interface should reside entirely in a single bank; however, interfaces that span multiple adjacent banks or the entire side of a device are also fully supported. In addition, you may also have wraparound memory interfaces, where the design uses two adjacent sides of the device and the memory interface logic resides in a device quadrant. In some cases, top or bottom bank interfaces have higher supported clock rates than left or right or wraparound interfaces.

### 1.1. Interface Pins

Any I/O banks that do not support transceiver operations in Arria® II, Arria V, Arria 10, Stratix® III, Stratix IV, and Stratix V devices support external memory interfaces. However, DQS (data strobe or data clock) and DQ (data) pins are listed in the device pin tables and fixed at specific locations in the device. You must adhere to these pin locations as these locations are optimized in routing to minimize skew and maximize margin. Always check the external memory interfaces chapters from the device handbooks for the number of DQS and DQ groups supported in a particular device and the pin table for the actual locations of the DQS and DQ pins.

The following table lists a summary of the number of pins required for various example memory interfaces. This table uses series OCT with calibration and parallel OCT with calibration, or dynamic calibrated OCT, when applicable, shown by the usage of R<sub>UP</sub> and R<sub>DN</sub> pins or RZQ pin.

**Table 1.** Pin Counts for Various Example External Memory Interfaces <sup>(1)</sup> <sup>(2)</sup>

External Memory Interface	FPGA DQS Group Size	Number of DQ Pins	Number of DQS/CQ /QK Pins	Number of Control Pins <sup>(19)</sup>	Number of Address Pins <sup>(3)</sup>	Number of Command Pins	Number of Clock Pins	R <sub>UP/R<sub>D</sub>N</sub> Pins <sup>(4)</sup>	R <sub>ZQ</sub> Pins <sup>(11)</sup>	Total Pins (with R <sub>UP/R<sub>D</sub>N</sub> pins)	Total Pins (with R <sub>ZQ</sub> pin)
LPDDR2	x8	8	2	1	10	2	2	N/A	1	N/A	26
		16	4	2	10	2	2	N/A	1	N/A	37
		72	18	9	10	2	2	N/A	1	N/A	114
LPDDR3	x8	16	4	2	10	2	2	N/A	1	N/A	37
		72	18	9	10	2	2	N/A	1	N/A	114
DDR4 SDRAM <sup>(12)</sup>	x4	4	2	0 <sup>(7)</sup>	17	11	2	N/A	1	N/A	37
	x8	8	2	1	17	11	2	N/A	1	N/A	42
		16	4	2	17	10 <sup>(13)</sup>	2	N/A	1	N/A	52
DDR3 SDRAM <sup>(5)</sup> <sup>(6)</sup>	x4	4	2	0 <sup>(7)</sup>	14	10	2	2	1	34	33
	x8	8	2	1	14	10	2	2	1	39	38
		16	4	2	14	10	2	2	1	50	49
DDR2 SDRAM <sup>(8)</sup>	x4	4	1	1 <sup>(7)</sup>	15	9	2	2	1	34	33
	x8	8	1 <sup>(9)</sup>	1	15	9	2	2	1	38	37
		16	2 <sup>(9)</sup>	2	15	9	2	2	1	48	47
DDR SDRAM <sup>(6)</sup>	x4	4	1	1 <sup>(7)</sup>	14	7	2	2	1	29	28
	x8	8	1	1	14	7	2	2	1	33	35
		16	2	2	14	7	2	2	1	43	42
QDR II + / II+ Xtreme SRAM <sup>(18)</sup>	x18	36	2	2	19	3 <sup>(10)</sup>	2 <sup>(15)</sup>	2	1	66	65
	x36	72	2	4	18	3 <sup>(10)</sup>	2 <sup>(15)</sup>	2	1	103	102
QDR II SRAM	x9	18	2	1	19	2	4 <sup>(16)</sup>	2	1	48	47
	x18	36	2	2	18	2	4 <sup>(16)</sup>	2	1	66	65
	x36	72	2	4	17	2	4 <sup>(16)</sup>	2	1	103	102
QDR IV SRAM <sup>(20)</sup>	x18	36	8	5	22	7	10 <sup>(17)</sup>	N/A	1	N/A	89
	x36	72	8	5	21	7	10 <sup>(17)</sup>	N/A	1	N/A	124
RLDRA M 3 CIO <sup>(14)</sup>	x9	18	4	2	20	8 <sup>(10)</sup>	6 <sup>(17)</sup>	N/A	1	N/A	59
		36	8	2	19	8 <sup>(10)</sup>	6 <sup>(17)</sup>	N/A	1	N/A	80
RLDRA M II CIO	x9	9	2	1	22	7 <sup>(10)</sup>	4 <sup>(17)</sup>	2	1	47	46

*continued...*



External Memory Interface	FPGA DQS Group Size	Number of DQ Pins	Number of DQS/CQ /QK Pins	Number of Control Pins (19)	Number of Address Pins (3)	Number of Command Pins	Number of Clock Pins	R <sub>UP</sub> /R <sub>DN</sub> Pins (4)	R <sub>ZQ</sub> Pins (11)	Total Pins (with R <sub>UP</sub> /R <sub>DN</sub> pins)	Total Pins (with R <sub>ZQ</sub> pin)
		18	4	1	21	7 (10)	4 (17)	2	1	57	56
	×18	36	4	1	20	7 (10)	6 (17)	2	1	76	75

Notes to table:

- These example pin counts are derived from memory vendor data sheets. Check the exact number of addresses and command pins of the memory devices in the configuration that you are using.
- PLL and DLL input reference clock pins are not counted in this calculation.
- The number of address pins depends on the memory device density.
- Some DQS or DQ pins are dual purpose and can also be required as R<sub>UP</sub>, R<sub>DN</sub>, or configuration pins. A DQS group is lost if you use these pins for configuration or as R<sub>UP</sub> or R<sub>DN</sub> pins for calibrated OCT. Pick R<sub>UP</sub> and R<sub>DN</sub> pins in a DQS group that is not used for memory interface purposes. You may need to place the DQS and DQ pins manually if you place the R<sub>UP</sub> and R<sub>DN</sub> pins in the same DQS group pins.
- The TDQS and TDQS# pins are not counted in this calculation, as these pins are not used in the memory controller.
- Numbers are based on 1-GB memory devices.
- Intel® FPGAs do not support DM pins in ×4 mode with differential DQS signaling.
- Numbers are based on 2-GB memory devices without using differential DQS, RDQS, and RDQS# pin support.
- Assumes single ended DQS mode. DDR2 SDRAM also supports differential DQS, which makes these DQS and DM numbers identical to DDR3 SDRAM.
- The QVLD pin that indicates read data valid from the QDR II+ SRAM or RLDRAM II device, is included in this number.
- R<sub>ZQ</sub> pins are supported by Arria V, Arria 10, Cyclone V, and Stratix V devices.
- Numbers are based on 2-GB discrete device with alert flag and address and command parity pins included.
- DDR4 x16 devices support only a bank group of 1.
- Numbers are based on a 576-MB device.
- These numbers include K and K# clock pins. The CQ and CQ# clock pins are calculated in a separate column.
- These numbers include K, K#, C, and C# clock pins. The CQ and CQ# clock pins are calculated in a separate column.
- These numbers include CK, CK#, DK, and DK# clock pins. QK and QK# clock pins are calculated in a separate column.
- This number is based on a 36864-kilobit device.
- For DDR,DDR2,DDR3,LDDR2, LPDDR3 SDRAM, RLDRAM 3, RLDRAM II, they are DM pins. For QDR II/II+/Extreme, they are BWS pins. For DDR4, they are DM/DBI pins. For QDR IV, they are DINVA[1:0], DINVB[1:0], and AINV.
- This number is based on a 144-Mbit device with address bus inversion and data bus inversion bits included.

**Note:**

Maximum interface width varies from device to device depending on the number of I/O pins and DQS or DQ groups available. Achievable interface width also depends on the number of address and command pins that the design requires. To ensure adequate PLL, clock, and device routing resources are available, you should always test fit any IP in the Quartus® Prime software before PCB sign-off.

Intel devices do not limit the width of external memory interfaces beyond the following requirements:

- Maximum possible interface width in any particular device is limited by the number of DQS groups available.
- Sufficient clock networks are available to the interface PLL as required by the IP.
- Sufficient spare pins exist within the chosen bank or side of the device to include all other address and command, and clock pin placement requirements.
- The greater the number of banks, the greater the skew, hence Intel recommends that you always generate a test project of your desired configuration and confirm that it meets timing.



### 1.1.1. Estimating Pin Requirements

You should use the Quartus Prime software for final pin fitting. However, you can estimate whether you have enough pins for your memory interface using the EMIF Device Selector (for Arria 10 and Stratix 10 devices) on [www.altera.com](http://www.altera.com), or by the following steps:

1. Find out how many read data pins are associated per read data strobe or clock pair, to determine which column of the DQS and DQ group availability ( $\times 4$ ,  $\times 8/\times 9$ ,  $\times 16/\times 18$ , or  $\times 32/\times 36$ ) refer to the pin table.
2. Check the device density and package offering information to see if you can implement the interface in one I/O bank or on one side or on two adjacent sides.

*Note:* If you target Arria II GX devices and you do not have enough I/O pins to have the memory interface on one side of the device, you may place them on the other side of the device. Arria II GX devices allow a memory interface to span across the top and bottom, or left and right sides of the device. For any interface that spans across two different sides, use the wraparound interface performance.

3. Calculate the number of other memory interface pins needed, including any other clocks (write clock or memory system clock), address, command, RUP, RDN, RZQ, and any other pins to be connected to the memory components. Ensure you have enough pins to implement the interface in one I/O bank or one side or on two adjacent sides.

*Note:* a. The DQS groups in Arria II GX devices reside on I/O modules, each consisting of 16 I/O pins. You can only use a maximum of 12 pins per I/O modules when the pins are used as DQS or DQ pins or HSTL/SSTL output or HSTL/SSTL bidirectional pins. When counting the number of available pins for the rest of your memory interface, ensure you do not count the leftover four pins per I/O modules used for DQS, DQ, address and command pins. The leftover four pins can be used as input pins only.

- b. Refer to the device pin-out tables and look for the blank space in the relevant DQS group column to identify the four pins that cannot be used in an I/O module for Arria II GX devices.
- c. If you enable Ping Pong PHY, the IP core exposes two independent Avalon interfaces to user logic, and a single external memory interface of double the width for the data bus and the CS#, CKE, ODT, and CK/CK# signals. The rest remain as if in single interface configuration.

You should test the proposed pin-outs with the rest of your design in the Quartus Prime software (with the correct I/O standard and OCT connections) before finalizing the pin-outs. There can be interactions between modules that are illegal in the Quartus Prime software that you might not know about unless you compile the design and use the Quartus Prime Pin Planner.

#### Related Information

[External Memory Interface Device Selector](#)



### 1.1.2. DDR, DDR2, DDR3, and DDR4 SDRAM Clock Signals

DDR, DDR2, DDR3, and DDR4 SDRAM devices use CK and CK# signals to clock the address and command signals into the memory. Furthermore, the memory uses these clock signals to generate the DQS signal during a read through the DLL inside the memory. The SDRAM data sheet specifies the following timings:

- $t_{DQSCK}$  is the skew between the CK or CK# signals and the SDRAM-generated DQS signal
- $t_{DSH}$  is the DQS falling edge from CK rising edge hold time
- $t_{DSS}$  is the DQS falling edge from CK rising edge setup time
- $t_{DQSS}$  is the positive DQS latching edge to CK rising edge

SDRAM have a write requirement ( $t_{DQSS}$ ) that states the positive edge of the DQS signal on writes must be within  $\pm 25\%$  ( $\pm 90^\circ$ ) of the positive edge of the SDRAM clock input. Therefore, you should generate the CK and CK# signals using the DDR registers in the IOE to match with the DQS signal and reduce any variations across process, voltage, and temperature. The positive edge of the SDRAM clock, CK, is aligned with the DQS write to satisfy  $t_{DQSS}$ .

DDR3 SDRAM can use a daisy-chained control address command (CAC) topology, in which the memory clock must arrive at each chip at a different time. To compensate for the flight-time skew between devices when using the CAC topology, you should employ write leveling.

### 1.1.3. DDR, DDR2, DDR3, and DDR4 SDRAM Command and Address Signals

Command and address signals in SDRAM devices are clocked into the memory device using the CK or CK# signal. These pins operate at single data rate (SDR) using only one clock edge. The number of address pins depends on the SDRAM device capacity. The address pins are multiplexed, so two clock cycles are required to send the row, column, and bank address.

For DDR, DDR2, and DDR3, the CS#, RAS#, CAS#, WE#, CKE, and ODT pins are SDRAM command and control pins. For DDR3 SDRAM, certain topologies such as RDIMM and LRDIMM include RESET#, PAR\_IN (1.5V LVCMOS I/O standard), and ERR\_OUT# (SSTL-15 I/O standard).

The DDR2 SDRAM command and address inputs do not have a symmetrical setup and hold time requirement with respect to the SDRAM clocks, CK, and CK#.

Although DDR4 operates in fundamentally the same way as other SDRAM, there are no longer dedicated pins for RAS#, CAS#, and WE#, as those are now shared with higher-order address pins. DDR4 still has CS#, CKE, ODT, and RESET# pins, similar to DDR3. DDR4 introduces some additional pins, including the ACT# (activate) pin and BG (bank group) pins. Depending on the memory format and the functions enabled, the following pins might also exist in DDR4: PAR (address command parity) pin and the ALERT# pin.

For Intel SDRAM high-performance controllers in Stratix III and Stratix IV devices, the command and address clock is a dedicated PLL clock output whose phase can be adjusted to meet the setup and hold requirements of the memory clock. The command and address clock is also typically half-rate, although a full-rate

implementation can also be created. The command and address pins use the DDIO output circuitry to launch commands from either the rising or falling edges of the clock. The chip select CS#, clock enable CKE, and ODT pins are only enabled for one memory clock cycle and can be launched from either the rising or falling edge of the command and address clock signal. The address and other command pins are enabled for two memory clock cycles and can also be launched from either the rising or falling edge of the command and address clock signal.

In Arria II GX devices, the command and address clock is either shared with the write\_clk\_2x or the mem\_clk\_2x clock.

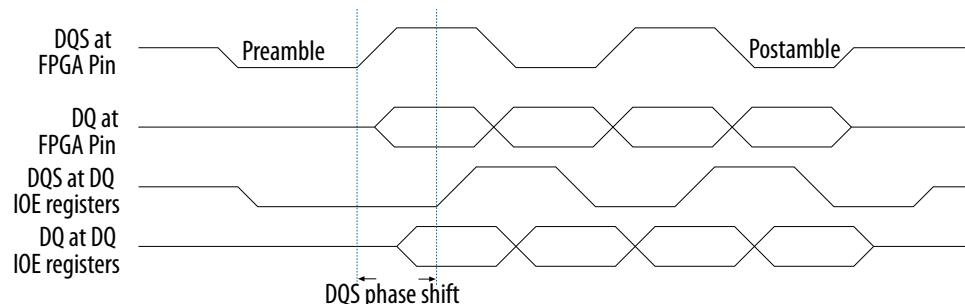
#### **1.1.4. DDR, DDR2, DDR3, and DDR4 SDRAM Data, Data Strobes, DM/DBI, and Optional ECC Signals**

DDR SDRAM uses bidirectional single-ended data strobe (DQS); DDR3 and DDR4 SDRAM use bidirectional differential data strobes. The DQSn pins in DDR2 SDRAM devices are optional but recommended for DDR2 SDRAM designs operating at more than 333 MHz. Differential DQS operation enables improved system timing due to reduced crosstalk and less simultaneous switching noise on the strobe output drivers. The DQ pins are also bidirectional.

Regardless of interface width, DDR SDRAM always operates in  $\times 8$  mode DQS groups. DQ pins in DDR2, DDR3, and DDR4 SDRAM interfaces can operate in either  $\times 4$  or  $\times 8$  mode DQS groups, depending on your chosen memory device or DIMM, regardless of interface width. The  $\times 4$  and  $\times 8$  configurations use one pair of bidirectional data strobe signals, DQS and DQSn, to capture input data. However, two pairs of data strobes, UDQS and UDQS# (upper byte) and LDQS and LDQS# (lower byte), are required by the  $\times 16$  configuration devices. A group of DQ pins must remain associated with its respective DQS and DQSn pins.

The DQ signals are edge-aligned with the DQS signal during a read from the memory and are center-aligned with the DQS signal during a write to the memory. The memory controller shifts the DQ signals by -90 degrees during a write operation to center align the DQ and DQS signals. The PHY IP delays the DQS signal during a read, so that the DQ and DQS signals are center aligned at the capture register. Intel devices use a phase-locked loop (PLL) to center-align the DQS signal with respect to the DQ signals during writes and Intel devices use dedicated DQS phase-shift circuitry to shift the incoming DQS signal during reads. The following figure shows an example where the DQS signal is shifted by 90 degrees for a read from the DDR2 SDRAM.

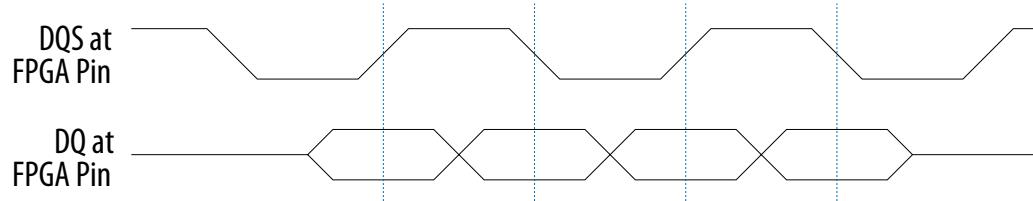
**Figure 1. Edge-aligned DQ and DQS Relationship During a DDR2 SDRAM Read in Burst-of-Four Mode**





The following figure shows an example of the relationship between the data and data strobe during a burst-of-four write.

**Figure 2. DQ and DQS Relationship During a DDR2 SDRAM Write in Burst-of-Four Mode**



The memory device's setup ( $t_{DS}$ ) and hold times ( $t_{DH}$ ) for the DQ and DM pins during writes are relative to the edges of DQS write signals and not the CK or CK# clock. Setup and hold requirements are not necessarily balanced in DDR2 and DDR3 SDRAM, unlike in DDR SDRAM devices.

The DQS signal is generated on the positive edge of the system clock to meet the  $t_{DQSS}$  requirement. DQ and DM signals use a clock shifted -90 degrees from the system clock, so that the DQS edges are centered on the DQ or DM signals when they arrive at the DDR2 SDRAM. The DQS, DQ, and DM board trace lengths need to be tightly matched (within 20 ps).

The SDRAM uses the DM pins during a write operation. Driving the DM pins low shows that the write is valid. The memory masks the DQ signals if the DM pins are driven high. To generate the DM signal, Intel recommends that you use the spare DQ pin within the same DQS group as the respective data, to minimize skew.

The DM signal's timing requirements at the SDRAM input are identical to those for DQ data. The DDR registers, clocked by the -90 degree shifted clock, create the DM signals.

DDR4 supports DM similarly to other SDRAM, except that in DDR4 DM is active LOW and bidirectional, because it supports Data Bus Inversion (DBI) through the same pin. DM is multiplexed with DBI by a Mode Register setting whereby only one function can be enabled at a time. DBI is an input/output identifying whether to store/output the true or inverted data. When enabled, if DBI is LOW, during a write operation the data is inverted and stored inside the DDR4 SDRAM; during a read operation, the data is inverted and output. The data is not inverted if DBI is HIGH. For Arria 10, the DBI (for DDR4) and the DM (for DDR3) pins in each DQS group must be paired with a DQ pin for proper operation.

Some SDRAM modules support error correction coding (ECC) to allow the controller to detect and automatically correct error in data transmission. The 72-bit SDRAM modules contain eight extra data pins in addition to 64 data pins. The eight extra ECC pins should be connected to a single DQS or DQ group on the FPGA.

### 1.1.5. DDR, DDR2, DDR3, and DDR4 SDRAM DIMM Options

Unbuffered DIMMs (UDIMMs) require one set of chip-select (CS#), on-die termination (ODT), clock-enable (CKE), and clock pair (CK/CKn) for every physical rank on the DIMM. Registered DIMMs use only one pair of clocks. DDR3 registered DIMMs require a minimum of two chip-select signals, while DDR4 requires only one.



Compared to the unbuffered DIMMs (UDIMM), registered and load-reduced DIMMs (RDIMMs and LRDIMMs, respectively) use at least two chip-select signals CS#[1:0] in DDR3 and DDR4. Both RDIMMs and LRDIMMs require an additional parity signal for address, RAS#, CAS#, and WE# signals. A parity error signal is asserted by the module whenever a parity error is detected.

LRDIMMs expand on the operation of RDIMMs by buffering the DQ/DQS bus. Only one electrical load is presented to the controller regardless of the number of ranks, therefore only one clock enable (CKE) and ODT signal are required for LRDIMMs, regardless of the number of physical ranks. Because the number of physical ranks may exceed the number of physical chip-select signals, DDR3 LRDIMMs provide a feature known as rank multiplication, which aggregates two or four physical ranks into one larger logical rank. Refer to LRDIMM buffer documentation for details on rank multiplication.

The following table shows UDIMM and RDIMM pin options for DDR, DDR2, and DDR3.

**Table 2. UDIMM and RDIMM Pin Options for DDR, DDR2, and DDR3**

Pins	UDIMM Pins (Single Rank)	UDIMM Pins (Dual Rank)	RDIMM Pins (Single Rank)	RDIMM Pins (Dual Rank)
Data	72 bit DQ[71:0] = {CB[7:0], DQ[63:0]}			
Data Mask	DM[8:0]	DM[8:0]	DM[8:0]	DM[8:0]
Data Strobe (1)	DQS[8:0] and DQS#[8:0]	DQS[8:0] and DQS#[8:0]	DQS[8:0] and DQS#[8:0]	DQS[8:0] and DQS#[8:0]
Address	BA[2:0], A[15:0]- 2 GB: A[13:0] 4 GB: A[14:0] 8 GB: A[15:0]	BA[2:0], A[15:0]- 2 GB: A[13:0] 4 GB: A[14:0] 8 GB: A[15:0]	BA[2:0], A[15:0]- 2 GB: A[13:0] 4 GB: A[14:0] 8 GB: A[15:0]	BA[2:0], A[15:0]- 2 GB: A[13:0] 4 GB: A[14:0] 8 GB: A[15:0]
Clock	CK0/CK0#	CK0/CK0#, CK1/CK1#	CK0/CK0#	CK0/CK0#
Command	ODT, CS#, CKE, RAS#, CAS#, WE#	ODT[1:0], CS#[1:0], CKE[1:0], RAS#, CAS#, WE#	ODT, CS#[1:0], CKE, RAS#, CAS#, WE# <sup>2</sup>	ODT[1:0], CS#[1:0], CKE[1:0], RAS#, CAS#, WE#
Parity	—	—	PAR_IN, ERR_OUT	PAR_IN, ERR_OUT
Other Pins	SA[2:0], SDA, SCL, EVENT#, RESET#			

Note to Table:

1. DQS#[8:0] is optional in DDR2 SDRAM and is not supported in DDR SDRAM interfaces.
2. For single rank DDR2 RDIMM, ignore CS#[1] because it is not used.

The following table shows LRDIMM pin options for DDR3.

**Table 3. LRDIMM Pin Options for DDR3**

Pins	LRDIMM Pins (x4, 2R)	LRDIMM (x4, 4R, RMF=1) <sup>3</sup>	LRDIMM Pins (x4, 4R, RMF=2)	LRDIMM Pins (x4, 8R, RMF=2)	LRDIMM Pins (x4, 8R, RMF=4)	LRDIMM (x8, 4R, RMF=1) <sup>3</sup>	LRDIMM Pins (x8, 4R, RMF=2)
Data	72 bit DQ [71:0]=	72 bit DQ [71:0]=	72 bit DQ [71:0]=	72 bit DQ [71:0]=	72 bit DQ [71:0]=	72 bit DQ [71:0]=	72 bit DQ [71:0]=

*continued...*



Pins	LRDIMM Pins (x4, 2R)	LRDIMM (x4, 4R, RMF=1) <sup>3</sup>	LRDIMM Pins (x4, 4R, RMF=2)	LRDIMM Pins (x4, 8R, RMF=2)	LRDIMM Pins (x4, 8R, RMF=4)	LRDIMM (x8, 4R, RMF=1) <sup>3</sup>	LRDIMM Pins (x8, 4R, RMF=2)
	{CB [7:0], DQ [63:0]}	{CB [7:0], DQ [63:0]}	{CB [7:0], DQ [63:0]}	{CB [7:0], DQ [63:0]}	{CB [7:0], DQ [63:0]}	{CB [7:0], DQ [63:0]}	{CB [7:0], DQ [63:0]}
Data Mask	—	—	—	—	—	DM[8:0]	DM[8:0]
Data Strobe	DQS[17:0] and DQS#[17:0]	DQS[17:0] and DQS#[17:0]	DQS[17:0] and DQS#[17:0]	DQS[17:0] and DQS#[17:0]	DQS[17:0] and DQS#[17:0]	DQS[8:0] and DQS#[8:0]	DQS[8:0] and DQS#[8:0]
Address	BA[2:0], A[15:0] -2GB:A[13:0] 4GB:A[14:0] 8GB:A[15:0]	BA[2:0], A[15:0] -2GB:A[13:0] 4GB:A[14:0] 8GB:A[15:0]	BA[2:0], A[16:0] -4GB:A[14:0] 8GB:A[15:0] 16GB:A[16:0]	BA[2:0], A[16:0] -4GB:A[14:0] 8GB:A[15:0] 16GB:A[16:0]	BA[2:0], A[17:0] -16GB:A[15:0] 32GB:A[16:0] 64GB:A[17:0]	BA[2:0], A[15:0] -2GB:A[13:0] 4GB:A[14:0] 8GB:A[15:0] 16GB:A[16:0]	BA[2:0], A[16:0] -4GB:A[14:0] 8GB:A[15:0] 16GB:A[16:0]
Clock	CK0/CK0#	CK0/CK0#	CK0/CK0#	CK0/CK0#	CK0/CK0#	CK0/CK0#	CK0/CK0#
Command	ODT, CS[1:0]#, CKE, RAS#, CAS#, WE#	ODT, CS[3:0]#, CKE, RAS#, CAS#, WE#	ODT, CS[2:0]#, CKE, RAS#, CAS#, WE#	ODT, CS[3:0]#, CKE, RAS#, CAS#, WE#	ODT, CS[3:0]#, CKE, RAS#, CAS#, WE#	ODT, CS[3:0]#, CKE, RAS#, CAS#, WE#	ODT, CS[2:0]#, CKE, RAS#, CAS#, WE#
Parity	PAR_IN, ERR_OUT	PAR_IN, ERR_OUT	PAR_IN, ERR_OUT	PAR_IN, ERR_OUT	PAR_IN, ERR_OUT	PAR_IN, ERR_OUT	PAR_IN, ERR_OUT
Other Pins	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#

Notes to Table:

1. DM pins are not used for LRDIMMs that are constructed using ×4 components.
2. S#[2] is treated as A[16] (whose corresponding pins are labeled as CS#[2] or RM[0]) and S#[3] is treated as A[17] (whose corresponding pins are labeled as CS#[3] or RM[1]) for certain rank multiplication configuration.
3. R = rank, RMF = rank multiplication factor.

The following table shows UDIMM, RDIMM, and LRDIMM pin options for DDR4.

**Table 4. UDIMM, RDIMM, and LRDIMM Pin Options for DDR4**

Pins	UDIMM Pins (Single Rank)	UDIMM Pins (Dual Rank)	RDIMM Pins (Single Rank)	RDIMM Pins (Dual Rank)	LRDIMM Pins (Dual Rank)	LRDIMM Pins (Quad Rank)
Data	72 bit DQ[71:0]= {CB[7:0], DQ[63:0]}					
Data Mask	DM#/ DBI#[8:0] <sup>(1)</sup>	DM#/ DBI#[8:0] <sup>(1)</sup>	DM#/ DBI#[8:0] <sup>(1)</sup>	DM#/ DBI#[8:0] <sup>(1)</sup>	—	—

*continued...*



Pins	UDIMM Pins (Single Rank)	UDIMM Pins (Dual Rank)	RDIMM Pins (Single Rank)	RDIMM Pins (Dual Rank)	LRDIMM Pins (Dual Rank)	LRDIMM Pins (Quad Rank)
Data Strobe	x8: DQS[8:0] and DQS#[8:0]	x8: DQS[8:0] and DQS#[8:0]	x8: DQS[8:0] and DQS#[8:0] x4: DQS[17:0] and DQS#[17:0]	x8: DQS[8:0] and DQS#[8:0] x4: DQS[17:0] and DQS#[17:0]	x4: DQS[17:0] and DQS#[17:0]	x4: DQS[17:0] and DQS#[17:0]
Address	BA[1:0], BG[1:0], A[16:0] - 4GB: A[14:0] 8GB: A[15:0] 16GB: A[16:0] (2)	BA[1:0], BG[1:0], A[16:0] - 8GB: A[14:0] 16GB: A[15:0] 32GB: A[16:0] (2)	BA[1:0], BG[1:0], x8: A[16:0] - 4GB: A[14:0] 8GB: A[15:0] 16GB: A[15:0] 32GB: A[16:0] (2) 32GB: A[17:0] (3)	BA[1:0], BG[1:0], x8: A[16:0] x4: A[17:0] - 8GB: A[14:0] 16GB: A[15:0] 32GB: A[16:0] (2) 64GB: A[17:0] (3)	BA[1:0], BG[1:0], A[17:0] - 16GB: A[15:0] 32GB: A[16:0] (2) 64GB: A[17:0] (3)	BA[1:0], BG[1:0], A[17:0] - 32GB: A[15:0] 64GB: A[16:0] (2) 128GB: A[17:0] (3)
Clock	CK0/CK0#	CK0/CK0#, CK1/CK1#	CK0/CK0#	CK0/CK0#	CK0/CK0#	CK0/CK0#
Command	ODT, CS#, CKE, ACT#, RAS#/A16, CAS#/A15, WE#/A14	ODT[1:0], CS#[1:0], CKE[1:0], RAS#/A16, ACT#, RAS#/A16, CAS#/A15, WE#/A14	ODT[1:0], CS#[1:0], CKE, ACT#, RAS#/A16, CAS#/A15, WE#/A14	ODT[1:0], CS#[1:0], CKE, ACT#, RAS#/A16, CAS#/A15, WE#/A14	ODT, CS#[3:0], CKE, ACT#, RAS#/A16, CAS#/A15, WE#/A14	ODT, CS#[3:0], CKE, ACT#, RAS#/A16, CAS#/A15, WE#/A14
Parity	PAR, ALERT#	PAR, ALERT#	PAR, ALERT#	PAR, ALERT#	PAR, ALERT#	PAR, ALERT#
Other Pins	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#	SA[2:0], SDA, SCL, EVENT#, RESET#

Notes to Table:

1. DM/DBI pins are available only for DIMMs constructed using x8 or greater components.
2. This density requires 4Gb x4 or 2Gb x8 DRAM components.
3. This density requires 8Gb x4 DRAM components.
4. This table assumes a single slot configuration. The Arria 10 memory controller can support up to 4 ranks per channel. A single slot interface may have up to 4 ranks, and a dual slot interface may have up to 2 ranks per slot. In either case, the total number of ranks, calculated as the number of slots multiplied by the number of ranks per slot, must be less than or equal to 4.

### 1.1.6. QDR II, QDR II+, and QDR II+ Xtreme SRAM Clock Signals

QDR II, QDR II+ and QDR II+ Xtreme SRAM devices have two pairs of clocks, listed below.

- Input clocks K and K#
- Echo clocks CQ and CQ#

In addition, QDR II devices have a third pair of input clocks, C and C#.



The positive input clock,  $K$ , is the logical complement of the negative input clock,  $K\#$ . Similarly,  $C$  and  $CQ$  are complements of  $C\#$  and  $CQ\#$ , respectively. With these complementary clocks, the rising edges of each clock leg latch the DDR data.

The QDR II SRAM devices use the  $K$  and  $K\#$  clocks for write access and the  $C$  and  $C\#$  clocks for read accesses only when interfacing more than one QDR II SRAM device. Because the number of loads that the  $K$  and  $K\#$  clocks drive affects the switching times of these outputs when a controller drives a single QDR II SRAM device,  $C$  and  $C\#$  are unnecessary. This is because the propagation delays from the controller to the QDR II SRAM device and back are the same. Therefore, to reduce the number of loads on the clock traces, QDR II SRAM devices have a single-clock mode, and the  $K$  and  $K\#$  clocks are used for both reads and writes. In this mode, the  $C$  and  $C\#$  clocks are tied to the supply voltage (VDD). Intel FPGA external memory IP supports only single-clock mode.

For QDR II, QDR II+, or QDR II+ Xtreme SRAM devices, the rising edge of  $K$  is used to capture synchronous inputs to the device and to drive out data through  $Q[x:0]$ , in similar fashion to QDR II SRAM devices in single clock mode. All accesses are initiated on the rising edge of  $K$ .

$CQ$  and  $CQ\#$  are the source-synchronous output clocks from the QDR II, QDR II+, or QDR II+ Xtreme SRAM device that accompanies the read data.

The Intel device outputs the  $K$  and  $K\#$  clocks, data, address, and command lines to the QDR II, QDR II+, or QDR II+ Xtreme SRAM device. For the controller to operate properly, the write data ( $D$ ), address ( $A$ ), and control signal trace lengths (and therefore the propagation times) should be equal to the  $K$  and  $K\#$  clock trace lengths.

You can generate  $K$  and  $K\#$  clocks using any of the PLL registers via the DDR registers. Because of strict skew requirements between  $K$  and  $K\#$  signals, use adjacent pins to generate the clock pair. The propagation delays for  $K$  and  $K\#$  from the FPGA to the QDR II, QDR II+, or QDR II+ Xtreme SRAM device are equal to the delays on the data and address ( $D$ ,  $A$ ) signals. Therefore, the signal skew effect on the write and read request operations is minimized by using identical DDR output circuits to generate clock and data inputs to the memory.

### 1.1.7. QDR II, QDR II+ and QDR II+ Xtreme SRAM Command Signals

QDR II, QDR II+ and QDR II+ Xtreme SRAM devices use the write port select (WPS#) signal to control write operations and the read port select (RPS#) signal to control read operations.

### 1.1.8. QDR II, QDR II+ and QDR II+ Xtreme SRAM Address Signals

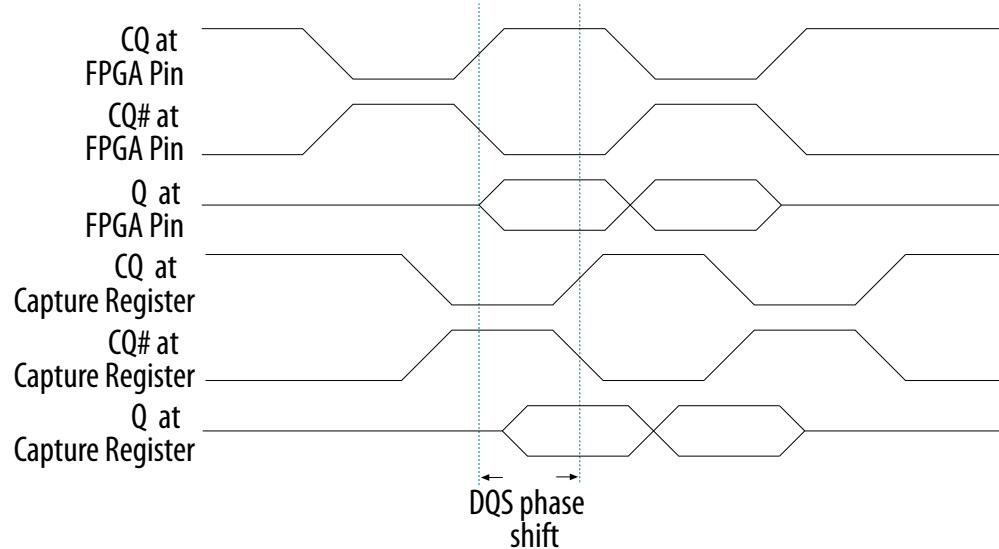
QDR II, QDR II+ and QDR II+ Xtreme SRAM devices use one address bus ( $A$ ) for both read and write accesses.

### 1.1.9. QDR II, QDR II+ and QDR II+ Xtreme SRAM Data, BWS, and QVLD Signals

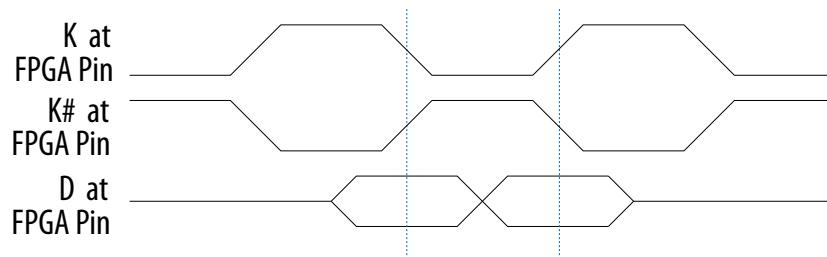
QDR II, QDR II+ and QDR II+ Xtreme SRAM devices use two unidirectional data buses: one for writes ( $D$ ) and one for reads ( $Q$ ).

At the pin, the read data is edge-aligned with the  $CQ$  and  $CQ\#$  clocks while the write data is center-aligned with the  $K$  and  $K\#$  clocks (see the following figures).

**Figure 3.** Edge-aligned CQ and Q Relationship During QDR II+ SRAM Read



**Figure 4.** Center-aligned K and D Relationship During QDR II+ SRAM Write



The byte write select signal ( $BWS\#$ ) indicates which byte to write into the memory device.

QDR II+ and QDR II+ Xtreme SRAM devices also have a QVLD pin that indicates valid read data. The QVLD signal is edge-aligned with the echo clock and is asserted high for approximately half a clock cycle before data is output from memory.

*Note:* The Intel FPGA external memory interface IP does not use the QVLD signal.

### 1.1.10. QDR IV SRAM Clock Signals

QDR IV SRAM devices have three pairs of differential clocks.



The three QDR IV differential clocks are as follows:

- Address and Command Input Clocks CK and CK#
- Data Input Clocks DK<sub>x</sub> and DK<sub>x</sub>#, where x can be A or B, referring to the respective ports
- Data Output Clocks, QK<sub>x</sub> and QK<sub>x</sub>#, where x can be A or B, referring to the respective ports

QDR IV SRAM devices have two independent bidirectional data ports, Port A and Port B, to support concurrent read/write transactions on both ports. These data ports are controlled by a common address port clocked by CK and CK# in double data rate. There is one pair of CK and CK# pins per QDR IV SRAM device.

DK<sub>x</sub> and DK<sub>x</sub># samples the DQ<sub>x</sub> inputs on both rising and falling edges. Similarly, QK<sub>x</sub> and QK<sub>x</sub># samples the DQ<sub>x</sub> outputs on both rising and falling edges.

QDR IV SRAM devices employ two sets of free running differential clocks to accompany the data. The DK<sub>x</sub> and DK<sub>x</sub># clocks are the differential input data clocks used during writes. The QK<sub>x</sub> and QK<sub>x</sub># clocks are the output data clocks used during reads. Each pair of DK<sub>x</sub> and DK<sub>x</sub>#, or QK<sub>x</sub> and QK<sub>x</sub># clocks are associated with either 9 or 18 data bits.

The polarity of the QKB and QKB# pins in the Intel FPGA external memory interface IP was swapped with respect to the polarity of the differential input buffer on the FPGA. In other words, the QKB pins on the memory side must be connected to the negative pins of the input buffers on the FPGA side, and the QKB# pins on the memory side must be connected to the positive pins of the input buffers on the FPGA side. Notice that the port names at the top-level of the IP already reflect this swap (that is, mem\_qkb is assigned to the negative buffer leg, and mem\_qkb\_n is assigned to the positive buffer leg).

QDR IV SRAM devices are available in x18 and x36 bus width configurations. The exact clock-data relationships are as follows:

- For x18 data bus width configuration, there are 9 data bits associated with each pair of write and read clocks. So, there are two pairs of DK<sub>x</sub> and DK<sub>x</sub># pins and two pairs of QK<sub>x</sub> or QK<sub>x</sub># pins.
- For x36 data bus width configuration, there are 18 data bits associated with each pair of write and read clocks. So, there are two pairs of DK<sub>x</sub> and DK<sub>x</sub># pins and two pairs of QK<sub>x</sub> or QK<sub>x</sub># pins.

There are t<sub>CKDK</sub> timing requirements for skew between CK and DK<sub>x</sub> or CK# and DK<sub>x</sub># .Similarly, there are t<sub>CKQK</sub> timing requirements for skew between CK and QK<sub>x</sub> or CK# and QK<sub>x</sub># .

### 1.1.11. QDR IV SRAM Commands and Addresses, AP, and AINV Signals

The CK and CK# signals clock the commands and addresses into the memory devices. There is one pair of CK and CK# pins per QDR IV SRAM device. These pins operate at double data rate using both rising and falling edge. The rising edge of CK latches the addresses for port A, while the falling edge of CK latches the addresses inputs for port B.

QDR IV SRAM devices have the ability to invert all address pins to reduce potential simultaneous switching noise. Such inversion is accomplished using the Address Inversion Pin for Address and Address Parity Inputs (AINV), which assumes an address parity of 0, and indicates whether the address bus and address parity are inverted.

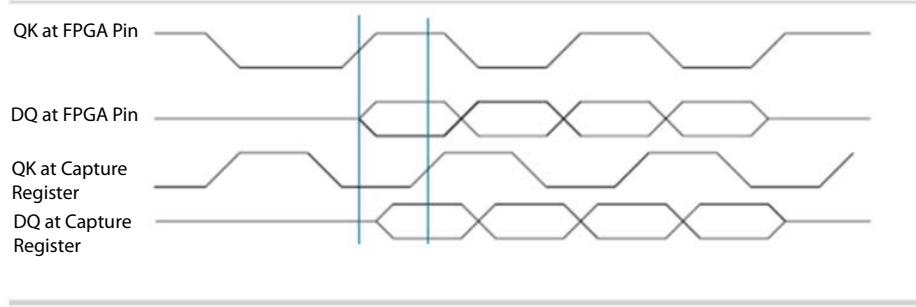
The above features are available as **Option Control** under **Configuration Register Settings** in Arria 10 EMIF IP. The commands and addresses must meet the memory address and command setup ( $t_{AS}$ ,  $t_{CS}$ ) and hold ( $t_{AH}$ ,  $t_{CH}$ ) time requirements.

### 1.1.12. QDR IV SRAM Data, DINv, and QVLD Signals

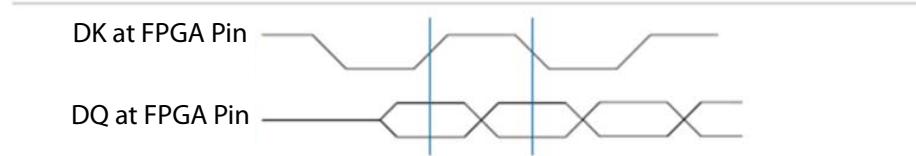
The read data is edge-aligned with the QKA or QKB# clocks while the write data is center-aligned with the DKA and DKB# clocks.

QK is shifted by the DLL so that the clock edges can be used to clock in the DQ at the capture register.

**Figure 5. Edge-Aligned DQ and QK Relationship During Read**



**Figure 6. Center-Aligned DQ and DK Relationship During Write**



The polarity of the QKB and QKB# pins in the Intel FPGA external memory interface IP was swapped with respect to the polarity of the differential input buffer on the FPGA. In other words, the QKB pins on the memory side need to be connected to the negative pins of the input buffers on the FPGA side, and the QKB# pins on the memory side need to be connected to the positive pins of the input buffers on the FPGA side. Notice that the port names at the top-level of the IP already reflect this swap (that is, `mem_qkb` is assigned to the negative buffer leg, and `mem_qkb_n` is assigned to the positive buffer leg).



The synchronous read/write input,  $RWx\#$ , is used in conjunction with the synchronous load input,  $LDx\#$ , to indicate a Read or Write Operation. For port A, these signals are sampled on the rising edge of  $CK$  clock, for port B, these signals are sampled on the falling edge of  $CK$  clock.

QDR IV SRAM devices have the ability to invert all data pins to reduce potential simultaneous switching noise, using the Data Inversion Pin for DQ Data Bus,  $DINVx$ . This pin indicates whether  $DQx$  pins are inverted or not.

To enable the data pin inversion feature, click **Configuration Register Settings > Option Control** in the Arria 10 or Stratix 10 EMIF IP.

QDR IV SRAM devices also have a  $QVLD$  pin which indicates valid read data. The  $QVLD$  signal is edge-aligned with  $QKx$  or  $QKx\#$  and is high approximately one-half clock cycle before data is output from the memory.

*Note:* The Intel ZFPGA external memory interface IP does not use the  $QVLD$  signal.

### 1.1.13. RLDRAM II and RLDRAM 3 Clock Signals

RLDRAM II and RLDRAM 3 devices use  $CK$  and  $CK\#$  signals to clock the command and address bus in single data rate (SDR). There is one pair of  $CK$  and  $CK\#$  pins per RLDRAM II or RLDRAM 3 device.

Instead of a strobe, RLDRAM II and RLDRAM 3 devices use two sets of free-running differential clocks to accompany the data. The  $DK$  and  $DK\#$  clocks are the differential input data clocks used during writes while the  $QK$  or  $QK\#$  clocks are the output data clocks used during reads. Even though  $QK$  and  $QK\#$  signals are not differential signals according to the RLDRAM II and RLDRAM 3 data sheets, Micron treats these signals as such for their testing and characterization. Each pair of  $DK$  and  $DK\#$ , or  $QK$  and  $QK\#$  clocks are associated with either 9 or 18 data bits.

The exact clock-data relationships are as follows:

- RLDRAM II: For  $\times 36$  data bus width configuration, there are 18 data bits associated with each pair of write and read clocks. So, there are two pairs of  $DK$  and  $DK\#$  pins and two pairs of  $QK$  or  $QK\#$  pins.
- RLDRAM 3: For  $\times 36$  data bus width configuration, there are 18 data bits associated with each pair of write clocks. There are 9 data bits associated with each pair of read clocks. So, there are two pairs of  $DK$  and  $DK\#$  pins and four pairs of  $QK$  and  $QK\#$  pins.
- RLDRAM II: For  $\times 18$  data bus width configuration, there are 18 data bits per one pair of write clocks and nine data bits per one pair of read clocks. So, there is one pair of  $DK$  and  $DK\#$  pins, but there are two pairs of  $QK$  and  $QK\#$  pins.
- RLDRAM 3: For  $\times 18$  data bus width configuration, there are 9 data bits per one pair of write clocks and nine data bits per one pair of read clocks. So, there are two pairs of  $DK$  and  $DK\#$  pins, and two pairs of  $QK$  and  $QK\#$  pins
- RLDRAM II: For  $\times 9$  data bus width configuration, there are nine data bits associated with each pair of write and read clocks. So, there is one pair of  $DK$  and  $DK\#$  pins and one pair of  $QK$  and  $QK\#$  pins each.
- RLDRAM 3: RLDRAM 3 does not have the  $\times 9$  data bus width configuration.

There are  $t_{CKDK}$  timing requirements for skew between CK and DK or CK# and DK#.

For both RLDRAM II and RLDRAM 3, because of the loads on these I/O pins, the maximum frequency you can achieve depends on the number of memory devices you are connecting to the Intel device. Perform SPICE or IBIS simulations to analyze the loading effects of the pin-pair on multiple RLDRAM II or RLDRAM 3 devices.

### **1.1.14. RLDRAM II and RLDRAM 3 Commands and Addresses**

The CK and CK# signals clock the commands and addresses into the memory devices.

These pins operate at single data rate using only one clock edge. RLDRAM II and RLDRAM 3 support both non-multiplexed and multiplexed addressing. Multiplexed addressing allows you to save a few user I/O pins while non-multiplexed addressing allows you to send the address signal within one clock cycle instead of two clock cycles. CS#, REF#, and WE# pins are input commands to the RLDRAM II or RLDRAM 3 device.

The commands and addresses must meet the memory address and command setup ( $t_{AS}$ ,  $t_{CS}$ ) and hold ( $t_{AH}$ ,  $t_{CH}$ ) time requirements.

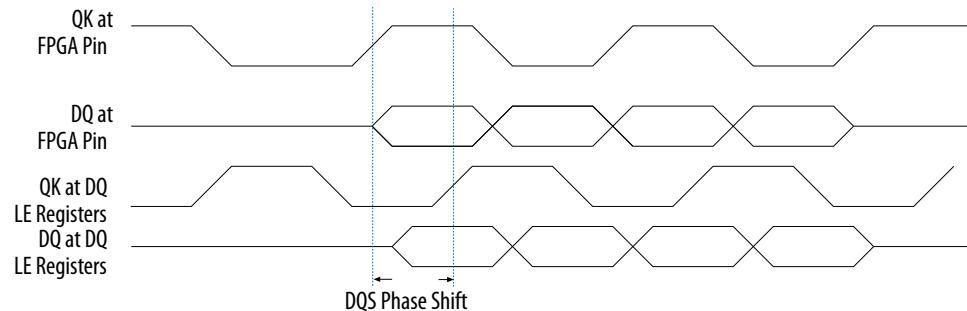
**Note:** The RLDRAM II and RLDRAM 3 external memory interface IP do not support multiplexed addressing.

### **1.1.15. RLDRAM II and RLDRAM 3 Data, DM and QVLD Signals**

The read data is edge-aligned with the QK or QK# clocks while the write data is center-aligned with the DK and DK# clocks (see the following figures). The memory controller shifts the DK and DK# signals to center align the DQ and DK or DK# signals during a write. It also shifts the QK signal during a read, so that the read data (DQ signals) and QK clock is center-aligned at the capture register.

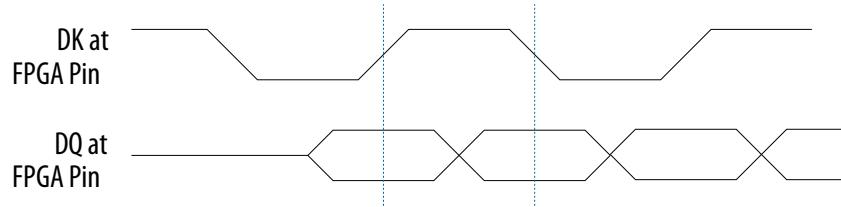
Intel devices use dedicated DQS phase-shift circuitry to shift the incoming QK signal during reads and use a PLL to center-align the DK and DK# signals with respect to the DQ signals during writes.

**Figure 7. Edge-aligned DQ and QK Relationship During RLDRAM II or RLDRAM 3 Read**





**Figure 8. Center-aligned DQ and DK Relationship During RLDRAM II or RLDRAM 3 Write**



For RLDRAM II and RLDRAM 3, data mask (DM) pins are used only during a write. The memory controller drives the DM signal low when the write is valid and drives it high to mask the DQ signals.

For RLDRAM II, there is one DM pin per memory device. The DQ input signal is masked when the DM signal is high.

For RLDRAM 3, there are two DM pins per memory device. DM0 is used to mask the lower byte for the x18 device and (DQ[8:0], DQ[26:18]) for the x36 device. DM1 is used to mask the upper byte for the x18 device and (DQ[17:9], DQ[35:27]) for the x36 device.

The DM timing requirements at the input to the memory device are identical to those for DQ data. The DDR registers, clocked by the write clock, create the DM signals. This reduces any skew between the DQ and DM signals.

The RLDRAM II or RLDRAM 3 device's setup time ( $t_{DS}$ ) and hold ( $t_{DH}$ ) time for the write DQ and DM pins are relative to the edges of the DK or DK# clocks. The DK and DK# signals are generated on the positive edge of system clock, so that the positive edge of CK or CK# is aligned with the positive edge of DK or DK# respectively to meet the tCKDK requirement. The DQ and DM signals are clocked using a shifted clock so that the edges of DK or DK# are center-aligned with respect to the DQ and DM signals when they arrive at the RLDRAM II or RLDRAM 3 device.

The clocks, data, and DM board trace lengths should be tightly matched to minimize the skew in the arrival time of these signals.

RLDRAM II and RLDRAM 3 devices also have a QVLD pin indicating valid read data. The QVLD signal is edge-aligned with QK or QK# and is high approximately half a clock cycle before data is output from the memory.

*Note:* The Intel FPGA external memory interface IP does not use the QVLD signal.

### 1.1.16. LPDDR2 and LPDDR3 Clock Signal

CK and CKn are differential clock inputs to the LPDDR2 and LPDDR3 interface. All the double data rate (DDR) inputs are sampled on both the positive and negative edges of the clock. Single data rate (SDR) inputs, CSn and CKE, are sampled at the positive clock edge.

The clock is defined as the differential pair which consists of CK and CKn. The positive clock edge is defined by the cross point of a rising CK and a falling CKn. The negative clock edge is defined by the cross point of a falling CK and a rising CKn.



The SDRAM data sheet specifies timing data for the following:

- $t_{DSH}$  is the DQS falling edge hold time from CK.
- $t_{DSS}$  is the DQS falling edge to the CK setup time.
- $t_{DQSS}$  is the Write command to the first DQS latching transition.
- $t_{DQSCK}$  is the DQS output access time from CK/CKn.

### 1.1.17. LPDDR2 and LPDDR3 Command and Address Signal

All LPDDR2 and LPDDR3 devices use double data rate architecture on the command/address bus to reduce the number of input pins in the system. The 10-bit command/address bus contains command, address, and bank/row buffer information. Each command uses one clock cycle, during which command information is transferred on both the positive and negative edges of the clock.

### 1.1.18. LPDDR2 and LPDDR3 Data, Data Strobe, and DM Signals

LPDDR2 and LPDDR3 devices use bidirectional and differential data strobes.

Differential DQS operation enables improved system timing due to reduced crosstalk and less simultaneous switching noise on the strobe output drivers. The DQ pins are also bidirectional. DQS is edge-aligned with the read data and centered with the write data.

DM is the input mask for the write data signal. Input data is masked when DM is sampled high coincident with that input data during a write access.

### 1.1.19. Maximum Number of Interfaces

The maximum number of interfaces supported for a given memory protocol varies, depending on the FPGA in use.

Unless otherwise noted, the calculation for the maximum number of interfaces is based on independent interfaces where the address or command pins are not shared. The maximum number of independent interfaces is limited to the number of PLLs each FPGA device has.

**Note:** You must share DLLs if the total number of interfaces exceeds the number of DLLs available in a specific FPGA device. You may also need to share PLL clock outputs depending on your clock network usage, refer to *PLLs and Clock Networks*.

**Note:** For information about the number of DQ and DQS in other packages, refer to the DQ and DQS tables in the relevant device handbook.

For interface information for Arria 10 and Stratix 10 devices, you can consult the EMIF Device Selector on [www.altera.com](http://www.altera.com).

Timing closure depends on device resource and routing utilization. For more information about timing closure, refer to the *Area and Timing Optimization Techniques* chapter in the *Quartus Prime Handbook*.



### Related Information

- [PLLs and Clock Networks](#) on page 71
- [Intel Arria 10 Device Handbook](#)
- [External Memory Interface Device Selector](#)
- [Quartus Prime Handbook](#)

#### 1.1.19.1. Maximum Number of DDR SDRAM Interfaces Supported per FPGA

The following table describes the maximum number of  $\times 8$  DDR SDRAM components that can fit in the smallest and biggest devices and pin packages assuming the device is blank.

Each interface of size  $n$ , where  $n$  is a multiple of 8, consists of:

- $n$  DQ pins (including error correction coding (ECC))
- $n/8$  DM pins
- $n/8$  DQS pins
- 18 address pins
- 6 command pins (CAS#, RAS#, WE#, CKE, and CS#)
- 1 CK, CK# pin pair for up to every three  $\times 8$  DDR SDRAM components

**Table 5. Maximum Number of DDR SDRAM Interfaces Supported per FPGA**

Device	Device Type	Package Pin Count	Maximum Number of Interfaces
Arria II GX	EP2AGX190 EP2AGX260	1,152	Four $\times 8$ interfaces or one $\times 72$ interface on each side (no DQ pins on left side)
	EP2AGX45 EP2AGX65	358	<ul style="list-style-type: none"><li>• On top side, one <math>\times 16</math> interface</li><li>• On bottom side, one <math>\times 16</math> interface</li><li>• On right side (no DQ pins on left side), one <math>\times 8</math> interface</li></ul>
Arria II GZ	EP2AGZ300 EP2AGZ350 EP2AGZ225	1,517	Four $\times 8$ interfaces or one $\times 72$ interface on each side
	EP2AGZ300 EP2AGZ350	780	<ul style="list-style-type: none"><li>• On top side, three <math>\times 8</math> interfaces or one <math>\times 64</math> interface</li><li>• On bottom side, three <math>\times 8</math> interfaces or one <math>\times 64</math> interface</li><li>• No DQ pins on the left and right sides</li></ul>
Stratix III	EP3SL340	1,760	<ul style="list-style-type: none"><li>• Two <math>\times 72</math> interfaces on both top and bottom sides</li><li>• One <math>\times 72</math> interface on both right and left sides</li></ul>
	EP3SE50	484	<ul style="list-style-type: none"><li>• Two <math>\times 8</math> interfaces on both top and bottom sides</li><li>• Three <math>\times 8</math> interface on both right and left sides</li></ul>
Stratix IV	EP4SGX290 EP4SGX360 EP4SGX530	1,932	<ul style="list-style-type: none"><li>• One <math>\times 72</math> interface on each side</li></ul>

*continued...*



Device	Device Type	Package Pin Count	Maximum Number of Interfaces
	EP4SE530 EP4SE820	1,760	or <ul style="list-style-type: none"><li>• One <math>\times 72</math> interface on each side and two additional <math>\times 72</math> wraparound interfaces, only if sharing DLL and PLL resources</li></ul>
	EP4SGX70 EP4SGX110 EP4SGX110 EP4SGX230	780	<ul style="list-style-type: none"><li>• Three <math>\times 8</math> interfaces or one <math>\times 64</math> interface on both top and bottom sides</li><li>• On left side, one <math>\times 48</math> interface or two <math>\times 8</math> interfaces</li><li>• No DQ pins on the right side</li></ul>

### Related Information

[External Memory Interface Device Selector](#)

#### 1.1.19.2. Maximum Number of DDR2 SDRAM Interfaces Supported per FPGA

The following table lists the maximum number of  $\times 8$  DDR2 SDRAM components that can be fitted in the smallest and biggest devices and pin packages assuming the device is blank.

Each interface of size  $n$ , where  $n$  is a multiple of 8, consists of:

- $n$  DQ pins (including ECC)
- $n/8$  DM pins
- $n/8$  DQS, DQSn pin pairs
- 18 address pins
- 7 command pins (CAS#, RAS#, WE#, CKE, ODT, and CS#)
- 1 CK, CK# pin pair up to every three  $\times 8$  DDR2 components

**Table 6. Maximum Number of DDR2 SDRAM Interfaces Supported per FPGA**

Device	Device Type	Package Pin Count	Maximum Number of Interfaces
Arria II GX	EP2AGX190 EP2AGX260	1,152	Four $\times 8$ interfaces or one $\times 72$ interface on each side (no DQ pins on left side)
	EP2AGX45 EP2AGX65	358	<ul style="list-style-type: none"><li>• One <math>\times 16</math> interface on both top and bottom sides</li><li>• On right side (no DQ pins on left side), one <math>\times 8</math> interface</li></ul>
Arria II GZ	EP2AGZ300 EP2AGZ350 EP2AGZ225	1,517	Four $\times 8$ interfaces or one $\times 72$ interface on each side
	EP2AGZ300 EP2AGZ350	780	<ul style="list-style-type: none"><li>• Three <math>\times 8</math> interfaces or one <math>\times 64</math> interface on both top and bottom sides</li><li>• No DQ pins on the left and right sides</li></ul>
Arria V	5AGXB1 5AGXB3 5AGXB5 5AGXB7 5AGTD3 5AGTD7	1,517	<ul style="list-style-type: none"><li>• Two <math>\times 72</math> interfaces on both top and bottom sides</li><li>• No DQ pins on left and right sides</li></ul>

*continued...*



Device	Device Type	Package Pin Count	Maximum Number of Interfaces
	5AGXA1 5AGXA3	672	<ul style="list-style-type: none"> <li>One <math>\times 56</math> interface or two <math>\times 24</math> interfaces on both top and bottom sides</li> <li>One <math>\times 32</math> interface on the right side</li> <li>No DQ pins on the left side</li> </ul>
	5AGXA5 5AGXA7	672	<ul style="list-style-type: none"> <li>One <math>\times 56</math> interface or two <math>\times 24</math> interfaces on both top and bottom sides</li> <li>No DQ pins on the left side</li> </ul>
Arria V GZ	5AGZE5 5AGZE7	1,517	<ul style="list-style-type: none"> <li>Three <math>\times 72</math> interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>
	5AGZE1 5AGZE3	780	<ul style="list-style-type: none"> <li>On top side, two <math>\times 8</math> interfaces</li> <li>On bottom side, four <math>\times 8</math> interfaces or one <math>\times 72</math> interface</li> <li>No DQ pins on left and right sides</li> </ul>
Cyclone V	5CGTD9 5CEA9 5CGXC9	1,152	<ul style="list-style-type: none"> <li>One <math>\times 72</math> interface or two <math>\times 32</math> interfaces on each of the top, bottom, and right sides</li> <li>No DQ pins on the left side</li> </ul>
	5CEA7 5CGTD7 5CGXC7	484	<ul style="list-style-type: none"> <li>One <math>\times 48</math> interface or two <math>\times 16</math> interfaces on both top and bottom sides</li> <li>One <math>\times 8</math> interface on the right side</li> <li>No DQ pins on the left side</li> </ul>
MAX 10 FPGA	10M50D672 10M40D672	762	One $\times 32$ interface on the right side
	10M50D256 10M40D256 10M25D256 10M16D256	256	One $\times 8$ interface on the right side
Stratix III	EP3SL340	1,760	<ul style="list-style-type: none"> <li>Two <math>\times 72</math> interfaces on both top and bottom sides</li> <li>One <math>\times 72</math> interface on both right and left sides</li> </ul>
	EP3SE50	484	<ul style="list-style-type: none"> <li>Two <math>\times 8</math> interfaces on both top and bottom sides</li> <li>Three <math>\times 8</math> interfaces on both right and left sides</li> </ul>
Stratix IV	EP4SGX290 EP4SGX360 EP4SGX530	1,932	<ul style="list-style-type: none"> <li>One <math>\times 72</math> interface on each side or</li> <li>One <math>\times 72</math> interface on each side and two additional <math>\times 72</math> wraparound interfaces only if sharing DLL and PLL resources</li> </ul>
	EP4SE530 EP4SE820	1,760	

*continued...*



Device	Device Type	Package Pin Count	Maximum Number of Interfaces
	EP4SGX70 EP4SGX110 EP4SGX180 EP4SGX230	780	<ul style="list-style-type: none"><li>Three <math>\times 8</math> interfaces or one <math>\times 64</math> interface on top and bottom sides</li><li>On left side, one <math>\times 48</math> interface or two <math>\times 8</math> interfaces</li><li>No DQ pins on the right side</li></ul>
Stratix V	5SGXA5 5SGXA7	1,932	<ul style="list-style-type: none"><li>Three <math>\times 72</math> interfaces on both top and bottom sides</li><li>No DQ pins on left and right sides</li></ul>
	5SGXA3 5SGXA4	780	<ul style="list-style-type: none"><li>On top side, two <math>\times 8</math> interfaces</li><li>On bottom side, four <math>\times 8</math> interfaces or one <math>\times 72</math> interface</li><li>No DQ pins on left and right sides</li></ul>

#### Related Information

[External Memory Interface Device Selector](#)

#### 1.1.19.3. Maximum Number of DDR3 SDRAM Interfaces Supported per FPGA

The following table lists the maximum number of  $\times 8$  DDR3 SDRAM components that can be fitted in the smallest and biggest devices and pin packages assuming the device is blank.

Each interface of size  $n$ , where  $n$  is a multiple of 8, consists of:

- $n$  DQ pins (including ECC)
- $n/8$  DM pins
- $n/8$  DQS, DQSn pin pairs
- 17 address pins
- 7 command pins (CAS#, RAS#, WE#, CKE, ODT, reset, and CS#)
- 1 CK, CK# pin pair

**Table 7. Maximum Number of DDR3 SDRAM Interfaces Supported per FPGA**

Device	Device Type	Package Pin Count	Maximum Number of Interfaces
Arria II GX	EP2AGX190 EP2AGX260	1,152	<ul style="list-style-type: none"><li>Four <math>\times 8</math> interfaces or one <math>\times 72</math> interface on each side</li><li>No DQ pins on left side</li></ul>
	EP2AGX45 EP2AGX65	358	<ul style="list-style-type: none"><li>One <math>\times 16</math> interface on both top and bottom sides</li><li>On right side, one <math>\times 8</math> interface</li><li>No DQ pins on left side</li></ul>
Arria II GZ	EP2AGZ300 EP2AGZ350 EP2AGZ225	1,517	Four $\times 8$ interfaces on each side
	EP2AGZ300 EP2AGZ350	780	<ul style="list-style-type: none"><li>Three <math>\times 8</math> interfaces on both top and bottom sides</li><li>No DQ pins on left and right sides</li></ul>

*continued...*



Device	Device Type	Package Pin Count	Maximum Number of Interfaces
Arria V	5AGXB1 5AGXB3 5AGXB5 5AGXB7 5AGTD3 5AGTD7	1,517	<ul style="list-style-type: none"> <li>Two <math>\times 72</math> interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>
	5AGXA1 5AGXA3	672	<ul style="list-style-type: none"> <li>One <math>\times 56</math> interface or two <math>\times 24</math> interfaces on top and bottom sides</li> <li>One <math>\times 32</math> interface on the right side</li> <li>No DQ pins on the left side</li> </ul>
	5AGXA5 5AGXA7	672	<ul style="list-style-type: none"> <li>One <math>\times 56</math> interface or two <math>\times 24</math> interfaces on both top and bottom sides</li> <li>No DQ pins on the left side</li> </ul>
Arria V GZ	5AGZE5 5AGZE7	1,517	<ul style="list-style-type: none"> <li>Two <math>\times 72</math> interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>
	5AGZE1 5AGZE3	780	<ul style="list-style-type: none"> <li>On top side, four <math>\times 8</math> interfaces or one <math>\times 72</math> interface</li> <li>On bottom side, four <math>\times 8</math> interfaces or one <math>\times 72</math> interface</li> <li>No DQ pins on left and right sides</li> </ul>
Cyclone V	5CGTD9 5CEA9 5CGXC9	1,152	<ul style="list-style-type: none"> <li>One <math>\times 72</math> interface or two <math>\times 32</math> interfaces on each of the top, bottom, and right sides</li> <li>No DQ pins on the left side</li> </ul>
	5CEA7 5CGTD7 5CGXC7	484	<ul style="list-style-type: none"> <li>One <math>\times 48</math> interface or two <math>\times 16</math> interfaces on both top and bottom sides</li> <li>One <math>\times 8</math> interface on the right side</li> <li>No DQ pins on the left side</li> </ul>
MAX 10 FPGA	10M50D672 10M40D672	762	One $\times 32$ interface on the right side
	10M50D256 10M40D256 10M25D256 10M16D256	256	One $\times 8$ interface on the right side
Stratix III	EP3SL340	1,760	<ul style="list-style-type: none"> <li>Two <math>\times 72</math> interfaces on both top and bottom sides</li> <li>One <math>\times 72</math> interface on both right and left sides</li> </ul>
	EP3SE50	484	<ul style="list-style-type: none"> <li>Two <math>\times 8</math> interfaces on both top and bottom sides</li> <li>Three <math>\times 8</math> interfaces on both right and left sides</li> </ul>
Stratix IV	EP4SGX290 EP4SGX360 EP4SGX530	1,932	<ul style="list-style-type: none"> <li>One <math>\times 72</math> interface on each side or</li> <li>One <math>\times 72</math> interface on each side and 2 additional <math>\times 72</math> wraparound interfaces only if sharing DLL and PLL resources</li> </ul>
	EP4SE530 EP4SE820	1,760	

*continued...*



Device	Device Type	Package Pin Count	Maximum Number of Interfaces
	EP4SGX70 EP4SGX110 EP4SGX180 EP4SGX230	780	<ul style="list-style-type: none"><li>Three <math>\times 8</math> interfaces or one <math>\times 64</math> interface on both top and bottom sides</li><li>On left side, one <math>\times 48</math> interface or two <math>\times 8</math> interfaces</li><li>No DQ pins on right side</li></ul>
Stratix V	5SGXA5 5SGXA7	1,932	<ul style="list-style-type: none"><li>Two <math>\times 72</math> interfaces (800 MHz) on both top and bottom sides</li><li>No DQ pins on left and right sides</li></ul>
	5SGXA3 5SGXA4	780	<ul style="list-style-type: none"><li>On top side, two <math>\times 8</math> interfaces</li><li>On bottom side, four <math>\times 8</math> interfaces</li><li>No DQ pins on left and right sides</li></ul>

### Related Information

[External Memory Interface Device Selector](#)

#### 1.1.19.4. Maximum Number of QDR II and QDR II+ SRAM Interfaces Supported per FPGA

The following table lists the maximum number of independent QDR II+ or QDR II SRAM interfaces that can be fitted in the smallest and biggest devices and pin packages assuming the device is blank.

One interface of  $\times 36$  consists of:

- 36 Q pins
- 36 D pins
- 1 K, K# pin pairs
- 1 CQ, CQ# pin pairs
- 19 address pins
- 4 BSWn pins
- WPSn, RPSn

One interface of  $\times 9$  consists of:

- 9 Q pins
- 9 D pins
- 1 K, K# pin pairs
- 1 CQ, CQ# pin pairs
- 21 address pins
- 1 BWSn pin
- WPSn, RPSn



**Table 8. Maximum Number of QDR II and QDR II+ SRAM Interfaces Supported per FPGA**

Device	Device Type	Package Pin Count	Maximum Number of Interfaces
Arria II GX	EP2AGX190 EP2AGX260	1,152	One $\times 36$ interface and one $\times 9$ interface on each side
	EP2AGX45 EP2AGX65	358	One $\times 9$ interface on each side No DQ pins on left side
Arria II GZ	EP2AGZ300 EP2AGZ350 EP2AGZ225	1,517	<ul style="list-style-type: none"> <li>Two <math>\times 36</math> interfaces and one <math>\times 9</math> interface on both top and bottom sides</li> <li>Four <math>\times 9</math> interfaces on right and left sides</li> </ul>
	EP2AGZ300 EP2AGZ350	780	<ul style="list-style-type: none"> <li>Three <math>\times 9</math> interfaces on both top and bottom sides</li> <li>No DQ pins on right and left sides</li> </ul>
Arria V	5AGXB1 5AGXB3 5AGXB5 5AGXB7 5AGTD3 5AGTD7	1,517	<ul style="list-style-type: none"> <li>Two <math>\times 36</math> interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>
	5AGXA1 5AGXA3	672	<ul style="list-style-type: none"> <li>Two <math>\times 9</math> interfaces on both top and bottom sides</li> <li>One <math>\times 9</math> interface on the right side</li> <li>No DQ pins on the left side</li> </ul>
	5AGXA5 5AGXA7	672	<ul style="list-style-type: none"> <li>Two <math>\times 9</math> interfaces on both top and bottom sides</li> <li>No DQ pins on the left side</li> </ul>
Arria V GZ	5AGZE5 5AGZE7	1,517	<ul style="list-style-type: none"> <li>Two <math>\times 36</math> interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>
	5AGZE1 5AGZE3	780	<ul style="list-style-type: none"> <li>On top side, one <math>\times 36</math> interface or three <math>\times 9</math> interfaces</li> <li>On bottom side, two <math>\times 9</math> interfaces</li> <li>No DQ pins on left and right sides</li> </ul>
Stratix III	EP3SL340	1,760	<ul style="list-style-type: none"> <li>Two <math>\times 36</math> interfaces and one <math>\times 9</math> interface on both top and bottom sides</li> <li>Five <math>\times 9</math> interfaces on both right and left sides</li> </ul>
	EP3SE50 EP3SL50 EP3SL70	484	<ul style="list-style-type: none"> <li>One <math>\times 9</math> interface on both top and bottom sides</li> <li>Two <math>\times 9</math> interfaces on both right and left sides</li> </ul>
Stratix IV	EP4SGX290 EP4SGX360 EP4SGX530	1,932	<ul style="list-style-type: none"> <li>Two <math>\times 36</math> interfaces on both top and bottom sides</li> <li>One <math>\times 36</math> interface on both right and left sides</li> </ul>
	EP4SE530 EP4SE820	1,760	
	EP4SGX70 EP4SGX110 EP4SGX180 EP4SGX230	780	<p>Two <math>\times 9</math> interfaces on each side No DQ pins on right side</p>
Stratix V	5SGXA5	1,932	<ul style="list-style-type: none"> <li>Two <math>\times 36</math> interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>

***continued...***



Device	Device Type	Package Pin Count	Maximum Number of Interfaces
	5SGXA7		
	5SGXA3 5SGXA4	780	<ul style="list-style-type: none"><li>On top side, one <math>\times 36</math> interface or three <math>\times 9</math> interfaces</li><li>On bottom side, two <math>\times 9</math> interfaces</li><li>No DQ pins on left and right sides</li></ul>

### Related Information

External Memory Interface Device Selector

#### 1.1.19.5. Maximum Number of RLDRAM II Interfaces Supported per FPGA

The following table lists the maximum number of independent RLDRAM II interfaces that can be fitted in the smallest and biggest devices and pin packages assuming the device is blank.

One common I/O  $\times 36$  interface consists of:

- 36 DQ
- 1 DM pin
- 2 DK, DK# pin pairs
- 2 QK, QK# pin pairs
- 1 CK, CK# pin pair
- 24 address pins
- 1 CS# pin
- 1 REF# pin
- 1 WE# pin

One common I/O  $\times 9$  interface consists of:

- 9 DQ
- 1 DM pins
- 1 DK, DK# pin pair
- 1 QK, QK# pin pair
- 1 CK, CK# pin pair
- 25 address pins
- 1 CS# pin
- 1 REF# pin
- 1 WE# pin

**Table 9. Maximum Number of RLDRAM II Interfaces Supported per FPGA**

Device	Device Type	Package Pin Count	Maximum Number of RLDRAM II CIO Interfaces
Arria II GZ	EP2AGZ300 EP2AGZ350	1,517	Two $\times 36$ interfaces on each side

*continued...*



Device	Device Type	Package Pin Count	Maximum Number of RLDRAM II CIO Interfaces
	EP2AGZ225		
	EP2AGZ300 EP2AGZ350	780	<ul style="list-style-type: none"> <li>Three ×9 interfaces or one ×36 interface on both top and bottom sides</li> <li>No DQ pins on the left and right sides</li> </ul>
Arria V	5AGXB1 5AGXB3 5AGXB5 5AGXB7 5AGTD3 5AGTD7	1,517	<ul style="list-style-type: none"> <li>Two ×36 interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>
	5AGXA1 5AGXA3	672	<ul style="list-style-type: none"> <li>One ×36 interface on both top and bottom sides</li> <li>One ×18 interface on the right side</li> <li>No DQ pins on the left side</li> </ul>
	5AGXA5 5AGXA7	672	<ul style="list-style-type: none"> <li>One ×36 interface on both top and bottom sides</li> <li>No DQ pins on the left side</li> </ul>
	5ZGZE5 5ZGZE7	1,517	<ul style="list-style-type: none"> <li>Four ×36 interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>
Arria V GZ	5AGZE1 5AGZE3	780	<ul style="list-style-type: none"> <li>On top side, three ×9 interfaces or two ×36 interfaces</li> <li>On bottom side, two ×9 interfaces or one ×36 interfaces</li> <li>No DQ pins on left and right sides</li> </ul>
	EP3SL340	1,760	<ul style="list-style-type: none"> <li>Four ×36 components on both top and bottom sides</li> <li>Three ×36 interfaces on both right and left sides</li> </ul>
Stratix III	EP3SE50 EP3SL50 EP3SL70	484	One ×9 interface on both right and left sides
	EP4SGX290 EP4SGX360 EP4SGX530	1,932	<ul style="list-style-type: none"> <li>Three ×36 interfaces on both top and bottom sides</li> <li>Two ×36 interfaces on both right and left sides</li> </ul>
	EP4SE530 EP4SE820	1,760	<ul style="list-style-type: none"> <li>Three ×36 interfaces on each side</li> </ul>
Stratix IV	EP4SGX70 EP4SGX110 EP4SGX180 EP4SGX230	780	One ×36 interface on each side (no DQ pins on right side)
	5SGXA5 5SGXA7	1,932	<ul style="list-style-type: none"> <li>Four ×36 interfaces on both top and bottom sides</li> <li>No DQ pins on left and right sides</li> </ul>
	5SGXA3 5SGXA4	780	<ul style="list-style-type: none"> <li>On top side, two ×9 interfaces or one ×18 interfaces</li> <li>On bottom side, three ×9 interfaces or two ×36 interfaces</li> <li>No DQ pins on left and right sides</li> </ul>

## Related Information

[External Memory Interface Device Selector](#)



### 1.1.19.6. Maximum Number of LPDDR2 SDRAM Interfaces Supported per FPGA

The following table lists the maximum number of x8 LPDDR2 SDRAM components that can fit in the smallest and largest devices and pin packages, assuming the device is blank.

Each interface of size  $n$ , where  $n$  is a multiple of 8, consists of:

- $n$  DQ pins (including ECC)
- $n/8$  DM pins
- $n/8$  DQS, DQS<sub>n</sub> pin pairs
- 10 address pins
- 2 command pins (CKE and CS<sub>n</sub>)
- 1 CK, CK# pin pair up to every three x8 LPDDR2 components

**Table 10. Maximum Number of LPDDR2 SDRAM Interfaces Supported per FPGA**

Device	Device Type	Package Pin Count	Maximum Number of LPDDR2 SDRAM Interfaces
Arria V	5AGXB1 5AGXB3 5AGXB5 5AGXB7 5AGTD3 5AGTD7	1,517	<ul style="list-style-type: none"><li>• One <math>\times 72</math> interface on both top and bottom sides</li><li>• No DQ pins on the left and right sides</li></ul>
	5AGXA1 5AGXA3	672	<ul style="list-style-type: none"><li>• One <math>\times 64</math> interface or two <math>\times 24</math> interfaces on both top and bottom sides</li><li>• One <math>\times 32</math> interface on the right side</li></ul>
	5AGXA5 5AGXA7	672	<ul style="list-style-type: none"><li>• One <math>\times 64</math> interface or two <math>\times 24</math> interfaces on both the top and bottom sides</li><li>• No DQ pins on the left side</li></ul>
Cyclone V	5CGTD9 5CEA9 5CGXC9	1,152	<ul style="list-style-type: none"><li>• One <math>\times 72</math> interface or two <math>\times 32</math> interfaces on each of the top, bottom, and right sides</li><li>• No DQ pins on the left side</li></ul>
	5CEA7 5CGTD7 5CGXC7	484	<ul style="list-style-type: none"><li>• One <math>\times 48</math> interface or two <math>\times 16</math> interfaces on both the top and bottom sides</li><li>• One <math>\times 8</math> interface on the right side</li><li>• No DQ pins on the left side</li></ul>
MAX 10 FPGA	10M50D672 10M40D672	762	One $\times 16$ interface on the right side
	10M50D256 10M40D256 10M25D256 10M16D256	256	One $\times 16$ interface on the right side

#### Related Information

[External Memory Interface Device Selector](#)



## 1.1.20. OCT Support

If the memory interface uses any FPGA OCT calibrated series, parallel, or dynamic termination for any I/O in your design, you need a calibration block for the OCT circuitry. This calibration block is not required to be within the same bank or side of the device as the memory interface pins. However, the block requires a pair of  $R_{UP}$  and  $R_{DN}$  or  $R_{ZQ}$  pins that must be placed within an I/O bank that has the same VCCIO voltage as the VCCIO voltage of the I/O pins that use the OCT calibration block.

The  $R_{ZQ}$  pin in Arria 10, Stratix 10, Arria V, Stratix V, and Cyclone V devices can be used as a general purpose I/O pin when it is not used to support OCT, provided the signal conforms to the bank voltage requirements.

The  $R_{UP}$  and  $R_{DN}$  pins in Arria II GX, Arria II GZ, MAX 10, Stratix III, and Stratix IV devices are dual functional pins that can also be used as DQ and DQS pins in when they are not used to support OCT, giving the following impacts on your DQS groups:

- If the  $R_{UP}$  and  $R_{DN}$  pins are part of a  $\times 4$  DQS group, you cannot use that DQS group in  $\times 4$  mode.
- If the  $R_{UP}$  and  $R_{DN}$  pins are part of a  $\times 8$  DQS group, you can only use this group in  $\times 8$  mode if any of the following conditions apply:
  - You are not using DM or BWSn pins.
  - You are not using a  $\times 8$  or  $\times 9$  QDR II SRAM device, as the  $R_{UP}$  and  $R_{DN}$  pins may have dual purpose function as the CQn pins. In this case, pick different pin locations for  $R_{UP}$  and  $R_{DN}$  pins, to avoid conflict with memory interface pin placement. You have the choice of placing the  $R_{UP}$  and  $R_{DN}$  pins in the same bank as the write data pin group or address and command pin group.
  - You are not using complementary or differential DQS pins.

**Note:** The Altera external memory interface IP does not support  $\times 8$  QDR II SRAM devices in the Quartus Prime software.

A DQS/DQ  $\times 8/\times 9$  group in Arria II GZ, Stratix III, and Stratix IV devices comprises 12 pins. A typical  $\times 8$  memory interface consists of one DQS, one DM, and eight DQ pins which add up to 10 pins. If you choose your pin assignment carefully, you can use the two extra pins for  $R_{UP}$  and  $R_{DN}$ . However, if you are using differential DQS, you do not have enough pins for  $R_{UP}$  and  $R_{DN}$  as you only have one pin leftover. In this case, as you do not have to put the OCT calibration block with the DQS or DQ pins, you can pick different locations for the  $R_{UP}$  and  $R_{DN}$  pins. As an example, you can place it in the I/O bank that contains the address and command pins, as this I/O bank has the same VCCIO voltage as the I/O bank containing the DQS and DQ pins.

There is no restriction when using  $\times 16/\times 18$  or  $\times 32/\times 36$  DQS groups that include the  $\times 4$  groups when pin members are used as  $R_{UP}$  and  $R_{DN}$  pins, as there are enough extra pins that can be used as DQS or DQ pins.

You must pick your DQS and DQ pins manually for the  $\times 8$ ,  $\times 9$ ,  $\times 16$  and  $\times 18$ , or  $\times 32$  and  $\times 36$  groups, if they are using  $R_{UP}$  and  $R_{DN}$  pins within the group. The Quartus Prime software might not place these pins optimally and might be unable to fit the design.



## 1.2. Guidelines for Intel Arria® 10 External Memory Interface IP

The Intel Arria® 10 device contains up to two I/O columns that can be used by external memory interfaces. The Arria 10 I/O subsystem resides in the I/O columns. Each column contains multiple I/O banks, each of which consists of four I/O lanes. An I/O lane is a group of twelve I/O ports.

The I/O column, I/O bank, I/O lane, adjacent I/O bank, and pairing pin for every physical I/O pin can be uniquely identified using the Bank Number and Index within I/O Bank values which are defined in each Arria 10 device pin-out file.

- The numeric component of the Bank Number value identifies the I/O column, while the letter represents the I/O bank.
- The Index within I/O Bank value falls within one of the following ranges: 0 to 11, 12 to 23, 24 to 35, or 36 to 47, and represents I/O lanes 1, 2, 3, and 4, respectively.
- The adjacent I/O bank is defined as the I/O bank with same column number but the letter is either before or after the respective I/O bank letter in the A-Z system.
- The pairing pin for an I/O pin is located in the same I/O bank. You can identify the pairing pin by adding one to its Index within I/O Bank number (if it is an even number), or by subtracting one from its Index within I/O Bank number (if it is an odd number).

For example, a physical pin with a Bank Number of 2K and Index within I/O Bank of 22, indicates that the pin resides in I/O lane 2, in I/O bank 2K, in column 2. The adjacent I/O banks are 2J and 2L. The pairing pin for this physical pin is the pin with an Index within I/O Bank of 23 and Bank Number of 2K.

### Related Information

[Restrictions on I/O Bank Usage for Arria 10 EMIF IP with HPS](#)

### 1.2.1. General Pin-Out Guidelines for Arria 10 EMIF IP

You should follow the recommended guidelines when performing pin placement for all external memory interface pins targeting Arria 10 devices, whether you are using the Altera hard memory controller or your own solution.

If you are using the Altera hard memory controller, you should employ the relative pin locations defined in the `<variation_name>/altera_emif_arch_nf_version number/<synth/sim>/<variation_name>_altera_emif_arch_nf_version number_<unique ID>_readme.txt` file, which is generated with your IP.

*Note:*

1. EMIF IP pin-out requirements for the Arria 10 Hard Processor Subsystem (HPS) are more restrictive than for a non-HPS memory interface. The HPS EMIF IP defines a fixed pin-out in the Quartus Prime IP file (.qip), based on the IP configuration. When targeting Arria 10 HPS, you do not need to make location assignments for external memory interface pins. To obtain the HPS-specific external memory interface pin-out, compile the interface in the Quartus Prime software. Alternatively, consult the device handbook or the device pin-out files. For information on how you can customize the HPS EMIF pin-out, refer to [Restrictions on I/O Bank Usage for Arria 10 EMIF IP with HPS](#).
2. Ping Pong PHY, PHY only, RLDRAMx , QDRx and LPDDR3 are not supported with HPS.



Observe the following general guidelines for placing pins for your Arria 10 external memory interface:

1. Ensure that the pins of a single external memory interface reside within a single I/O column.
2. An external memory interface can occupy one or more banks in the same I/O column. When an interface must occupy multiple banks, ensure that those banks are adjacent to one another.
3. Be aware that any pin in the same bank that is not used by an external memory interface is available for use as a general purpose I/O of compatible voltage and termination settings.
4. All address and command pins and their associated clock pins (CK and CK#) must reside within a single bank. The bank containing the address and command pins is identified as the address and command bank.
5. To minimize latency, when the interface uses more than two banks, you must select the center bank of the interface as the address and command bank.
6. The address and command pins and their associated clock pins in the address and command bank must follow a fixed pin-out scheme, as defined in the *Arria 10 External Memory Interface Pin Information File*, which is available on [www.altera.com](http://www.altera.com).

You do not have to place every address and command pin manually. If you assign the location for one address and command pin, the Fitter automatically places the remaining address and command pins.

*Note:* The pin-out scheme is a hardware requirement that you must follow, and can vary according to the topology of the memory device. Some schemes require three lanes to implement address and command pins, while others require four lanes. To determine which scheme to follow, refer to the messages window during parameterization of your IP, or to the `<variation_name>/altera_emif_arch_nf_<version>/<synth/sim>` `<variation_name>_altera_emif_arch_nf_<version>_<unique ID>_readme.txt` file after you have generated your IP.

7. An unused I/O lane in the address and command bank can serve to implement a data group, such as a x8 DQS group. The data group must be from the same controller as the address and command signals.
8. An I/O lane must not be used by both address and command pins and data pins.
9. Place read data groups according to the DQS grouping in the pin table and pin planner. Read data strobes (such as DQS and DQS#) or read clocks (such as CQ and CQ# / QK and QK#) must reside at physical pins capable of functioning as DQS/CQ and DQSn/CQn for a specific read data group size. You must place the associated read data pins (such as DQ and Q), within the same group.



*Note:* a. Unlike other device families, there is no need to swap CQ/CQ# pins in certain QDR II and QDR II+ latency configurations.

b. QDR-IV requires that the polarity of all QKB/QKB# pins be swapped with respect to the polarity of the differential buffer inputs on the FPGA to ensure correct data capture on port B. All QKB pins on the memory device must be connected to the negative pins of the input buffers on the FPGA side, and all QKB# pins on the memory device must be connected to the positive pins of the input buffers on the FPGA side. Notice that the port names at the top-level of the IP already reflect this swap (that is, mem\_qkb is assigned to the negative buffer leg, and mem\_qkb\_n is assigned to the positive buffer leg).

10. You can use a single I/O lane to implement two x4 DQS groups. The pin table specifies which pins within an I/O lane can be used for the two pairs of DQS and DQS# signals. In addition, for x4 DQS groups you must observe the following rules:

- There must be an even number of x4 groups in an external memory interface.
- DQS group 0 and DQS group 1 must be placed in the same I/O lane. Similarly, DQS group 2 and group 3 must be in the same I/O lane. Generally, DQS group X and DQS group X+1 must be in the same I/O lane, where X is an even number.

11. You should place the write data groups according to the DQS grouping in the pin table and pin planner. Output-only data clocks for QDR II, QDR II+, and QDR II+ Extreme, and RLDRAM 3 protocols need not be placed on DQS/DQSn pins, but must be placed on a differential pin pair. They must be placed in the same I/O bank as the corresponding DQS group.

*Note:* For RLDRAM 3, x36 device, DQ[8:0] and DQ[26:18] are referenced to DK0/DK0#, and DQ[17:9] and DQ[35:27] are referenced to DK1/DK1#.

12. For protocols and topologies with bidirectional data pins where a write data group consists of multiple read data groups, you should place the data groups and their respective write and read clock in the same bank to improve I/O timing.

You do not need to specify the location of every data pin manually. If you assign the location for the read capture strobe/clock pin pairs, the Fitter will automatically place the remaining data pins.

13. Ensure that DM/BWS pins are paired with a write data pin by placing one in an I/O pin and another in the pairing pin for that I/O pin. It is recommended—though not required—that you follow the same rule for DBI pins, so that at a later date you have the freedom to repurpose the pin as DM.

*Note:*

1. x4 mode does not support DM/DBI, or Arria 10 EMIF IP for HPS.
2. If you are using an Arria 10 EMIF IP-based RLDRAM II or RLDRAM 3 external memory interface, you should ensure that all the pins in a DQS group (that is, DQ, DM, DK, and QK) are placed in the same I/O bank. This requirement facilitates timing closure and is necessary for successful compilation of your design.

### Multiple Interfaces in the Same I/O Column

To place multiple interfaces in the same I/O column, you must ensure that the global reset signals (global\_reset\_n) for each individual interface all come from the same input pin or signal.



### I/O Banks Selection

- For each memory interface, select consecutive I/O banks.
  - A memory interface can only span across I/O banks in the same I/O column.
  - Because I/O bank 2A is also employed for configuration-related operations, you can use it to construct external memory interfaces only when the following conditions are met:
    - The pins required for configuration related use (such as configuration bus for Fast Passive Parallel mode or control signals for Partial Reconfiguration) are never shared with pins selected for EMIF use, even after configuration is complete.
    - The I/O voltages are compatible.
    - The design has achieved a successful fit in the Quartus Prime software.
- Refer to the Arria 10 Device Handbook and the *Configuration Function* column of the Pin-Out files for more information about pins and configuration modes.
- The number of I/O banks that you require depends on the memory interface width.
  - The 3V I/O bank does not support dynamic OCT or calibrated OCT. To place a memory interface in a 3V I/O bank, ensure that calibrated OCT is disabled for the address/command signals, the memory clock signals, and the data bus signals, during IP generation.
  - In some device packages, the number of I/O pins in some LVDS I/O banks is less than 48 pins.

### Address/Command Pins Location

- All address/command pins for a controller must be in a single I/O bank.
- If your interface uses multiple I/O banks, the address/command pins must use the middle bank. If the number of banks used by the interface is even, any of the two middle I/O banks can be used for address/command pins.
- Address/command pins and data pins cannot share an I/O lane but can share an I/O bank.
- The address/command pin locations for the soft and hard memory controllers are predefined. In the *External Memory Interface Pin Information for Devices* spreadsheet, each index in the "Index within I/O bank" column denotes a dedicated address/command pin function for a given protocol. The index number of the pin specifies to which I/O lane the pin belongs:
  - I/O lane 0—Pins with index 0 to 11
  - I/O lane 1—Pins with index 12 to 23
  - I/O lane 2—Pins with index 24 to 35
  - I/O lane 3—Pins with index 36 to 47
- For memory topologies and protocols that require only three I/O lanes for the address/command pins, use I/O lanes 0, 1, and 2.
- Unused address/command pins in an I/O lane can be used as general-purpose I/O pins.



## CK Pins Assignment

Assign the clock pin (CK pin) according to the number of I/O banks in an interface:

- The number of I/O banks is odd—assign one CK pin to the middle I/O bank.
- The number of I/O banks is even—assign the CK pin to any one of the middle two I/O banks.

Although the Fitter can automatically select the required I/O banks, Intel recommends that you make the selection manually to reduce the pre-fit run time.

## PLL Reference Clock Pin Placement

Place the PLL reference clock pin in the address/command bank. Other I/O banks may not have free pins that you can use as the PLL reference clock pin:

- If you are sharing the PLL reference clock pin between several interfaces, the I/O banks must be consecutive.

The Arria 10 External Memory Interface IP does not support PLL cascading.

## RZQ Pin Placement

You may place the RZQ pin in any I/O bank in an I/O column with the correct VCCIO and VCCPT for the memory interface I/O standard in use. The recommended location is in the address/command I/O bank.

## DQ and DQS Pins Assignment

Intel recommends that you assign the DQS pins to the remaining I/O lanes in the I/O banks as required:

- Constrain the DQ and DQS signals of the same DQS group to the same I/O lane.
- DQ signals from two different DQS groups cannot be constrained to the same I/O lane.

If you do not specify the DQS pins assignment, the Fitter will automatically select the DQS pins.

## Sharing an I/O Bank Across Multiple Interfaces

If you are sharing an I/O bank across multiple external memory interfaces, follow these guidelines:

- The interfaces must use the same protocol, voltage, data rate, frequency, and PLL reference clock.
- You cannot use an I/O bank as the address/command bank for more than one interface. The memory controller and sequencer cannot be shared.
- You cannot share an I/O lane. There is only one DQS input per I/O lane, and an I/O lane can only connect to one memory controller.

## Ping Pong PHY Implementation

The Ping Pong PHY feature instantiates two hard memory controllers—one for the primary interface and one for the secondary interface. The hard memory controller I/O bank of the primary interface is used for address and command and is always adjacent



and above the hard memory controller I/O bank of the secondary interface. All four lanes of the primary hard memory controller I/O bank are used for address and command.

When you use Ping Pong PHY, the EMIF IP exposes two independent Avalon-MM interfaces to user logic; these interfaces correspond to the two hard memory controllers inside the interface. Each Avalon-MM interface has its own set of clock and reset signals. Refer to *Qsys Interfaces* for more information on the additional signals exposed by Ping Pong PHY interfaces.

For more information on Ping Pong PHY in Arria 10, refer to *Functional Description—Arria 10 EMIF*, in this handbook. For pin allocation information for Arria 10 devices, refer to *External Memory Interface Pin Information for Arria 10 Devices* on [www.altera.com](http://www.altera.com).

### **Additional Requirements for DDR3 and DDR4 Ping-Pong PHY Interfaces**

If you are using Ping Pong PHY with a DDR3 or DDR4 external memory interface on an Arria 10 device, follow these guidelines:

- The address and command I/O bank must not contain any DQS group.
- I/O banks that are above the address and command I/O bank must contain only data pins of the primary interface—that is, the interface with the lower DQS group indices.
- The I/O bank immediately below the address and command I/O bank must contain at least one DQS group of the secondary interface—that is, the interface with the higher DQS group indices. This I/O bank can, but is not required to, contain DQS groups of the primary interface.
- I/O banks that are two or more banks below the address and command I/O bank must contain only data pins of the secondary interface.

### **Related Information**

- [Pin-Out Files for Intel FPGAs](#)
- [Functional Description—Arria 10 EMIF](#)
- [External Memory Interface Pin Information for Arria 10 Devices](#)
- [Restrictions on I/O Bank Usage for Arria 10 EMIF IP with HPS](#)

## **1.2.2. Resource Sharing Guidelines for Arria 10 EMIF IP**

In Arria 10, different external memory interfaces can share PLL reference clock pins, core clock networks, I/O banks, and hard Nios processors. Each I/O bank has DLL and PLL resources, therefore these do not need to be shared. The Fitter automatically merges DLL and PLL resources when a bank is shared by different external memory interfaces, and duplicates them for a multi-I/O-bank external memory interface.

### **Multiple Interfaces in the Same I/O Column**

To place multiple interfaces in the same I/O column, you must ensure that the global reset signals (`global_reset_n`) for each individual interface all come from the same input pin or signal.



### PLL Reference Clock Pin

To conserve pin usage and enable core clock network and I/O bank sharing, you can share a PLL reference clock pin between multiple external memory interfaces. Sharing of a PLL reference clock pin also implies sharing of the reference clock network.

Observe the following guidelines for sharing the PLL reference clock pin:

1. To share a PLL reference clock pin, connect the same signal to the `pll_ref_clk` port of multiple external memory interfaces in the RTL code.
2. Place related external memory interfaces in the same I/O column.
3. Place related external memory interfaces in adjacent I/O banks. If you leave an unused I/O bank between the I/O banks used by the external memory interfaces, that I/O bank cannot be used by any other external memory interface with a different PLL reference clock signal.

**Note:**

The `pll_ref_clk` pin can be placed in the address and command I/O bank or in a data I/O bank, there is no impact on timing. However, for greatest flexibility during debug (such as when creating designs with narrower interfaces), the recommended placement is in the address and command I/O bank.

### Core Clock Network

To access all external memory interfaces synchronously and to reduce global clock network usage, you may share the same core clock network with other external memory interfaces.

Observe the following guidelines for sharing the core clock network:

1. To share a core clock network, connect the `clks_sharing_master_out` of the master to the `clks_sharing_slave_in` of all slaves in the RTL code.
2. Place related external memory interfaces in the same I/O column.
3. Related external memory interface must have the same rate, memory clock frequency, and PLL reference clock.
4. If you are sharing core clocks between a Ping Pong PHY and a hard controller that have the same protocol, rate, and frequency, the Ping Pong PHY must be the core clock master.

### I/O Bank

To reduce I/O bank utilization, you may share an I/O Bank with other external memory interfaces.

Observe the following guidelines for sharing an I/O Bank:

1. Related external memory interfaces must have the same protocol, rate, memory clock frequency, and PLL reference clock.
2. You cannot use a given I/O bank as the address and command bank for more than one external memory interface.
3. You cannot share an I/O lane between external memory interfaces, but an unused pin can serve as a general purpose I/O pin, of compatible voltage and termination standards.



### Hard Nios Processor

All external memory interfaces residing in the same I/O column will share the same hard Nios processor. The shared hard Nios processor calibrates the external memory interfaces serially.

### Reset Signal

When multiple external memory interfaces occupy the same I/O column, they must share the same IP reset signal.

## 1.3. Guidelines for Intel Stratix® 10 External Memory Interface IP

Intel Stratix® 10 devices contain up to three I/O columns that external memory interfaces can use. The Stratix 10 I/O subsystem resides in the I/O columns. Each column contains multiple I/O banks, each of which consists of four I/O lanes. An I/O lane is a group of twelve I/O ports.

The I/O column, I/O bank, I/O lane, adjacent I/O bank, and pairing pin for every physical I/O pin can be uniquely identified by the Bank Number and Index within I/O Bank values, which are defined in each Stratix 10 device pin-out file.

- The numeric component of the Bank Number value identifies the I/O column, while the letter represents the I/O bank.
- The Index within I/O Bank value falls within one of the following ranges: 0 to 11, 12 to 23, 24 to 35, or 36 to 47, and represents I/O lanes 1, 2, 3, and 4, respectively.
- The adjacent I/O bank is defined as the I/O bank with same column number but the letter is either before or after the respective I/O bank letter in the A-Z system.
- The pairing pin for an I/O pin is located in the same I/O bank. You can identify the pairing pin by adding one to its Index within I/O Bank number (if it is an even number), or by subtracting one from its Index within I/O Bank number (if it is an odd number).

For example, a physical pin with a Bank Number of 2M and Index within I/O Bank of 22, indicates that the pin resides in I/O lane 2, in I/O bank 2M, in column 2. The adjacent I/O banks are 2L and 2N. The pairing pin for this physical pin is the pin with an Index within I/O Bank of 23 and Bank Number of 2M

.

### 1.3.1. General Pin-Out Guidelines for Stratix 10 EMIF IP

You should follow the recommended guidelines when placing pins for all external memory interface pins targeting Stratix 10 devices, whether you are using the hard memory controller or your own solution.

If you are using the hard memory controller, you should employ the relative pin locations defined in the `<variation_name>/altera_emif_arch_nd_version number/<synth/sim>/<variation_name>_altera_emif_arch_nd_version number_<unique ID>.readme.txt` file, which is generated with your IP.

**Note:**

1. EMIF IP pin-out requirements for the Stratix 10 Hard Processor Subsystem (HPS) are more restrictive than for a non-HPS memory interface. The HPS EMIF IP defines a fixed pin-out in the Quartus Prime IP file (.qip), based on the IP configuration. When targeting Stratix 10 HPS, you do not need to make location assignments for external memory interface pins. To obtain the HPS-specific external memory interface pin-out, compile the interface in the Quartus Prime software. Alternatively, consult the device handbook or the device pin-out files. For information on how you can customize the HPS EMIF pin-out, refer to *Restrictions on I/O Bank Usage for Stratix 10 EMIF IP with HPS*.
2. Ping Pong PHY, PHY only, RLDRAMx, QDRx and LPDDR3 are not supported with HPS.

Observe the following guidelines when placing pins for your Stratix 10 external memory interface:

1. Ensure that the pins of a single external memory interface reside within a single I/O column.
2. An external memory interface can occupy one or more banks in the same I/O column. When an interface must occupy multiple banks, ensure that those banks are adjacent to one another. (That is, the banks must contain the same column number and letter before or after the respective I/O bank letter.)
3. Be aware that any pin in the same bank that is not used by an external memory interface is available for use as a general purpose I/O of compatible voltage and termination settings.
4. All address and command pins and their associated clock pins (CK and CK#) must reside within a single bank. The bank containing the address and command pins is identified as the address and command bank.
5. To minimize latency, when the interface uses more than two banks, you must select the center bank of the interface as the address and command bank.
6. The address and command pins and their associated clock pins in the address and command bank must follow a fixed pin-out scheme, as defined in the *Stratix 10 External Memory Interface Pin Information File*, which is available on [www.altera.com](http://www.altera.com).

You do not have to place every address and command pin manually. If you assign the location for one address and command pin, the Fitter automatically places the remaining address and command pins.

**Note:** The pin-out scheme is a hardware requirement that you must follow, and can vary according to the topology of the memory device. Some schemes require three lanes to implement address and command pins, while others require four lanes. To determine which scheme to follow, refer to the messages window during parameterization of your IP, or to the `<variation_name>/altera_emif_arch_nd_<version>/<synth_sim>/<variation_name>_altera_emif_arch_nd_<version>_<unique_ID>_readme.txt` file after you have generated your IP.

7. An unused I/O lane in the address and command bank can serve to implement a data group, such as a x8 DQS group. The data group must be from the same controller as the address and command signals.
8. An I/O lane must not be used by both address and command pins and data pins.



9. Place read data groups according to the DQS grouping in the pin table and Pin Planner. Read data strobes (such as DQS and DQS#) or read clocks (such as CQ and CQ# / QK and QK#) must reside at physical pins capable of functioning as DQS/CQ and DQSn/CQn for a specific read data group size. You must place the associated read data pins (such as DQ and Q), within the same group.

*Note:*

  - a. Unlike other device families, there is no need to swap CQ/CQ# pins in certain QDR II and QDR II+ latency configurations.
  - b. QDR-IV requires that the polarity of all QKB/QKB# pins be swapped with respect to the polarity of the differential buffer inputs on the FPGA to ensure correct data capture on port B. All QKB pins on the memory device must be connected to the negative pins of the input buffers on the FPGA side, and all QKB# pins on the memory device must be connected to the positive pins of the input buffers on the FPGA side. Notice that the port names at the top-level of the IP already reflect this swap (that is, mem\_qkb is assigned to the negative buffer leg, and mem\_qkb\_n is assigned to the positive buffer leg).
10. You can implement two x4 DQS groups with a single I/O lane. The pin table specifies which pins within an I/O lane can be used for the two pairs of DQS and DQS# signals. In addition, for x4 DQS groups you must observe the following rules:
  - There must be an even number of x4 groups in an external memory interface.
  - DQS group 0 and DQS group 1 must be placed in the same I/O lane. Similarly, DQS group 2 and group 3 must be in the same I/O lane. Generally, DQS group X and DQS group X+1 must be in the same I/O lane, where X is an even number.
11. You should place the write data groups according to the DQS grouping in the pin table and pin planner. Output-only data clocks for QDR II, QDR II+, and QDR II+ Extreme, and RLDRAM 3 protocols need not be placed on DQS/DQSn pins, but must be placed on a differential pin pair. They must be placed in the same I/O bank as the corresponding DQS group.

*Note:* For RLDRAM 3, x36 device, DQ[8:0] and DQ[26:18] are referenced to DK0/DK0#, and DQ[17:9] and DQ[35:27] are referenced to DK1/DK1#.
12. For protocols and topologies with bidirectional data pins where a write data group consists of multiple read data groups, you should place the data groups and their respective write and read clock in the same bank to improve I/O timing.

You do not need to specify the location of every data pin manually. If you assign the location for the read capture strobe/clock pin pairs, the Fitter will automatically place the remaining data pins.
13. Ensure that DM/BWS pins are paired with a write data pin by placing one in an I/O pin and another in the pairing pin for that I/O pin. It is recommended—though not required—that you follow the same rule for DBI pins, so that at a later date you have the freedom to repurpose the pin as DM.

*Note:*

1. x4 mode does not support DM/DBI, or Stratix 10 EMIF IP for HPS.
2. If you are using a Stratix 10 EMIF IP-based RLDRAM 3 external memory interface, you should ensure that all the pins in a DQS group (that is, DQ, DM, DK, and QK) are placed in the same I/O bank. This requirement facilitates timing closure and is necessary for successful compilation of your design.



## Multiple Interfaces in the Same I/O Column

To place multiple interfaces in the same I/O column, you must ensure that the global reset signals (global\_reset\_n) for each individual interface all come from the same input pin or signal.

### I/O Banks Selection

- For each memory interface, select adjacent I/O banks. (That is, select banks that contain the same column number and letter before or after the respective I/O bank letter.)
- A memory interface can only span across I/O banks in the same I/O column.
- The number of I/O banks that you require depends on the memory interface width.
- In some device packages, the number of I/O pins in some LVDS I/O banks is less than 48 pins.

### Address/Command Pins Location

- All address/command pins for a controller must be in a single I/O bank.
- If your interface uses multiple I/O banks, the address/command pins must use the middle bank. If the number of banks used by the interface is even, any of the two middle I/O banks can be used for address/command pins.
- Address/command pins and data pins cannot share an I/O lane but can share an I/O bank.
- The address/command pin locations for the soft and hard memory controllers are predefined. In the *External Memory Interface Pin Information for Devices* spreadsheet, each index in the "Index within I/O bank" column denotes a dedicated address/command pin function for a given protocol. The index number of the pin specifies to which I/O lane the pin belongs:
  - I/O lane 0—Pins with index 0 to 11
  - I/O lane 1—Pins with index 12 to 23
  - I/O lane 2—Pins with index 24 to 35
  - I/O lane 3—Pins with index 36 to 47
- For memory topologies and protocols that require only three I/O lanes for the address/command pins, use I/O lanes 0, 1, and 2.
- Unused address/command pins in an I/O lane can serve as general-purpose I/O pins.

### CK Pins Assignment

Assign the clock pin (CK pin) according to the number of I/O banks in an interface:

- The number of I/O banks is odd—assign one CK pin to the middle I/O bank.
- The number of I/O banks is even—assign the CK pin to any one of the middle two I/O banks.

Although the Fitter can automatically select the required I/O banks, Intel recommends that you make the selection manually to reduce the pre-fit run time.



### PLL Reference Clock Pin Placement

Place the PLL reference clock pin in the address/command bank. Other I/O banks may not have free pins that you can use as the PLL reference clock pin:

- If you are sharing the PLL reference clock pin between several interfaces, the I/O banks must be adjacent. (That is, the banks must contain the same column number and letter before or after the respective I/O bank letter.)

The Stratix 10 External Memory Interface IP does not support PLL cascading.

### RZQ Pin Placement

You may place the RZQ pin in any I/O bank in an I/O column with the correct  $V_{CCIO}$  and  $V_{CCPT}$  for the memory interface I/O standard in use. However, it is recommended to place the RZQ pin in the address/command I/O bank, for greater flexibility during debug if a narrower interface project is required for testing.

### DQ and DQS Pins Assignment

Intel recommends that you assign the DQS pins to the remaining I/O lanes in the I/O banks as required:

- Constrain the DQ and DQS signals of the same DQS group to the same I/O lane.
- DQ signals from two different DQS groups cannot be constrained to the same I/O lane.

If you do not specify the DQS pins assignment, the Fitter will select the DQS pins automatically.

### Sharing an I/O Bank Across Multiple Interfaces

If you are sharing an I/O bank across multiple external memory interfaces, follow these guidelines:

- The interfaces must use the same protocol, voltage, data rate, frequency, and PLL reference clock.
- You cannot use an I/O bank as the address/command bank for more than one interface. The memory controller and sequencer cannot be shared.
- You cannot share an I/O lane. There is only one DQS input per I/O lane, and an I/O lane can connect to only one memory controller.

### Ping Pong PHY Implementation

The Ping Pong PHY feature instantiates two hard memory controllers—one for the primary interface and one for the secondary interface. The hard memory controller I/O bank of the primary interface is used for address and command and is always adjacent (contains the same column number and letter before or after the respective I/O bank letter) and above the hard memory controller I/O bank of the secondary interface. All four lanes of the primary hard memory controller I/O bank are used for address and command.

When you use Ping Pong PHY, the EMIF IP exposes two independent Avalon-MM interfaces to user logic; these interfaces correspond to the two hard memory controllers inside the interface. Each Avalon-MM interface has its own set of clock and reset signals. Refer to *Qsys Interfaces* for more information on the additional signals exposed by Ping Pong PHY interfaces.



For more information on Ping Pong PHY in Stratix 10, refer to *Functional Description—Stratix 10 EMIF*, in this handbook. For pin allocation information for Stratix 10 devices, refer to *External Memory Interface Pin Information for Stratix 10 Devices* on [www.altera.com](http://www.altera.com).

### Additional Requirements for DDR3 and DDR4 Ping-Pong PHY Interfaces

If you are using Ping Pong PHY with a DDR3 or DDR4 external memory interface on a Stratix 10 device, follow these guidelines:

- The address and command I/O bank must not contain any DQS group.
- I/O banks that are above the address and command I/O bank must contain only data pins of the primary interface—that is, the interface with the lower DQS group indices.
- The I/O bank immediately below the address and command I/O bank must contain at least one DQS group of the secondary interface—that is, the interface with the higher DQS group indices. This I/O bank can, but is not required to, contain DQS groups of the primary interface.
- I/O banks that are two or more banks below the address and command I/O bank must contain only data pins of the secondary interface.

### 1.3.2. Resource Sharing Guidelines for Stratix 10 EMIF IP

In Stratix 10, different external memory interfaces can share PLL reference clock pins, core clock networks, I/O banks, and hard Nios processors. Each I/O bank has DLL and PLL resources, therefore these do not need to be shared. The Fitter automatically merges DLL and PLL resources when a bank is shared by different external memory interfaces, and duplicates them for a multi-I/O-bank external memory interface.

#### PLL Reference Clock Pin

To conserve pin usage and enable core clock network and I/O bank sharing, you can share a PLL reference clock pin between multiple external memory interfaces; the interfaces must be of the same protocol, rate, and frequency. Sharing of a PLL reference clock pin also implies sharing of the reference clock network.

Observe the following guidelines for sharing the PLL reference clock pin:

1. To share a PLL reference clock pin, connect the same signal to the `pll_ref_clk` port of multiple external memory interfaces in the RTL code.
2. Place related external memory interfaces in the same I/O column.
3. Place related external memory interfaces in adjacent I/O banks. If you leave an unused I/O bank between the I/O banks used by the external memory interfaces, that I/O bank cannot be used by any other external memory interface with a different PLL reference clock signal.

#### Note:

You can place the `pll_ref_clk` pin in the address and command I/O bank or in a data I/O bank, there is no impact on timing. However, the recommendation is to place it in the address/command I/O bank.

#### Core Clock Network

To access all external memory interfaces synchronously and to reduce global clock network usage, you may share the same core clock network with other external memory interfaces.



Observe the following guidelines for sharing the core clock network:

1. To share a core clock network, connect the `clks_sharing_master_out` of the master to the `clks_sharing_slave_in` of all slaves in the RTL code.
2. Place related external memory interfaces in the same I/O column.
3. Related external memory interfaces must have the same rate, memory clock frequency, and PLL reference clock.

### I/O Bank

To reduce I/O bank utilization, you may share an I/O Bank with other external memory interfaces.

Observe the following guidelines for sharing an I/O Bank:

1. Related external memory interfaces must have the same protocol, rate, memory clock frequency, and PLL reference clock.
2. You cannot use a given I/O bank as the address and command bank for more than one external memory interface.
3. You cannot share an I/O lane between external memory interfaces, but an unused pin can serve as a general purpose I/O pin, of compatible voltage and termination standards.

### Hard Nios Processor

All external memory interfaces residing in the same I/O column will share the same hard Nios processor. The shared hard Nios processor calibrates the external memory interfaces serially.

### Reset Signal

When multiple external memory interfaces occupy the same I/O column, they must share the same IP reset signal.

## 1.4. Guidelines for UniPHY-based External Memory Interface IP

Intel recommends that you place all the pins for one memory interface (attached to one controller) on the same side of the device. For projects where I/O availability is limited and you must spread the interface on two sides of the device, place all the input pins on one side and the output pins on an adjacent side of the device, along with their corresponding source-synchronous clock.

### 1.4.1. General Pin-out Guidelines for UniPHY-based External Memory Interface IP

For best results in laying out your UniPHY-based external memory interface, you should observe the following guidelines.

**Note:** For a unidirectional data bus as in QDR II and QDR II+ SRAM interfaces, do not split a read data pin group or a write data pin group onto two sides. You should also not split the address and command group onto two sides either, especially when you are interfacing with QDR II and QDR II+ SRAM burst-length-of-two devices, where the address signals are double data rate. Failure to adhere to these rules might result in timing failure.



In addition, there are some exceptions for the following interfaces:

- $\times 36$  emulated QDR II and QDR II+ SRAM in Arria II, Stratix III, and Stratix IV devices.
- RLDRAM II and RLDRAM 3 CIO devices.
- QDR II/+ SDRAM burst-length-of-two devices.
- You must compile the design in the Quartus Prime software to ensure that you are not violating signal integrity and Quartus Prime placement rules, which is critical when you have transceivers in the same design.

The following are general guidelines for placing pins optimally for your memory interfaces:

1. For Arria II GZ, Arria V, Cyclone V, Stratix III, Stratix IV, and Stratix V designs, if you are using OCT, the RUP and RDN, or RZQ pins must be in any bank with the same I/O voltage as your memory interface signals and often use two DQS or DQ pins from a group. If you decide to place the RUP and RDN, or RZQ pins in a bank where the DQS and DQ groups are used, place these pins first and then determine how many DQ pins you have left, to find out if your data pins can fit in the remaining pins. Refer to *OCT Support for Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone V, Stratix III, Stratix IV, and Stratix V Devices*.
2. Use the PLL that is on the same side of the memory interface. If the interface is spread out on two adjacent sides, you may use the PLL that is located on either adjacent side. You must use the dedicated input clock pin to that particular PLL as the reference clock for the PLL. The input of the memory interface PLL cannot come from the FPGA clock network.
3. The Intel FPGA IP uses the output of the memory interface PLL as the DLL input reference clock. Therefore, ensure you select a PLL that can directly feed a suitable DLL.

**Note:** Alternatively, you can use an external pin to feed into the DLL input reference clock. The available pins are also listed in the External Memory Interfaces chapter of the relevant device family handbook. You can also activate an unused PLL clock output, set it at the desired DLL frequency, and route it to a PLL dedicated output pin. Connect a trace on the PCB from this output pin to the DLL reference clock pin, but be sure to include any signal integrity requirements such as terminations.

4. Read data pins require the usage of DQS and DQ group pins to have access to the DLL control signals.

**Note:** In addition, QVLD pins in RLDRAM II and RLDRAM 3 DRAM, and QDR II+ SRAM must use DQS group pins, when the design uses the QVLD signal. None of the Intel FPGA IP uses QVLD pins as part of read capture, so theoretically you do not need to connect the QVLD pins if you are using the Intel solution. It is good to connect it anyway in case the Intel solution gets updated to use QVLD pins.

5. In differential clocking (DDR3/DDR2 SDRAM, RLDRAM II, and RLDRAM 3 interfaces), connect the positive leg of the read strobe or clock to a DQS pin, and the negative leg of the read strobe or clock to a DQSn pin. For QDR II or QDR II+ SRAM devices with 2.5 or 1.5 cycles of read latency, connect the CQ pin to a DQS pin, and the CQn pin to a CQn pin (and not the DQSn pin). For QDR II or QDR II+ SRAM devices with 2.0 cycles of read latency, connect the CQ pin to a CQn pin, and the CQn pin to a DQS pin.



6. Write data (if unidirectional) and data mask pins (DM or BWSn) pins must use DQS groups. While the DLL phase shift is not used, using DQS groups for write data minimizes skew, and must use the SW and TCCS timing analysis methodology.
7. Assign the write data strobe or write data clock (if unidirectional) in the corresponding DQS/DQSn pin with the write data groups that place in DQ pins (except in RLDRAM II and RLDRAM 3 CIO devices). Refer to the *Pin-out Rule Exceptions* for your memory interface protocol.

*Note:* When interfacing with a DDR, or DDR2, or DDR3 SDRAM without leveling, put the CK and CK# pairs in a single  $\times 4$  DQS group to minimize skew between clocks and maximize margin for the  $t_{DQSS}$ ,  $t_{DSS}$ , and  $t_{DSH}$  specifications from the memory devices.
8. Assign any address pins to any user I/O pin. To minimize skew within the address pin group, you should assign the address pins in the same bank or side of the device.
9. Assign the command pins to any I/O pins and assign the pins in the same bank or device side as the other memory interface pins, especially address and memory clock pins. The memory device usually uses the same clock to register address and command signals.
  - In QDR II and QDR II+ SRAM interfaces where the memory clock also registers the write data, assign the address and command pins in the same I/O bank or same side as the write data pins, to minimize skew.
  - For more information about assigning memory clock pins for different device families and memory standards, refer to *Pin Connection Guidelines Tables*.

#### Related Information

- [Pin Connection Guidelines Tables](#) on page 61
- [Additional Guidelines for Arria V GZ and Stratix V Devices](#) on page 66
- [OCT Support](#) on page 37
- [Pin-out Rule Exceptions for  \$\times 36\$  Emulated QDR II and QDR II+ SRAM Interfaces in Arria II, Stratix III and Stratix IV Devices](#) on page 53
- [Pin-out Rule Exceptions for QDR II and QDR II+ SRAM Burst-length-of-two Interfaces](#) on page 60
- [Pin-out Rule Exceptions for RLDRAM II and RLDRAM 3 Interfaces](#) on page 58

#### 1.4.2. Pin-out Rule Exceptions for $\times 36$ Emulated QDR II and QDR II+ SRAM Interfaces in Arria II, Stratix III and Stratix IV Devices

A few packages in the Arria II, Arria V GZ, Stratix III, Stratix IV, and Stratix V device families do not offer any  $\times 32/\times 36$  DQS groups where one read clock or strobe is associated with 32 or 36 read data pins. This limitation exists in the following I/O banks:

- All I/O banks in U358- and F572-pin packages for all Arria II GX devices
- All I/O banks in F484-pin packages for all Stratix III devices
- All I/O banks in F780-pin packages for all Arria II GZ, Stratix III, and Stratix IV devices; top and side I/O banks in F780-pin packages for all Stratix V and Arria V GZ devices



- All I/O banks in F1152-pin packages for all Arria II GZ, Stratix III, and Stratix IV devices, except EP4SGX290, EP4SGX360, EP4SGX530, EPAGZ300, and EPAGZ350 devices
- Side I/O banks in F1517- and F1760-pin packages for all Stratix III devices
- All I/O banks in F1517-pin for EP4SGX180, EP4SGX230, EP4S40G2, EP4S40G5, EP4S100G2, EP4S100G5, and EPAGZ225 devices
- Side I/O banks in F1517-, F1760-, and F1932-pin packages for all Arria II GZ and Stratix IV devices

This limitation limits support for  $\times 36$  QDR II and QDR II+ SRAM devices. To support these memory devices, this following section describes how you can emulate the  $\times 32/\times 36$  DQS groups for these devices.

- The maximum frequency supported in  $\times 36$  QDR II and QDR II+ SRAM interfaces using  $\times 36$  emulation is lower than the maximum frequency when using a native  $\times 36$  DQS group.

**Note:** The F484-pin package in Stratix III devices cannot support  $\times 32/\times 36$  DQS group emulation, as it does not support  $\times 16/\times 18$  DQS groups.

To emulate a  $\times 32/\times 36$  DQS group, combine two  $\times 16/\times 18$  DQS groups together. For  $\times 36$  QDR II and QDR II+ SRAM interfaces, the 36-bit wide read data bus uses two  $\times 16/\times 18$  groups; the 36-bit wide write data uses another two  $\times 16/\times 18$  groups or four  $\times 8/\times 9$  groups. The CQ and CQn signals from the QDR II and QDR II+ SRAM device traces are then split on the board to connect to two pairs of CQ/CQn pins in the FPGA. You might then need to split the QVLD pins also (if you are connecting them). These connections are the only connections on the board that you need to change for this implementation. There is still only one pair of K and Kn connections on the board from the FPGA to the memory (see the following figure). Use an external termination for the CQ/CQn signals at the FPGA end. You can use the FPGA OCT features on the other QDR II interface signals with  $\times 36$  emulation. In addition, there may be extra assignments to be added with  $\times 36$  emulation.

**Note:** Other QDR II and QDR II+ SRAM interface rules also apply for this implementation.

You may also combine four  $\times 9$  DQS groups (or two  $\times 9$  DQS groups and one  $\times 18$  group) on the same side of the device, if not the same I/O bank, to emulate a  $\times 36$  write data group, if you need to fit the QDR II interface in a particular side of the device that does not have enough  $\times 18$  DQS groups available for write data pins. Intel does not recommend using  $\times 4$  groups as the skew may be too large, as you need eight  $\times 4$  groups to emulate the  $\times 36$  write data bits.

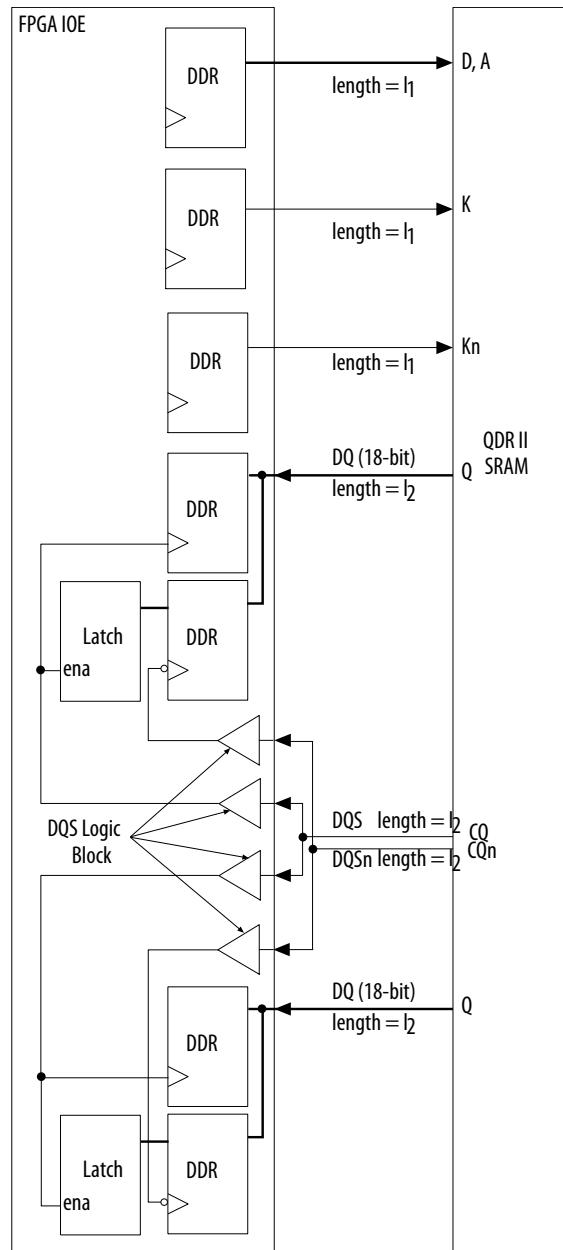
You cannot combine four  $\times 9$  groups to create a  $\times 36$  read data group as the loading on the CQ pin is too large and hence the signal is degraded too much.

When splitting the CQ and CQn signals, the two trace lengths that go to the FPGA pins must be as short as possible to reduce reflection. These traces must also have the same trace delay from the FPGA pin to the Y or T junction on the board. The total trace delay from the memory device to each pin on the FPGA should match the Q trace delay (I2).

**Note:** You must match the trace delays. However, matching trace length is only an approximation to matching actual delay.



**Figure 9. Board Trace Connection for Emulated x36 QDR II and QDR II+ SRAM Interface**



#### 1.4.2.1. Timing Impact on x36 Emulation

With x36 emulation, the CQ/CQn signals are split on the board, so these signals see two loads (to the two FPGA pins)—the DQ signals still only have one load. The difference in loading gives some slew rate degradation, and a later CQ/CQn arrival time at the FPGA pin.



The slew rate degradation factor is taken into account during timing analysis when you indicate in the UniPHY Preset Editor that you are using  $\times 36$  emulation mode. However, you must determine the difference in CQ/CQn arrival time as it is highly dependent on your board topology.

The slew rate degradation factor for  $\times 36$  emulation assumes that CQ/CQn has a slower slew rate than a regular  $\times 36$  interface. The slew rate degradation is assumed not to be more than 500 ps (from 10% to 90%  $V_{CCIO}$  swing). You may also modify your board termination resistor to improve the slew rate of the  $\times 36$ -emulated CQ/CQn signals. If your modified board does not have any slew rate degradation, you do not need to enable the  $\times 36$  emulation timing in the UniPHY-based controller parameter editor.

For more information about how to determine the CQ/CQn arrival time skew, refer to *Determining the CQ/CQn Arrival Time Skew*.

Because of this effect, the maximum frequency supported using  $\times 36$  emulation is lower than the maximum frequency supported using a native  $\times 36$  DQS group.

#### Related Information

[Determining the CQ/CQn Arrival Time Skew](#) on page 56

#### 1.4.2.2. Rules to Combine Groups

For devices that do not have four  $\times 16/\times 18$  groups in a single side of the device to form two  $\times 36$  groups for read and write data, you can form one  $\times 36$  group on one side of the device, and another  $\times 36$  group on the other side of the device. All the read groups have to be on the same edge (column I/O or row I/O) and all write groups have to be on the same type of edge (column I/O or row I/O), so you can have an interface with the read group in column I/O and the write group in row I/O. The only restriction is that you cannot combine an  $\times 18$  group from column I/O with an  $\times 18$  group from row IO to form a  $\times 36$ -emulated group.

For vertical migration with the  $\times 36$  emulation implementation, check if migration is possible and enable device migration in the Quartus Prime software.

**Note:** I/O bank 1C in both Stratix III and Stratix IV devices has dual-function configuration pins. Some of the DQS pins may not be available for memory interfaces if these are used for device configuration purposes.

Each side of the device in these packages has four remaining  $\times 8/\times 9$  groups. You can combine four of the remaining for the write side (only) if you want to keep the  $\times 36$  QDR II and QDR II+ SRAM interface on one side of the device, by changing the **Memory Interface Data Group** default assignment, from the default **18** to **9**.

For more information about rules to combine groups for your target device, refer to the External Memory Interfaces chapter in the respective device handbooks.

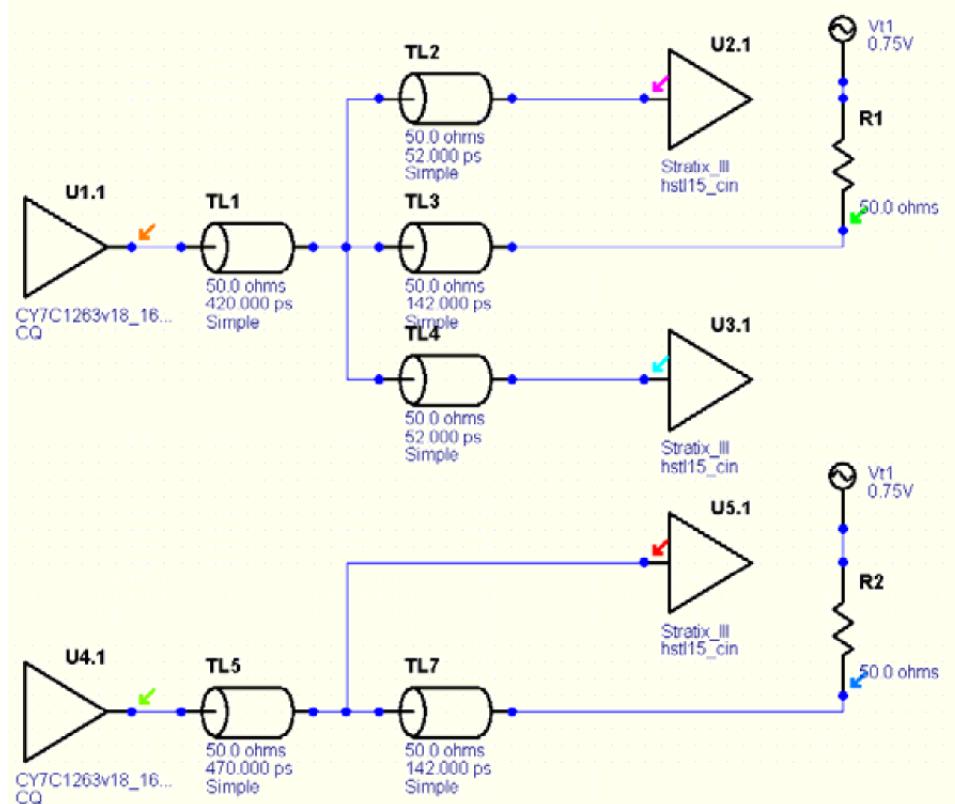
#### 1.4.2.3. Determining the CQ/CQn Arrival Time Skew

Before compiling a design in the Quartus Prime software, you need to determine the CQ/CQn arrival time skew based on your board simulation. You then need to apply this skew in the **report\_timing.tcl** file of your QDR II and QDR II+ SRAM interface in the Quartus Prime software.



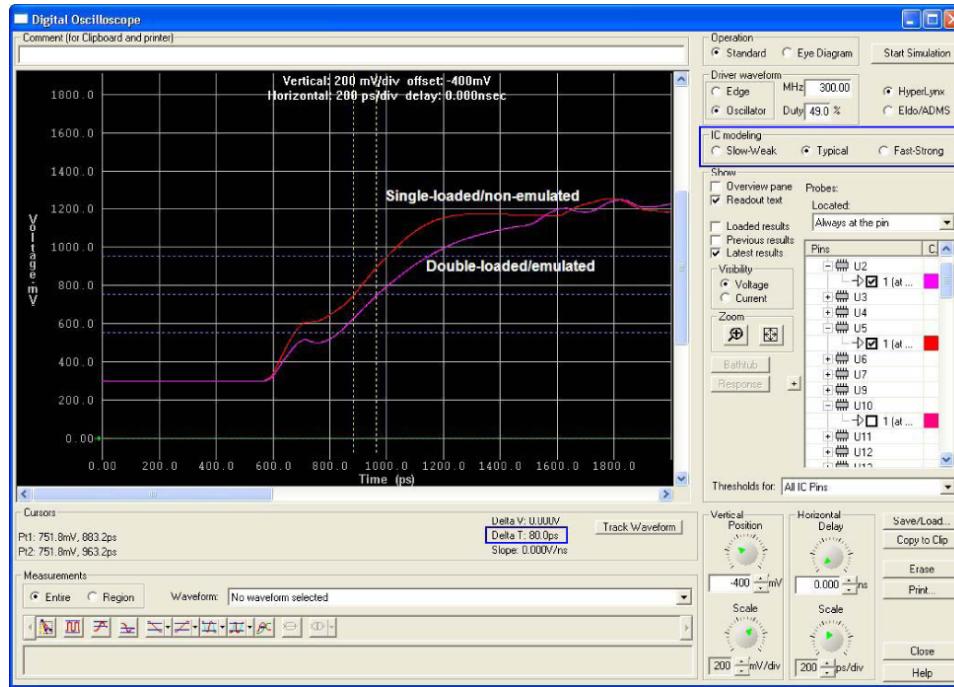
The following figure shows an example of a board topology comparing an emulated case where CQ is double-loaded and a non-emulated case where CQ only has a single load.

**Figure 10. Board Simulation Topology Example**



Run the simulation and look at the signal at the FPGA pin. The following figure shows an example of the simulation results from the preceding figure. As expected, the double-loaded emulated signal, in pink, arrives at the FPGA pin later than the single-loaded signal, in red. You then need to calculate the difference of this arrival time at VREF level (0.75 V in this case). Record the skew and rerun the simulation in the other two cases (slow-weak and fast-strong). To pick the largest and smallest skew to be included in Quartus Prime timing analysis, follow these steps:

1. Open the `<variation_name>_report_timing.tcl` and search for `tmin_additional_dqs_variation`.
2. Set the minimum skew value from your board simulation to `tmin_additional_dqs_variation`.
3. Set the maximum skew value from your board simulation to `tmax_additional_dqs_variation`.
4. Save the `.tcl` file.

**Figure 11. Board Simulation Results**


### 1.4.3. Pin-out Rule Exceptions for RLDRAM II and RLDRAM 3 Interfaces

RLDRAM II and RLDRAM 3 CIO devices have one bidirectional bus for the data, but there are two different sets of clocks: one for read and one for write. As the QK and QK# already occupies the DQS and DQSn pins needed for read, placement of DK and DK# pins are restricted due to the limited number of pins in the FPGA. This limitations causes the exceptions to the previous rules, which are discussed below.

The address or command pins of RLDRAM II must be placed in a DQ-group because these pins are driven by the PHY clock. Half-rate RLDRAM II interfaces and full-rate RLDRAM 3 interfaces use the PHY clock for both the DQ pins and the address or command pins.

#### 1.4.3.1. Interfacing with x9 RLDRAM II CIO Devices

RLDRAM 3 devices do not have the x9 configuration.

RLDRAM II devices have the following pins:

- 2 pins for QK and QK# signals
- 9 DQ pins (in a x8/x9 DQS group)
- 2 pins for DK and DK# signals
- 1 DM pin
- 14 pins total (15 if you have a QVLD)



In the FPGA, the  $\times 8/\times 9$  DQS group consists of 12 pins: 2 for the read clocks and 10 for the data. In this case, move the QVLD (if you want to keep this connected even though this is not used in the Intel FPGA memory interface solution) and the DK and DK# pins to the adjacent DQS group. If that group is in use, move to any available user I/O pins in the same I/O bank.

#### **1.4.3.2. Interfacing with $\times 18$ RLDRAM II and RLDRAM 3 CIO Devices**

This topic describes interfacing with  $\times 18$  RLDRAM II and RLDRAM 3 devices.

RLDRAM II devices have the following pins:

- 4 pins for QK/QK# signals
- 18 DQ pins (in  $\times 8/\times 9$  DQS group)
- 2 pins for DK/DK# signals
- 1 DM pin
- 25 pins total (26 if you have a QVLD)

In the FPGA, you use two  $\times 8/\times 9$  DQS group totaling 24 pins: 4 for the read clocks and 18 for the read data.

Each  $\times 8/\times 9$  group has one DQ pin left over that can either use QVLD or DM, so one  $\times 8/\times 9$  group has the DM pin associated with that group and one  $\times 8/\times 9$  group has the QVLD pin associated with that group.

RLDRAM 3 devices have the following pins:

- 4 pins for QK/QK# signals
- 18 DQ pins (in  $\times 8/\times 9$  DQS group)
- 4 pins for DK/DK# signals
- 2 DM pins
- 28 pins total (29 if you have a QVLD)

In the FPGA, you use two  $\times 8/\times 9$  DQS group totaling 24 pins: 4 for the read clocks and 18 for the read data.

Each  $\times 8/\times 9$  group has one DQ pin left over that can either use QVLD or DM, so one  $\times 8/\times 9$  group has the DM pin associated with that group and one  $\times 8/\times 9$  group has the QVLD pin associated with that group.

#### **1.4.3.3. Interfacing with RLDRAM II and RLDRAM 3 $\times 36$ CIO Devices**

This topic describes interfacing with RLDRAM II and RLDRAM 3  $\times 36$  CIO devices.

RLDRAM II devices have the following pins:

- 4 pins for QK/QK# signals
- 36 DQ pins (in  $\times 16/\times 18$  DQS group)
- 4 pins for DK/DK# signals
- 1 DM pins
- 46 pins total (47 if you have a QVLD)



In the FPGA, you use two  $\times 16/\times 18$  DQS groups totaling 48 pins: 4 for the read clocks and 36 for the read data. Configure each  $\times 16/\times 18$  DQS group to have:

- Two QK/QK# pins occupying the DQS/DQSn pins
- Pick two DQ pins in the  $\times 16/\times 18$  DQS groups that are DQS and DQSn pins in the  $\times 4$  or  $\times 8/\times 9$  DQS groups for the DK and DK# pins
- 18 DQ pins occupying the DQ pins
- There are two DQ pins leftover that you can use for QVLD or DM pins. Put the DM pin in the group associated with DK[1] and the QVLD pin in the group associated with DK[0].
- Check that DM is associated with DK[1] for your chosen memory component.

RLDRAM 3 devices have the following pins:

- 8 pins for QK/QK# signals
- 36 DQ pins (in  $\times 8/\times 9$  DQS group)
- 4 pins for DK/DK# signals
- 2 DM pins
- 48 pins total (49 if you have a QVLD)

In the FPGA, you use four  $\times 8/\times 9$  DQS groups.

In addition, observe the following placement rules for RDRAM 3 interfaces:

For  $\times 18$  devices:

- Use two  $\times 8/\times 9$  DQS groups. Assign the QK/QK# pins and the DQ pins of the same read group to the same DQS group.
- DQ, DM, and DK/DK# pins belonging to the same write group should be assigned to the same I/O sub-bank, for timing closure.
- Whenever possible, assign CK/CK# pins to the same I/O sub-bank as the DK/DK# pins, to improve tCKDK timing.

For  $\times 36$  devices:

- Use four  $\times 8/\times 9$  DQS groups. Assign the QK/QK# pins and the DQ pins of the same read group to the same DQS group.
- DQ, DM, and DK/DK# pins belonging to the same write group should be assigned to the same I/O sub-bank, for timing closure.
- Whenever possible, assign CK/CK# pins to the same I/O sub-bank as the DK/DK# pins, to improve tCKDK timing.

#### 1.4.4. Pin-out Rule Exceptions for QDR II and QDR II+ SRAM Burst-length-of-two Interfaces

If you are using the QDR II and QDR II+ SRAM burst-length-of-two devices, you may want to place the address pins in a DQS group to minimize skew, because these pins are now double data rate too.



The address pins typically do not exceed 22 bits, so you may use one  $\times 18$  DQS groups or two  $\times 9$  DQS groups on the same side of the device, if not the same I/O bank. In Arria V GZ, Stratix III, Stratix IV, and Stratix V devices, one  $\times 18$  group typically has 22 DQ bits and 2 pins for DQS/DQSn pins, while one  $\times 9$  group typically has 10 DQ bits with 2 pins for DQS/DQSn pins. Using  $\times 4$  DQS groups should be a last resort.

### 1.4.5. Pin Connection Guidelines Tables

The following table lists the FPGA pin utilization for DDR, DDR2, and DDR3 SDRAM without leveling interfaces.

**Table 11. FPGA Pin Utilization for DDR, DDR2, and DDR3 SDRAM without Leveling Interfaces**

Interface Pin Description	Memory Device Pin Name	FPGA Pin Utilization			
		Arria II GX	Arria II GZ, Stratix III, and Stratix IV	Arria V, Cyclone V, and Stratix V	MAX 10 FPGA
Memory System Clock	CK and CK# <sup>(1)</sup> <sup>(2)</sup>	If you are using single-ended DQS signaling, place any unused DQ or DQS pins with DIFFOUT capability located in the same bank or on the same side as the data pins.  If you are using differential DQS signaling in UniPHY IP, place on DIFFOUT in the same single DQ group of adequate width to minimize skew.	If you are using single-ended DQS signaling, place any DIFFOUT pins in the same bank or on the same side as the data pins  If you are using differential DQS signaling in UniPHY IP, place any DIFFOUT pins in the same bank or on the same side as the data pins. If there are multiple CK/CK# pairs, place them on DIFFOUT in the same single DQ group of adequate width. For example, DIMMs requiring three memory clock pin-pairs must use a $\times 4$ DQS group.	If you are using single-ended DQS signaling, place any unused DQ or DQS pins with DIFFOUT capability in the same bank or on the same side as the data pins.  If you are using differential DQS signaling, place any unused DQ or DQS pins with DIFFOUT capability for the mem_clk[n:0] and mem_clk_n[n:0] signals (where n>=0). CK and CK# pins must use a pin pair that has DIFFOUT capability.  CK and CK# pins can be in the same group as other DQ or DQS pins. CK and CK# pins can be placed such that one signal of the differential pair is in a DQ group and the other signal is not.  If there are multiple CK and CK# pin pairs, place them on DIFFOUT in the	Place any differential I/O pin pair ( DIFFIO) in the same bank or on the same side as the data pins.

*continued...*



Interface Pin Description	Memory Device Pin Name	FPGA Pin Utilization			
		Arria II GX	Arria II GZ, Stratix III, and Stratix IV	Arria V, Cyclone V, and Stratix V	MAX 10 FPGA
				same single DQ group of adequate width.	
Clock Source	—	Dedicated PLL clock input pin with direct connection to the PLL (not using the global clock network). For Arria II GX, Arria II GZ, Arria V GZ, Stratix III, Stratix IV and Stratix V Devices, also ensure that the PLL can supply the input reference clock to the DLL. Otherwise, refer to alternative DLL input reference clocks (see <i>General Pin-out Guidelines</i> ).			
Reset	—	Dedicated clock input pin to accommodate the high fan-out signal.			
Data	DQ	DQ in the pin table, marked as Q in the Quartus Prime Pin Planner. Each DQ group has a common background color for all of the DQ and DM pins, associated with DQS (and DQSn) pins.			
Data mask	DM				
Data strobe	DQS or DQS and DQSn (DDR2 and DDR2 SDRAM only)	DQS (S in the Quartus Prime Pin Planner) for single-ended DQS signaling or DQS and DQSn (S and Sbar in the Quartus Prime Pin Planner) for differential DQS signaling. DDR2 supports either single-ended or differential DQS signaling. DDR3 SDRAM mandates differential DQS signaling.			
Address and command	A[], BA[], CAS#, CKE, CS#, ODT, RAS#, WE#, RESET#	Any user I/O pin. To minimize skew, you must place the address and command pins in the same bank or side of the device as the CK/CK# pins, DQ, DQS, or DM pins. The reset# signal is only available in DDR3 SDRAM interfaces. Intel devices use the SSTL-15 I/O standard on the RESET# signal to meet the voltage requirements of 1.5 V CMOS at the memory device. Intel recommends that you do not terminate the RESET# signal to VTT.			
Notes to Table: 1. The first CK/CK# pair refers to mem_clk[0] or mem_clk_n[0] in the IP core. 2. The restriction on the placement for the first CK/CK# pair is required because this placement allows the mimic path that the IP VT tracking uses to go through differential I/O buffers to mimic the differential DQS signals.					

### Related Information

[General Pin-out Guidelines for UniPHY-based External Memory Interface IP](#) on page 51

#### 1.4.5.1. DDR3 SDRAM With Leveling Interface Pin Utilization Applicable for Arria V GZ, Stratix III, Stratix IV, and Stratix V Devices

The following table lists the FPGA pin utilization for DDR3 SDRAM with leveling interfaces.



**Table 12. DDR3 SDRAM With Leveling Interface Pin Utilization Applicable for Arria V GZ, Stratix III, Stratix IV, and Stratix V Devices**

Interface Pin Description	Memory Device Pin Name	FPGA Pin Utilization
Data	DQ	DQ in the pin table, marked as Q in the Quartus Prime Pin Planner. Each DQ group has a common background color for all of the DQ and DM pins, associated with DQS (and DQSn) pins. The $\times 4$ DIMM has the following mapping between DQS and DQ pins: <ul style="list-style-type: none"> <li>• DQS[0] maps to DQ[3:0]</li> <li>• DQS[9] maps to DQ[7:4]</li> <li>• DQS[1] maps to DQ[11:8]</li> <li>• DQS[10] maps to DQ[15:12]</li> </ul> The DQS pin index in other DIMM configurations typically increases sequentially with the DQ pin index (DQS[0]: DQ[3:0]; DQS[1]: DQ[7:4]; DQS[2]: DQ[11:8]). In this DIMM configuration, the DQS pins are indicated this way to ensure pin out is compatible with both $\times 4$ and $\times 8$ DIMMs.
Data Mask	DM	
Data Strobe	DQS and DQSn	DQS and DQSn (S and Sbar in the Quartus Prime Pin Planner)
Address and Command	A[], BA[], CAS#, CKE, CS#, ODT, RAS#, WE#,	Any user I/O pin. To minimize skew, you should place address and command pins in the same bank or side of the device as the following pins: CK/CK# pins, DQ, DQS, or DM pins.
	RESET#	Intel recommends that you use the 1.5V CMOS I/O standard on the RESET# signal. If your board is already using the SSTL-15 I/O standard, you do not terminate the RESET# signal to VTT.
Memory system clock	CK and CK#	For controllers with UniPHY IP, you can assign the memory clock to any unused DIFF_OUT pins in the same bank or on the same side as the data pins. However, for Arria V GZ and Stratix V devices, place the memory clock pins to any unused DQ or DQS pins. Do not place the memory clock pins in the same DQ group as any other DQ or DQS pins.  If there are multiple CK/CK# pin pairs using Arria V GZ or Stratix V devices, you must place them on DIFFOUT in the same single DQ groups of adequate width. For example, DIMMs requiring three memory clock pin-pairs must use a $\times 4$ DQS group.  Placing the multiple CK/CK# pin pairs on DIFFOUT in the same single DQ groups for Stratix III and Stratix IV devices improves timing.
Clock Source	—	Dedicated PLL clock input pin with direct (not using a global clock net) connection to the PLL and optional DLL required by the interface.
Reset	—	Dedicated clock input pin to accommodate the high fan-out signal.

#### 1.4.5.2. QDR II and QDR II+ SRAM Pin Utilization for Arria II, Arria V, Stratix III, Stratix IV, and Stratix V Devices

The following table lists the FPGA pin utilization for QDR II and QDR II+ SRAM interfaces.

**Table 13. QDR II and QDR II+ SRAM Pin Utilization for Arria II, Arria V, Stratix III, Stratix IV, and Stratix V Devices**

Interface Pin Description	Memory Device Pin Name	FPGA Pin Utilization
Read Clock	CQ and CQ# <sup>(1)</sup>	For QDR II SRAM devices with 1.5 or 2.5 cycles of read latency or QDR II+ SRAM devices with 2.5 cycles of read latency, connect CQ to DQS pin (S in the Quartus Prime Pin Planner), and CQn to CQn pin (Qbar in the Quartus Prime Pin Planner). For QDR II or QDR II+ SRAM devices with 2.0 cycles of read latency, connect CQ to CQn pin (Qbar in the Quartus Prime Pin Planner), and CQn to DQS pin (S in the Quartus Prime Pin Planner). Arria V devices do not use CQn. The CQ rising and falling edges are used to clock the read data, instead of separate CQ and CQn signals.
Read Data	Q	DQ pins (Q in the Quartus Prime Pin Planner). Ensure that you are using the DQ pins associated with the chosen read clock pins (DQS and CQn pins). QVLD pins are only available for QDR II+ SRAM devices and note that Intel FPGA IP does not use the QVLD pin.
Data Valid	QVLD	
Memory and Write Data Clock	K and K#	Differential or pseudo-differential DQ, DQS, or DQSn pins in or near the write data group.
Write Data	D	DQ pins. Ensure that you are using the DQ pins associated with the chosen memory and write data clock pins (DQS and DQSn pins).
Byte Write Select	BWS#, NWS#	
Address and Command	A, WPS#, RPS#	Any user I/O pin. To minimize skew, you should place address and command pins in the same bank or side of the device as the following pins: K and K# pins, DQ, DQS, BWS#, and NWS# pins. If you are using burst-length-of-two devices, place the address signals in a DQS group pin as these signals are now double data rate.
Clock source	—	Dedicated PLL clock input pin with direct (not using a global clock net) connection to the PLL and optional DLL required by the interface.
Reset	—	Dedicated clock input pin to accommodate the high fan-out signal

Note to table:

1. For Arria V designs with integer latency, connect the CQ# signal to the CQ/CQ# pins from the pin table and ignore the polarity in the Pin Planner. For Arria V designs with fractional latency, connect the CQ signal to the CQ/CQ# pins from the pin table.

#### 1.4.5.3. RLDRAM II CIO Pin Utilization for Arria II GZ, Arria V, Stratix III, Stratix IV, and Stratix V Devices

The following table lists the FPGA pin utilization for RLDRAM II CIO and RLDRAM 3 interfaces.



**Table 14. RLDRAm II CIO Pin Utilization for Arria II GZ, Arria V, Stratix III, Stratix IV, and Stratix V Devices and RLDRAm 3 Pin Utilization for Arria V GZ and Stratix V Devices**

Interface Pin Description	Memory Device Pin Name	FPGA Pin Utilization
Read Clock	QK and QK# <sup>(1)</sup>	DQS and DQSn pins (S and Sbar in the Quartus Prime Pin Planner)
Data	Q	DQ pins (Q in the Quartus Prime Pin Planner). Ensure that you are using the DQ pins associated with the chosen read clock pins (DQS and DQSn pins). Intel FPGA IP does not use the QVLD pin. You may leave this pin unconnected on your board.
Data Valid	QVLD	You may not be able to fit these pins in a DQS group. For more information about how to place these pins, refer to "Exceptions for RLDRAm II and RLDRAm 3 Interfaces" on page 3-34.
Data Mask	DM	
Write Data Clock	DK and DK#	DQ pins in the same DQS group as the read data (Q) pins or in adjacent DQS group or in the same bank as the address and command pins. For more information, refer to <i>Exceptions for RLDRAm II and RLDRAm 3 Interfaces</i> . DK/DK# must use differential output-capable pins. For Nios-based configuration, the DK pins must be in a DQ group but the DK pins do not have to be in the same group as the data or QK pins.
Memory Clock	CK and CK#	Any differential output-capable pins. For Arria V GZ and Stratix V devices, place any unused DQ or DQS pins with DIFFOUT capability. Place the memory clock pins either in the same bank as the DK or DK# pins to improve DK versus CK timing, or in the same bank as the address and command pins to improve address command timing. Do not place CK and CK# pins in the same DQ group as any other DQ or DQS pins.
Address and Command	A, BA, CS#, REF#, WE#	Any user I/O pins. To minimize skew, you should place address and command pins in the same bank or side of the device as the following pins: CK/CK# pins, DQ, DQS, and DM pins.
Clock source	—	Dedicated PLL clock input pin with direct (not using a global clock net) connection to the PLL and optional DLL required by the interface.
Reset	—	Dedicated clock input pin to accommodate the high fan-out signal
Note to Table:		
1. For Arria V devices, refer to the pin table for the QK and QK# pins. Connect QK and QK# signals to the QK and QK# pins from the pin table and ignore the polarity in the Pin Planner.		

#### Related Information

[Pin-out Rule Exceptions for RLDRAm II and RLDRAm 3 Interfaces](#) on page 58

#### 1.4.5.4. LPDDR2 Pin Utilization for Arria V, Cyclone V, and MAX 10 FPGA Devices

The following table lists the FPGA pin utilization for LPDDR2 SDRAM.

**Table 15. LPDDR2 Pin Utilization for Arria V, Cyclone V, and MAX 10 FPGA Devices**

Interface Pin Description	Memory Device Pin Name	FPGA Pin Utilization
Memory Clock	CK, CKn	Differential clock inputs. All double data rate (DDR) inputs are sampled on both positive and negative edges of the CK signal. Single data rate (SDR) inputs are sampled at the positive

*continued...*



Interface Pin Description	Memory Device Pin Name	FPGA Pin Utilization
		clock edge. Place any unused DQ or DQS pins with DIFFOUT capability for the mem_clk[:n:0] and mem_clk_n[:n:0] signals (where n>=0). Do not place CK and CK# pins in the same group as any other DQ or DQS pins. If there are multiple CK and CK# pin pairs, place them on DIFFOUT in the same single DQ group of adequate width.
Address and Command	CA0-CA9 CSn CKE	Unidirectional DDR command and address bus inputs. Chip Select: CSn is considered to be part of the command code. Clock Enable: CKE HIGH activates and CKE LOW deactivates internal clock signals and therefore device input buffers and output drivers. Place address and command pins in any DDR-capable I/O pin. To minimize skew, Intel recommends using address and command pins in the same bank or side of the device as the CK/CK#, DQ, DQS, or DM pins..
Data	DQ0-DQ7 (x8) DQ0-DQ15 (x16) DQ0-DQ31 (x32)	Bidirectional data bus. Pins are used as data inputs and outputs. DQ in the pin table is marked as Q in the Pin Planner. Each DQ group has a common background color for all of the DQ and DM pins associated with DQS (and DQSn) pins. Place on DQ group pin marked Q in the Pin Planner.
Data Strobe	DQS, DQSn	Data Strobe. The data strobe is bidirectional (used for read and write data) and differential (DQS and DQSn). It is output with read data and input with write data. Place on DQS and DQSn (S and Sbar in the Pin Planner) for differential DQS signaling.
Data Mask	DM0 (x8) DM0-DM1 (x16) DM0-DM3 (x32)	Input Data Mask. DM is the input mask signal for write data. Input data is masked when DM is sampled HIGH coincident with that input data during a write access. DM is sampled on both edges of DQS. DQ in the pin table is marked as Q in the Pin Planner. Each DQ group has a common background color for all of the DQ and DM pins, associated with DQS (and DQSn) pins. Place on DQ group pin marked Q in the Pin Planner.
Clock Source	—	Dedicated PLL clock input pin with direct (not using a global clock net) connection to the PLL and optional DLL required by the interface.
Reset	—	Dedicated clock input pin to accommodate the high fan-out signal.

#### 1.4.5.5. Additional Guidelines for Arria V GZ and Stratix V Devices

This section provides guidelines for improving timing for Arria V GZ and Stratix V devices and the rules that you must follow to overcome timing failures.

##### Performing Manual Pin Placement

The following table lists rules that you can follow to perform proper manual pin placement and avoid timing failures.

The rules are categorized as follows:

- **Mandatory**—This rule is mandatory and cannot be violated as it would result in a no-fit error.
- **Recommended**—This rule is recommended and if violated the implementation is legal but the timing is degraded.
- **Highly Recommended**—This rule is not mandatory but is highly recommended because disregarding this rule might result in timing violations.



**Table 16. Manual Pin Placement Rules**

Rules	Frequency	Device	Reason
<b>Mandatory</b>			
Must place all CK, CK#, address, control, and command pins of an interface in the same I/O sub-bank.	> 800 MHz	All	For optimum timing, clock and data output paths must share as much hardware as possible. For write data pins (for example, DQ/DQS), the best timing is achieved through the DQS Groups.
Must not split interface between top and bottom sides	Any	All	Because PLLs and DLLs on the top edge cannot access the bottom edge of a device and vice-versa.
Must not place pins from separate interfaces in the same I/O sub-banks unless the interfaces share PLL or DLL resources.	Any	All	All pins require access to the same leveling block.
Must not share the same PLL input reference clock unless the interfaces share PLL or DLL resources.	Any	All	Because sharing the same PLL input reference clock forces the same ff-PLL to be used. Each ff-PLL can drive only one PHY clock tree and interfaces not sharing a PLL cannot share a PHY clock tree.
<b>Recommended</b>			
Place all CK, CK#, address, control, and command pins of an interface in the same I/O sub-bank.	<800 MHz	All	Place all CK/CK#, address, control, and command pins in the same I/O sub-bank when address and command timing is critical. For optimum timing, clock and data output paths should share as much hardware as possible. For write data pins (for example, DQ/DQS), the best timing is achieved through the DQS Groups.
Avoid using I/Os at the device corners (for example, sub-bank "A").	Any	A7 (1)	The delay from the FPGA core fabric to the I/O periphery is higher toward the sub-banks in the corners. By not using I/Os at the device corners, you can improve core timing closure.
	>=800 MHz	All	Corner I/O pins use longer delays, therefore avoiding corner I/O pins is recommended for better memory clock performance.
Avoid straddling an interface across the center PLL.	Any	All	Straddling the center PLL causes timing degradation, because it increases the length of the PHY clock tree and increases jitter. By not straddling the center PLL, you can improve core timing closure.
Use the center PLL(f-PLL1) for a wide interface that must straddle across center PLL.	>= 800 MHz	All	Using a non-center PLL results in driving a sub-bank in the opposite quadrant due to long PHY clock tree delay.
Place the DQS/DQS# pins such that all DQ groups of the same interface are next to each other and do not span across the center PLL.	Any	All	To ease core timing closure. If the pins are too far apart then the core logic is also placed apart which results in difficult timing closure.
Place CK, CK#, address, control, and command pins in the same quadrant as DQ groups for improved timing in general.	Any	All	
<b>Highly Recommended</b>			
<i>continued...</i>			



Rules	Frequency	Device	Reason
Place all CK, CK#, address, control, and command pins of an interface in the same I/O sub-bank.	>= 800 MHz	All	For optimum timing, clock and data output paths should share as much hardware as possible. For write data pins (for example, DQ/DQS), the best timing is achieved through the DQS Groups.
Use center PLL and ensure that the PLL input reference clock pin is placed at a location that can drive the center PLL.	>= 800 MHz	All	Using a non-center PLL results in driving a sub-bank in the opposite quadrant due to long PHY clock tree delay.
If center PLL is not accessible, place pins in the same quadrant as the PLL.	>= 800 MHz	All	
<p>Note to Table:</p> <ol style="list-style-type: none"><li>1. This rule is currently applicable to A7 devices only. This rule might be applied to other devices in the future if they show the same failure.</li></ol>			

#### 1.4.5.6. Additional Guidelines for Arria V ( Except Arria V GZ) Devices

This section provides guidelines on how to improve timing for Arria V devices and the rules that you must follow to overcome timing failures.

##### Performing Manual Pin Placement

The following table lists rules you can follow to perform proper manual pin placement and avoid timing failures.

The rules are categorized as follows:

- **Mandatory**—This rule is mandatory and cannot be violated as it would result in a no-fit error.
- **Recommended**—This rule is recommended and if violated the implementation is legal but the timing is degraded.

**Table 17. Manual Pin Placement Rules for Arria V (Except Arria V GZ) Devices**

Rules	Frequency	Device	Reason
<b>Mandatory</b>			
Must place all CK, CK#, address, control, and command pins of an interface on the same device edge as the DQ groups.	All	All	For optimum timing, clock and data output ports must share as much hardware as possible.
Must not place pins from separate interfaces in the same I/O sub-banks unless the interfaces share PLL or DLL resources. To share resources, the interfaces must use the same memory protocol, frequency, controller rate, and phase requirements.	All	All	All pins require access to the same PLL/DLL block.
Must not split interface between top, bottom, and right sides.	All	All	PHYCLK network support interfaces at the same side of the I/O banks only. PHYCLK networks do not support split interface.

*continued...*



Rules	Frequency	Device	Reason
<b>Recommended</b>			
Place the DQS/DQS# pins such that all DQ groups of the same interface are next to each other and do not span across the center PLL.	All	All	To ease core timing closure. If the pins are too far apart then the core logic is also placed apart which results in difficult timing closure.
Place all pins for a memory interface in an I/O bank and use the nearest PLL to that I/O bank for the memory interface.	All	All	Improve timing performance by reducing the PHY clock tree delay.

**Note:** Not all hard memory controllers on a given device package necessarily have the same address widths; some hard memory controllers have 16-bit address capability, while others have only 15-bit addresses.

#### 1.4.5.7. Additional Guidelines for MAX 10 Devices

The following additional guidelines apply when you implement an external memory interface for a MAX 10 device.

##### I/O Pins Not Available for DDR3 or LPDDR2 External Memory Interfaces (Preliminary)

The I/O pins named in the following table are not available for use when implementing a DDR3 or LPDDR2 external memory interface for a MAX 10 device.

	F256	U324	F484	F672
10M16	N16	R15	U21	—
	P16	P15	U22	—
	—	R18	M21	—
	—	P18	L22	—
	—	—	F21	—
	—	—	F20	—
	—	E16	E19	—
	—	D16	F18	—
10M25	N16	—	U21	—
	P16	—	U22	—
	—	—	M21	—
	—	—	L22	—
	—	—	F21	—
	—	—	F20	—
	—	—	E19	—
	—	—	F18	—
	—	—	F17	—
	—	—	E17	—

*continued...*

	<b>F256</b>	<b>U324</b>	<b>F484</b>	<b>F672</b>
10M50	—	—	—	W23
	—	—	—	W24
	—	—	—	U25
	—	—	—	U24
	N16	—	U21	T24
	P16	—	U22	R25
	—	—	M21	R24
	—	—	L22	P25
	—	—	F21	K23
	—	—	F20	K24
	—	—	E19	J23
	—	—	F18	H23
	—	—	F17	G23
	—	—	E17	F23
	—	—	—	G21
	—	—	—	G22

### **Additional Restrictions on I/O Pin Availability**

The following restrictions are in addition to those represented in the above table.

- When implementing a DDR3 or LPDDR2 external memory interface, you can use only 75 percent of the remaining I/O pins in banks 5 and 6 for normal I/O operations.
- When implementing a DDR2 external memory interface, 25 percent of the remaining I/O pins in banks 5 and 6 can be assigned only as input pins.

### **MAX 10 Board Design Considerations**

- For DDR2, DDR3, and LPDDR2 interfaces, the maximum board skew between pins must be lower than 40 ps. This guideline applies to all pins (address, command, clock, and data).
- To minimize unwanted inductance from the board via, Intel recommends that you keep the PCB via depth for  $V_{CCIO}$  banks below 49.5 mil.
- For devices with DDR3 interface implementation, onboard termination is required for the DQ, DQS, and address signals. Intel recommends that you use termination resistor value of  $80 \Omega$  to  $V_{TT}$ .
- For the DQ, address, and command pins, keep the PCB trace routing length less than six inches for DDR3, or less than three inches for LPDDR2.

### **Power Supply Variation for LPDDR2 Interfaces**

For an LPDDR2 interface that targets 200 MHz, constrain the memory device I/O and core power supply variation to within  $\pm 3\%$ .



#### 1.4.5.8. Additional Guidelines for Cyclone V Devices

This topic provides guidelines for improving performance for Cyclone V devices.

##### I/O Pins Connect to Ground for Hard Memory Interface Operation

According to the Cyclone V pin-out file, there are some general I/O pins that are connected to ground for hard memory interface operation. These I/O pins should be grounded to reduce crosstalk from neighboring I/O pins and to ensure the performance of the hard memory interface.

The grounded user I/O pins can also be used as regular I/O pins if you run short of available I/O pins; however, the hard memory interface performance will be reduced if these pins are not connected to ground.

#### 1.4.6. PLLs and Clock Networks

The exact number of clocks and PLLs required in your design depends greatly on the memory interface frequency, and on the IP that your design uses.

For example, you can build simple DDR slow-speed interfaces that typically require only two clocks: system and write. You can then use the rising and falling edges of these two clocks to derive four phases ( $0^\circ$ ,  $90^\circ$ ,  $180^\circ$ , and  $270^\circ$ ). However, as clock speeds increase, the timing margin decreases and additional clocks are required, to optimize setup and hold and meet timing. Typically, at higher clock speeds, you need to have dedicated clocks for resynchronization, and address and command paths.

Intel FPGA memory interface IP uses one PLL, which generates the various clocks needed in the memory interface data path and controller, and provides the required phase shifts for the write clock and address and command clock. The PLL is instantiated when you generate the Intel FPGA memory IPs.

By default, the memory interface IP uses the PLL to generate the input reference clock for the DLL, available in all supported device families. This method eliminates the need of an extra pin for the DLL input reference clock.

The input reference clock to the DLL can come from certain input clock pins or clock output from certain PLLs.

**Note:** Intel recommends using integer PLLs for memory interfaces; handbook specifications are based on integer PLL implementations.

For the actual pins and PLLs connected to the DLLs, refer to the *External Memory Interfaces* chapter of the relevant device family handbook.

You must use the PLL located in the same device quadrant or side as the memory interface and the corresponding dedicated clock input pin for that PLL, to ensure optimal performance and accurate timing results from the Quartus Prime software.

The input clock to the PLL can fan out to logic other than the PHY, so long as the clock input pin to the PLL is a dedicated input clock path, and you ensure that the clock domain transfer between UniPHY and the core logic is clocked by the reference clock going into a global clock.



#### 1.4.6.1. Number of PLLs Available in Intel Device Families

The following table lists the number of PLLs available in Intel device families.

**Table 18. Number of PLLs Available in Intel Device Families**

Device Family	Enhanced PLLs Available
Arria II GX	4-6
Arria II GZ	3-8
Arria V	16-24
Arria V GZ (fPLL)	22-28
Cyclone V	4-8
MAX 10 FPGA	1-4
Stratix III	4-12
Stratix IV	3-12
Stratix V (fPLL)	22-28

Note to Table:

- For more details, refer to the *Clock Networks and PLL* chapter of the respective device family handbook.

#### 1.4.6.2. Number of Enhanced PLL Clock Outputs and Dedicated Clock Outputs Available in Intel Device Families

The following table lists the number of enhanced PLL clock outputs and dedicated clock outputs available in Intel device families.

**Table 19. Number of Enhanced PLL Clock Outputs and Dedicated Clock Outputs Available in Intel Device Families <sup>(1)</sup>**

Device Family	Number of Enhanced PLL Clock Outputs	Number Dedicated Clock Outputs
Arria II GX <sup>(2)</sup>	7 clock outputs each	1 single-ended or 1 differential pair 3 single-ended or 3 differential pair total <sup>(3)</sup>
Arria V	18 clock outputs each	4 single-ended or 2 single-ended and 1 differential pair
Stratix III	Left/right: 7 clock outputs Top/bottom: 10 clock outputs	Left/right: 2 single-ended or 1 differential pair Top/bottom: 6 single-ended or 4 single-ended and 1 differential pair
Arria II GZ and Stratix IV	Left/right: 7 clock outputs Top/bottom: 10 clock outputs	Left/right: 2 single-ended or 1 differential pair Top/bottom: 6 single-ended or 4 single-ended and 1 differential pair
Arria V GZ and Stratix V	18 clock outputs each	4 single-ended or 2 single-ended and 1 differential pair

Notes to Table:

- For more details, refer to the *Clock Networks and PLL* chapter of the respective device family handbook.
- PLL\_5 and PLL\_6 of Arria II GX devices do not have dedicated clock outputs.
- The same PLL clock outputs drives three single-ended or three differential I/O pairs, which are only supported in PLL\_1 and PLL\_3 of the EP2AGX95, EP2AGX125, EP2AGX190, and EP2AGX260 devices.



#### 1.4.6.3. Number of Clock Networks Available in Intel Device Families

The following table lists the number of clock networks available in Intel device families.

**Table 20. Number of Clock Networks Available in Intel Device Families <sup>(1)</sup>**

Device Family	Global Clock Network	Regional Clock Network
Arria II GX	16	48
Arria II GZ	16	64–88
Arria V	16	88
Arria V GZ	16	92
Cyclone V	16	N/A
MAX 10 FPGA	10	
Stratix III	16	64–88
Stratix IV	16	64–88
Stratix V	16	92

Note to Table:

- For more information on the number of available clock network resources per device quadrant to better understand the number of clock networks available for your interface, refer to the *Clock Networks and PLL* chapter of the respective device family handbook.

**Note:** You must decide whether you need to share clock networks, PLL clock outputs, or PLLs if you are implementing multiple memory interfaces.

#### 1.4.6.4. Clock Network Usage in UniPHY-based Memory Interfaces—DDR2 and DDR3 SDRAM <sup>(1) (2)</sup>

The following table lists clock network usage in UniPHY-based memory interfaces for DDR2 and DDR3 protocols.

**Table 21. Clock Network Usage in UniPHY-based Memory Interfaces—DDR2 and DDR3 SDRAM**

Device	DDR3 SDRAM		DDR2 SDRAM	
	Half-Rate		Half-Rate	
	Number of full-rate clock	Number of half-rate clock	Number of full-rate clock	Number of half-rate clock
Stratix III	3 global	1 global 1 regional	1 global 2 global	1 global 1 regional
Arria II GZ and Stratix IV	3 global	1 global 1 regional	1 regional 2 regional	1 global 1 regional
Arria V GZ and Stratix V	1 global 2 regional	2 global	1 regional 2 regional	2 global

Notes to Table:

- There are two additional regional clocks, `pll_avl_clk` and `pll_config_clk` for DDR2 and DDR3 SDRAM with UniPHY memory interfaces.
- In multiple interface designs with other IP, the clock network might need to be modified to get a design to fit. For more information, refer to the *Clock Networks and PLLs* chapter in the respective device handbooks.



#### 1.4.6.5. Clock Network Usage in UniPHY-based Memory Interfaces—RLDRAM II, and QDR II and QDR II+ SRAM

The following table lists clock network usage in UniPHY-based memory interfaces for RLDRAM II, QDR II, and QDR II+ protocols.

**Table 22. Clock Network Usage in UniPHY-based Memory Interfaces—RLDRAM II, and QDR II and QDR II+ SRAM**

Device	RLDRAM II			QDR II/QDR II+ SRAM		
	Half-Rate		Full-Rate	Half-Rate		Full-Rate
	Number of full-rate clock	Number of half-rate clock	Number of full-rate clock	Number of full-rate clock	Number of half-rate clock	Number of full-rate clock
Arria II GX	—	—	—	2 global	2 global	4 global
Stratix III	2 regional	1 global 1 regional	1 global 2 regional	1 global 1 regional	2 regional	1 global 2 regional
Arria II GZ and Stratix IV	2 regional	1 global 1 regional	1 global 2 regional	1 global 1 regional	2 regional	1 global 2 regional

**Note:** For more information about the clocks used in UniPHY-based memory standards, refer to the *Functional Description—UniPHY* chapter in volume 3 of the External Memory Interface Handbook.

#### Related Information

Functional Description—UniPHY

#### 1.4.6.6. PLL Usage for DDR, DDR2, and DDR3 SDRAM Without Leveling Interfaces

The following table lists PLL usage for DDR, DDR2, and DDR3 protocols without leveling interfaces.

**Table 23. PLL Usage for DDR, DDR2, and DDR3 SDRAM Without Leveling Interfaces**

Clock	Arria II GX Devices	Stratix III and Stratix IV Devices
C0	<ul style="list-style-type: none"><li>phy_clk_1x in half-rate designs</li><li>aux_half_rate_clk</li><li>PLL scan_clk</li></ul>	<ul style="list-style-type: none"><li>phy_clk_1x in half-rate designs</li><li>aux_half_rate_clk</li><li>PLL scan_clk</li></ul>
C1	<ul style="list-style-type: none"><li>phy_clk_1x in full-rate designs</li><li>aux_full_rate_clk</li><li>mem_clk_2x to generate DQS and CK/CK# signals</li><li>ac_clk_2x</li><li>cs_n_clk_2x</li></ul>	<ul style="list-style-type: none"><li>mem_clk_2x</li></ul>
C2	<ul style="list-style-type: none"><li>Unused</li></ul>	<ul style="list-style-type: none"><li>phy_clk_1x in full-rate designs</li><li>aux_full_rate_clk</li></ul>
C3	<ul style="list-style-type: none"><li>write_clk_2x (for DQ)</li><li>ac_clk_2x</li><li>cs_n_clk_2x</li></ul>	<ul style="list-style-type: none"><li>write_clk_2x</li></ul>

*continued...*



Clock	Arria II GX Devices	Stratix III and Stratix IV Devices
C4	<ul style="list-style-type: none"> <li>resync_clk_2x</li> </ul>	<ul style="list-style-type: none"> <li>resync_clk_2x</li> </ul>
C5	<ul style="list-style-type: none"> <li>measure_clk_2x</li> </ul>	<ul style="list-style-type: none"> <li>measure_clk_1x</li> </ul>
C6	—	<ul style="list-style-type: none"> <li>ac_clk_1x</li> </ul>

#### 1.4.6.7. PLL Usage for DDR3 SDRAM With Leveling Interfaces

The following table lists PLL usage for DDR3 protocols with leveling interfaces.

**Table 24. PLL Usage for DDR3 SDRAM With Leveling Interfaces**

Clock	Stratix III and Stratix IV Devices
C0	<ul style="list-style-type: none"> <li>phy_clk_1x in half-rate designs</li> <li>aux_half_rate_clk</li> <li>PLL scan_clk</li> </ul>
C1	<ul style="list-style-type: none"> <li>mem_clk_2x</li> </ul>
C2	<ul style="list-style-type: none"> <li>aux_full_rate_clk</li> </ul>
C3	<ul style="list-style-type: none"> <li>write_clk_2x</li> </ul>
C4	<ul style="list-style-type: none"> <li>resync_clk_2x</li> </ul>
C5	<ul style="list-style-type: none"> <li>measure_clk_1x</li> </ul>
C6	<ul style="list-style-type: none"> <li>ac_clk_1x</li> </ul>

## 1.5. Using PLL Guidelines

When using PLL for external memory interfaces, you must consider the following guidelines:

- For the clock source, use the clock input pin specifically dedicated to the PLL that you want to use with your external memory interface. The input and output pins are only fully compensated when you use the dedicated PLL clock input pin. If the clock source for the PLL is not a dedicated clock input pin for the dedicated PLL, you would need an additional clock network to connect the clock source to the PLL block. Using additional clock network may increase clock jitter and degrade the timing margin.
- Pick a PLL and PLL input clock pin that are located on the same side of the device as the memory interface pins.
- Share the DLL and PLL static clocks for multiple memory interfaces provided the controllers are on the same or adjacent side of the device and run at the same memory clock frequency.
- If your design uses a dedicated PLL to only generate a DLL input reference clock, you must set the PLL mode to **No Compensation** in the Quartus Prime software to minimize the jitter, or the software forces this setting automatically. The PLL does not generate other output, so it does not need to compensate for any clock path.



- If your design cascades PLL, the source (upstream) PLL must have a low-bandwidth setting, while the destination (downstream) PLL must have a high-bandwidth setting to minimize jitter. Intel does not recommend using cascaded PLLs for external memory interfaces because your design gets accumulated jitters. The memory output clock may violate the memory device jitter specification.
- Use cascading PLLs at your own risk. For more information, refer to "PLL Cascading".
- If you are using Arria II GX devices, for a single memory instance that spans two right-side quadrants, use a middle-side PLL as the source for that interface.
- If you are using Arria II GZ, Arria V GZ, Stratix III, Stratix IV, or Stratix V devices, for a single memory instance that spans two top or bottom quadrants, use a middle top or bottom PLL as the source for that interface. The ten dual regional clocks that the single interface requires must not block the design using the adjacent PLL (if available) for a second interface.

#### Related Information

[PLL Cascading](#) on page 76

## 1.6. PLL Cascading

Arria II GZ PLLs, Stratix III PLLs, Stratix IV PLLs, Stratix V and Arria V GZ fractional PLLs (fPLLs), and the two middle PLLs in Arria II GX EP2AGX95, EP2AGX125, EP2AGX190, and EP2AGX260 devices can be cascaded using either the global or regional clock trees, or the cascade path between two adjacent PLLs.

*Note:*

Use cascading PLLs at your own risk. You should use faster memory devices to maximize timing margins.

The UniPHY IP supports PLL cascading using the cascade path without any additional timing derating when the bandwidth and compensation rules are followed. The timing constraints and analysis assume that there is no additional jitter due to PLL cascading when the upstream PLL uses no compensation and low bandwidth, and the downstream PLL uses no compensation and high bandwidth.

The UniPHY IP does not support PLL cascading using the global and regional clock networks. You can implement PLL cascading at your own risk without any additional guidance and specifications from Intel. The Quartus Prime software does issue a critical warning suggesting use of the cascade path to minimize jitter, but does not explicitly state that Intel does not support cascading using global and regional clock networks.

Some Arria II GX devices (EP2AGX95, EP2AGX125, EP2AGX190, and EP2AGX260) have direct cascade path for two middle right PLLs. Arria II GX PLLs have the same bandwidth options as Stratix IV GX left and right PLLs.

The Arria 10 External Memory Interface IP does not support PLL cascading.

## 1.7. DLL

The Intel FPGA memory interface IP uses one DLL. The DLL is located at the corner of the device and can send the control signals to shift the DQS pins on its adjacent sides for Stratix-series devices, or DQS pins in any I/O banks in Arria II GX devices.



For example, the top-left DLL can shift DQS pins on the top side and left side of the device. The DLL generates the same phase shift resolution for both sides, but can generate different phase offset to the two different sides, if needed. Each DQS pin can be configured to use or ignore the phase offset generated by the DLL.

The DLL cannot generate two different phase offsets to the same side of the device. However, you can use two different DLLs to for this functionality.

DLL reference clocks must come from either dedicated clock input pins located on either side of the DLL or from specific PLL output clocks. Any clock running at the memory frequency is valid for the DLLs.

To minimize the number of clocks routed directly on the PCB, typically this reference clock is sourced from the memory controllers PLL. In general, DLLs can use the PLLs directly adjacent to them (corner PLLs when available) or the closest PLL located in the two sides adjacent to its location.

**Note:** By default, the DLL reference clock in Intel FPGA external memory IP is from a PLL output.

When designing for 780-pin packages with EP3SE80, EP3SE110, EP3SL150, EP4SE230, EP4SE360, EP4SGX180, and EP4SGX230 devices, the PLL to DLL reference clock connection is limited. DLL2 is isolated from a direct PLL connection and can only receive a reference clock externally from pins CLK[11:4]p in EP3SE80, EP3SE110, EP3SL150, EP4SE230, and EP4SE360 devices. In EP4SGX180 and EP4SGX230 devices, DLL2 and DLL3 are not directly connected to PLL. DLL2 and DLL3 receive a reference clock externally from pins CLK[7:4]p and CLK[15:12]p respectively.

For more DLL information, refer to the respective device handbooks.

The DLL reference clock should be the same frequency as the memory interface, but the phase is not important.

The required DQS capture phase is optimally chosen based on operating frequency and external memory interface type (DDR, DDR2, DDR3 SDRAM, and QDR II SRAM, or RLDRAM II). As each DLL supports two possible phase offsets, two different memory interface types operating at the same frequency can easily share a single DLL. More may be possible, depending on the phase shift required.

Intel FPGA memory IP always specifies a default optimal phase setting, to override this setting, refer to *Implementing and Parameterizing Memory IP*.

When sharing DLLs, your memory interfaces must be of the same frequency. If the required phase shift is different amongst the multiple memory interfaces, you can use a different delay chain in the DQS logic block or use the DLL phase offset feature.

To simplify the interface to IP connections, multiple memory interfaces operating at the same frequency usually share the same system and static clocks as each other where possible. This sharing minimizes the number of dedicated clock nets required and reduces the number of different clock domains found within the same design.

As each DLL can directly drive four banks, but each PLL only has complete C (output) counter coverage of two banks (using dual regional networks), situations can occur where a second PLL operating at the same frequency is required. As cascaded PLLs increase jitter and reduce timing margin, you are advised to first ascertain if an alternative second DLL and PLL combination is not available and more optimal.



Select a DLL that is available for the side of the device where the memory interface resides. If you select a PLL or a PLL input clock reference pin that can also serve as the DLL input reference clock, you do not need an extra input pin for the DLL input reference clock.

#### Related Information

[Implementing and Parameterizing Memory IP](#) on page 184

## 1.8. Other FPGA Resources

The Intel FPGA memory interface IP uses FPGA fabric, including registers and the Memory Block to implement the memory interface.

For resource utilization examples to ensure that you can fit your other modules in the device, refer to the “Resource Utilization” section in the *Introduction to UniPHY IP* chapter of the External Memory Interface Handbook.

One OCT calibration block is used if you are using the FPGA OCT feature in the memory interface. The OCT calibration block uses two pins (RUP and RDN), or single pin (RZQ) (“OCT Support for Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone V, Stratix III, Stratix IV, and Stratix V Devices”). You can select any of the available OCT calibration block as you do not need to place this block in the same bank or device side of your memory interface. The only requirement is that the I/O bank where you place the OCT calibration block uses the same VCCIO voltage as the memory interface. You can share multiple memory interfaces with the same OCT calibration block if the VCCIO voltage is the same.

#### Related Information

[OCT Support](#) on page 37

## 1.9. Document Revision History

Date	Version	Changes
May 2017	2017.05.08	<ul style="list-style-type: none"><li>Added <i>Guidelines for Stratix 10 External Memory Interface IP</i>, <i>General Pin-Out Guidelines for Stratix 10 EMIF IP</i>, and <i>Resource Sharing Guidelines for Stratix 10 EMIF IP</i> sections.</li><li>Rebranded as Intel.</li></ul>
October 2016	2016.10.31	<ul style="list-style-type: none"><li>Removed paragraph from <i>Address and Command</i> description in several pin utilization tables.</li></ul>
May 2016	2016.05.02	<ul style="list-style-type: none"><li>Modified <i>Data Strobe</i> and <i>Address</i> data in <i>UDIMM, RDIMM, and LRDIMM Pin Options for DDR4</i> table in <i>DDR, DDR2, DDR3, and DDR4 SDRAM DIMM Options</i>. Added notes to table.</li></ul>

*continued...*



Date	Version	Changes
November 2015	2015.11.02	<ul style="list-style-type: none"> <li>Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i>.</li> <li>Modified <i>I/O Banks Selection</i>, <i>PLL Reference Clock</i> and <i>RZQ Pins Placement</i>, and <i>Ping Pong PHY Implementation</i> sections in <i>General Pin-Out Guidelines for Arria 10 EMIF IP</i>.</li> <li>Added <i>Additional Requirements for DDR3 and DDR4 Ping-Pong PHY Interfaces</i> in <i>General Pin-Out Guidelines for Arria 10 EMIF IP</i>.</li> <li>Removed references to OCT Blocks from <i>Resource Sharing Guidelines for Arria 10 EMIF IP</i> section.</li> <li>Added LPDDR3.</li> </ul>
May 2015	2015.05.04	<ul style="list-style-type: none"> <li>Removed the F672 package of the 10M25 device.</li> <li>Updated the additional guidelines for MAX 10 devices to improve clarity.</li> <li>Added related information link to the <i>MAX 10 FPGA Signal Integrity Design Guidelines</i> for the <i>Additional Guidelines for MAX 10 Devices</i> topic.</li> </ul>
December 2014	2014.12.15	<ul style="list-style-type: none"> <li><i>General Pin-Out Guidelines for Arria 10 EMIF IP</i> section: <ul style="list-style-type: none"> <li>Added note to step 10.</li> <li>Removed steps 13 and 14.</li> <li>Added a bullet point to <i>Address/Command Pins Location</i>.</li> <li>Added <i>Ping Pong PHY Implementation</i></li> <li>Added parenthetical comment to fifth bullet point in <i>I/O Banks Selection</i></li> <li>Added note following the procedure, advising that all pins in a DQS group should reside in the same I/O bank, for RLDRAM II and RLDRAM 3 interfaces.</li> </ul> </li> <li>Added <i>QDR IV SRAM Clock Signals</i>, <i>QDR IV SRAM Commands and Addresses</i>, <i>AP</i>, and <i>AINV Signals</i>, and <i>QDR IV SRAM Data</i>, <i>DINV</i>, and <i>QVLD Signals</i> topics.</li> <li>Added note to <i>Estimating Pin Requirements</i> section.</li> <li><i>DDR, DDR2, DDR3, and DDR4 SDRAM DIMM Options</i> section: <ul style="list-style-type: none"> <li>Added <i>UDIMM</i>, <i>RDIMM</i>, and <i>LRDIMM Pin Options</i> for <i>DDR4</i> table.</li> <li>Changed notes to <i>LRDIMM Pin Options for DDR, DDR2, and DDR3</i> table.</li> <li>Removed reference to Chip ID pin.</li> </ul> </li> </ul>

***continued...***



Date	Version	Changes
August 2014	2014.08.15	<ul style="list-style-type: none"><li>• Made several changes to <i>Pin Counts for Various Example Memory Interfaces</i> table:<ul style="list-style-type: none"><li>— Added DDR4 SDRAM and RLDRAM 3 CIO.</li><li>— Removed x72 rows from table entries for DDR, DDR2, and DDR3.</li><li>— Added Arria 10 to note 11.</li><li>— Added notes 12-18.</li></ul></li><li>• Added DDR4 to descriptions of:<ul style="list-style-type: none"><li>— Clock signals</li><li>— Command and address signals</li><li>— Data, data strobe, DM/DBI, and optional ECC signals</li><li>— SDRAM DIMM options</li></ul></li><li>• Added QDR II+ Xtreme to descriptions of:<ul style="list-style-type: none"><li>— SRAM clock signals</li><li>— SRAM command signals</li><li>— SRAM address signals</li><li>— SRAM data, BWS, and QVLD signals</li></ul></li><li>• Changed title of section <i>OCT Support for Arria II GX, Arria II GZ, Arria V, Arria V GZ, Cyclone V, Stratix III, Stratix IV, and Stratix V Devices</i> to <i>OCT Support</i>.</li><li>• Reorganized chapter to have separate sections for <i>Guidelines for Arria 10 External Memory Interface IP</i> and <i>Guidelines for UniPHY-based External Memory Interface IP</i>.</li><li>• Revised Arria 10-specific guidelines.</li></ul>
December 2013	2013.12.16	<ul style="list-style-type: none"><li>• Removed references to ALTMEMPHY and HardCopy.</li><li>• Removed references to Cyclone III and Cyclone IV devices.</li></ul>
November 2012	6.0	<ul style="list-style-type: none"><li>• Added Arria V GZ information.</li><li>• Added RLDRAM 3 information.</li><li>• Added LRDIMM information.</li></ul>
June 2012	5.0	<ul style="list-style-type: none"><li>• Added LPDDR2 information.</li><li>• Added Cyclone V information.</li><li>• Added Feedback icon.</li></ul>
November 2011	4.0	<ul style="list-style-type: none"><li>• Moved and reorganized <i>Planning Pin and Resource</i> section to Volume 2:Design Guidelines.</li><li>• Added <i>Additional Guidelines for Arria V GZ and Stratix V Devices</i> section.</li><li>• Added Arria V and Cyclone V information.</li></ul>
June 2011	3.0	<ul style="list-style-type: none"><li>• Moved <i>Select a Device</i> and <i>Memory IP Planning</i> chapters to Volume 1.</li><li>• Added information about interface pins.</li><li>• Added guidelines for using PLL.</li></ul>
December 2010	2.1	<ul style="list-style-type: none"><li>• Added a new section on controller efficiency.</li><li>• Added Arria II GX and Stratix V information.</li></ul>
July 2010	2.0	Updated information about UniPHY-based interfaces and Stratix V devices.
April 2010	1.0	Initial release.

## 2. DDR2, DDR3, and DDR4 SDRAM Board Design Guidelines

The following topics provide guidelines for improving the signal integrity of your system and for successfully implementing a DDR2, DDR3, or DDR4 SDRAM interface on your system.

The following areas are discussed:

- comparison of various types of termination schemes, and their effects on the signal quality on the receiver
- proper drive strength setting on the FPGA to optimize the signal integrity at the receiver
- effects of different loading types, such as components versus DIMM configuration, on signal quality

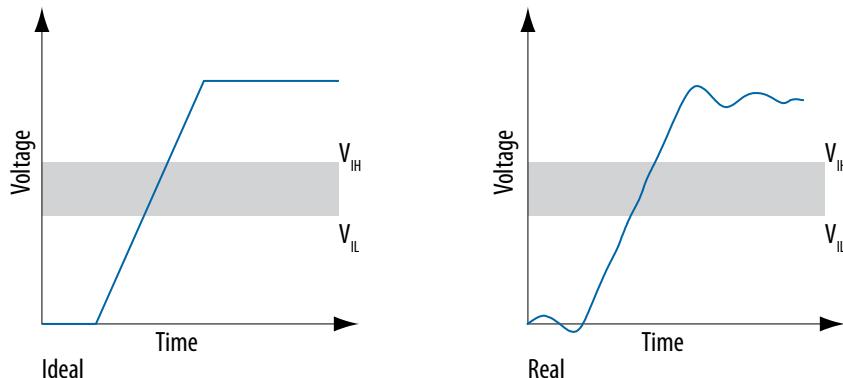
It is important to understand the trade-offs between different types of termination schemes, the effects of output drive strengths, and different loading types, so that you can swiftly navigate through the multiple combinations and choose the best possible settings for your designs.

The following key factors affect signal quality at the receiver:

- Leveling and dynamic ODT
- Proper use of termination
- Layout guidelines

As memory interface performance increases, board designers must pay closer attention to the quality of the signal seen at the receiver because poorly transmitted signals can dramatically reduce the overall data-valid margin at the receiver. The following figure shows the differences between an ideal and real signal seen by the receiver.

**Figure 12. Ideal and Real Signal at the Receiver**



Intel Corporation. All rights reserved. Intel, the Intel logo, Altera, Arria, Cyclone, Empirion, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.



## 2.1. Leveling and Dynamic Termination

DDR3 and DDR4 SDRAM DIMMs, as specified by JEDEC, always use a fly-by topology for the address, command, and clock signals.

Intel recommends that for full DDR3 or DDR4 SDRAM compatibility when using discrete DDR3 or DDR4 SDRAM components, you should mimic the JEDEC DDR3 or DDR4 fly-by topology on your custom printed circuit boards (PCB).

**Note:** Arria® II, Arria V GX, Arria V GT, Arria V SoC, Cyclone® V, and Cyclone V SoC devices do not support DDR3 SDRAM with read or write leveling, so these devices do not support standard DDR3 SDRAM DIMMs or DDR3 SDRAM components using the standard DDR3 SDRAM fly-by address, command, and clock layout topology.

**Table 25. Device Family Topology Support**

Device	I/O Support
Arria II	Non-leveling
Arria V GX, Arria V GT, Arria V SoC	Non-leveling
Arria V GZ	Leveling
Cyclone V GX, Cyclone V GT, Cyclone V SoC	Non-leveling
Stratix III	Leveling
Stratix IV	Leveling
Stratix V	Leveling
Arria 10	Leveling
Stratix 10	Leveling

### Related Information

[www.JEDEC.org](http://www.JEDEC.org)

## 2.1.1. Read and Write Leveling

A major difference between DDR2 and DDR3/DDR4 SDRAM is the use of leveling. To improve signal integrity and support higher frequency operations, the JEDEC committee defined a fly-by termination scheme used with clocks, and command and address bus signals.

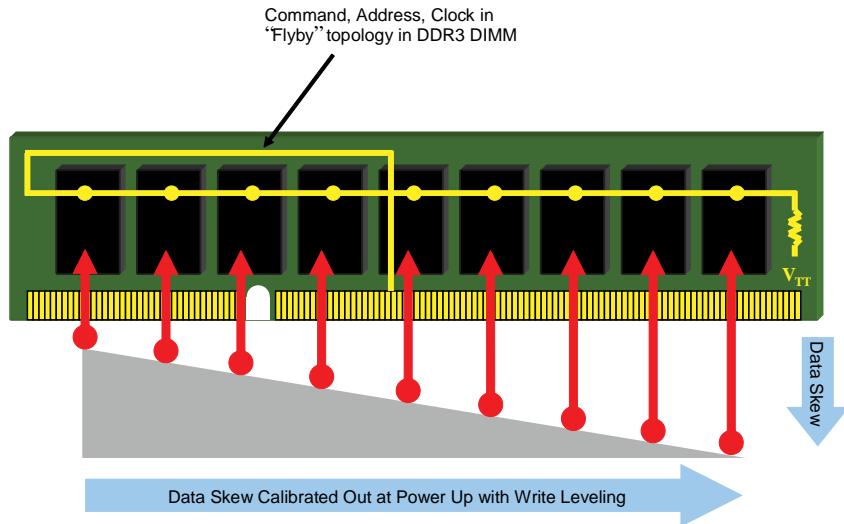
**Note:** This section describes read and write leveling in terms of a comparison between DDR3 and DDR2. Leveling in DDR4 is fundamentally similar to DDR3. Refer to the DDR4 JEDEC specifications for more information.

The following section describes leveling in DDR3, and is equally applicable to DDR4.

Fly-by topology reduces simultaneous switching noise (SSN) by deliberately causing flight-time skew between the data and strobes at every DRAM as the clock, address, and command signals traverse the DIMM, as shown in the following figure.



Figure 13. DDR3 DIMM Fly-By Topology Requiring Write Leveling



The flight-time skew caused by the fly-by topology led the JEDEC committee to introduce the write leveling feature on the DDR3 SDRAMs. Controllers must compensate for this skew by adjusting the timing per byte lane.

During a write, DQS groups launch at separate times to coincide with a clock arriving at components on the DIMM, and must meet the timing parameter between the memory clock and DQS defined as tDQSS of  $\pm 0.25$  tCK.

During the read operation, the memory controller must compensate for the delays introduced by the fly-by topology. The Stratix® III, Stratix IV, and Stratix V FPGAs have alignment and synchronization registers built in the I/O element to properly capture the data.

In DDR2 SDRAM, there are only two drive strength settings, full or reduced, which correspond to the output impedance of 18-ohm and 40-ohm, respectively. These output drive strength settings are static settings and are not calibrated; consequently, the output impedance varies as the voltage and temperature drifts.

The DDR3 SDRAM uses a programmable impedance output buffer. There are two drive strength settings, 34-ohm and 40-ohm. The 40-ohm drive strength setting is currently a reserved specification defined by JEDEC, but available on the DDR3 SDRAM, as offered by some memory vendors. Refer to the data sheet of the respective memory vendors for more information about the output impedance setting. You select the drive strength settings by programming the memory mode register defined by mode register 1 (MR1). To calibrate output driver impedance, an external precision resistor, RZQ, connects the ZQ pin and VSSQ. The value of this resistor must be 240-ohm  $\pm 1\%$ .

If you are using a DDR3 SDRAM DIMM, RZQ is soldered on the DIMM so you do not need to layout your board to account for it. Output impedance is set during initialization. To calibrate output driver impedance after power-up, the DDR3 SDRAM needs a calibration command that is part of the initialization and reset procedure and is updated periodically when the controller issues a calibration command.

In addition to calibrated output impedance, the DDR3 SDRAM also supports calibrated parallel ODT through the same external precision resistor, RZQ, which is possible by using a merged output driver structure in the DDR3 SDRAM, which also helps to improve pin capacitance in the DQ and DQS pins. The ODT values supported in DDR3 SDRAM are 20-ohm, 30-ohm, 40-ohm, 60-ohm, and 120-ohm, assuming that RZQ is 240-ohm.

#### **Related Information**

[www.JEDEC.org](http://www.JEDEC.org)

### **2.1.2. Dynamic ODT**

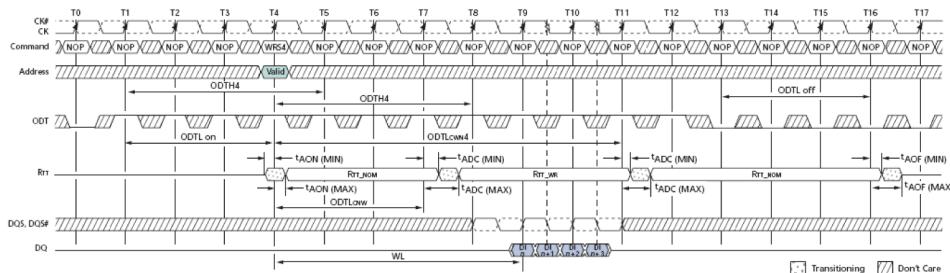
Dynamic ODT is a feature in DDR3 and DDR4 SDRAM that is not available in DDR2 SDRAM. Dynamic ODT can change the ODT setting without issuing a mode register set (MRS) command.

**Note:** This topic highlights the dynamic ODT feature in DDR3. To learn about dynamic ODT in DDR4, refer to the JEDEC DDR4 specifications.

When you enable dynamic ODT, and there is no write operation, the DDR3 SDRAM terminates to a termination setting of RTT\_NOM; when there is a write operation, the DDR3 SDRAM terminates to a setting of RTT\_WR. You can preset the values of RTT\_NOM and RTT\_WR by programming the mode registers, MR1 and MR2.

The following figure shows the behavior of ODT when you enable dynamic ODT.

**Figure 14. Dynamic ODT: Behavior with ODT Asserted Before and After the Write**



In the multi-load DDR3 SDRAM configuration, dynamic ODT helps reduce the jitter at the module being accessed, and minimizes reflections from any secondary modules.

For more information about using the dynamic ODT on DDR3 SDRAM, refer to the application note by Micron, *TN-41-04 DDR3 Dynamic On-Die Termination*.

In addition to RTT\_NOM and RTT\_WR, DDR4 has RTT\_PARK which applies a specified termination value when the ODT signal is low.

#### **Related Information**

[www.JEDEC.org](http://www.JEDEC.org)

### **2.1.3. Dynamic On-Chip Termination**

Dynamic OCT is available in Arria V, Arria 10, Cyclone V, Stratix III, Stratix IV, Stratix V, and Stratix 10.



The dynamic OCT scheme enables series termination (RS) and parallel termination (RT) to be dynamically turned on and off during the data transfer. The series and parallel terminations are turned on or off depending on the read and write cycle of the interface. During the write cycle, the RS is turned on and the RT is turned off to match the line impedance. During the read cycle, the RS is turned off and the RT is turned on as the FPGA implements the far-end termination of the bus.

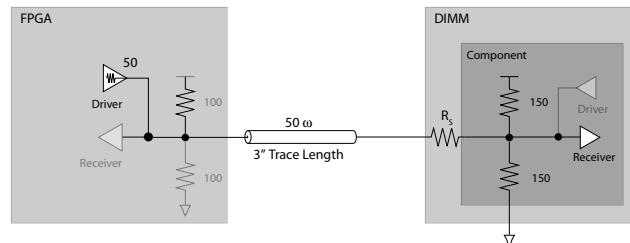
For more information about dynamic OCT, refer to the I/O features chapters in the devices handbook for your Intel device.

### 2.1.3.1. FPGA Writing to Memory

The benefit of using dynamic series OCT is that when driver is driving the transmission line, it "sees" a matched transmission line with no external resistor termination.

The following figure shows dynamic series OCT scheme when the FPGA is writing to the memory.

**Figure 15. Dynamic Series OCT Scheme with ODT on the Memory**



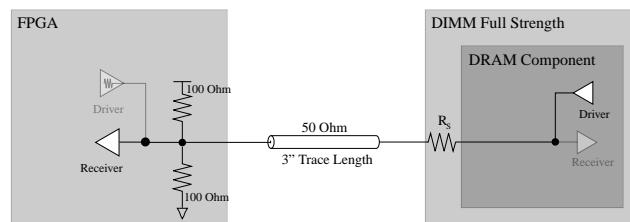
Refer to the memory vendors when determining the over- and undershoot. They typically specify a maximum limit on the input voltage to prevent reliability issues.

### 2.1.3.2. FPGA Reading from Memory

The following figure shows the dynamic parallel termination scheme when the FPGA is reading from memory.

When the SDRAM DIMM is driving the transmission line, the ringing and reflection is minimal because the FPGA-side termination 50-ohm pull-up resistor is matched with the transmission line.

**Figure 16. Dynamic Parallel OCT Scheme with Memory-Side Series Resistor**



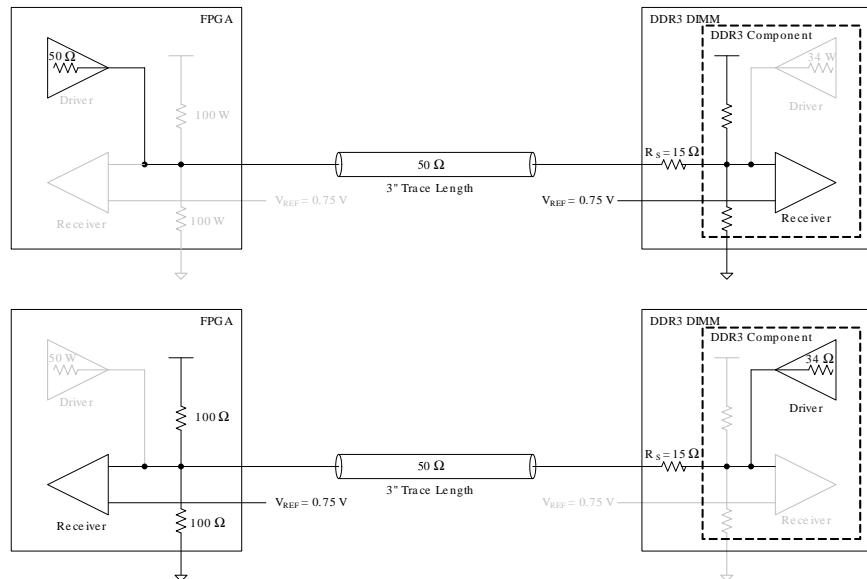
### 2.1.4. Dynamic On-Chip Termination in Stratix III and Stratix IV Devices

Stratix III and Stratix IV devices support on-off dynamic series and parallel termination for a bidirectional I/O in all I/O banks. Dynamic OCT is a new feature in Stratix III and Stratix IV FPGA devices.

You enable dynamic parallel termination only when the bidirectional I/O acts as a receiver and disable it when the bidirectional I/O acts as a driver. Similarly, you enable dynamic series termination only when the bidirectional I/O acts as a driver and is disable it when the bidirectional I/O acts as a receiver. The default setting for dynamic OCT is series termination, to save power when the interface is idle—no active reads or writes.

**Note:** The dynamic control operation of the OCT is separate to the output enable signal for the buffer. UniPHY IP can enable parallel OCT only during read cycles, saving power when the interface is idle.

**Figure 17. Dynamic OCT Between Stratix III and Stratix IV FPGA Devices**



Dynamic OCT is useful for terminating any high-performance bidirectional path because signal integrity is optimized depending on the direction of the data. In addition, dynamic OCT also eliminates the need for external termination resistors when used with memory devices that support ODT (such as DDR3 SDRAM), thus reducing cost and easing board layout.

However, dynamic OCT in Stratix III and Stratix IV FPGA devices is different from dynamic ODT in DDR3 SDRAM mentioned in previous sections and these features should not be assumed to be identical.

For detailed information about the dynamic OCT feature in the Stratix III FPGA, refer to the *Stratix III Device I/O Features* chapter in volume 1 of the *Stratix III Device Handbook*.

For detailed information about the dynamic OCT feature in the Stratix IV FPGA, refer to the *I/O Features in Stratix IV Devices* chapter in volume 1 of the *Stratix IV Device Handbook*.

### Related Information

- [Stratix III Device I/O Features](#)
- [I/O Features in Stratix IV Devices](#)



## 2.1.5. Dynamic OCT in Stratix V Devices

Stratix V devices also support the dynamic OCT feature and provide more flexibility. Stratix V OCT calibration uses one RZQ pin that exists in every OCT block.

You can use any one of the following as a reference resistor on the RZQ pin to implement different OCT values:

- 240-ohm reference resistor—to implement RS OCT of 34-ohm, 40-ohm, 48-ohm, 60-ohm, and 80-ohm; and RT OCT resistance of 20-ohm, 30-ohm, 40-ohm, and 120-ohm□
- 100-ohm reference resistor—to implement RS OCT of 25-ohm and 50-ohm; and RT OCT resistance of 50-ohm□

For detailed information about the dynamic OCT feature in the Stratix V FPGA, refer to the *I/O Features in Stratix V Devices* chapter in volume 1 of the *Stratix V Device Handbook*.

### Related Information

[I/O Features in Stratix V Devices](#)

## 2.1.6. Dynamic On-Chip Termination (OCT) in Arria 10 and Stratix 10 Devices

Depending upon the Rs (series) and Rt (parallel) OCT values that you want, you should choose appropriate values for the RZQ resistor and connect this resistor to the RZQ pin of the FPGA.

- Select a 240-ohm reference resistor to ground to implement Rs OCT values of 34-ohm, 40-ohm, 48-ohm, 60-ohm, and 80-ohm, and Rt OCT resistance values of 20-ohm, 30-ohm, 34-ohm, 40-ohm, 60-ohm, 80-ohm, 120-ohm and 240 ohm.
- Select a 100-ohm reference resistor to ground to implement Rs OCT values of 25-ohm and 50-ohm, and an RT OCT resistance of 50-ohm.

Check the FPGA I/O tab of the parameter editor to determine the I/O standards and termination values supported for data, address and command, and memory clock signals.

## 2.2. DDR2 Terminations and Guidelines

This section provides information for DDR2 SDRAM interfaces.

### 2.2.1. Termination for DDR2 SDRAM

DDR2 adheres to the JEDEC standard of governing Stub-Series Terminated Logic (SSTL), JESD8-15a, which includes four different termination schemes.

Two commonly used termination schemes of SSTL are:

- Single parallel terminated output load with or without series resistors (Class I, as stated in JESD8-15a)
- Double parallel terminated output load with or without series resistors (Class II, as stated in JESD8-15a)

Depending on the type of signals you choose, you can use either termination scheme. Also, depending on your design's FPGA and SDRAM memory devices, you may choose external or internal termination schemes.

To reduce system cost and simplify printed circuit board layout, you may choose not to have any parallel termination on the transmission line, and use point-to-point connections between the memory interface and the memory. In this case, you may take advantage of internal termination schemes such as on-chip termination (OCT) on the FPGA side and on-die termination (ODT) on the SDRAM side when it is offered on your chosen device.

#### Related Information

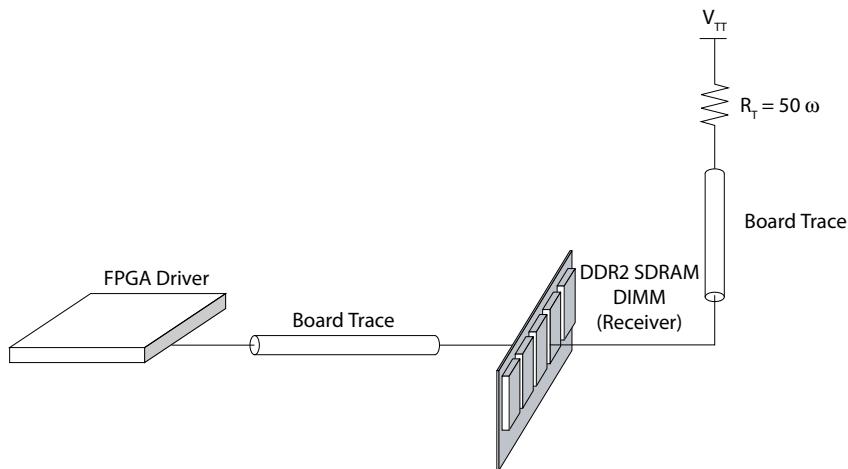
[DDR3 Terminations in Arria V, Cyclone V, Stratix III, Stratix IV, and Stratix V](#) on page 97

##### 2.2.1.1. External Parallel Termination

If you use external termination, you must study the locations of the termination resistors to determine which topology works best for your design.

The following two figures illustrate the most common termination topologies: fly-by topology and non-fly-by topology, respectively.

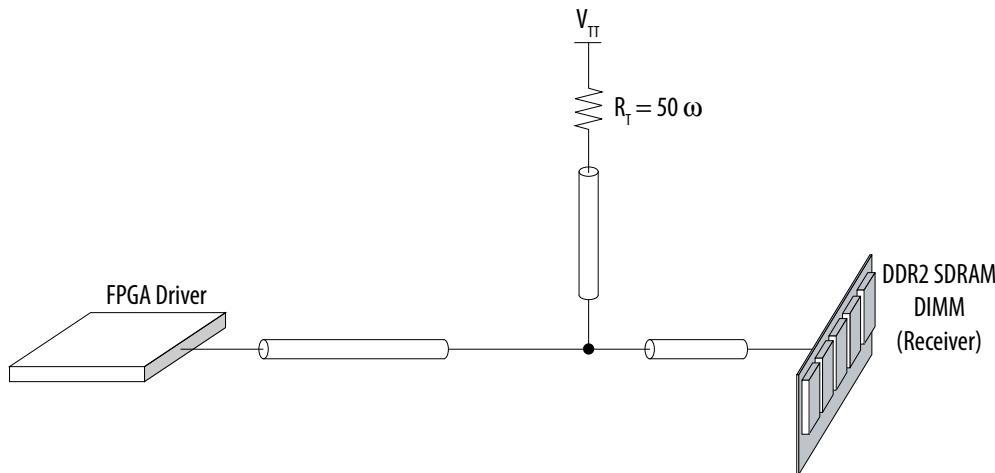
**Figure 18. Fly-By Placement of a Parallel Resistor**



With fly-by topology, you place the parallel termination resistor after the receiver. This termination placement resolves the undesirable unterminated stub found in the non-fly-by topology. However, using this topology can be costly and complicate routing.



**Figure 19. Non-Fly-By Placement of a Parallel Resistor**



With non-fly-by topology, the parallel termination resistor is placed between the driver and receiver (closest to the receiver). This termination placement is easier for board layout, but results in a short stub, which causes an unterminated transmission line between the terminating resistor and the receiver. The unterminated transmission line results in ringing and reflection at the receiver.

If you do not use external termination, DDR2 offers ODT and Intel FPGAs have varying levels of OCT support. You should explore using ODT and OCT to decrease the board power consumption and reduce the required board space.

### 2.2.1.2. On-Chip Termination

OCT technology is offered on Arria II GX, Arria II GZ, Arria V, Arria 10, Cyclone V, MAX 10, Stratix III, Stratix IV, and Stratix V devices.

The following table summarizes the extent of OCT support for devices earlier than Arria 10. This table provides information about SSTL-18 standards because SSTL-18 is the supported standard for DDR2 memory interface by Intel FPGAs.

For Arria II, Stratix III and Stratix IV devices, on-chip series (RS) termination is supported only on output and bidirectional buffers. The value of RS with calibration is calibrated against a 25-ohm resistor for class II and 50-ohm resistor for class I connected to RUP and RDN pins and adjusted to  $\pm 1\%$  of 25-ohm or 50-ohm. On-chip parallel (RT) termination is supported only on inputs and bidirectional buffers. The value of RT is calibrated against 100-ohm connected to the RUP and RDN pins. Calibration occurs at the end of device configuration. Dynamic OCT is supported only on bidirectional I/O buffers.

For Arria V, Cyclone V, and Stratix V devices, RS and RT values are calibrated against the on-board resistor RZQ. If you want 25 or 50 ohm values for your RS and RT, you must connect a 100 ohm resistor with a tolerance of  $+/-1\%$  to the RZQ pin.

For more information about on-chip termination, refer to the device handbook for the device that you are using.

**Table 26. On-Chip Termination Schemes**

Termination Scheme	SSTL-18	FPGA Device						
		Arria II GX	Arria II GZ	Arria V	Cyclone V	MAX 10	Stratix III and Stratix IV	Stratix V (1)
		Column and Row I/O	Column I/O					
On-Chip Series Termination without Calibration	Class I	50	50	50	50	50	50	50
	Class II	25	25	25	25	25	25	25
On-Chip Series Termination with Calibration	Class I	50	50	50	50	50	50	50
	Class II	25	25	25	25	25	25	25
On-Chip Parallel Termination with Calibration	Class I and Class II	—	50	50	50	—	50	50

Note to Table:

- Row I/O is not available for external memory interfaces in Stratix V devices.

### 2.2.1.3. Recommended Termination Schemes

The following table provides the recommended termination schemes for major DDR2 memory interface signals.

Signals include data (DQ), data strobe (DQS/DQSn), data mask (DM), clocks (mem\_clk/mem\_clk\_n), and address and command signals.

When interfacing with multiple DDR2 SDRAM components where the address, command, and memory clock pins are connected to more than one load, follow these steps:

1. Simulate the system to get the new slew-rate for these signals.
2. Use the derated tIS and tIH specifications from the DDR2 SDRAM data sheet based on the simulation results.
3. If timing deration causes your interface to fail timing requirements, consider signal duplication of these signals to lower their loading, and hence improve timing.

**Note:** Intel uses Class I and Class II termination in this table to refer to drive strength, and not physical termination.

**Note:** You must simulate your design for your system to ensure correct operation.

**Table 27. Termination Recommendations (1)**

Device Family	Signal Type	SSTL 18 IO Standard (2) (3) (4) (5) (6)	FPGA-End Discrete Termination	Memory-End Termination 1 (Rank/DIMM)	Memory I/O Standard
<b>Arria II GX</b>					
DDR2 component	DQ	Class I R50 CAL	50-ohm Parallel to VTT discrete	ODT75 (7)	HALF (8)
	DQS DIFF (13)	DIFF Class R50 CAL	50-ohm Parallel to VTT discrete	ODT75 (7)	HALF (8)
	DQS SE (12)	Class I R50 CAL	50-ohm Parallel to VTT discrete	ODT75 (7)	HALF (8)
	DM	Class I R50 CAL	N/A	ODT75 (7)	N/A
	Address and command	Class I MAX	N/A	56-ohm parallel to VTT discrete	N/A
	Clock	DIFF Class I R50 CAL	N/A	×1 = 100-ohm differential (10) ×2 = 200-ohm differential (11)	N/A
DDR2 DIMM	DQ	Class I R50 CAL	50-ohm Parallel to VTT discrete	ODT75 (7)	FULL (9)
	DQS DIFF (13)	DIFF Class I R50 CAL	50-ohm Parallel to VTT discrete	ODT75 (7)	FULL (9)
	DQS SE (12)	Class I R50 CAL	50-ohm Parallel to VTT discrete	ODT75 (7)	FULL (9)
	DM	Class I R50 CAL	N/A	ODT75 (7)	N/A
	Address and command	Class I MAX	N/A	56-ohm parallel to VTT discrete	N/A
	Clock	DIFF Class I R50 CAL	N/A	N/A = on DIMM	N/A
<b>Arria V and Cyclone V</b>					
DDR2 component	DQ	Class I R50/P50 DYN CAL	N/A	ODT75 (7)	HALF (8)
	DQS DIFF (13)	DIFF Class I R50/P50 DYN CAL	N/A	ODT75 (7)	HALF (8)
	DQS SE (12)	Class I R50/P50 DYN CAL	N/A	ODT75 (7)	HALF (8)
	DM	Class I R50 CAL	N/A	ODT75 (7)	N/A
	Address and command	Class I MAX	N/A	56-ohm parallel to VTT discrete	N/A
	Clock	DIFF Class I R50 NO CAL	N/A	×1 = 100-ohm differential (10) ×2 = 200-ohm differential (11)	N/A
DDR2 DIMM	DQ	Class I R50/P50 DYN CAL	N/A	ODT75 (7)	FULL (9)
	DQS DIFF (13)	DIFF Class I R50/P50 DYN CAL	N/A	ODT75 (7)	FULL (9)

*continued...*



Device Family	Signal Type	SSTL 18 IO Standard <sup>(2)(3)</sup> <sup>(4)(5)(6)</sup>	FPGA-End Discrete Termination	Memory-End Termination 1 (Rank/DIMM)	Memory I/O Standard
Arria II GZ, Stratix III, Stratix IV, and Stratix V	DQS SE <sup>(12)</sup>	Class I R50/P50 DYN CAL	N/A	ODT75 <sup>(7)</sup>	FULL <sup>(9)</sup>
	DM	Class I R50 CAL	N/A	ODT75 <sup>(7)</sup>	N/A
	Address and command	Class I MAX	N/A	56-ohm parallel to VTT discrete	N/A
	Clock	DIFF Class I R50 NO CAL	N/A	N/A = on DIMM	N/A
<b>Arria II GZ, Stratix III, Stratix IV, and Stratix V</b>					
DDR2 component	DQ	Class I R50/P50 DYN CAL	N/A	ODT75 <sup>(7)</sup>	HALF <sup>(8)</sup>
	DQS DIFF <sup>(13)</sup>	DIFF Class I R50/P50 DYN CAL	N/A	ODT75 <sup>(7)</sup>	HALF <sup>(8)</sup>
	DQS SE <sup>(12)</sup>	DIFF Class I R50/P50 DYN CAL	N/A	ODT75 <sup>(7)</sup>	HALF <sup>(8)</sup>
	DM	Class I R50 CAL	N/A	ODT75 <sup>(7)</sup>	N/A
	Address and command	Class I MAX	N/A	56-ohm Parallel to VTT discrete	N/A
	Clock	DIFF Class I R50 NO CAL	N/A	x1 = 100-ohm differential <sup>(10)</sup> x2 = 200-ohm differential <sup>(11)</sup>	N/A
DDR2 DIMM	DQ	Class I R50/P50 DYN CAL	N/A	ODT75 <sup>(7)</sup>	FULL <sup>(9)</sup>
	DQS DIFF <sup>(13)</sup>	DIFF Class I R50/P50 DYN CAL	N/A	ODT75 <sup>(7)</sup>	FULL <sup>(9)</sup>
	DQS SE <sup>(12)</sup>	Class I R50/P50 DYN CAL	N/A	ODT75 <sup>(7)</sup>	FULL <sup>(9)</sup>
	DM	Class I R50 CAL	N/A	ODT75 <sup>(7)</sup>	N/A
	Address and command	Class I MAX	N/A	56-ohm Parallel to VTT discrete	N/A
	Clock	DIFF Class I R50 NO CAL	N/A	N/A = on DIMM	N/A
<b>MAX 10</b>					
DDR2 component	DQ/DQS	Class I 12 mA	50-ohm Parallel to VTT discrete	ODT75 <sup>(7)</sup>	HALF <sup>(8)</sup>
	DM	Class I 12 mA	N/A	80-ohm Parallel to VTT discrete	N/A
	Address and command	Class I MAX	N/A		N/A
	Clock	Class I 12 mA	N/A	x1 = 100-ohm differential <sup>(10)</sup> x2 = 200-ohm differential <sup>(11)</sup>	N/A
Notes to Table:					
1. N/A is not available. 2. R is series resistor. 3. P is parallel resistor.					



Device Family	Signal Type	SSTL 18 IO Standard <sup>(2) (3) (4) (5) (6)</sup>	FPGA-End Discrete Termination	Memory-End Termination 1 (Rank/DIMM)	Memory I/O Standard
4. DYN is dynamic OCT. 5. NO CAL is OCT without calibration. 6. CAL is OCT with calibration. 7. ODT75 vs. ODT50 on the memory has the effect of opening the eye more, with a limited increase in overshoot/undershoot. 8. HALF is reduced drive strength. 9. FULL is full drive strength. 10.x1 is a single-device load. 11.x2 is two-device load. For example, you can feed two out of nine devices on a single rank DIMM with a single clock pair —except for MAX 10, which doesn't support DIMMs. 12.DQS SE is single-ended DQS. 13.DQS DIFF is differential DQS					

## 2.2.2. DDR2 Design Layout Guidelines

The general layout guidelines in the following topic apply to DDR2 SDRAM interfaces.

These guidelines will help you plan your board layout, but are not meant as strict rules that must be adhered to. Intel recommends that you perform your own board-level simulations to ensure that the layout you choose for your board allows you to achieve your desired performance.

For more information about how the memory manufacturers route these address and control signals on their DIMMs, refer to the Cadence PCB browser from the Cadence website, at [www.cadence.com](http://www.cadence.com). The various JEDEC example DIMM layouts are available from the JEDEC website, at [www.jedec.org](http://www.jedec.org).

For more information about board skew parameters, refer to Board Skews in the Implementing and Parameterizing Memory IP chapter. For assistance in calculating board skew parameters, refer to the board skew calculator tool, which is available at the Intel website.

**Note:**

1. The following layout guidelines include several +/- length based rules. These length based guidelines are for first order timing approximations if you cannot simulate the actual delay characteristic of the interface. They do not include any margin for crosstalk.
2. To ensure reliable timing closure to and from the periphery of the device, signals to and from the periphery should be registered before any further logic is connected.

Intel recommends that you get accurate time base skew numbers for your design when you simulate the specific implementation.

### Related Information

<http://www.jedec.org/download/DesignFiles/DDR2/default1.cfm>

## 2.2.3. General Layout Guidelines

The following table lists general board design layout guidelines. These guidelines are Intel recommendations, and should not be considered as hard requirements. You should perform signal integrity simulation on all the traces to verify the signal integrity



of the interface. You should extract the slew rate and propagation delay information, enter it into the IP and compile the design to ensure that timing requirements are met.

**Table 28. General Layout Guidelines**

Parameter	Guidelines
Impedance	<ul style="list-style-type: none"><li>All unused via pads must be removed, because they cause unwanted capacitance.</li><li>Trace impedance plays an important role in the signal integrity. You must perform board level simulation to determine the best characteristic impedance for your PCB. For example, it is possible that for multi rank systems 40 ohms could yield better results than a traditional 50 ohm characteristic impedance.</li></ul>
Decoupling Parameter	<ul style="list-style-type: none"><li>Use 0.1 uF in 0402 size to minimize inductance</li><li>Make VTT voltage decoupling close to termination resistors</li><li>Connect decoupling caps between VTT and ground</li><li>Use a 0.1 uF cap for every other VTT pin and 0.01 uF cap for every VDD and VDDQ pin</li><li>Verify the capacitive decoupling using the Intel Power Distribution Network Design Tool</li></ul>
Power	<ul style="list-style-type: none"><li>Route GND and V<sub>CC</sub> as planes</li><li>Route VCCIO for memories in a single split plane with at least a 20-mil (0.020 inches, or 0.508 mm) gap of separation</li><li>Route VTT as islands or 250-mil (6.35-mm) power traces</li><li>Route oscillators and PLL power as islands or 100-mil (2.54-mm) power traces</li></ul>
General Routing	All specified delay matching requirements include PCB trace delays, different layer propagation velocity variance, and crosstalk. To minimize PCB layer propagation variance, Intel recommends that signals from the same net group always be routed on the same layer. <ul style="list-style-type: none"><li>Use 45° angles (<i>not</i> 90° corners)</li><li>Avoid T-Junctions for critical nets or clocks</li><li>Avoid T-junctions greater than 250 mils (6.35 mm)</li><li>Disallow signals across split planes</li><li>Restrict routing other signals close to system reset signals</li><li>Avoid routing memory signals closer than 0.025 inch (0.635 mm) to PCI or system clocks</li></ul>

#### Related Information

[Power Distribution Network Design Tool](#)

#### 2.2.4. Layout Guidelines for DDR2 SDRAM Interface

Unless otherwise specified, the following guidelines apply to the following topologies:

- DIMM—UDIMM topology
- DIMM—RDIMM topology
- Discrete components laid out in UDIMM topology
- Discrete components laid out in RDIMM topology

Trace lengths for CLK and DQS should tightly match for each memory component. To match the trace lengths on the board, a balanced tree topology is recommended for clock and address and command signal routing. In addition to matching the trace lengths, you should ensure that DDR timing is passing in the Report DDR Timing report. For Stratix devices, this timing is shown as Write Leveling tDQSS timing. For Arria and Cyclone devices, this timing is shown as CK vs DQS timing



For a table of device family topology support, refer to *Leveling and Dynamic ODT*.

The following table lists DDR2 SDRAM layout guidelines. These guidelines are Intel recommendations, and should not be considered as hard requirements. You should perform signal integrity simulation on all the traces to verify the signal integrity of the interface. You should extract the slew rate and propagation delay information, enter it into the IP and compile the design to ensure that timing requirements are met.

**Note:** The following layout guidelines also apply to DDR3 SDRAM without leveling interfaces.

**Table 29. DDR2 SDRAM Layout Guidelines (1)**

Parameter	Guidelines
DIMMs	If you consider a normal DDR2 unbuffered, unregistered DIMM, essentially you are planning to perform the DIMM routing directly on your PCB. Therefore, each address and control pin routes from the FPGA (single pin) to all memory devices must be on the same side of the FPGA.
General Routing	<ul style="list-style-type: none"> <li>All data, address, and command signals must have matched length traces <math>\pm 50</math> ps.</li> <li>All signals within a given <b>Byte Lane Group</b> should be matched length with maximum deviation of <math>\pm 10</math> ps and routed in the same layer.</li> </ul>
Clock Routing	<ul style="list-style-type: none"> <li>A 4.7 K-ohm resistor to ground is recommended for each Clock Enable signal. You can place the resistor at either the memory end or the FPGA end of the trace.</li> <li>Route clocks on inner layers with outer-layer run lengths held to under 500 mils (12.7 mm)</li> <li>These signals should maintain a 10-mil (0.254 mm) spacing from other nets</li> <li>Clocks should maintain a length-matching between clock pairs of <math>\pm 5</math> ps.</li> <li>Differential clocks should maintain a length-matching between P and N signals of <math>\pm 2</math> ps, routed in parallel.</li> <li>Space between different pairs should be at least three times the space between the differential pairs and must be routed differentially (5-mil trace, 10-15 mil space on centers), and equal to the signals in the Address/Command Group or up to 100 mils (2.54 mm) longer than the signals in the Address/Command Group.</li> <li>Trace lengths for CLK and DQS should closely match for each memory component. To match trace lengths on the board, a balanced tree topology is recommended for clock and address and command signal routing. For Stratix device families, ensure that Write Leveling tDQSS is passing in the DDR timing report; for Arria and Cyclone device families, verify that CK vs DQS timing is passing in the DDR timing report.</li> </ul>
Address and Command Routing	<ul style="list-style-type: none"> <li>Unbuffered address and command lines are more susceptible to cross-talk and are generally noisier than buffered address or command lines. Therefore, un-buffered address and command signals should be routed on a different layer than data signals (DQ) and data mask signals (DM) and with greater spacing.</li> <li>Do not route differential clock (CK) and clock enable (CKE) signals close to address signals.</li> </ul>

*continued...*

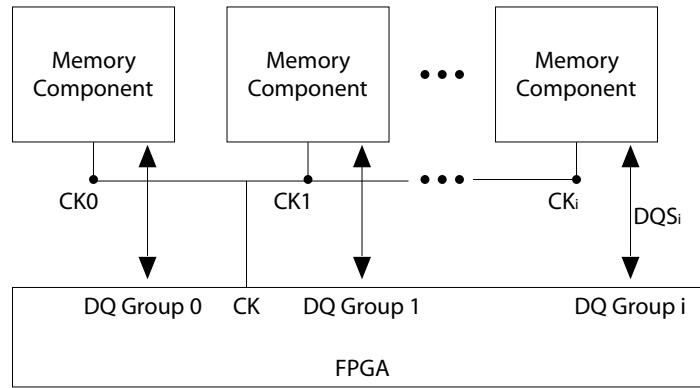


Parameter	Guidelines
DQ, DM, and DQS Routing Rules	<ul style="list-style-type: none"><li>• Keep the distance from the pin on the DDR2 DIMM or component to the termination resistor pack (VTT) to less than 500 mils for DQS[<math>x</math>] Data Groups.</li><li>• Keep the distance from the pin on the DDR2 DIMM or component to the termination resistor pack (VTT) to less than 1000 mils for the ADR_CMD_CTL Address Group.</li><li>• Parallelism rules for the DQS[<math>x</math>] Data Groups are as follows:<ul style="list-style-type: none"><li>— 4 mils for parallel runs &lt; 0.1 inch (approximately 1<math>\times</math> spacing relative to plane distance)</li><li>— 5 mils for parallel runs &lt; 0.5 inch (approximately 1<math>\times</math> spacing relative to plane distance)</li><li>— 10 mils for parallel runs between 0.5 and 1.0 inches (approximately 2<math>\times</math> spacing relative to plane distance)</li><li>— 15 mils for parallel runs between 1.0 and 6.0 inch (approximately 3<math>\times</math> spacing relative to plane distance)</li></ul></li><li>• Parallelism rules for the ADR_CMD_CTL group and CLOCKS group are as follows:<ul style="list-style-type: none"><li>— 4 mils for parallel runs &lt; 0.1 inch (approximately 1<math>\times</math> spacing relative to plane distance)</li><li>— 10 mils for parallel runs &lt; 0.5 inch (approximately 2<math>\times</math> spacing relative to plane distance)</li><li>— 15 mils for parallel runs between 0.5 and 1.0 inches (approximately 3<math>\times</math> spacing relative to plane distance)</li><li>— 20 mils for parallel runs between 1.0 and 6.0 inches (approximately 4<math>\times</math> spacing relative to plane distance)</li></ul></li><li>• All signals are to maintain a 20-mil separation from other, non-related nets.</li><li>• All signals must have a total length of &lt; 6 inches.</li><li>• Trace lengths for CLK and DQS should closely match for each memory component. To match trace lengths on the board, a balanced tree topology is recommended for clock and address and command signal routing. For Stratix device families, ensure that Write Leveling tDQSS is passing in the DDR timing report; for Arria and Cyclone device families, verify that CK vs DQS timing is passing in the DDR timing report.</li></ul>
Termination Rules	<ul style="list-style-type: none"><li>• When pull-up resistors are used, fly-by termination configuration is recommended. Fly-by helps reduce stub reflection issues.</li><li>• Pull-ups should be within 0.5 to no more than 1 inch.</li><li>• Pull up is typically 56-ohms.</li><li>• If using resistor networks:<ul style="list-style-type: none"><li>• Do not share R-pack series resistors between address/command and data lines (DQ, DQS, and DM) to eliminate crosstalk within pack.</li><li>• Series and pull up tolerances are 1–2%.</li><li>• Series resistors are typically 10 to 20-ohm.</li><li>• Address and control series resistor typically at the FPGA end of the link.</li><li>• DM, DQS, DQ series resistor typically at the memory end of the link (or just before the first DIMM).</li></ul></li></ul>

*continued...*



Parameter	Guidelines
	<ul style="list-style-type: none"> <li>If termination resistor packs are used:</li> <li>The distance to your memory device should be less than 750 mils.</li> <li>The distance from your FPGA device should be less than 1250 mils.</li> </ul>
Quartus Prime Software Settings for Board Layout	<ul style="list-style-type: none"> <li>To perform timing analyses on board and I/O buffers, use third party simulation tool to simulate all timing information such as skew, ISI, crosstalk, and type the simulation result into the UniPHY board setting panel.</li> <li>Do not use advanced I/O timing model (AIOT) or board trace model unless you do not have access to any third party tool. AIOT provides reasonable accuracy but tools like HyperLynx provides better result. In operations with higher frequency, it is crucial to properly simulate all signal integrity related uncertainties.</li> <li>The Quartus Prime software does timing check to find how fast the controller issues a write command after a read command, which limits the maximum length of the DQ/DQS trace. Check the turnaround timing in the Report DDR timing report and ensure the margin is positive before board fabrication. Functional failure happens if the margin is less than 0.</li> </ul>
Note to Table:	
1. For point-to-point and DIMM interface designs, refer to the Micron website, <a href="http://www.micron.com">www.micron.com</a> .	

**Figure 20. Balanced Tree Topology**

$CK_i$  = Clock signal propagation delay to device  $i$

$DQS_i$  = DQ/DQS signals propagation delay to group  $i$

#### Related Information

- External Memory Interface Spec Estimator
- [www.micron.com](http://www.micron.com)
- Leveling and Dynamic Termination on page 82

### 2.3. DDR3 Terminations in Arria V, Cyclone V, Stratix III, Stratix IV, and Stratix V

DDR3 DIMMs have terminations on all unidirectional signals, such as memory clocks, and addresses and commands; thus eliminating the need for them on the FPGA PCB. In addition, using the ODT feature on the DDR3 SDRAM and the dynamic OCT feature

of Stratix III, Stratix IV, and Stratix V FPGAs completely eliminates any external termination resistors; thus simplifying the layout for the DDR3 SDRAM interface when compared to that of the DDR2 SDRAM interface.

The following topics describe the correct way to terminate a DDR3 SDRAM interface together with Stratix III, Stratix IV, and Stratix V FPGA devices.

**Note:** If you are using a DDR3 SDRAM without leveling interface, refer to "Board Termination for DDR2 SDRAM". Note also that Arria V and Cyclone V devices do not support DDR3 with leveling.

### Related Information

[Termination for DDR2 SDRAM](#) on page 87

## 2.3.1. Terminations for Single-Rank DDR3 SDRAM Unbuffered DIMM

The most common implementation of the DDR3 SDRAM interface is the unbuffered DIMM (UDIMM). You can find DDR3 SDRAM UDIMMs in many applications, especially in PC applications.

The following table lists the recommended termination and drive strength setting for UDIMM and Stratix III, Stratix IV, and Stratix V FPGA devices.

**Note:** These settings are just recommendations for you to get started. Simulate with real board and try different settings to get the best SI.

**Table 30. Drive Strength and ODT Setting Recommendations for Single-Rank UDIMM**

Signal Type	SSTL 15 I/O Standard <sup>(1)</sup>	FPGA End On-Board Termination <sup>(2)</sup>	Memory End Termination for Write	Memory Driver Strength for Read
DQ	Class I R50C/G50C <sup>(3)</sup>	—	60-ohm ODT <sup>(4)</sup>	40-ohm <sup>(4)</sup>
DQS	Differential Class I R50C/G50C <sup>(3)</sup>	—	60-ohm ODT <sup>(4)</sup>	40-ohm <sup>(4)</sup>
DM	Class I R50C <sup>(3)</sup>	—	60-ohm ODT <sup>(4)</sup>	40-ohm <sup>(4)</sup>
Address and Command	Class I with maximum drive strength	—	39-ohm on-board termination to $V_{DD}$ <sup>(5)</sup>	
CK/CK#	Differential Class I R50C	—	On-board <sup>(5)</sup> 2.2 pf compensation cap before the first component; 36-ohm termination to $V_{DD}$ for each arm (72-ohm differential); add 0.1 uF just before $V_{DD}$ .	

Notes to Table:

1. UniPHY IP automatically implements these settings.
2. Intel recommends that you use dynamic on-chip termination (OCT) for Stratix III and Stratix IV device families.
3. R50C is series with calibration for write, G50C is parallel 50 with calibration for read.
4. You can specify these settings in the parameter editor.
5. For DIMM, these settings are already implemented on the DIMM card; for component topology, Intel recommends that you mimic termination scheme on the DIMM card on your board.

You can implement a DDR3 SDRAM UDIMM interface in several permutations, such as single DIMM or multiple DIMMs, using either single-ranked or dual-ranked UDIMMs. In addition to the UDIMM's form factor, these termination recommendations are also valid for small-outline (SO) DIMMs and MicroDIMMs.



### 2.3.2. Terminations for Multi-Rank DDR3 SDRAM Unbuffered DIMM

You can implement a DDR3 SDRAM UDIMM interface in several permutations, such as single DIMM or multiple DIMMs, using either single-ranked or dual-ranked UDIMMs. In addition to the UDIMM's form factor, these termination recommendations are also valid for small-outline (SO) DIMMs and MicroDIMMs.

The following table lists the different permutations of a two-slot DDR3 SDRAM interface and the recommended ODT settings on both the memory and controller when writing to memory.

**Table 31. DDR3 SDRAM ODT Matrix for Writes (1) (2)**

Slot 1	Slot 2	Write To	Controller OCT (3)	Slot 1		Slot 2	
				Rank 1	Rank 2	Rank 1	Rank 2
DR	DR	Slot 1	Series 50-ohm	120-ohm (4)	ODT off	ODT off	40-ohm (4)
		Slot 2	Series 50-ohm	ODT off	40-ohm (4)	120-ohm (4)	ODT off
SR	SR	Slot 1	Series 50-ohm	120-ohm (4)	Unpopulated	40-ohm (4)	Unpopulated
		Slot 2	Series 50-ohm	40-ohm (4)	Unpopulated	120-ohm (4)	Unpopulated
DR	Empty	Slot 1	Series 50-ohm	120-ohm (4)	ODT off	Unpopulated	Unpopulated
Empty	DR	Slot 2	Series 50-ohm	Unpopulated	Unpopulated	120-ohm (4)	ODT off
SR	Empty	Slot 1	Series 50-ohm	120-ohm (4)	Unpopulated	Unpopulated	Unpopulated
Empty	SR	Slot 2	Series 50-ohm	Unpopulated	Unpopulated	120-ohm (4)	Unpopulated

Notes to Table:

1. SR: single-ranked DIMM; DR: dual-ranked DIMM.
2. These recommendations are taken from the DDR3 ODT and Dynamic ODT session of the JEDEC DDR3 2007 Conference, Oct 3-4, San Jose, CA.
3. The controller in this case is the FPGA.
4. Dynamic ODT is required. For example, the ODT of Slot 2 is set to the lower ODT value of 40-ohms when the memory controller is writing to Slot 1, resulting in termination and thus minimizing any reflection from Slot 2. Without dynamic ODT, Slot 2 will not be terminated.

The following table lists the different permutations of a two-slot DDR3 SDRAM interface and the recommended ODT settings on both the memory and controller when reading from memory.

**Table 32. DDR3 SDRAM ODT Matrix for Reads (1) (2)**

Slot 1	Slot 2	Read From	Controller OCT (3)	Slot 1		Slot 2	
				Rank 1	Rank 2	Rank 1	Rank 2
DR	DR	Slot 1	Parallel 50-ohm	ODT off	ODT off	ODT off	40-ohm (4)
		Slot 2	Parallel 50-ohm	ODT off	40-ohm (4)	ODT off	ODT off

*continued...*

<b>Slot 1</b>	<b>Slot 2</b>	<b>Read From</b>	<b>Controller OCT<sup>(3)</sup></b>	<b>Slot 1</b>		<b>Slot 2</b>	
				<b>Rank 1</b>	<b>Rank 2</b>	<b>Rank 1</b>	<b>Rank 2</b>
SR	SR	Slot 1	Parallel 50-ohm	ODT off	Unpopulated	40-ohm <sup>(4)</sup>	Unpopulated
		Slot 2	Parallel 50-ohm	40-ohm <sup>(4)</sup>	Unpopulated	ODT off	Unpopulated
DR	Empty	Slot 1	Parallel 50-ohm	ODT off	ODT off	Unpopulated	Unpopulated
Empty	DR	Slot 2	Parallel 50-ohm	Unpopulated	Unpopulated	ODT off	ODT off
SR	Empty	Slot 1	Parallel 50-ohm	ODT off	Unpopulated	Unpopulated	Unpopulated
Empty	SR	Slot 2	Parallel 50-ohm	Unpopulated	Unpopulated	ODT off	Unpopulated

Notes to Table:

1. SR: single-ranked DIMM; DR: dual-ranked DIMM.
2. These recommendations are taken from the DDR3 ODT and Dynamic ODT session of the JEDEC DDR3 2007 Conference, Oct 3-4, San Jose, CA.
3. The controller in this case is the FPGA. JEDEC typically recommends 60-ohms, but this value assumes that the typical motherboard trace impedance is 60-ohms and that the controller supports this termination. Intel recommends using a 50-ohm parallel OCT when reading from the memory.

### 2.3.3. Terminations for DDR3 SDRAM Registered DIMM

The difference between a registered DIMM (RDIMM) and a UDIMM is that the clock, address, and command pins of the RDIMM are registered or buffered on the DIMM before they are distributed to the memory devices. For a controller, each clock, address, or command signal has only one load, which is the register or buffer. In a UDIMM, each controller pin must drive a fly-by wire with multiple loads.

You do not need to terminate the clock, address, and command signals on your board because these signals are terminated at the register. However, because of the register, these signals become point-to-point signals and have improved signal integrity making the drive strength requirements of the FPGA driver pins more relaxed. Similar to the signals in a UDIMM, the DQS, DQ, and DM signals on a RDIMM are not registered. To terminate these signals, refer to "DQS, DQ, and DM for DDR3 SDRAM UDIMM".

### 2.3.4. Terminations for DDR3 SDRAM Load-Reduced DIMM

RDIMM and LRDIMM differ in that DQ, DQS, and DM signals are registered or buffered in the LRDIMM. The LRDIMM buffer IC is a superset of the RDIMM buffer IC. The buffer IC isolates the memory interface signals from loading effects of the memory chip. Reduced electrical loading allows a system to operate at higher frequency and higher density.

**Note:** If you want to use your DIMM socket for UDIMM and RDIMM/LRDIMM, you must create the necessary redundant connections on the board from the FPGA to the DIMM socket. For example, the number of chip select signals required for a single-rank UDIMM is one, but for single-rank RDIMM the number of chip selects required is two. RDIMM and LRDIMM have parity signals associated with the address and command bus which UDIMM does not have. Consult the DIMM manufacturer's data sheet for detailed information about the necessary pin connections for various DIMM topologies.



### 2.3.5. Terminations for DDR3 SDRAM Components With Leveling

The following topics discusses terminations used to achieve optimum performance for designing the DDR3 SDRAM interface using discrete DDR3 SDRAM components.

In addition to using DDR3 SDRAM DIMM to implement your DDR3 SDRAM interface, you can also use DDR3 SDRAM components. However, for applications that have limited board real estate, using DDR3 SDRAM components reduces the need for a DIMM connector and places components closer, resulting in denser layouts.

#### 2.3.5.1. DDR3 SDRAM Components With or Without Leveling

The DDR3 SDRAM UDIMM is laid out to the JEDEC specification. The JEDEC specification is available from either the JEDEC Organization website ([www.JEDEC.org](http://www.JEDEC.org)) or from the memory vendors. However, when you are designing the DDR3 SDRAM interface using discrete SDRAM components, you may desire a layout scheme that is different than the DIMM specification.

You have the following options:

- Mimic the standard DDR3 SDRAM DIMM, using a fly-by topology for the memory clocks, address, and command signals. This option needs read and write leveling, so you must use the UniPHY IP with leveling.
- Mimic a standard DDR2 SDRAM DIMM, using a balanced (symmetrical) tree-type topology for the memory clocks, address, and command signals. Using this topology results in unwanted stubs on the command, address, and clock, which degrades signal integrity and limits the performance of the DDR3 SDRAM interface.

#### Related Information

- Layout Guidelines for DDR3 and DDR4 SDRAM Interfaces on page 113
- [www.JEDEC.org](http://www.JEDEC.org)

### 2.4. DDR3 and DDR4 on Arria 10 and Stratix 10 Devices

The following topics describe considerations specific to DDR3 and DDR4 external memory interface protocols on Arria 10 and Stratix 10 devices.

#### Related Information

[www.JEDEC.org](http://www.JEDEC.org)

#### 2.4.1. Dynamic On-Chip Termination (OCT) in Arria 10 and Stratix 10 Devices

Depending upon the Rs (series) and Rt (parallel) OCT values that you want, you should choose appropriate values for the RZQ resistor and connect this resistor to the RZQ pin of the FPGA.

- Select a 240-ohm reference resistor to ground to implement Rs OCT values of 34-ohm, 40-ohm, 48-ohm, 60-ohm, and 80-ohm, and Rt OCT resistance values of 20-ohm, 30-ohm, 34-ohm, 40-ohm, 60-ohm, 80-ohm, 120-ohm and 240 ohm.
- Select a 100-ohm reference resistor to ground to implement Rs OCT values of 25-ohm and 50-ohm, and an RT OCT resistance of 50-ohm.



Check the FPGA I/O tab of the parameter editor to determine the I/O standards and termination values supported for data, address and command, and memory clock signals.

### 2.4.2. Dynamic On-Die Termination (ODT) in DDR4

In DDR4, in addition to the Rtt\_nom and Rtt\_wr values, which are applied during read and write respectively, a third option called Rtt\_park is available. When Rtt\_park is enabled, a selected termination value is set in the DRAM when ODT is driven low.

Rtt\_nom and Rtt\_wr work the same as in DDR3, which is described in *Dynamic ODT for DDR3*.

Refer to the DDR4 JEDEC specification or your memory vendor data sheet for details about available termination values and functional description for dynamic ODT in DDR4 devices.

For DDR4 LRDIMM, if SPD byte 152 calls for different values of Rtt\_Park to be used for package ranks 0 and 1 versus package ranks 2 and 3, set the value to the larger of the two impedance settings.

### 2.4.3. Choosing Terminations on Arria 10 Devices

To determine optimal on-chip termination (OCT) and on-die termination (ODT) values for best signal integrity, you should simulate your memory interface in HyperLynx or a similar tool.

If the optimal OCT and ODT termination values as determined by simulation are not available in the list of available values in the parameter editor, select the closest available termination values for OCT and ODT.

Refer to *Dynamic On-Chip Termination (OCT) in Arria 10 Devices* for examples of various OCT modes. Refer to the *Arria 10 Device Handbook* for more information about OCT. For information on available ODT choices, refer to your memory vendor data sheet.

#### Related Information

[Dynamic On-Chip Termination \(OCT\) in Arria 10 and Stratix 10 Devices](#) on page 87

### 2.4.4. On-Chip Termination Recommendations for DDR3 and DDR4 on Arria 10 Devices

- Output mode (drive strength) for Address/Command/Clock and Data Signals: Depending upon the I/O standard that you have selected, you would have a range of selections expressed in terms of ohms or milamps. A value of 34 to 40 ohms or 12 mA is a good starting point for output mode drive strength.
- Input mode (parallel termination) for Data and Data Strobe signals: A value of 40 or 60 ohms is a good starting point for FPGA side input termination.

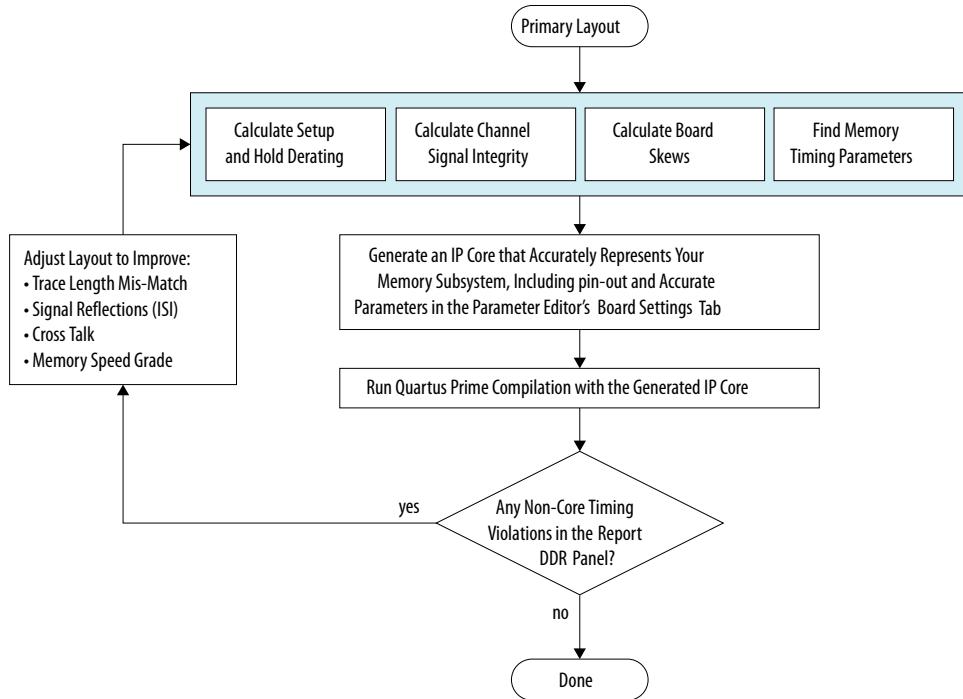


## 2.5. Layout Approach

For all practical purposes, you can regard the TimeQuest timing analyzer's report on your memory interface as definitive for a given set of memory and board timing parameters.

You will find timing under **Report DDR** in TimeQuest and on the **Timing Analysis** tab in the parameter editor.

The following flowchart illustrates the recommended process to follow during the board design phase, to determine timing margin and make iterative improvements to your design.



### Board Skew

For information on calculating board skew parameters, refer to *Implementing and Parameterizing Memory IP*, in the *External Memory Interface Handbook*.

The Board Skew Parameter Tool is an interactive tool that can help you calculate board skew parameters if you know the absolute delay values for all the memory related traces.

### Memory Timing Parameters

For information on the memory timing parameters to be entered into the parameter editor, refer to the datasheet for your external memory device.

### Related Information

[Board Skew Parameter Tool](#)

## 2.6. Channel Signal Integrity Measurement

As external memory interface data rates increase, so does the importance of proper channel signal integrity measurement. By measuring the actual channel loss during the layout process and including that data in your parameterization, a realistic assessment of margins is achieved.

### 2.6.1. Importance of Accurate Channel Signal Integrity Information

Default values for channel loss (or eye reduction) can be used when calculating timing margins, however those default values may not accurately reflect the channel loss in your system. If the channel loss in your system is different than the default values, the calculated timing margins will vary accordingly.

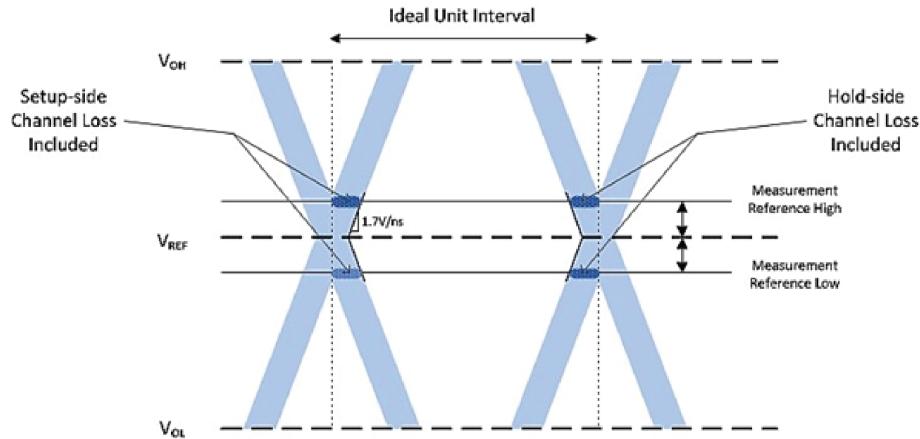
If your actual channel loss is greater than the default channel loss, and if you rely on default values, the available timing margins for the entire system will be lower than the values calculated during compilation. By relying on default values that do not accurately reflect your system, you may be led to believe that you have good timing margin, while in reality, your design may require changes to achieve good channel signal integrity.

### 2.6.2. Understanding Channel Signal Integrity Measurement

To measure channel signal integrity you need to measure the channel loss for various signals. For a particular signal or signal trace, channel loss is defined as loss of the eye width at  $+/- V_{IH}$ (ac and dc)  $+/- V_{IL}$ (ac and dc).  $V_{IH}/V_{IL}$  above or below  $V_{REF}$  is used to align with various requirements of the timing model for memory interfaces.

The example below shows a reference eye diagram where the channel loss on the setup- or leading-side of the eye is equal to the channel loss on the hold- or lagging-side of the eye; however, it does not necessarily have to be that way. Because Intel's calibrating PHY will calibrate to the center of the read and write eye, the Board Settings tab has parameters for the total extra channel loss for Write DQ and Read DQ. For address and command signals which are not-calibrated, the Board Settings tab allows you to enter setup- and hold-side channel losses that are not equal, allowing the Quartus Prime software to place the clock statically within the center of the address and command eye.

**Figure 21. Equal Setup and Hold-side Losses**



### 2.6.3. How to Enter Calculated Channel Signal Integrity Values

You should enter calculated channel loss values in the **Channel Signal Integrity** section of the **Board** (or **Board Timing**) tab of the parameter editor.

#### Arria V, Cyclone V, and Stratix V

For 28nm families, fixed values are assigned to different signals within the timing analysis algorithms of the Quartus Prime software. The following table shows the values for different signal groups:

Signal Group	Assumed Channel Loss
Address/Command (output)	250 ps
Write (output)	350 ps
Read Capture (input)	225 ps

If your calculated values are higher than the assumed channel loss, you must enter the positive difference; if your calculated values are lower than the assumed channel loss, you must enter the negative difference. For example, if the measured channel loss for reads for your system is 250 ps then you should enter 25 ps as the read channel loss.

#### Arria 10 and Stratix 10

For Arria 10 and Stratix 10 EMIF IP, the default channel loss displayed in the parameter editor is based on the selected configuration (different values for single rank versus dual rank), and on internal Intel reference boards. You should replace the default value with the value that you calculate.

## 2.6.4. Guidelines for Calculating DDR3 Channel Signal Integrity

### Address and Command ISI and Crosstalk

Simulate the address/command and control signals and capture eye at the DRAM pins, using the memory clock as the trigger for the memory interface's address/command and control signals. Measure the setup and hold channel losses at the voltage thresholds mentioned in the memory vendor's data sheet.

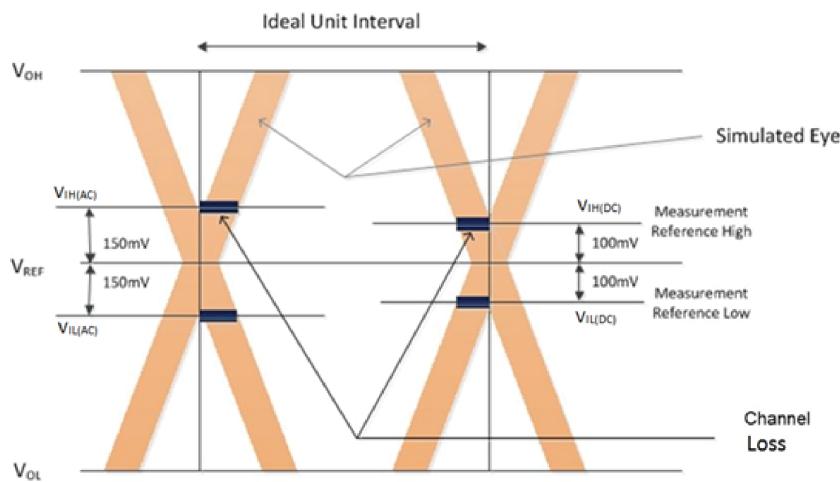
Address and command channel loss = Measured loss on the setup side + measured loss on the hold side.

$$V_{REF} = V_{DD}/2 = 0.75 \text{ mV for DDR3}$$

You should select the  $V_{IH}$  and  $V_{IL}$  voltage levels appropriately for the DDR3L memory device that you are using. Check with your memory vendor for the correct voltage levels, as the levels may vary for different speed grades of device.

The following figure illustrates a DDR3 example where  $V_{IH(AC)}/V_{IL(AC)}$  is  $\pm 150 \text{ mV}$  and  $V_{IH(DC)}/V_{IL(DC)}$  is  $\pm 100 \text{ mV}$ .

**Figure 22.**



### Write DQ ISI and Crosstalk

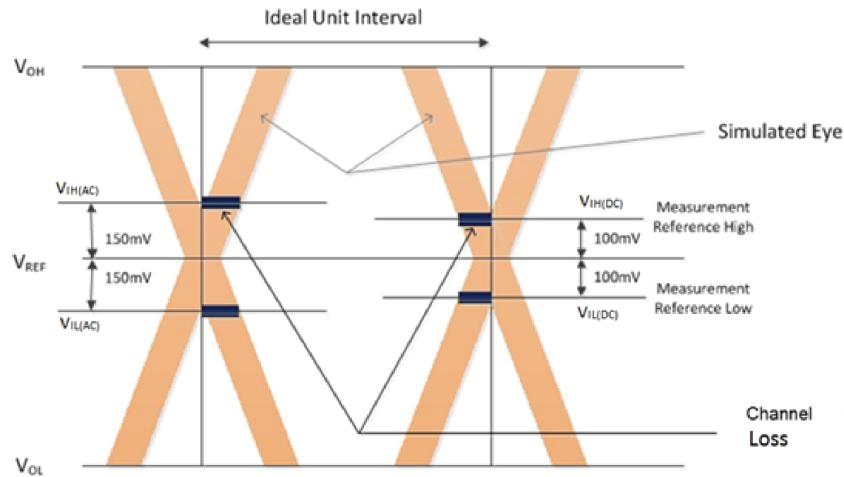
Simulate the write DQ signals and capture eye at the DRAM pins, using DQ Strobe (DQS) as a trigger for the DQ signals of the memory interface simulation. Measure the setup and hold channel losses at the  $V_{IH}$  and  $V_{IL}$  mentioned in the memory vendor's data sheet. The following figure illustrates a DDR3 example where  $V_{IH(AC)}/V_{IL(AC)}$  is  $\pm 150 \text{ mV}$  and  $V_{IH(DC)}/V_{IL(DC)}$  is  $\pm 100 \text{ mV}$ .

Write Channel Loss = Measured Loss on the Setup side + Measured Loss on the Hold side

$$V_{REF} = V_{DD}/2 = 0.75 \text{ mV for DDR3}$$



**Figure 23.**



#### Read DQ ISI and Crosstalk

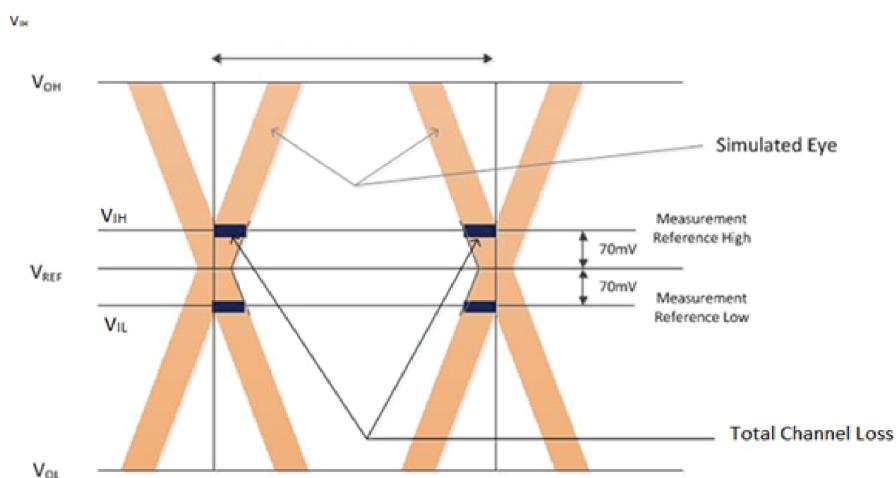
Simulate read DQ signals and capture eye at the FPGA die. Do not measure at the pin, because you might see unwanted reflections that could create a false representation of the eye opening at the input buffer of the FPGA. Use DQ Strobe (DQS) as a trigger for the DQ signals of your memory interface simulation. Measure the eye opening at  $\pm 70$  mV ( $V_{IH}/V_{IL}$ ) with respect to  $V_{REF}$ .

Read Channel Loss =  $(UI) - (\text{Eye opening at } \pm 70 \text{ mV with respect to } V_{REF})$

UI = Unit interval. For example, if you are running your interface at 800 Mhz, the effective data is 1600 Mbps, giving a unit interval of  $1/1600 = 625 \text{ ps}$

$V_{REF} = VDD/2 = 0.75 \text{ mV for DDR3}$

**Figure 24.**

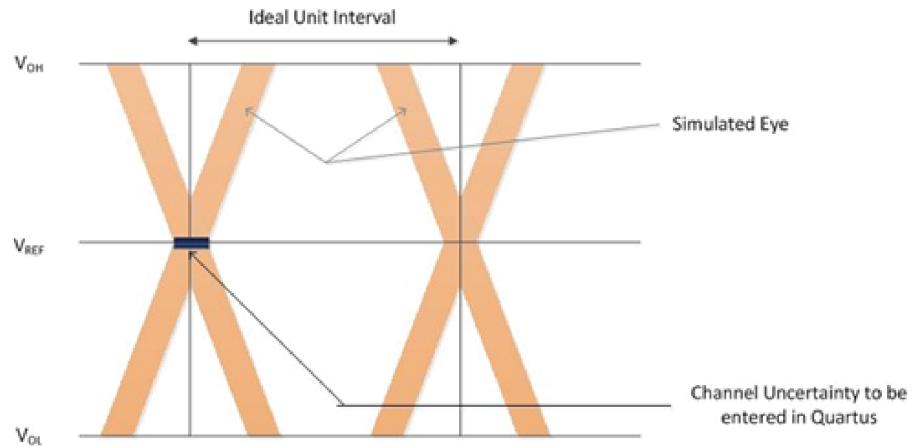


### Write/Read DQS ISI and Crosstalk

Simulate the Write/Read DQS and capture eye, and measure the uncertainty at  $V_{REF}$ .

$V_{REF} = V_{DD}/2 = 0.75 \text{ mV}$  for DDR3

**Figure 25.**



### 2.6.5. Guidelines for Calculating DDR4 Channel Signal Integrity

#### Address and Command ISI and Crosstalk

Simulate the address/command and control signals and capture eye at the DRAM pins, using the memory clock as the trigger for the memory interface's address/command and control signals. Measure the setup and hold channel losses at the voltage thresholds mentioned in the memory vendor's data sheet.

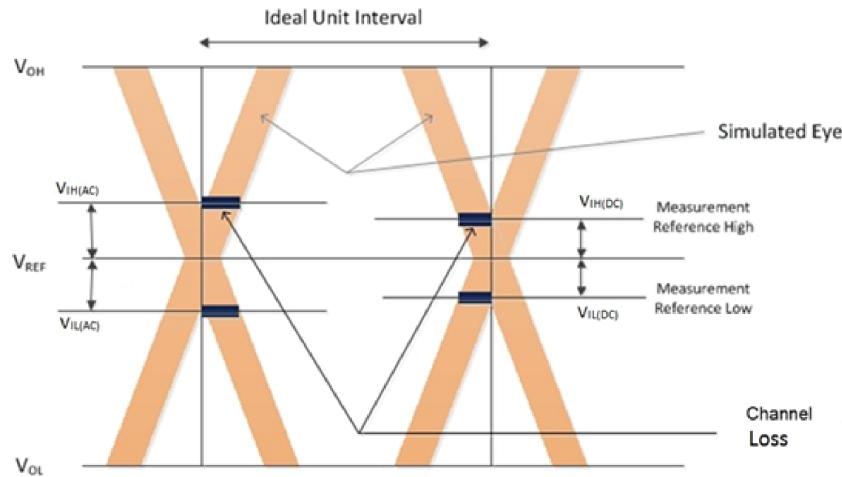
Address and command channel loss = Measured loss on the setup side + measured loss on the hold side.

$V_{REF} = V_{DD}/2 = 0.75 \text{ mV}$  for address/command for DDR4.

You should select the  $V_{IH}$  and  $V_{IL}$  voltage levels appropriately for the DDR4 memory device that you are using. Check with your memory vendor for the correct voltage levels, as the levels may vary for different speed grades of device.

The following figure illustrates a DDR4-1200 example, where  $V_{IH(AC)}/V_{IL(AC)}$  is  $\pm 100 \text{ mV}$  and  $V_{IH(DC)}/V_{IL(DC)}$  is  $\pm 75 \text{ mV}$ .

Select the  $V_{IH(AC)}$ ,  $V_{IL(AC)}$ ,  $V_{IH(DC)}$ , and  $V_{IL(DC)}$  for the speed grade of DDR4 memory device from the memory vendor's data sheet.

**Figure 26.**

### Write DQ ISI and Crosstalk

Simulate the write DQ signals and capture eye at the DRAM pins, using DQ Strobe (DQS) as a trigger for the DQ signals of the memory interface simulation. Measure the setup and hold channel losses at the  $V_{IH}$  and  $V_{IL}$  mentioned in the memory vendor's data sheet

Write Channel Loss = Measured Loss on the Setup side + Measured Loss on the Hold side.

or

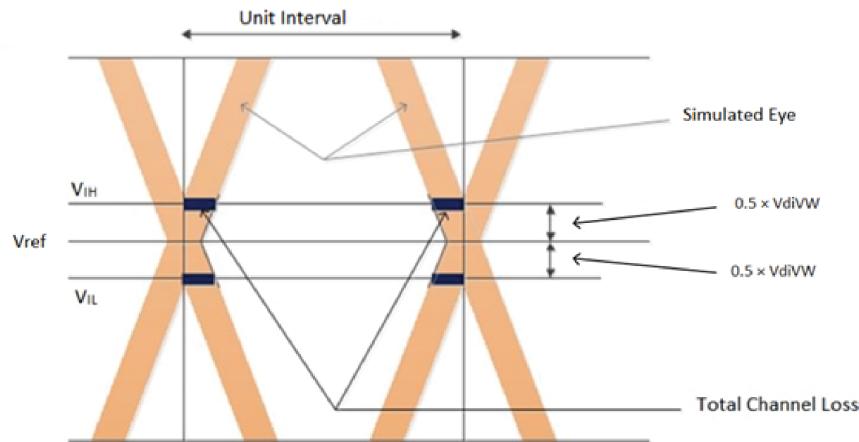
Write Channel Loss = UI - (Eye opening at  $V_{IH}$  or  $V_{IL}$ ).

$V_{REF}$  = Voltage level where the eye opening is highest.

$V_{IH} = V_{REF} + (0.5 \times V_{diVW})$ .

$V_{IL} = V_{REF} - (0.5 \times V_{diVW})$ .

Where  $V_{diVW}$  varies by frequency of operation; you can find the  $V_{diVW}$  value in your memory vendor's data sheet.

**Figure 27.**


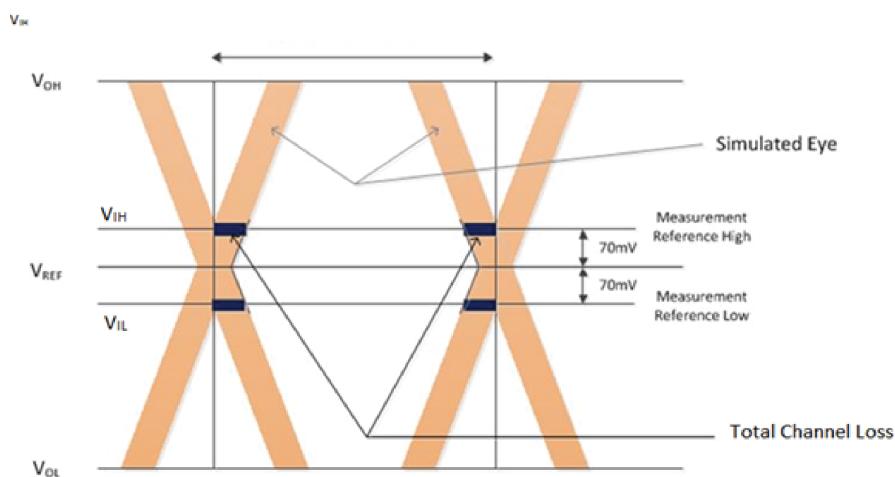
### Read DQ ISI and Crosstalk

Simulate read DQ signals and capture eye at the FPGA die. Do not measure at the pin, because you might see unwanted reflections that could create a false representation of the eye opening at the input buffer of the FPGA. Use DQ Strobe (DQS) as a trigger for the DQ signals of your memory interface simulation. Measure the eye opening at +/- 70 mV ( $V_{IH}/V_{IL}$ ) with respect to  $V_{REF}$ .

Read Channel Loss = (UI) - (Eye opening at +/- 70 mV with respect to  $V_{REF}$ .)

UI = Unit interval. For example, if you are running your interface at 800 Mhz, the effective data is 1600 Mbps, giving a unit interval of  $1/1600 = 625$  ps.

$V_{REF}$  = Voltage level where the eye opening is highest.

**Figure 28.**


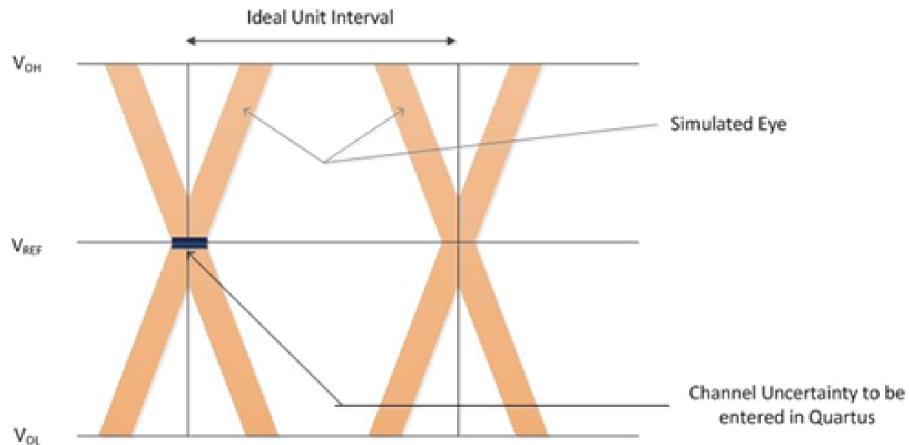


### Write/Read DQS ISI and Crosstalk

Simulate write and read DQS and capture eye. Measure the uncertainty at  $V_{REF}$ .

$V_{REF}$  = Voltage level where the eye opening is the highest.

**Figure 29.**



## 2.7. Design Layout Guidelines

The general layout guidelines in the following topic apply to DDR3 and DDR4 SDRAM interfaces.

These guidelines will help you plan your board layout, but are not meant as strict rules that must be adhered to. Intel recommends that you perform your own board-level simulations to ensure that the layout you choose for your board allows you to achieve your desired performance.

For more information about how the memory manufacturers route these address and control signals on their DIMMs, refer to the Cadence PCB browser from the Cadence website, at [www.cadence.com](http://www.cadence.com). The various JEDEC example DIMM layouts are available from the JEDEC website, at [www.jedec.org](http://www.jedec.org).

For more information about board skew parameters, refer to Board Skews in the Implementing and Parameterizing Memory IP chapter. For assistance in calculating board skew parameters, refer to the board skew calculator tool, which is available at the Intel website.

*Note:*

1. The following layout guidelines include several  $+/-$  length based rules. These length based guidelines are for first order timing approximations if you cannot simulate the actual delay characteristic of the interface. They do not include any margin for crosstalk.
2. To ensure reliable timing closure to and from the periphery of the device, signals to and from the periphery should be registered before any further logic is connected.



Intel recommends that you get accurate time base skew numbers for your design when you simulate the specific implementation.

### Related Information

- [www.JEDEC.org](http://www.JEDEC.org)
- [www.cadence.com](http://www.cadence.com)
- [www.mentor.com](http://www.mentor.com)
- Board Skew Parameters Tool
- <http://www.jedec.org/download/DesignFiles/DDR2/default1.cfm>

## 2.7.1. General Layout Guidelines

The following table lists general board design layout guidelines. These guidelines are Intel recommendations, and should not be considered as hard requirements. You should perform signal integrity simulation on all the traces to verify the signal integrity of the interface. You should extract the slew rate and propagation delay information, enter it into the IP and compile the design to ensure that timing requirements are met.

**Table 33. General Layout Guidelines**

Parameter	Guidelines
Impedance	<ul style="list-style-type: none"><li>• All unused via pads must be removed, because they cause unwanted capacitance.</li><li>• Trace impedance plays an important role in the signal integrity. You must perform board level simulation to determine the best characteristic impedance for your PCB. For example, it is possible that for multi rank systems 40 ohms could yield better results than a traditional 50 ohm characteristic impedance.</li></ul>
Decoupling Parameter	<ul style="list-style-type: none"><li>• Use 0.1 uF in 0402 size to minimize inductance</li><li>• Make VTT voltage decoupling close to termination resistors</li><li>• Connect decoupling caps between VTT and ground</li><li>• Use a 0.1 uF cap for every other VTT pin and 0.01 uF cap for every VDD and VDDQ pin</li><li>• Verify the capacitive decoupling using the Intel Power Distribution Network Design Tool</li></ul>
Power	<ul style="list-style-type: none"><li>• Route GND and <math>V_{CC}</math> as planes</li><li>• Route VCCIO for memories in a single split plane with at least a 20-mil (0.020 inches, or 0.508 mm) gap of separation</li><li>• Route VTT as islands or 250-mil (6.35-mm) power traces</li><li>• Route oscillators and PLL power as islands or 100-mil (2.54-mm) power traces</li></ul>
General Routing	All specified delay matching requirements include PCB trace delays, different layer propagation velocity variance, and crosstalk. To minimize PCB layer propagation variance, Intel recommends that signals from the same net group always be routed on the same layer. <ul style="list-style-type: none"><li>• Use 45° angles (<i>not</i> 90° corners)</li><li>• Avoid T-Junctions for critical nets or clocks</li><li>• Avoid T-junctions greater than 250 mils (6.35 mm)</li><li>• Disallow signals across split planes</li><li>• Restrict routing other signals close to system reset signals</li><li>• Avoid routing memory signals closer than 0.025 inch (0.635 mm) to PCI or system clocks</li></ul>

### Related Information

[Power Distribution Network Design Tool](#)



## 2.7.2. Layout Guidelines for DDR3 and DDR4 SDRAM Interfaces

The following table lists DDR3 and DDR4 SDRAM layout guidelines.

Unless otherwise specified, the guidelines in the following table apply to the following topologies:

- DIMM—UDIMM topology
- DIMM—RDIMM topology
- DIMM—LRDIMM topology
- Not all versions of the Quartus Prime software support LRDIMM.
- Discrete components laid out in UDIMM topology
- Discrete components laid out in RDIMM topology

These guidelines are recommendations, and should not be considered as hard requirements. You should perform signal integrity simulation on all the traces to verify the signal integrity of the interface.

Unless stated otherwise, the following guidelines apply to all devices that support DDR3 or DDR4, including Arria 10 and Stratix 10.

For information on the simulation flow for 28nm products, refer to [http://www.alterawiki.com/wiki/Measuring\\_Channel\\_Signal\\_Integrity](http://www.alterawiki.com/wiki/Measuring_Channel_Signal_Integrity).

For information on the simulation flow for Arria 10 products, refer to [http://www.alterawiki.com/wiki/Arria\\_10\\_EMIF\\_Simulation\\_Guidance](http://www.alterawiki.com/wiki/Arria_10_EMIF_Simulation_Guidance).

<http://www.altera.com/technology/memory/estimator/mem-emif-index.html>

For supported frequencies and topologies, refer to the *External Memory Interface Spec Estimator* <http://www.altera.com/technology/memory/estimator/mem-emif-index.html>.

For frequencies greater than 800 MHz, when you are calculating the delay associated with a trace, you must take the FPGA package delays into consideration. For more information, refer to *Package Deskew*.

For device families that do not support write leveling, refer to *Layout Guidelines for DDR2 SDRAM Interfaces*.

**Table 34. DDR3 and DDR4 SDRAM Layout Guidelines (1)**

Parameter	Guidelines
Decoupling Parameter	<ul style="list-style-type: none"><li>• Make VTT voltage decoupling close to the components and pull-up resistors.</li><li>• Connect decoupling caps between VTT and VDD using a 0.1 <math>\mu</math>F cap for every other VTT pin.</li><li>• Use a 0.1 <math>\mu</math>F cap and 0.01 <math>\mu</math>F cap for every VDDQ pin.</li></ul>
Maximum Trace Length (2)	<ul style="list-style-type: none"><li>• Even though there are no hard requirements for minimum trace length, you need to simulate the trace to ensure the signal integrity. Shorter routes result in better timing.</li><li>• For DIMM topology only:</li><li>• Maximum trace length for all signals from FPGA to the first DIMM slot is 4.5 inches.</li><li>• Maximum trace length for all signals from DIMM slot to DIMM slot is 0.425 inches.</li></ul>

*continued...*



Parameter	Guidelines
	<ul style="list-style-type: none"><li>For discrete components only:</li><li>Maximum trace length for address, command, control, and clock from FPGA to the first component must not be more than 7 inches.</li><li>Maximum trace length for DQ, DQS, DQS#, and DM from FPGA to the first component is 5 inches.</li></ul>
General Routing	<ul style="list-style-type: none"><li>Route over appropriate VCC and GND planes.</li><li>Keep signal routing layers close to GND and power planes.</li></ul>
Spacing Guidelines	<ul style="list-style-type: none"><li>Avoid routing two signal layers next to each other. Always make sure that the signals related to memory interface are routed between appropriate GND or power layers.</li><li>For DQ/DQS/DM traces: Maintain at least 3H spacing between the edges (air-gap) for these traces. (Where H is the vertical distance to the closest return path for that particular trace.)</li><li>For Address/Command/Control traces: Maintain at least 3H spacing between the edges (air-gap) these traces. (Where H is the vertical distance to the closest return path for that particular trace.)</li><li>For Clock traces: Maintain at least 5H spacing between two clock pair or a clock pair and any other memory interface trace. (Where H is the vertical distance to the closest return path for that particular trace.)</li></ul>
Clock Routing	<ul style="list-style-type: none"><li>Route clocks on inner layers with outer-layer run lengths held to under 500 mils (12.7 mm).</li><li>Route clock signals in a daisy chain topology from the first SDRAM to the last SDRAM. The maximum length of the first SDRAM to the last SDRAM must not exceed 0.69 tCK for DDR3 and 1.5 tCK for DDR4. For different DIMM configurations, check the appropriate JEDEC specification.</li><li>These signals should maintain the following spacings:</li><li>Clocks should maintain a length-matching between clock pairs of <math>\pm 5</math> ps.</li><li>Clocks should maintain a length-matching between positive (<math>p</math>) and negative (<math>n</math>) signals of <math>\pm 2</math> ps, routed in parallel.</li><li>Space between different pairs should be at least two times the trace width of the differential pair to minimize loss and maximize interconnect density.</li><li>To avoid mismatched transmission line to via, Intel recommends that you use Ground Signal Signal Ground (GSSG) topology for your clock pattern—GND  CLKP CKLN GND.</li><li>Route all addresses and commands to match the clock signals to within <math>\pm 20</math> ps to each discrete memory component. Refer to the following figure.</li></ul>
Address and Command Routing	<ul style="list-style-type: none"><li>Route address and command signals in a daisy chain topology from the first SDRAM to the last SDRAM. The maximum length of the first SDRAM to the last SDRAM must not be more than 0.69 tCK for DDR3 and 1.5 tCK for DDR4. For different DIMM configurations, check the appropriate JEDEC specifications.</li><li>UDIMMs are more susceptible to cross-talk and are generally noisier than buffered DIMMs. Therefore, route address and command signals of UDIMMs on a different layer than data signals (DQ) and data mask signals (DM) and with greater spacing.</li><li>Do not route differential clock (CK) and clock enable (CKE) signals close to address signals.</li><li>Route all addresses and commands to match the clock signals to within <math>\pm 20</math> ps to each discrete memory component. Refer to the following figure.</li></ul>

*continued...*



Parameter	Guidelines
DQ, DM, and DQS Routing Rules	<ul style="list-style-type: none"> <li>All the trace length matching requirements are from the FPGA package ball to the SDRAM package ball, which means you must consider trace mismatching on different DIMM raw cards.</li> <li>Match in length all DQ, DQS, and DM signals within a given byte-lane group with a maximum deviation of <math>\pm 10</math> ps.</li> <li>Ensure to route all DQ, DQS, and DM signals within a given byte-lane group on the same layer to avoid layer to layer transmission velocity differences, which otherwise increase the skew within the group.</li> <li>Do not count on FPGAs to deskew for more than 20 ps of DQ group skew. The skew algorithm only removes the following possible uncertainties: <ul style="list-style-type: none"> <li>Minimum and maximum die IOE skew or delay mismatch</li> <li>Minimum and maximum device package skew or mismatch</li> <li>Board delay mismatch of 20 ps</li> <li>Memory component DQ skew mismatch</li> <li>Increasing any of these four parameters runs the risk of the deskew algorithm limiting, failing to correct for the total observed system skew. If the algorithm cannot compensate without limiting the correction, timing analysis shows reduced margins.</li> </ul> </li> <li>For memory interfaces with leveling, the timing between the DQS and clock signals on each device calibrates dynamically to meet tDQSS. To make sure the skew is not too large for the leveling circuit's capability, follow these rules: <ul style="list-style-type: none"> <li>Propagation delay of clock signal must not be shorter than propagation delay of DQS signal at every device: <math>(CK_i) - DQS_i &gt; 0</math>; <math>0 &lt; i &lt;</math> number of components – 1 . For DIMMs, ensure that the CK trace is longer than the longest DQS trace at the DIMM connector.</li> <li>Total skew of CLK and DQS signal between groups is less than one clock cycle: <math>(CK_i + DQS_i)_{\text{max}} - (CK_i + DQS_i)_{\text{min}} &lt; 1 \times tCK</math>(If you are using a DIMM topology, your delay and skew must take into consideration values for the actual DIMM.)</li> </ul> </li> </ul>
Spacing Guidelines	<ul style="list-style-type: none"> <li>Avoid routing two signal layers next to each other. Always ensure that the signals related to the memory interface are routed between appropriate GND or power layers.</li> <li>For DQ/DQS/DM traces: Maintain at least 3H spacing between the edges (air-gap) of these traces, where H is the vertical distance to the closest return path for that particular trace.</li> <li>For Address/Command/Control traces: Maintain at least 3H spacing between the edges (air-gap) of these traces, where H is the vertical distance to the closest return path for that particular trace.</li> <li>For Clock traces: Maintain at least 5H spacing between two clock pairs or a clock pair and any other memory interface trace, where H is the vertical distance to the closest return path for that particular trace.</li> </ul>
Quartus Prime Software Settings for Board Layout	<ul style="list-style-type: none"> <li>To perform timing analyses on board and I/O buffers, use third party simulation tool to simulate all timing information such as skew, ISI, crosstalk, and type the simulation result into the UniPHY board setting panel.</li> <li>Do not use advanced I/O timing model (AIOT) or board trace model unless you do not have access to any third party tool. AIOT provides reasonable accuracy but tools like HyperLynx provide better results.</li> </ul>
Notes to Table:	
<ol style="list-style-type: none"> <li>For point-to-point and DIMM interface designs, refer to the Micron website, <a href="http://www.micron.com">www.micron.com</a>.</li> <li>For better efficiency, the UniPHY IP requires faster turnarounds from read commands to write.</li> </ol>	

### Related Information

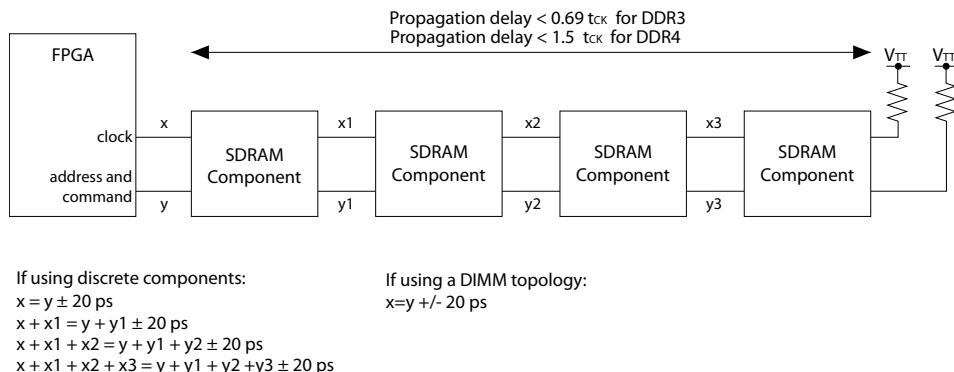
- [Layout Guidelines for DDR2 SDRAM Interface](#) on page 94
- [Package Deskeew](#) on page 121
- [External Memory Interface Spec Estimator](#)
- [www.micron.com](http://www.micron.com)

### 2.7.3. Length Matching Rules

The following topics provide guidance on length matching for different types of DDR3 signals.

Route all addresses and commands to match the clock signals to within  $\pm 20$  ps to each discrete memory component. The following figure shows the DDR3 and DDR4 SDRAM component routing guidelines for address and command signals.

**Figure 30. DDR3 and DDR4 SDRAM Component Address and Command Routing Guidelines**



The timing between the DQS and clock signals on each device calibrates dynamically to meet tDQSS. The following figure shows the delay requirements to align DQS and clock signals. To ensure that the skew is not too large for the leveling circuit's capability, follow these rules:

- Propagation delay of clock signal must not be shorter than propagation delay of DQS signal at every device:

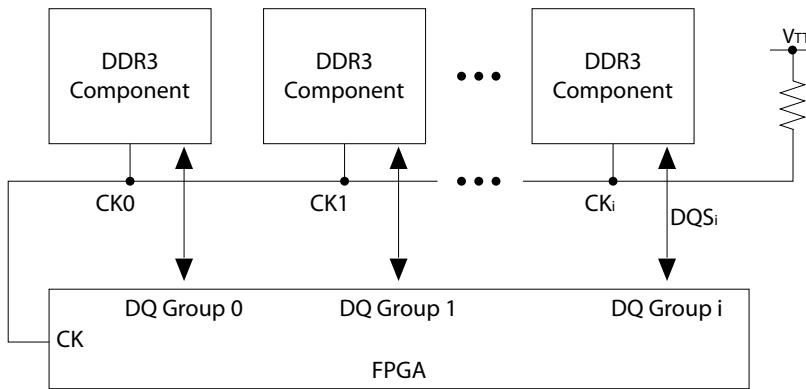
$$CK_i - DQS_i > 0; 0 < i < \text{number of components} - 1$$

- Total skew of CLK and DQS signal between groups is less than one clock cycle:

$$(CK_i + DQS_i)_{\max} - (CK_i + DQS_i)_{\min} < 1 \times tCK$$



**Figure 31. Delaying DQS Signal to Align DQS and Clock**



$CK_i$  = Clock signal propagation delay to device  $i$

$DQS_i$  = DQ/DQS signals propagation delay to group  $i$

**Clk pair matching**—If you are using a DIMM (UDIMM, RDIMM, or LRDIMM) topology, match the trace lengths up to the DIMM connector. If you are using discrete components, match the lengths for all the memory components connected in the fly-by chain.

**DQ group length matching**—If you are using a DIMM (UDIMM, RDIMM, or LRDIMM) topology, apply the DQ group trace matching rules described in the guideline table earlier up to the DIMM connector. If you are using discrete components, match the lengths up to the respective memory components.

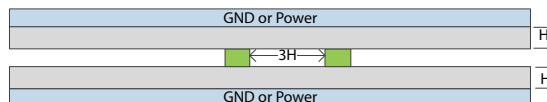
When you are using DIMMs, it is assumed that lengths are tightly matched within the DIMM itself. You should check that appropriate traces are length-matched within the DIMM.

## 2.7.4. Spacing Guidelines

This topic provides recommendations for minimum spacing between board traces for various signal traces.

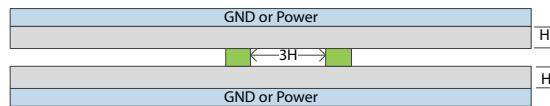
### Spacing Guidelines for DQ, DQS, and DM Traces

Maintain a minimum of  $3H$  spacing between the edges (air-gap) of these traces. (Where  $H$  is the vertical distance to the closest return path for that particular trace.)



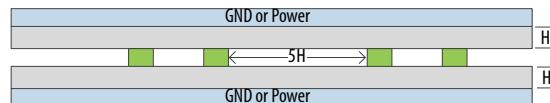
### Spacing Guidelines for Address and Command and Control Traces

Maintain at least  $3H$  spacing between the edges (air-gap) of these traces. (Where  $H$  is the vertical distance to the closest return path for that particular trace.)



### Spacing Guidelines for Clock Traces

Maintain at least 5H spacing between two clock pair or a clock pair and any other memory interface trace. (Where H is the vertical distance to the closest return path for that particular trace.)



## 2.7.5. Layout Guidelines for DDR3 and DDR4 SDRAM Wide Interface (>72 bits)

The following topics discuss different ways to lay out a wider DDR3 or DDR4 SDRAM interface to the FPGA. Choose the topology based on board trace simulation and the timing budget of your system.

The UniPHY IP supports up to a 144-bit wide DDR3 interface. You can either use discrete components or DIMMs to implement a wide interface (any interface wider than 72 bits). Intel recommends using leveling when you implement a wide interface with DDR3 components.

When you lay out for a wider interface, all rules and constraints discussed in the previous sections still apply. The DQS, DQ, and DM signals are point-to-point, and all the same rules discussed in *Design Layout Guidelines* apply.

The main challenge for the design of the fly-by network topology for the clock, command, and address signals is to avoid signal integrity issues, and to make sure you route the DQS, DQ, and DM signals with the chosen topology.

### Related Information

[Design Layout Guidelines on page 111](#)

### 2.7.5.1. Fly-By Network Design for Clock, Command, and Address Signals

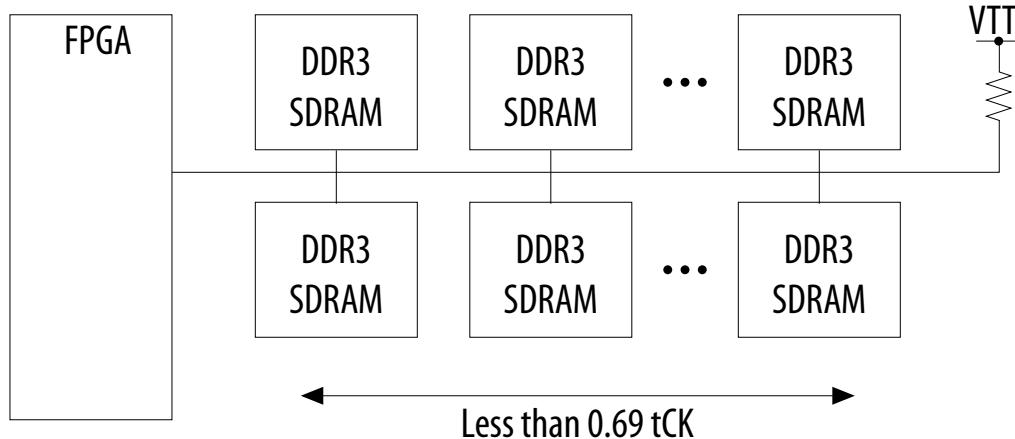
The UniPHY IP requires the flight-time skew between the first DDR3 SDRAM component and the last DDR3 SDRAM component to be less than 0.69 tCK for memory clocks. This constraint limits the number of components you can have for each fly-by network.

If you design with discrete components, you can choose to use one or more fly-by networks for the clock, command, and address signals.

The following figure shows an example of a single fly-by network topology.



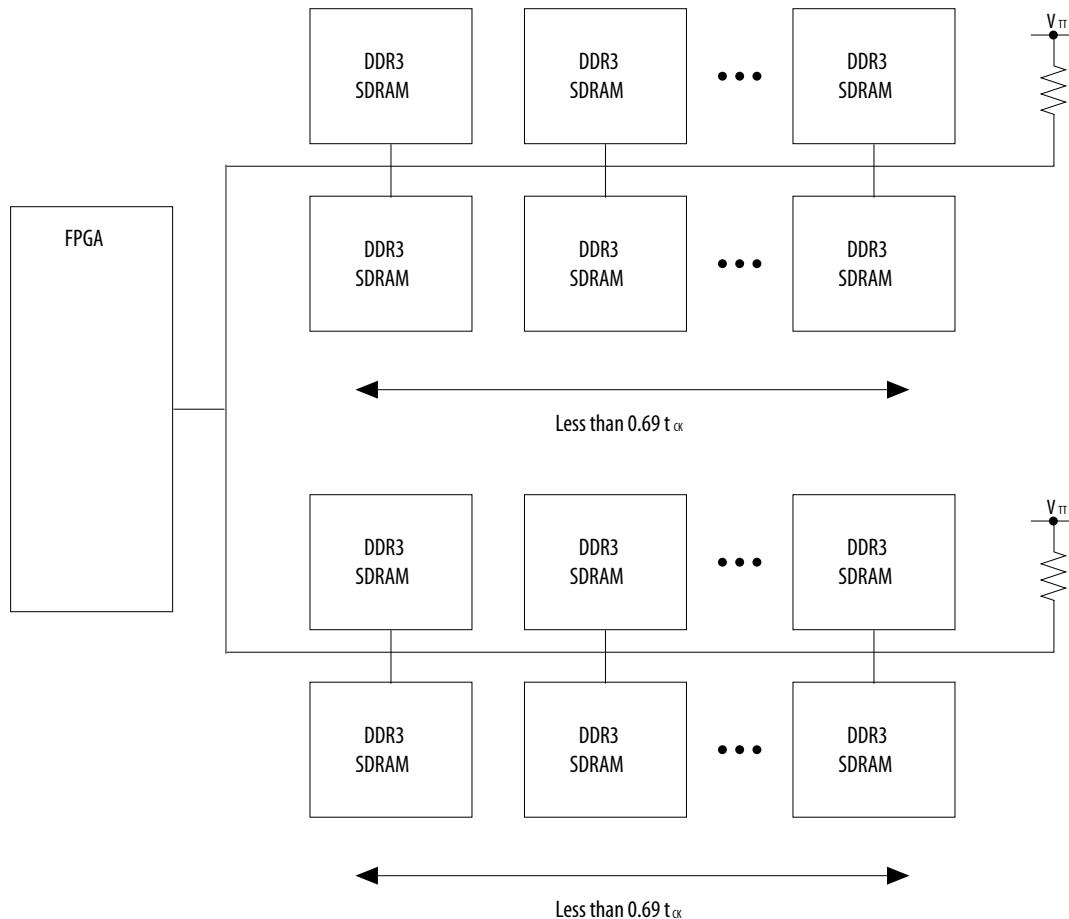
Figure 32. Single Fly-By Network Topology



Every DDR3 SDRAM component connected to the signal is a small load that causes discontinuity and degrades the signal. When using a single fly-by network topology, to minimize signal distortion, follow these guidelines:

- Use  $\times 16$  device instead  $\times 4$  or  $\times 8$  to minimize the number of devices connected to the trace.
- Keep the stubs as short as possible.
- Even with added loads from additional components, keep the total trace length short; keep the distance between the FPGA and the first DDR3 SDRAM component less than 5 inches.
- Simulate clock signals to ensure a decent waveform.

The following figure shows an example of a double fly-by network topology. This topology is not rigid but you can use it as an alternative option. The advantage of using this topology is that you can have more DDR3 SDRAM components in a system without violating the 0.69 tCK rule. However, as the signals branch out, the components still create discontinuity.

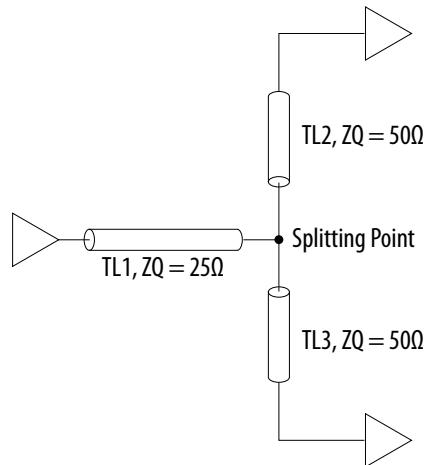
**Figure 33. Double Fly-By Network Topology**


You must perform simulations to find the location of the split, and the best impedance for the traces before and after the split.

The following figure shows a way to minimize the discontinuity effect. In this example, keep TL2 and TL3 matches in length. Keep TL1 longer than TL2 and TL3, so that it is easier to route all the signals during layout.



**Figure 34. Minimizing Discontinuity Effect**



You can also consider using a DIMM on each branch to replace the components. Because the trade impedance on the DIMM card is 40-ohm to 60-ohm, perform a board trace simulation to control the reflection to within the level your system can tolerate.

By using the new features of the DDR3 SDRAM controller with UniPHY and the Stratix III, Stratix IV, or Stratix V devices, you simplify your design process. Using the fly-by daisy chain topology increases the complexity of the datapath and controller design to achieve leveling, but also greatly improves performance and eases board layout for DDR3 SDRAM.

You can also use the DDR3 SDRAM components without leveling in a design if it may result in a more optimal solution, or use with devices that support the required electrical interface standard, but do not support the required read and write leveling functionality.

## 2.8. Package Deskew

Trace lengths inside the device package are not uniform for all package pins. The nonuniformity of package traces can affect system timing for high frequencies. In the Quartus II software version 12.0 and later, and the Quartus Prime software, a package deskew option is available.

If you do not enable the package deskew option, the Quartus Prime software uses the package delay numbers to adjust skews on the appropriate signals; you do not need to adjust for package delays on the board traces. If you do enable the package deskew option, the Quartus Prime software does not use the package delay numbers for timing analysis, and you must deskew the package delays with the board traces for the appropriate signals for your design.

### 2.8.1. Package Deskew Recommendation for Stratix V Devices

Package deskew is not required for any memory protocol operating at 800 MHz or below.



For DDR3 and RLDRAM3 designs operating above 800 MHz, you should run timing analysis with accurately entered board skew parameters in the parameter editor. If **Report DDR** reports non-core timing violations, you should then perform the steps in the following topics, and modify your board layout. Package deskew is not required for any protocols other than DDR3 and RLDRAM 3.

### 2.8.2. DQ/DQS/DM Deskew

To get the package delay information, follow these steps:

1. Select the **FPGA DQ/DQS Package Skews Deskewed on Board** checkbox on the **Board Settings** tab of the parameter editor.
2. Generate your IP.
3. Instantiate your IP in the project.
4. Run **Analysis and Synthesis** in the Quartus Prime software. (Skip this step if you are using an Arria 10 device.)
5. Run the `<core_name>.p0_pin_assignment.tcl` script. (Skip this step if you are using an Arria 10 device.)
6. Compile your design.
7. Refer to the **All Package Pins** compilation report, or find the pin delays displayed in the `<core_name>.pin` file.

### 2.8.3. Address and Command Deskew

Deskew address and command delays as follows:

1. Select the **FPGA Address/Command Package Skews Deskewed on Board** checkbox on the **Board Settings** tab of the parameter editor.
2. Generate your IP.
3. Instantiate your IP in the project.
4. Run **Analysis and Synthesis** in the Quartus Prime software. (Skip this step if you are using an Arria 10 device.)
5. Run the `<core_name>.p0_pin_assignment.tcl` script. (Skip this step if you are using an Arria 10 device.)
6. Compile your design.
7. Refer to the **All Package Pins** compilation report, or find the pin delays displayed in the `<core_name>.pin` file.

### 2.8.4. Package Deskew Recommendations for Arria 10 and Stratix 10 Devices

The following table shows package deskew recommendations for all protocols supported on Arria 10 devices.

As operating frequencies increase, it becomes increasingly critical to perform package deskew. The frequencies listed in the table are the *minimum* frequencies for which you must perform package deskew.



If you plan to use a listed protocol at the specified frequency or higher, you must perform package deskew. For example, you must perform package deskew if you plan to use dual-rank DDR4 at 800 MHz or above.

Protocol	Minimum Frequency (MHz) for Which to Perform Package Deskew		
	Single Rank	Dual Rank	Quad Rank
DDR4	933	800	667
DDR3	933	800	667
LPDDR3	667	533	Not required
QDR IV	933	Not applicable	Not applicable
RLDRAM 3	933	667	Not applicable
RLDRAM II	Not required	Not applicable	Not applicable
QDR II, II+, II+ Xtreme	Not required	Not applicable	Not applicable

## 2.8.5. Deskew Example

Consider an example where you want to deskew an interface with 4 DQ pins, 1 DQS pin, and 1 DQSn pin.

Let's assume an operating frequency of 667 MHz, and the package lengths for the pins reported in the **.pin** file as follows:

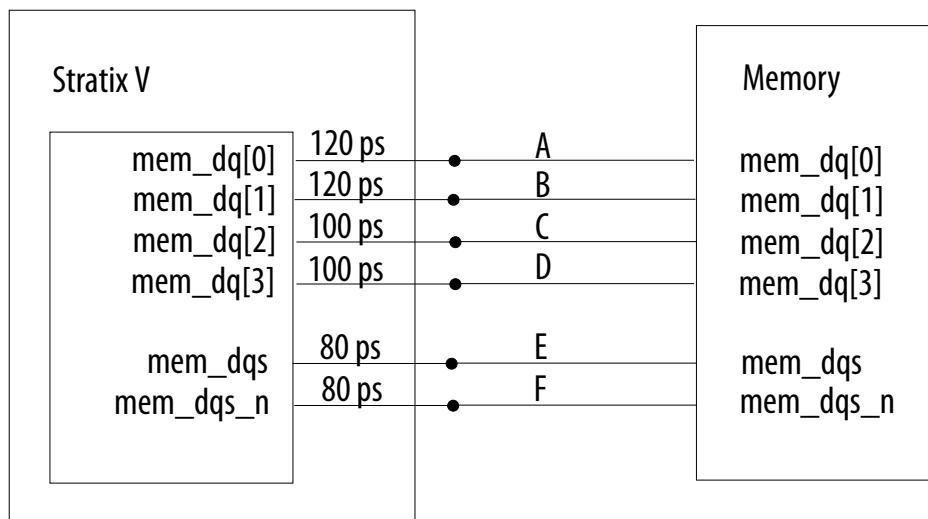
```

dq[ 0] = 120 ps
dq[ 1] = 120 ps
dq[ 2] = 100 ps
dq[ 3] = 100 ps
dqs   = 80 ps
dqs_n = 80 ps

```

The following figure illustrates this example.

**Figure 35. Deskew Example**

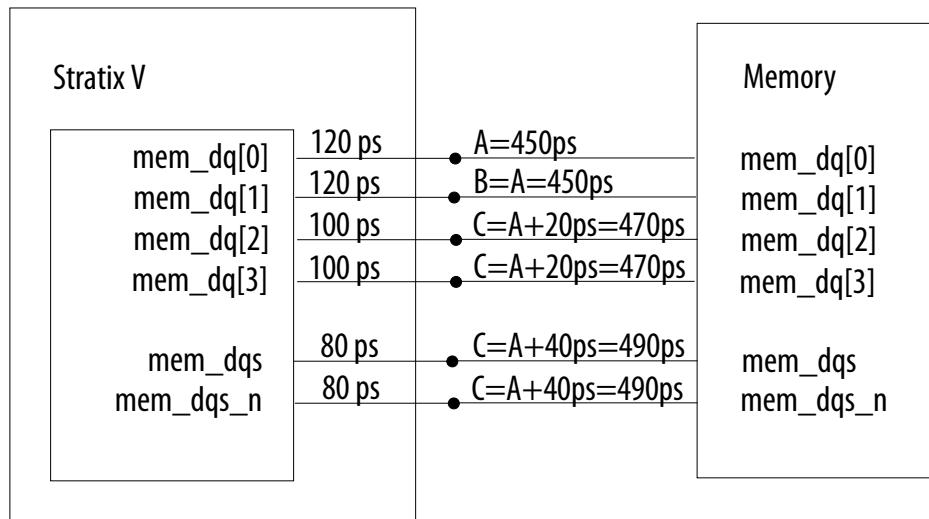


When you perform length matching for all the traces in the DQS group, you must take package delays into consideration. Because the package delays of traces A and B are 40 ps longer than the package delays of traces E and F, you would need to make the board traces for E and F 40 ps longer than the board traces for A and B.

A similar methodology would apply to traces C and D, which should be 20 ps longer than the lengths of traces A and B.

The following figure shows this scenario with the length of trace A at 450 ps.

**Figure 36. Deskew Example with Trace Delay Calculations**



When you enter the board skews into the Board Settings tab of the DDR3 parameter editor, you should calculate the board skew parameters as the sums of board delay and corresponding package delay. If a pin does not have a package delay (such as address and command pins), you should use the board delay only.

The example of the preceding figure shows an ideal case where board skews are perfectly matched. In reality, you should allow plus or minus 10 ps of skew mismatch within a DQS group (DQ/DQS/DM).

## 2.8.6. Package Migration

Package delays can be different for the same pin in different packages. If you want to use multiple migratable packages in your system, you should compensate for package skew as described in this topic. The information in this topic applies to Arria 10, Stratix V, and Stratix 10 devices.

### Scenario 1

Your PCB is designed for multiple migratable devices, but you have only one device with which to go to production.



Assume two migratable packages, device A and device B, and that you want to go to production with device A. Follow these steps:

1. Perform package deskew for device A.
2. Compile your design for device A, with the **Package Skew** option enabled.
3. Note the skews in the <core\_name>.pin file for device A. Deskew these package skews with board trace lengths as described in the preceding examples.
4. Recompile your design for device A.
5. For device B, open the parameter editor and deselect the **Package Deskew** option.
6. Calculate board skew parameters, only taking into account the board traces for device B, and enter that value into the parameter editor for device B.
7. Regenerate the IP and recompile the design for device B.
8. Verify that timing requirements are met for both device A and device B.

### Scenario 2

Your PCB is designed for multiple migratable devices, and you want to go to production with all of them.

Assume you have device A and device B, and plan to use both devices in production. Follow these steps:

1. Do not perform any package deskew compensation for either device.
2. Compile a Quartus Prime design for device A with the **Package Deskew** option disabled, and ensure that all board skews are entered accurately.
3. Verify that the **Report DDR** timing report meets your timing requirements.
4. Compile a Quartus Prime design for device B with the **Package Deskew** option disabled, and ensure that all board skews are entered accurately.
5. Verify that the **Report DDR** timing report meets your timing requirements.

## 2.8.7. Package Deskew for RLDRAM II and RLDRAM 3

You should follow Intel's package deskew guidance if you are using Arria 10, Stratix 10, or Stratix V devices.

For more information on package deskew, refer to *Package Deskew*.

### Related Information

[Package Deskew](#)

## 2.9. Document Revision History



Date	Version	Changes
May 2017	2017.05.08	<ul style="list-style-type: none"><li>Added <i>Channel Signal Integrity Measurement</i> section.</li><li>Added Stratix 10 to several sections.</li><li>Removed QDR-IV future support note from <i>Package Deskeew Recommendations for Arria 10 and Stratix 10 Devices</i> section.</li><li>Rebranded as Intel.</li></ul>
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	<ul style="list-style-type: none"><li>Minor change to <i>Clock Routing</i> description in the <i>DDR2 SDRAM Layout Guidelines</i> table in <i>Layout Guidelines for DDR2 SDRAM Interface</i>.</li><li>Added maximum length of the first SDRAM to the last SDRAM for clock routing and address and command routing for DDR4, in <i>Layout Guidelines for DDR3 and DDR4 SDRAM Interfaces</i>.</li><li>Removed <i>DRAM Termination Guidance</i> from <i>Layout Guidelines for DDR3 and DDR4 SDRAM Interfaces</i>.</li><li>Added DDR4 support to <i>Length Matching Rules</i>.</li></ul>
November 2015	2015.11.02	<ul style="list-style-type: none"><li>Minor additions to procedure steps in <i>DQ/DQS/DM Deskeew</i> and <i>Address and Command Deskeew</i>.</li><li>Added reference to Micron Technical Note in <i>Layout Guidelines for DDR3 and DDR4 SDRAM Interfaces</i>.</li><li>Changed title of <i>Board Termination for DDR2 SDRAM</i> to <i>Termination for DDR2 SDRAM</i> and <i>Board Termination for DDR3 SDRAM</i> to <i>Termination for DDR3 SDRAM</i>.</li><li>Changed title of <i>Leveling and Dynamic ODT</i> to <i>Leveling and Dynamic Termination</i>.</li><li>Added DDR4 support in <i>Dynamic ODT</i>.</li><li>Removed topics pertaining to older device families.</li><li>Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i>.</li></ul>
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	<ul style="list-style-type: none"><li>Added MAX 10 to <i>On-Chip Termination</i> topic.</li><li>Added MAX 10 to <i>Termination Recommendations</i> table in <i>Recommended Termination Schemes</i> topic.</li></ul>
August 2014	2014.08.15	<ul style="list-style-type: none"><li>Added Arria V Soc and Cyclone V SoC devices to note in <i>Leveling and Dynamic ODT</i> section.</li><li>Added DDR4 to <i>Read and Write Leveling</i> section.</li><li>Revised text in <i>On-Chip Termination</i> section.</li><li>Added text to note in <i>Board Termination for DDR3 SDRAM</i> section.</li><li>Added <i>Layout Approach</i> information in the <i>DDR3 and DDR4 on Arria 10 Devices</i> section.</li><li>Recast expressions of length-matching measurements throughout <i>DDR2 SDRAM Layout Guidelines</i> table.</li><li>Made several changes to <i>DDR3 and DDR4 SDRAM Layout Guidelines</i> table:<ul style="list-style-type: none"><li>Added <i>Spacing Guidelines</i> section.</li><li>Removed millimeter approximations from lengths expressed in picoseconds.</li><li>Revised Guidelines for <i>Clock Routing</i>, <i>Address and Command Routing</i>, and <i>DQ, DM, and DQS Routing Rules</i> sections.</li></ul></li><li>Added <i>Spacing Guidelines</i> information to <i>Design Layout Guidelines</i> section.</li></ul>
December 2013	2013.12.16	<ul style="list-style-type: none"><li>Review and minor updates of content.</li><li>Consolidated General Layout Guidelines.</li><li>Added DDR3 and DDR4 information for Arria 10 devices.</li><li>Updated chapter title to include DDR4 support.</li><li>Removed references to ALTMEMPHY.</li></ul>

*continued...*



Date	Version	Changes
		<ul style="list-style-type: none"> <li>Removed references to Cyclone III and Cyclone IV devices.</li> <li>Removed references to Stratix II devices.</li> <li>Corrected Vtt to Vdd in <i>Memory Clocks for DDR3 SDRAM UDIMM</i> section.</li> </ul>
November 2012	5.0	<ul style="list-style-type: none"> <li>Updated <i>Layout Guidelines for DDR2 SDRAM Interface</i> and <i>Layout Guidelines for DDR3 SDRAM Interface</i>.</li> <li>Added LRDIMM support.</li> <li>Added Package Deskew section.</li> </ul>
June 2012	4.1	Added Feedback icon.
November 2011	4.0	Added Arria V and Cyclone V information.
June 2011	3.0	<ul style="list-style-type: none"> <li>Merged DDR2 and DDR3 chapters to <i>DDR2 and DDR3 SDRAM Interface Termination and Layout Guidelines</i> and updated with leveling information.</li> <li>Added Stratix V information.</li> </ul>
December 2010	2.1	Added <i>DDR3 SDRAM Interface Termination, Drive Strength, Loading, and Board Layout Guidelines</i> chapter with Stratix V information.
July 2010	2.0	Updated Arria II GX information.
April 2010	1.0	Initial release.

## 3. Dual-DIMM DDR2 and DDR3 SDRAM Board Design Guidelines

---

The following topics describe guidelines for implementing dual unbuffered DIMM (UDIMM) DDR2 and DDR3 SDRAM interfaces.

The following topics discuss the impact on signal integrity of the data signal with the following conditions in a dual-DIMM configuration:

- Populating just one slot versus populating both slots
- Populating slot 1 versus slot 2 when only one DIMM is used
- On-die termination (ODT) setting of 75-ohm versus an ODT setting of 150-ohm

For detailed information about a single-DIMM DDR2 SDRAM interface, refer to the *DDR2 and DDR3 SDRAM Board Design Guidelines* chapter.

### Related Information

[DDR2, DDR3, and DDR4 SDRAM Board Design Guidelines](#) on page 81

### 3.1. General Layout Guidelines

The following table lists general board design layout guidelines. These guidelines are Intel recommendations, and should not be considered as hard requirements. You should perform signal integrity simulation on all the traces to verify the signal integrity of the interface. You should extract the slew rate and propagation delay information, enter it into the IP and compile the design to ensure that timing requirements are met.



**Table 35. General Layout Guidelines**

Parameter	Guidelines
Impedance	<ul style="list-style-type: none"> <li>All unused via pads must be removed, because they cause unwanted capacitance.</li> <li>Trace impedance plays an important role in the signal integrity. You must perform board level simulation to determine the best characteristic impedance for your PCB. For example, it is possible that for multi rank systems 40 ohms could yield better results than a traditional 50 ohm characteristic impedance.</li> </ul>
Decoupling Parameter	<ul style="list-style-type: none"> <li>Use 0.1 uF in 0402 size to minimize inductance</li> <li>Make VTT voltage decoupling close to termination resistors</li> <li>Connect decoupling caps between VTT and ground</li> <li>Use a 0.1 uF cap for every other VTT pin and 0.01 uF cap for every VDD and VDDQ pin</li> <li>Verify the capacitive decoupling using the Intel Power Distribution Network Design Tool</li> </ul>
Power	<ul style="list-style-type: none"> <li>Route GND and V<sub>CC</sub> as planes</li> <li>Route VCCIO for memories in a single split plane with at least a 20-mil (0.020 inches, or 0.508 mm) gap of separation</li> <li>Route VTT as islands or 250-mil (6.35-mm) power traces</li> <li>Route oscillators and PLL power as islands or 100-mil (2.54-mm) power traces</li> </ul>
General Routing	<p>All specified delay matching requirements include PCB trace delays, different layer propagation velocity variance, and crosstalk. To minimize PCB layer propagation variance, Intel recommends that signals from the same net group always be routed on the same layer.</p> <ul style="list-style-type: none"> <li>Use 45° angles (<i>not</i> 90° corners)</li> <li>Avoid T-Junctions for critical nets or clocks</li> <li>Avoid T-junctions greater than 250 mils (6.35 mm)</li> <li>Disallow signals across split planes</li> <li>Restrict routing other signals close to system reset signals</li> <li>Avoid routing memory signals closer than 0.025 inch (0.635 mm) to PCI or system clocks</li> </ul>

#### Related Information

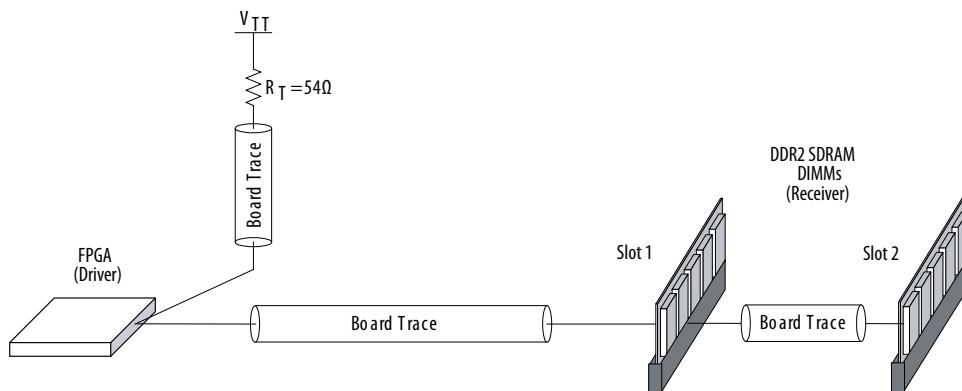
[Power Distribution Network Design Tool](#)

## 3.2. Dual-Slot Unbuffered DDR2 SDRAM

This topic describes guidelines for implementing a dual slot unbuffered DDR2 SDRAM interface, operating at up to 400-MHz and 800-Mbps data rates.

The following figure shows a typical DQS, DQ, and DM signal topology for a dual-DIMM interface configuration using the ODT feature of the DDR2 SDRAM components.

**Figure 37. Dual-DIMM DDR2 SDRAM Interface Configuration**



The simulations in this section use a Stratix® II device-based board. Because of limitations of this FPGA device family, simulations are limited to 266 MHz and 533 Mbps so that comparison to actual hardware results can be directly made.

### 3.2.1. Overview of ODT Control

When there is only a single-DIMM on the board, the ODT control is relatively straightforward. During write to the memory, the ODT feature of the memory is turned on; during read from the memory, the ODT feature of the memory is turned off. However, when there are multiple DIMMs on the board, the ODT control becomes more complicated.

With a dual-DIMM interface on the system, the controller has different options for turning the memory ODT on or off during read or write. The following table lists the DDR2 SDRAM ODT control during write to the memory. These DDR2 SDRAM ODT controls are recommended by Samsung Electronics. The JEDEC DDR2 specification was updated to include optional support for  $R_{TT}(\text{nominal}) = 50\text{-ohm}$ .

For more information about the DDR2 SDRAM ODT controls recommended by Samsung, refer to the *Samsung DDR2 Application Note: ODT (On Die Termination) Control*.

**Table 36. DDR2 SDRAM ODT Control—Writes<sup>(1)</sup>**

Slot 1 <sup>(2)</sup>	Slot 2 <sup>(2)</sup>	Write To	FPGA	Module in Slot 1		Module in Slot 2	
				Rank 1	Rank 2	Rank 3	Rank 4
DR	DR	Slot 1	Series 50-ohms	Infinite	Infinite	75 or 50-ohm	Infinite
		Slot 2	Series 50-ohms	75 or 50-ohm	Infinite	Infinite	Infinite
SR	SR	Slot 1	Series 50-ohms	Infinite	Unpopulated	75 or 50-ohm	Unpopulated

*continued...*



Slot 1 (2)	Slot 2 (2)	Write To	FPGA	Module in Slot 1		Module in Slot 2	
				Rank 1	Rank 2	Rank 3	Rank 4
		Slot 2	Series 50-ohms	75 or 50-ohm	Unpopulated	Infinite	Unpopulated
DR	Empty	Slot 1	Series 50-ohms	150-ohm	Infinite	Unpopulated	Unpopulated
Empty	DR	Slot 2	Series 50-ohms	Unpopulated	Unpopulated	150-ohm	Infinite
SR	Empty	Slot 1	Series 50-ohms	150-ohm	Unpopulated	Unpopulated	Unpopulated
Empty	SR	Slot 2	Series 50-ohms	Unpopulated	Unpopulated	150-ohm	Unpopulated

Notes to Table:

- For DDR2 at 400 MHz and 533 Mbps = 75-ohm; for DDR2 at 667 MHz and 800 Mbps = 50-ohm.
- SR = single ranked; DR = dual ranked.

**Table 37. DDR2 SDRAM ODT Control—Reads (1)**

Slot 1 (2)	Slot 2 (2)	Read From	FPGA	Module in Slot 1		Module in Slot 2	
				Rank 1	Rank 2	Rank 3	Rank 4
DR	DR	Slot 1	Parallel 50-ohms	Infinite	Infinite	75 or 50-ohm	Infinite
		Slot 2	Parallel 50-ohms	75 or 50-ohm	Infinite	Infinite	Infinite
SR	SR	Slot 1	Parallel 50-ohms	Infinite	Unpopulated	75 or 50-ohm	Unpopulated
		Slot 2	Parallel 50-ohms	75 or 50-ohm	Unpopulated	Infinite	Unpopulated
DR	Empty	Slot 1	Parallel 50-ohms	Infinite	Infinite	Unpopulated	Unpopulated
Empty	DR	Slot 2	Parallel 50-ohms	Unpopulated	Unpopulated	Infinite	Infinite
SR	Empty	Slot 1	Parallel 50-ohms	Infinite	Unpopulated	Unpopulated	Unpopulated
Empty	SR	Slot 2	Parallel 50-ohms	Unpopulated	Unpopulated	Infinite	Unpopulated

Notes to Table:

- For DDR2 at 400 MHz and 533 Mbps = 75-ohm; for DDR2 at 667 MHz and 800 Mbps = 50-ohm.
- SR = single ranked; DR = dual ranked.

### 3.2.2. DIMM Configuration

Although populating both memory slots is common in a dual-DIMM memory system, there are some instances when only one slot is populated.

For example, some systems are designed to have a certain amount of memory initially and as applications get more complex, the system can be easily upgraded to accommodate more memory by populating the second memory slot without redesigning the system. The following topics discuss a dual-DIMM system where the

dual-DIMM system only has one slot populated at one time and a dual-DIMM system where both slots are populated. ODT controls recommended by memory vendors, as well as other possible ODT settings are evaluated for usefulness in an FPGA system.

### **3.2.3. Dual-DIMM Memory Interface with Slot 1 Populated**

The following topics focus on a dual-DIMM memory interface where slot 1 is populated and slot 2 is unpopulated.

These topics examine the impact on the signal quality due to an unpopulated DIMM slot and compares it to a single-DIMM memory interface.

#### **3.2.3.1. FPGA Writing to Memory**

In the DDR2 SDRAM, the ODT feature has two settings: 150-ohms and 75-ohms.

The recommended ODT setting for a dual DIMM configuration with one slot occupied is 150-ohm.

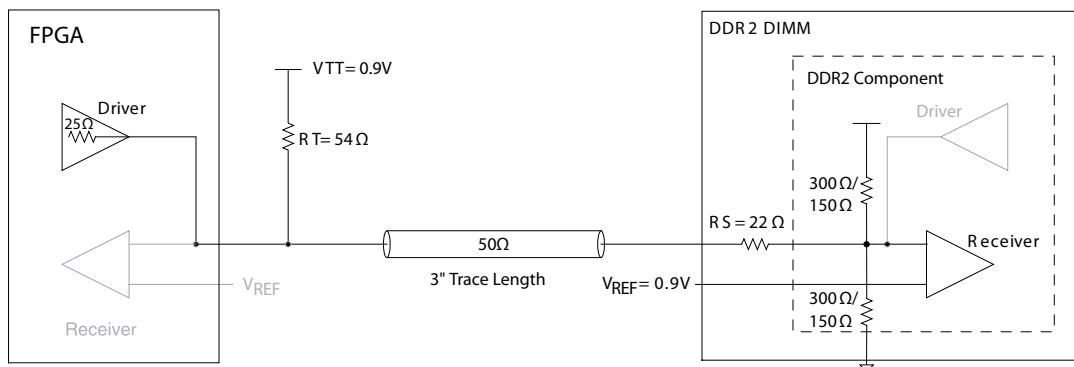
**Note:** On DDR2 SDRAM devices running at 333 MHz/667 Mbps and above, the ODT feature supports an additional setting of 50-ohm.

Refer to the respective memory decathlete for additional information about the ODT settings in DDR2 SDRAM devices.

#### **3.2.3.2. Write to Memory Using an ODT Setting of 150-ohm**

The following figure shows a double parallel termination scheme (Class II) using ODT on the memory with a memory-side series resistor when the FPGA is writing to the memory using a 25-ohm OCT drive strength setting on the FPGA.

**Figure 38. Double Parallel Termination Scheme (Class II) Using ODT on DDR2 SDRAM DIMM with Memory-Side Series Resistor**



#### **Related Information**

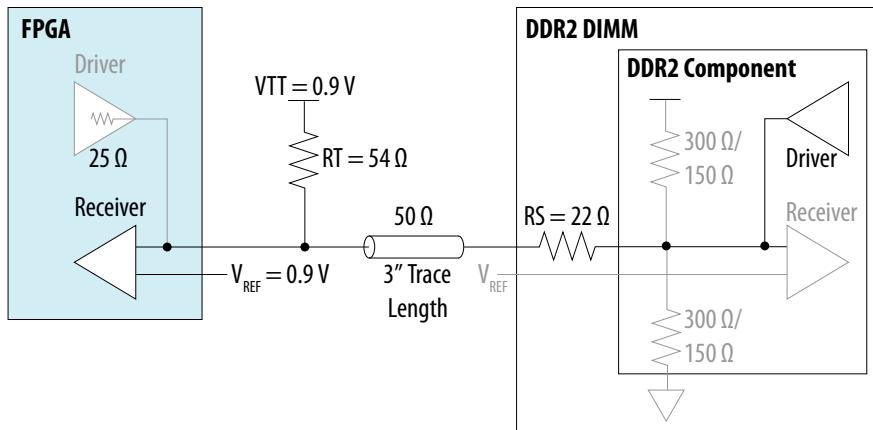
[DDR2, DDR3, and DDR4 SDRAM Board Design Guidelines](#) on page 81



### 3.2.3.3. Reading from Memory

During read from the memory, the ODT feature is turned off. Thus, there is no difference between using an ODT setting of 150-ohm and 75-ohm. As such, the termination scheme becomes a single parallel termination scheme (Class I) where there is an external resistor on the FPGA side and a series resistor on the memory side as shown in the following figure.

**Figure 39. Single Parallel Termination Scheme (Class I) Using External Resistor and Memory-Side Series Resistor**



#### Related Information

[DDR2, DDR3, and DDR4 SDRAM Board Design Guidelines](#) on page 81

### 3.2.4. Dual-DIMM with Slot 2 Populated

The following topics focus on a dual-DIMM memory interface where slot 2 is populated and slot 1 is unpopulated. Specifically, these topics discuss the impact of location of the DIMM on the signal quality.

#### 3.2.4.1. FPGA Writing to Memory

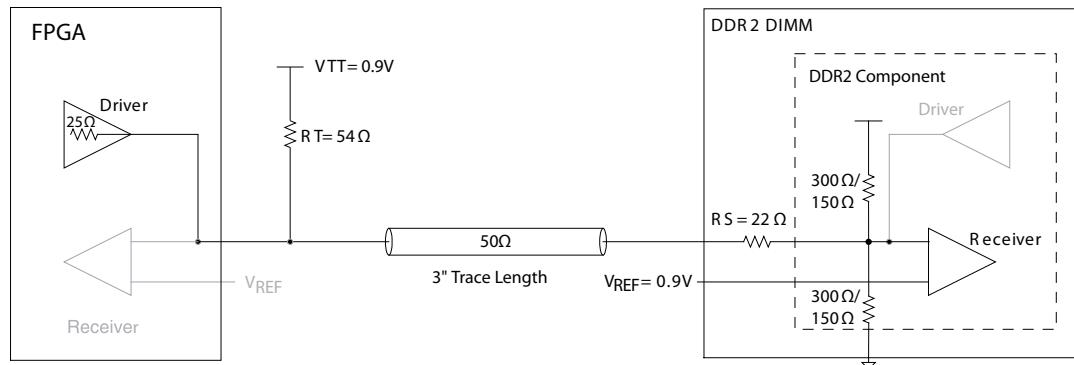
The following topics explore the differences between populating slot 1 and slot 2 of the dual-DIMM memory interface.

Previous topics focused on the dual-DIMM memory interface where slot 1 is populated resulting in the memory being located closer to the FPGA. When slot 2 is populated, the memory is located further away from the FPGA, resulting in additional trace length that potentially affects the signal quality seen by the memory. The following topics explore the differences between populating slot 1 and slot 2 of the dual-DIMM memory interface.

#### 3.2.4.2. Write to Memory Using an ODT Setting of 150-ohm

The following figure shows the double parallel termination scheme (Class II) using ODT on the memory with the memory-side series resistor when the FPGA is writing to the memory using a 25-ohm OCT drive strength setting on the FPGA.

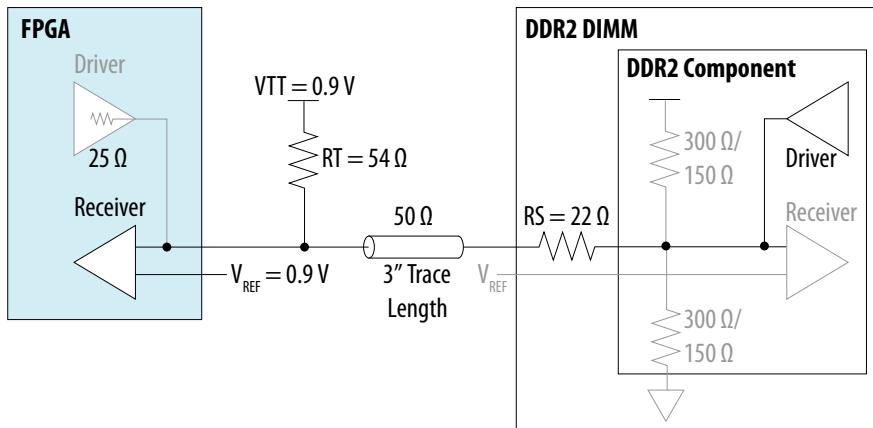
**Figure 40. Double Parallel Termination Scheme (Class II) Using ODT on DDR2 SDRAM DIMM with Memory-side Series Resistor**



### 3.2.4.3. Reading from Memory

During read from memory, the ODT feature is turned off, thus there is no difference between using an ODT setting of 150-ohm and 75-ohm. As such, the termination scheme becomes a single parallel termination scheme (Class I) where there is an external resistor on the FPGA side and a series resistor on the memory side, as shown in the following figure.

**Figure 41. Single Parallel Termination Scheme (Class I) Using External Resistor and Memory-Side Series Resistor**



## 3.2.5. Dual-DIMM Memory Interface with Both Slot 1 and Slot 2 Populated

The following topics focus on a dual-DIMM memory interface where both slot 1 and slot 2 are populated. As such, you can write to either the memory in slot 1 or the memory in slot 2.

### 3.2.5.1. FPGA Writing to Memory

The following topics explore the use of the 150-ohm setting and compares the results to that of the recommended 75-ohm.

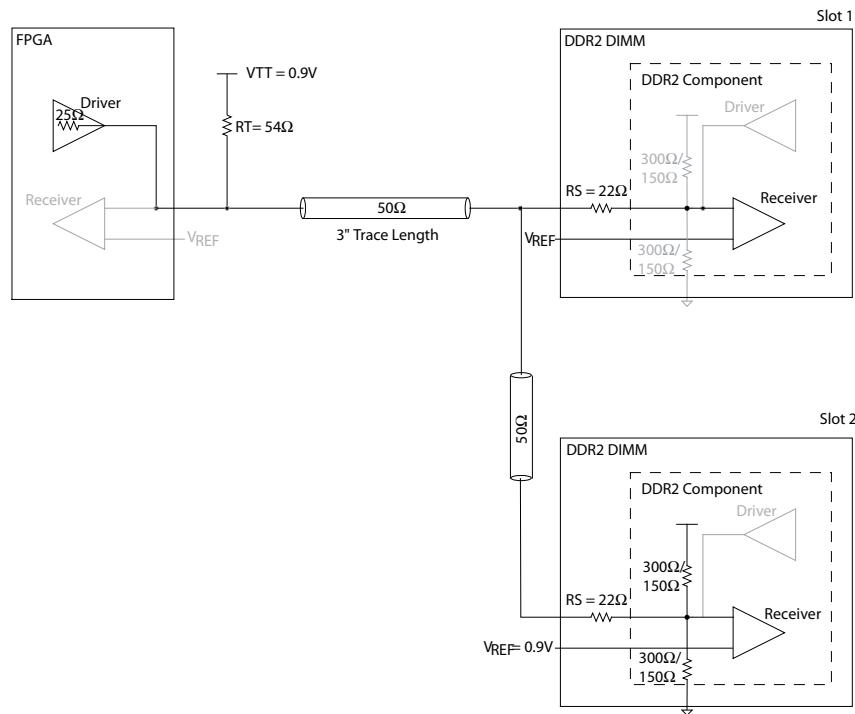


In Table 5–1, the recommended ODT setting for a dual DIMM configuration with both slots occupied is 75-ohm. Because there is an option for an ODT setting of 150-ohm, this section explores the usage of the 150-ohm setting and compares the results to that of the recommended 75-ohm.

### 3.2.5.2. Write to Memory in Slot 1 Using an ODT Setting of 75-ohm

The following figure shows the double parallel termination scheme (Class II) using ODT on the memory with the memory-side series resistor when the FPGA is writing to the memory using a 25-ohm OCT drive strength setting on the FPGA. In this scenario, the FPGA is writing to the memory in slot 1 and the ODT feature of the memory at slot 2 is turned on.

**Figure 42. Double Parallel Termination Scheme (Class II) Using ODT on DDR2 SDRAM DIMM with a Memory-Side Series Resistor**



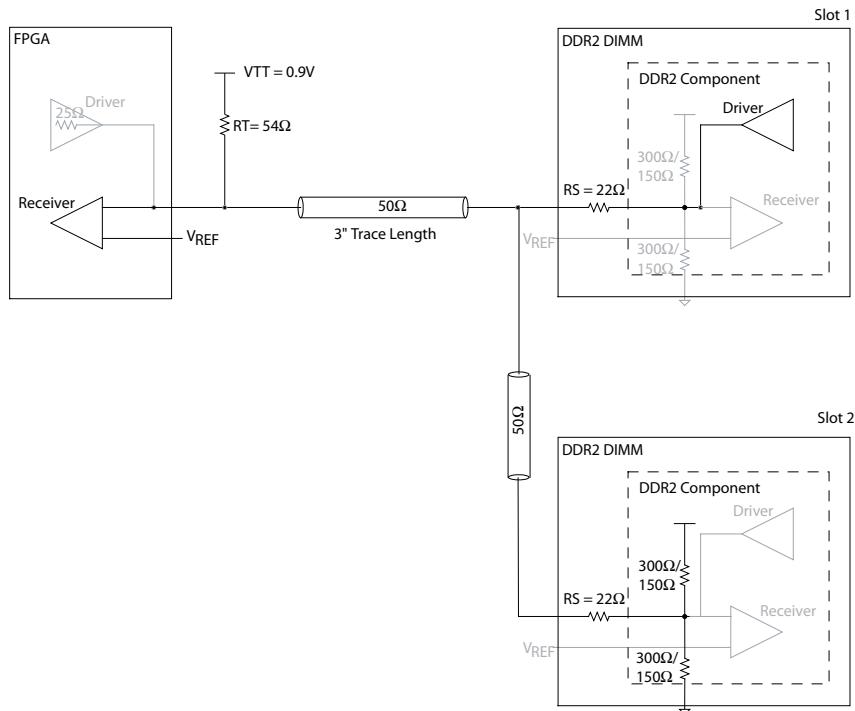
### 3.2.5.3. Reading From Memory

In Table 5–2, the recommended ODT setting for a dual-DIMM configuration with both slots occupied is to turn on the ODT feature using a setting of 75-ohm on the slot that is not read from. As there is an option for an ODT setting of 150-ohm, this section explores the usage of the 150-ohm setting and compares the results to that of the recommended 75-ohm.

### Read From Memory in Slot 1 Using an ODT Setting of 75-ohms on Slot 2

The following figure shows the double parallel termination scheme (Class II) using ODT on the memory with the memory-side series resistor when the FPGA is reading from the memory using a full drive strength setting on the memory. In this scenario, the FPGA is reading from the memory in slot 1 and the ODT feature of the memory at slot 2 is turned on.

**Figure 43. Double Parallel Termination Scheme (Class II) Using External Resistor and Memory-Side Series Resistor and ODT Feature Turned On**

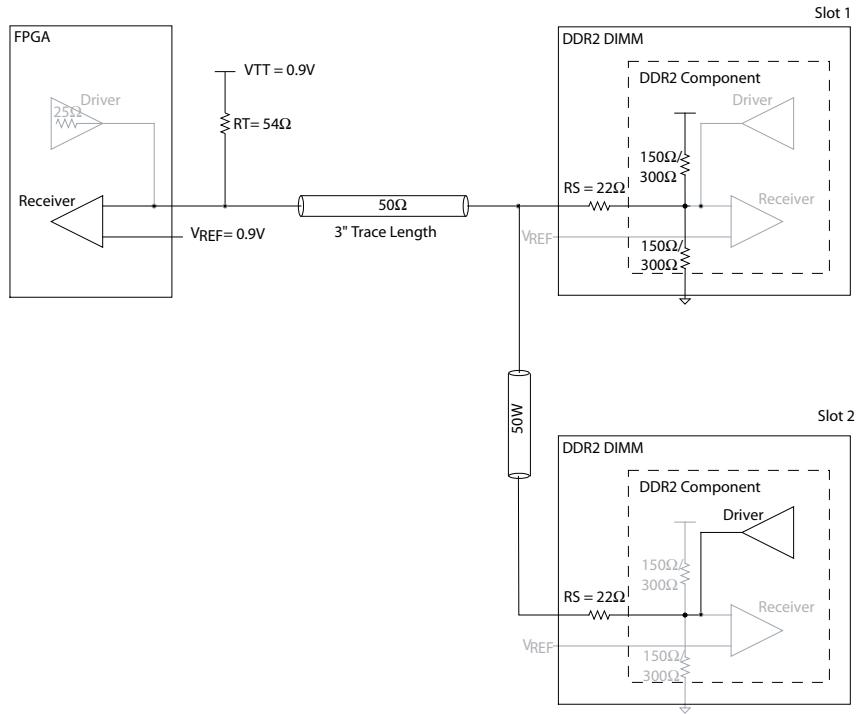


### Read From Memory in Slot 2 Using an ODT Setting of 75-ohms on Slot 1

In this scenario, the FPGA is reading from the memory in slot 2 and the ODT feature of the memory at slot 1 is turned on.



**Figure 44. Double Parallel Termination Scheme (Class II) Using External Resistor and a Memory-Side Series Resistor and ODT Feature Turned On**

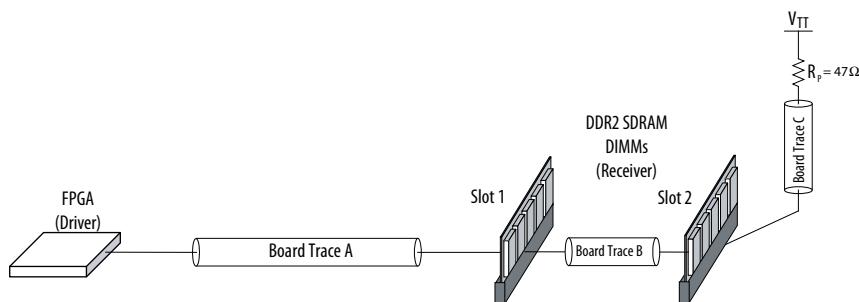


### 3.2.6. Dual-DIMM DDR2 Clock, Address, and Command Termination and Topology

The address and command signals on a DDR2 SDRAM interface are unidirectional signals that the FPGA memory controller drives to the DIMM slots. These signals are always Class-I terminated at the memory end of the line, as shown in the following figure.

Always place DDR2 SDRAM address and command Class-I termination after the last DIMM. The interface can have one or two DIMMs, but never more than two DIMMs total.

**Figure 45. Multi DIMM DDR2 Address and Command Termination Topology**



In the above figure, observe the following points:

- Board trace A = 1.9 to 4.5 inches (48 to 115 mm)
- Board trace B = 0.425 inches (10.795 mm)
- Board trace C = 0.2 to 0.55 inches (5 to 13 mm)
- Total of board trace A + B + C = 2.5 to 5 inches (63 to 127 mm)
- $R_p$  = 36 to 56-ohm
- Length match all address and command signals to +250 mils (+5 mm) or +/- 50 ps of memory clock length at the DIMM.

You may place a compensation capacitor directly before the first DIMM slot 1 to improve signal quality on the address and command signal group. If you fit a capacitor, Intel recommends a value of 24 pF.

For more information, refer to *Micron TN47-01*.

### 3.2.7. Control Group Signals

The control group of signals: chip select CS#, clock enable CKE, and ODT are always 1T regardless of whether you implement a full-rate or half-rate design.

As the signals are also SDR, the control group signals operate at a maximum frequency of  $0.5 \times$  the data rate. For example, in a 400-MHz design, the maximum control group frequency is 200 MHz.

### 3.2.8. Clock Group Signals

Depending on the specific form factor, DDR2 SDRAM DIMMs have two or three differential clock pairs, to ensure that the loading on the clock signals is not excessive. The clock signals are always terminated on the DIMMs and hence no termination is required on your PCB.

Additionally, each DIMM slot is required to have its own dedicated set of clock signals. Hence clock signals are always point-to-point from the FPGA PHY to each individual DIMM slot. Individual memory clock signals should never be shared between two DIMM slots.

A typical two slot DDR2 DIMM design therefore has six differential memory clock pairs —three to the first DIMM and three to the second DIMM. All six memory clock pairs must be delay matched to each other to  $\pm 25$  mils ( $\pm 0.635$  mm) and  $\pm 10$  mils ( $\pm 0.254$  mm) for each CLK to CLK# signal.

You may place a compensation capacitor between each clock pair directly before the DIMM connector, to improve the clock slew rates. As FPGA devices have fully programmable drive strength and slew rate options, this capacitor is usually not required for FPGA design. However, Intel advises that you simulate your specific implementation to ascertain if this capacitor is required or not. If fitted the best value is typically 5 pF.

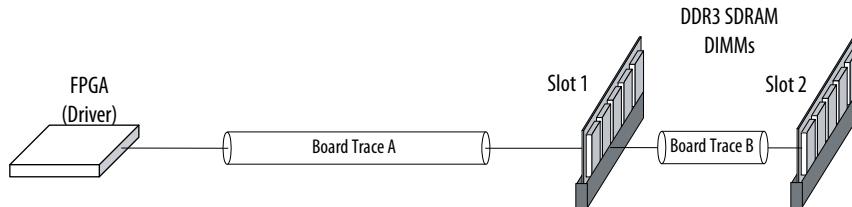
## 3.3. Dual-Slot Unbuffered DDR3 SDRAM

The following topics detail the system implementation of a dual slot unbuffered DDR3 SDRAM interface, operating at up to 400 MHz and 800 Mbps data rates.



The following figure shows a typical DQS, DQ, and DM, and address and command signal topology for a dual-DIMM interface configuration, using the ODT feature of the DDR3 SDRAM components combined with the dynamic OCT features available in Stratix III and Stratix IV devices.

**Figure 46. Multi DIMM DDR3 DQS, DQ, and DM, and Address and Command Termination Topology**



In the above figure, observe the following points:

- Board trace A = 1.9 to 4.5 inches (48 to 115 mm)
- Board trace B = 0.425 inches (10.795 mm)
- This topology to both DIMMs is accurate for DQS, DQ, and DM, and address and command signals
- This topology is not correct for CLK and CLK# and control group signals (CS#, CKE, and ODT), which are always point-to-point single rank only.

### 3.3.1. Comparison of DDR3 and DDR2 DQ and DQS ODT Features and Topology

DDR3 and DDR2 SDRAM systems are quite similar. The physical topology of the data group of signals may be considered nearly identical.

The FPGA end (driver) I/O standard changes from SSTL18 for DDR2 to SSTL15 for DDR3, but all other OCT settings are identical. DDR3 offers enhanced ODT options for termination and drive-strength settings at the memory end of the line.

For more information, refer to the DDR3 SDRAM ODT matrix for writes and the DDR3 SDRAM ODT matrix for reads tables in the *DDR2 and DDR3 SDRAM Board Design Guidelines* chapter.

#### Related Information

[DDR2, DDR3, and DDR4 SDRAM Board Design Guidelines](#) on page 81

### 3.3.2. Dual-DIMM DDR3 Clock, Address, and Command Termination and Topology

One significant difference between DDR3 and DDR2 DIMM based interfaces is the address, command and clock signals. DDR3 uses a daisy chained based architecture when using JEDEC standard modules.

The address, command, and clock signals are routed on each module in a daisy chain and feature a fly-by termination on the module. Impedance matching is required to make the dual-DIMM topology work effectively—40 to 50-ohm traces should be targeted on the main board.



### 3.3.2.1. Address and Command Signals

Two UDIMMs result in twice the effective load on the address and command signals, which reduces the slew rate and makes it more difficult to meet setup and hold timing ( $t_{IS}$  and  $t_{IH}$ ). However, address and command signals operate at half the interface rate and are SDR. Hence a 400-Mbps data rate equates to an address and command fundamental frequency of 100 MHz.

### 3.3.2.2. Control Group Signals

The control group signals (chip Select CS#, clock enable CKE, and ODT) are only ever single rank. A dual-rank capable DDR3 DIMM slot has two copies of each signal, and a dual-DIMM slot interface has four copies of each signal.

The signal quality of these signals is identical to a single rank case. The control group of signals, are always 1T regardless of whether you implement a full-rate or half-rate design. As the signals are also SDR, the control group signals operate at a maximum frequency of  $0.5 \times$  the data rate. For example, in a 400 MHz design, the maximum control group frequency is 200 MHz.

### 3.3.2.3. Clock Group Signals

Like the control group signals, the clock signals in DDR3 SDRAM are only ever single rank loaded. A dual-rank capable DDR3 DIMM slot has two copies of the signal, and a dual-slot interface has four copies of the `mem_clk` and `mem_clk_n` signals.

For more information about a DDR3 two-DIMM system design, refer to Micron *TN-41-08: DDR3 Design Guide for Two-DIMM Systems*.

### 3.3.3. FPGA OCT Features

Many FPGA devices offer OCT. Depending on the chosen device family, series (output), parallel (input) or dynamic (bidirectional) OCT may be supported.

For more information specific to your device family, refer to the respective I/O features chapter in the relevant device handbook.

Use series OCT in place of the near-end series terminator typically used in both Class I or Class II termination schemes that both DDR2 and DDR3 type interfaces use.

Use parallel OCT in place of the far-end parallel termination typically used in Class I termination schemes on unidirectional input only interfaces. For example, QDR-II type interfaces, when the FPGA is at the far end.

Use dynamic OCT in place of both the series and parallel termination at the FPGA end of the line. Typically use dynamic OCT for DQ and DQS signals in both DDR2 and DDR3 type interfaces. As the parallel termination is dynamically disabled during writes, the FPGA driver only ever drives into a Class I transmission line. When combined with dynamic ODT at the memory, a truly dynamic Class I termination scheme exists where both reads and writes are always fully Class I terminated in each direction. Hence, you can use a fully dynamic bidirectional Class I termination scheme instead of a static discretely terminated Class II topology, which saves power, printed circuit board (PCB) real estate, and component cost.



### 3.3.3.1. Arria V, Cyclone V, Stratix III, Stratix IV, and Stratix V Devices

Arria® V, Cyclone® V, Stratix III, Stratix IV, and Stratix V devices feature full dynamic OCT termination capability. Intel advises that you use this feature combined with the SDRAM ODT to simplify PCB layout and save power.

### 3.3.3.2. Arria II GX Devices

Arria II GX devices do not support dynamic OCT. Intel recommends that you use series OCT with SDRAM ODT. Use parallel discrete termination at the FPGA end of the line when necessary.

For more information, refer to the *DDR2 and DDR3 SDRAM Board Design Guidelines* chapter.

#### Related Information

[DDR2, DDR3, and DDR4 SDRAM Board Design Guidelines](#) on page 81

## 3.4. Document Revision History

Date	Version	Changes
May 2017	2017.5.08	Rebranded as Intel.
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	Maintenance release.
November 2015	2015.11.02	Maintenance release.
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	Removed <i>Address and Command Signals</i> section from <i>Dual-DIMM DDR2 Clock, Address, and Command Termination and Topology</i>
December 2013	2013.12.16	<ul style="list-style-type: none"> <li>• Reorganized content.</li> <li>• Consolidated General Layout Guidelines.</li> <li>• Removed references to ALTMEMPHY.</li> <li>• Removed references to Stratix II devices.</li> </ul>
June 2012	4.1	Added Feedback icon.
November 2011	4.0	Added Arria V and Cyclone V information.
June 2011	3.0	Added Stratix V information.
December 2010	2.1	Maintenance update.
July 2010	2.0	Updated Arria II GX information.
April 2010	1.0	Initial release.

## 4. LPDDR2 and LPDDR3 SDRAM Board Design Guidelines

The following topics provide guidelines to improve your system's signal integrity and to successfully implement an LPDDR2 or LPDDR3 SDRAM interface in your system.

### 4.1. LPDDR2 Guidance

The LPDDR2 SDRAM Controller with UniPHY intellectual property (IP) enables you to implement LPDDR2 SDRAM interfaces with Arria® V and Cyclone® V devices.

The following topics focus on key factors that affect signal integrity:

- I/O standards
- LPDDR2 configurations
- Signal terminations
- Printed circuit board (PCB) layout guidelines

#### I/O Standards

LPDDR2 SDRAM interface signals use HSUL-12 JEDEC I/O signaling standards, which provide low power and low emissions. The HSUL-12 JEDEC I/O standard is mainly for point-to-point unterminated bus topology. This standard eliminates the need for external series or parallel termination resistors in LPDDR2 SDRAM implementation. With this standard, termination power is greatly reduced and programmable drive strength is used to match the impedance.

To select the most appropriate standard for your interface, refer to the the *Device Datasheet for Arria V Devices* chapter in the *Arria V Device Handbook*, or the *Device Datasheet for Cyclone V Devices* chapter in the *Cyclone V Device Handbook*.

#### Related Information

- [Arria V Device Datasheet](#)
- [Cyclone V Device Datasheet](#)

#### 4.1.1. LPDDR2 SDRAM Configurations

The LPDDR2 SDRAM Controller with UniPHY IP supports interfaces for LPDDR2 SDRAM with a single device, and multiple devices up to a maximum width of 32 bits.

When using multiple devices, a balanced-T topology is recommended for the signal connected from single point to multiple point, to maintain equal flight time.

You should connect a 200 ohm differential termination resistor between CK/CK# in multiple device designs as shown in the second figure below, to maintain an effective resistance of 100 ohms.

---

Intel Corporation. All rights reserved. Intel, the Intel logo, Altera, Arria, Cyclone, Empirion, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

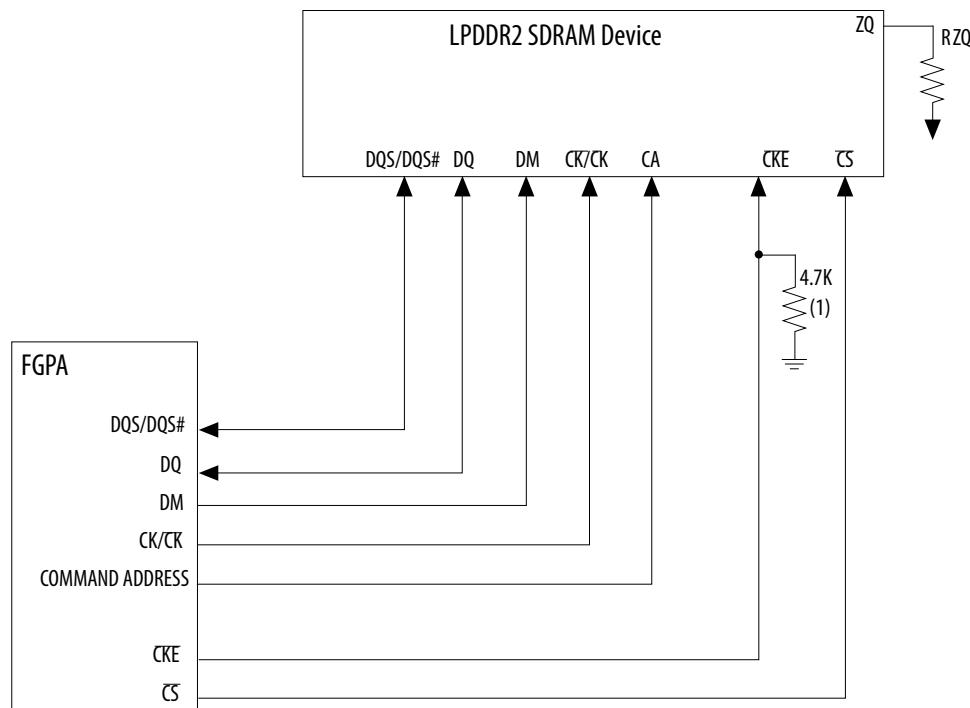
\*Other names and brands may be claimed as the property of others.



You should also simulate your multiple device design to obtain the optimum drive strength settings and ensure correct operation.

The following figure shows the main signal connections between the FPGA and a single LPDDR2 SDRAM component.

**Figure 47. Configuration with a Single LPDDR2 SDRAM Component**

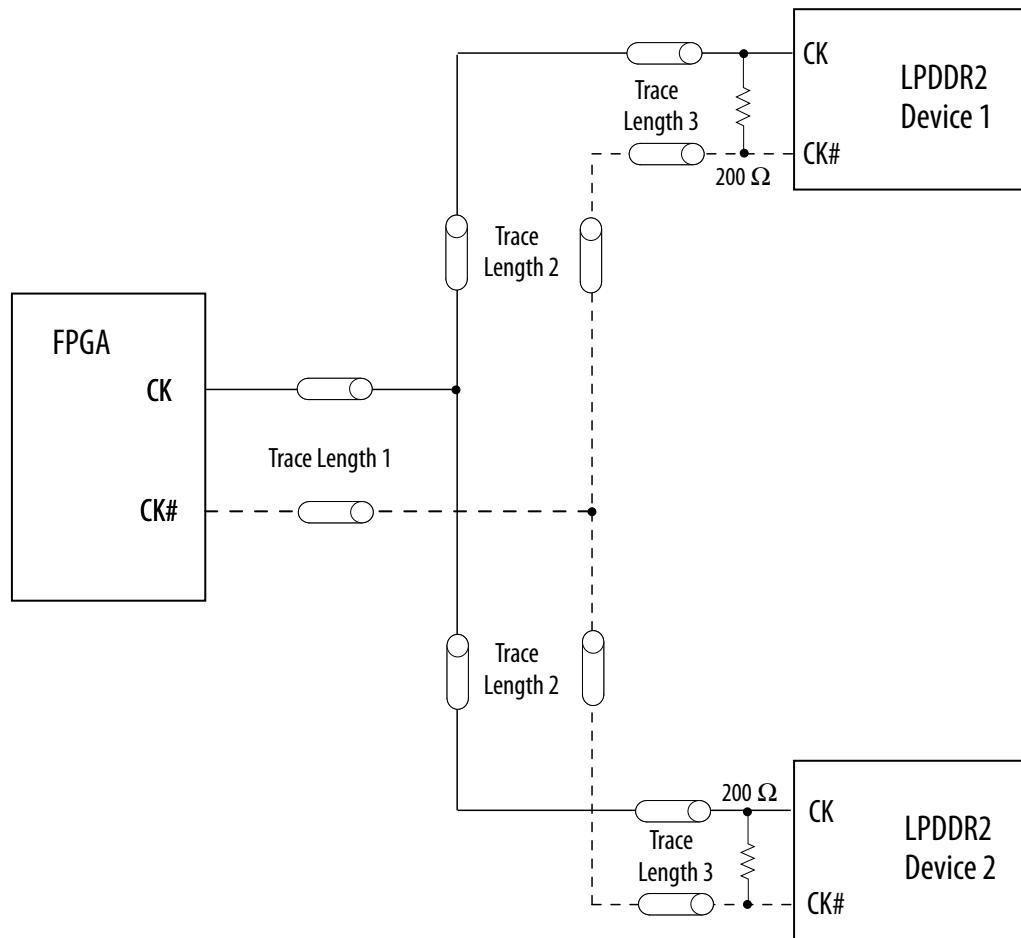


Note to Figure:

1. Use external discrete termination, as shown for CKE, but you may require a pull-down resistor to GND. Refer to the LPDDR2 SDRAM device data sheet for more information about LPDDR2 SDRAM power-up sequencing.

The following figure shows the differential resistor placement for CK/CK# for multi-point designs.

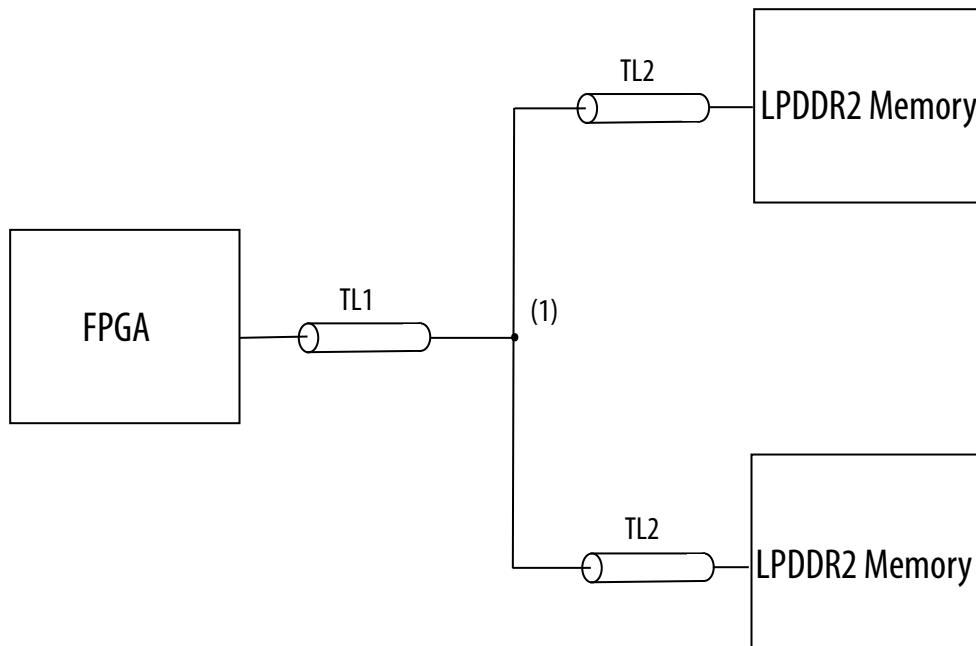
**Figure 48. CK Differential Resistor Placement for Multi Point Design**



Note to Figure:

1. Place 200-ohm differential resistors near the memory devices at the end of the last board trace segments.

The following figure shows the detailed balanced topology recommended for the address and command signals in the multi-point design.

**Figure 49. Address Command Balanced-T Topology**

Notes to Figure:

1. Split the trace close to the memory devices to minimize signal reflections and impedance nonuniformity.
2. Keep the TL2 traces as short as possible, so that the memory devices appear as a single load.

#### 4.1.2. OCT Signal Terminations for Arria V and Cyclone V Devices

Arria V and Cyclone V devices offer OCT technology. The following table lists the extent of OCT support for each device.

**Table 38. On-Chip Termination Schemes**

Termination Scheme	I/O Standard	Arria V and Cyclone V
On-Chip Series Termination without Calibration	HSUL-12	34/40/48/60/80
On-Chip Series Termination with Calibration	HSUL-12	34/40/48/60/80

On-chip series ( $R_S$ ) termination supports output buffers, and bidirectional buffers only when they are driving output signals. LPDDR2 SDRAM interfaces have bidirectional data paths. The UniPHY IP uses series OCT for memory writes but no parallel OCT for memory reads because Arria V and Cyclone V support only on-chip series termination in the HSUL-12 I/O standard.

For Arria V and Cyclone V devices, the HSUL-12 I/O calibrated terminations are calibrated against 240 ohm 1% resistors connected to the  $R_{ZQ}$  pins in an I/O bank with the same  $V_{CCIO}$  as the LPDDR2 interface.

Calibration occurs at the end of the device configuration.



LPDDR2 SDRAM memory components have a ZQ pin which connects through a resistor  $R_{ZQ}$  (240 ohm) to ground. The output signal impedances for LPDDR2 SDRAM are 34.3 ohm, 40 ohm, 48 ohm, 60 ohm, 80 ohm, and 120 ohm. The output signal impedance is set by mode register during initialization. Refer to the LPDDR2 SDRAM device data sheet for more information.

For information about OCT, refer to the *I/O Features in Arria V Devices* chapter in the *Arria V Device Handbook*, or the *I/O Features in Cyclone V Devices* chapter in the *Cyclone V Device Handbook*.

The following section shows HyperLynx simulation eye diagrams to demonstrate signal termination options. Intel strongly recommends signal terminations to optimize signal integrity and timing margins, and to minimize unwanted emissions, reflections, and crosstalk.

All of the eye diagrams shown in this section are for a 50 ohm trace with a propagation delay of 509 ps which is approximately a 2.8-inch trace on a standard FR4 PCB. The signal I/O standard is HSUL-12.

The eye diagrams in this section show the best case achievable and do not take into account PCB vias, crosstalk, and other degrading effects such as variations in the PCB structure due to manufacturing tolerances.

*Note:* Simulate your design to ensure correct operation.

#### Related Information

- [I/O Features in Arria V Devices](#)
- [I/O Features in Cyclone V Devices](#)

#### 4.1.2.1. Outputs from the FPGA to the LPDDR2 Component

The following output signals are from the FPGA to the LPDDR2 SDRAM component:

- write data (DQ)
- data mask (DM)
- data strobe (DQS/DQS#)
- command address
- command (CS, and CKE)
- clocks (CK/CK#)

No far-end memory termination is needed when driving output signals from FPGA to LPDDR2 SDRAM. Cyclone V and Arria V devices offer the OCT series termination for impedance matching.

#### 4.1.2.2. Input to the FPGA from the LPDDR2 SDRAM Component

The LPDDR2 SDRAM component drives the following input signals into the FPGA:

- read data
- DQS



LPDDR2 SDRAM provides the flexibility to adjust drive strength to match the impedance of the memory bus, eliminating the need for termination voltage (VTT) and series termination resistors.

The programmable drive strength options are 34.3 ohms, 40 ohms (default), 48 ohms, 60 ohms, 80 ohms, and 120 ohms. You must perform board simulation to determine the best option for your board layout.

**Note:** By default, LPDDR2 SDRAM UniPHY IP uses 40 ohm drive strength.

#### 4.1.2.3. Termination Schemes

The following table lists the recommended termination schemes for major LPDDR2 SDRAM memory interface signals.

These signals include data (DQ), data strobe (DQS), data mask (DM), clocks (CK, and CK#), command address (CA), and control (CS#, and CKE).

**Table 39. Termination Recommendations for Arria V and Cyclone V Devices**

Signal Type	HSUL-12 Standard <sup>(1) (2)</sup>	Memory End Termination
DQS/DQS#	R34 CAL	ZQ40
Data (Write)	R34 CAL	-
Data (Read)	-	ZQ40
Data Mask (DM)	R34 CAL	-
CK/CK# Clocks	R34 CAL	$\times 1 = -$ <sup>(4)</sup> $\times 2 = 200$ -ohm□Differential <sup>(5)</sup>
Command Address (CA),	R34 CAL	-
Chip Select (CS#)	R34 CAL	-
Clock Enable (CKE) <sup>(3)</sup>	R34 CAL	4.7 K-ohm□parallel to GND

Notes to Table:

1. R is effective series output impedance.
2. CAL is OCT with calibration.
3. Intel recommends that you use a 4.7 K-ohm□parallel to GND if your design meets the power sequencing requirements of the LPDDR2 SDRAM component. Refer to the LPDDR2 SDRAM data sheet for further information.
4.  $\times 1$  is a single-device load.
5.  $\times 2$  is a double-device load. An alternative option is to use a 100 -ohm differential termination at the trace split.

**Note:** The recommended termination schemes in the above table are based on 2.8 inch maximum trace length analysis. You may add the external termination resistor or adjust the drive strength to improve signal integrity for longer trace lengths.

Recommendations for external termination are as follows:

- Class I termination (50 ohms parallel to VTT at the memory end) — Unidirectional signal (Command Address, control, and CK/CK# signals)
- Class II termination (50 ohms parallel to VTT at both ends) — Bidirectional signal ( DQ and DQS/DQS# signal)

Intel recommends that you simulate your design to ensure good signal integrity.



### 4.1.3. General Layout Guidelines

The following table lists general board design layout guidelines. These guidelines are Intel recommendations, and should not be considered as hard requirements. You should perform signal integrity simulation on all the traces to verify the signal integrity of the interface. You should extract the slew rate and propagation delay information, enter it into the IP and compile the design to ensure that timing requirements are met.

**Table 40. General Layout Guidelines**

Parameter	Guidelines
Impedance	<ul style="list-style-type: none"><li>All unused via pads must be removed, because they cause unwanted capacitance.</li><li>Trace impedance plays an important role in the signal integrity. You must perform board level simulation to determine the best characteristic impedance for your PCB. For example, it is possible that for multi rank systems 40 ohms could yield better results than a traditional 50 ohm characteristic impedance.</li></ul>
Decoupling Parameter	<ul style="list-style-type: none"><li>Use 0.1 uF in 0402 size to minimize inductance</li><li>Make VTT voltage decoupling close to termination resistors</li><li>Connect decoupling caps between VTT and ground</li><li>Use a 0.1 uF cap for every other VTT pin and 0.01 uF cap for every VDD and VDDQ pin</li><li>Verify the capacitive decoupling using the Intel Power Distribution Network Design Tool</li></ul>
Power	<ul style="list-style-type: none"><li>Route GND and V<sub>CC</sub> as planes</li><li>Route VCCIO for memories in a single split plane with at least a 20-mil (0.020 inches, or 0.508 mm) gap of separation</li><li>Route VTT as islands or 250-mil (6.35-mm) power traces</li><li>Route oscillators and PLL power as islands or 100-mil (2.54-mm) power traces</li></ul>
General Routing	All specified delay matching requirements include PCB trace delays, different layer propagation velocity variance, and crosstalk. To minimize PCB layer propagation variance, Intel recommends that signals from the same net group always be routed on the same layer. <ul style="list-style-type: none"><li>Use 45° angles (<i>not</i> 90° corners)</li><li>Avoid T-Junctions for critical nets or clocks</li><li>Avoid T-junctions greater than 250 mils (6.35 mm)</li><li>Disallow signals across split planes</li><li>Restrict routing other signals close to system reset signals</li><li>Avoid routing memory signals closer than 0.025 inch (0.635 mm) to PCI or system clocks</li></ul>

#### Related Information

[Power Distribution Network Design Tool](#)

### 4.1.4. LPDDR2 Layout Guidelines

The following table lists the LPDDR2 SDRAM general routing layout guidelines.

**Note:** The following layout guidelines include several +/- length-based rules. These length-based guidelines are for first order timing approximations if you cannot simulate the actual delay characteristics of your PCB implementation. They do not include any margin for crosstalk. Intel recommends that you get accurate time base skew numbers when you simulate your specific implementation.

**Table 41. LPDD2 Layout Guidelines**

Parameter	Guidelines
General Routing	<ul style="list-style-type: none"> <li>If you must route signals of the same net group on different layers with the same impedance characteristic, simulate your worst case PCB trace tolerances to ascertain actual propagation delay differences. Typical layer to layer trace delay variations are of 15 ps/inch order.</li> <li>Avoid T-junctions greater than 75 ps (approximately 25 mils, 6.35 mm).</li> <li>Match all signals within a given DQ group with a maximum skew of <math>\pm 10</math> ps and route on the same layer.</li> </ul>
Clock Routing	<ul style="list-style-type: none"> <li>Route clocks on inner layers with outer-layer run lengths held to under 150 ps.</li> <li>These signals should maintain a 10-mil (0.254 mm) spacing from other nets.</li> <li>Clocks should maintain a length-matching between clock pairs of <math>\pm 5</math> ps.</li> <li>Differential clocks should maintain a length-matching between P and N signals of <math>\pm 2</math> ps.</li> <li>Space between different clock pairs should be at least three times the space between the traces of a differential pair.</li> </ul>
Address and Command Routing	<ul style="list-style-type: none"> <li>To minimize crosstalk, route address, and command signals on a different layer than the data and data mask signals.</li> <li>Do not route the differential clock (CK/CK#) and clock enable (CKE) signals close to the address signals.</li> </ul>
External Memory Routing Rules	<ul style="list-style-type: none"> <li>Apply the following parallelism rules for the LPDDR2 SDRAM data groups: <ul style="list-style-type: none"> <li>4 mils for parallel runs &lt; 0.1 inch (approximately 1x spacing relative to plane distance).</li> <li>5 mils for parallel runs &lt; 0.5 inch (approximately 1x spacing relative to plane distance).</li> <li>10 mils for parallel runs between 0.5 and 1.0 inches (approximately 2x spacing relative to plane distance).</li> <li>15 mils for parallel runs between 1.0 and 2.8 inch (approximately 3x spacing relative to plane distance).</li> </ul> </li> <li>Apply the following parallelism rules for the address/command group and clocks group: <ul style="list-style-type: none"> <li>4 mils for parallel runs &lt; 0.1 inch (approximately 1x spacing relative to plane distance)</li> <li>10 mils for parallel runs &lt; 0.5 inch (approximately 2x spacing relative to plane distance)</li> <li>15 mils for parallel runs between 0.5 and 1.0 inches (approximately 3x spacing relative to plane distance)</li> <li>20 mils for parallel runs between 1.0 and 2.8 inches (approximately 4x spacing relative to plane distance)</li> </ul> </li> </ul>
Maximum Trace Length	<ul style="list-style-type: none"> <li>Keep traces as short as possible. The maximum trace length of all signals from the FPGA to the LPDDR2 SDRAM components should be less than 509 ps. Intel recommends that you simulate your design to ensure good signal integrity.</li> </ul>
Trace Matching Guidance	<p>The following layout approach is recommended, based on the preceding guidelines:</p> <ol style="list-style-type: none"> <li>Route the differential clocks (CK/CK#) and data strobe (DQS/DQS#) with a length-matching between P and N signals of <math>\pm 2</math> ps.</li> <li>Route the DQS /DQS# associated with a DQ group on the same PCB layer. Match these DQS pairs to within <math>\pm 5</math> ps.</li> <li>Set the DQS/DQS# as the target trace propagation delay for the associated data and data mask signals.</li> <li>Route the data and data mask signals for the DQ group ideally on the same layer as the associated DQS/DQS# to within <math>\pm 10</math> ps skew of the target DQS/DQS#.</li> <li>Route the CK/CK# clocks and set as the target trace propagation delays for the DQ group. Match the CK/CK# clock to within <math>\pm 50</math> ps of all the DQS/DQS#.</li> <li>Route the address/control signal group (address, CS, CKE) ideally on the same layer as the CK/CK# clocks, to within <math>\pm 20</math> ps skew of the CK/CK# traces.</li> </ol>

*continued...*

Parameter	Guidelines
	<p>This layout approach provides a good starting point for a design requirement of the highest clock frequency supported for the LPDDR2 SDRAM interface.</p> <p><i>Note:</i> You should create your project in the Quartus® Prime software with a fully implemented LPDDR2 interface, and observe the interface timing margins to determine the actual margins for your design.</p>

Although the recommendations in this chapter are based on simulations, you can apply the same general principles when determining the best termination scheme, drive strength setting, and loading style to any board design. Even armed with this knowledge, it is still critical that you simulate your design with IBIS or HSPICE models, to determine the quality of signal integrity in your design.

#### Related Information

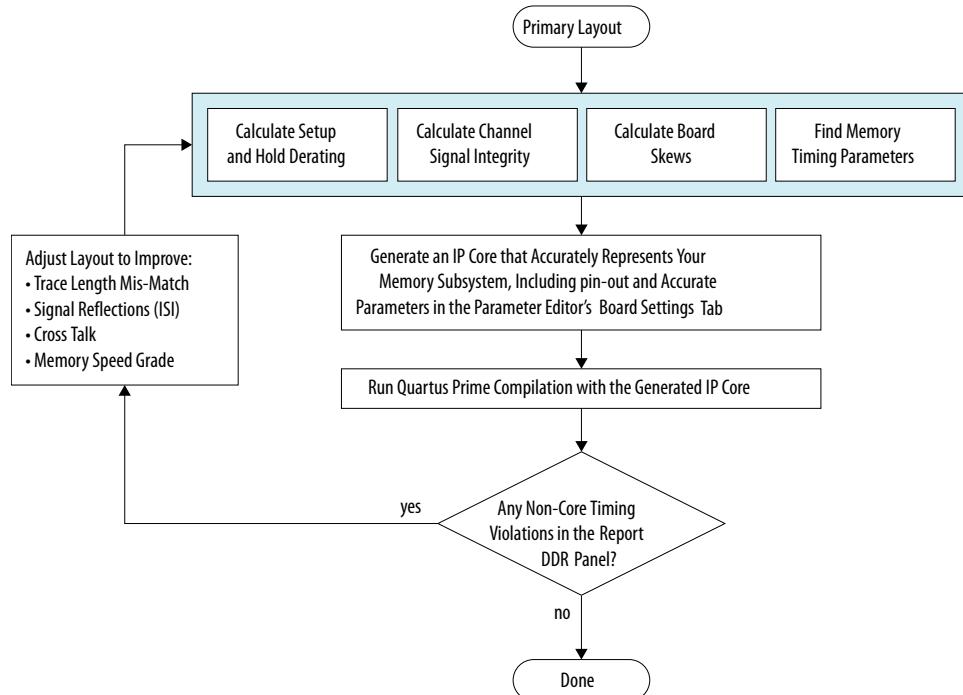
[Intel Power Distribution Network \(PDN\) Design tool](#)

## 4.2. LPDDR3 Guidance

The LPDDR3 SDRAM Controller intellectual property (IP) enables you to implement LPDDR3 SDRAM interfaces with Arria® 10 and Stratix® 10 devices.

For all practical purposes, you can regard the TimeQuest timing analyzer's report on your memory interface as definitive for a given set of memory and board timing parameters. You can find timing information under **Report DDR** in TimeQuest and on the **Timing Analysis** tab in the parameter editor.

The following flowchart illustrates the recommended process to follow during the design phase, to determine timing margin and make iterative improvements to your design.





## 4.2.1. Signal Integrity, Board Skew, and Board Setting Parameters

### Channel Signal Integrity

For information on determining channel signal integrity, refer to the wiki page: [http://www.alterawiki.com/wiki/Arria\\_10\\_EMIF\\_Simulation\\_Guidance](http://www.alterawiki.com/wiki/Arria_10_EMIF_Simulation_Guidance).

### Board Skew

For information on calculating board skew parameters, refer to *Implementing and Parameterizing Memory IP*. The Board Skew Parameter Tool is an interactive tool that can help you calculate board skew parameters if you know the absolute delay values for all the memory related traces.

### Arria 10 Board Setting Parameters

For Board Setting and layout approach information for Arria 10 devices, refer to the wiki page: [http://www.alterawiki.com/wiki/Arria\\_10\\_EMIF\\_Simulation\\_Guidance](http://www.alterawiki.com/wiki/Arria_10_EMIF_Simulation_Guidance).

## 4.2.2. LPDDR3 Layout Guidelines

The following table lists the LPDDR3 SDRAM general routing layout guidelines.

**Table 42. LPDDR3 Layout Guidelines**

Parameter	Guidelines
Max Length Discrete	500 ps.
Data Group Skew	Match DM and DQ within 5 ps of DQS.
Address/Command vs Clock Skew	Match Address/Command signals within 10 ps of mem CK.
Package Skew Matching	Yes.
Clock matching	<ul style="list-style-type: none"> <li>• 2 ps within a clock pair</li> <li>• 5 ps between clock pairs</li> </ul>
Spacing Guideline Data/Data Strobe/Address/Command	3H spacing between any Data and Address/Command traces, where H is distance to the nearest return path.
Spacing Guideline Mem Clock	5H spacing between mem clock and any other signal, where H is distance to the nearest return path.

## 4.2.3. Package Deskew

Trace lengths inside the device package are not uniform for all package pins. The nonuniformity of package traces can affect system timing for high frequencies. In the Quartus II software version 12.0 and later, and the Quartus Prime software, a package deskew option is available.

If you do not enable the package deskew option, the Quartus Prime software uses the package delay numbers to adjust skews on the appropriate signals; you do not need to adjust for package delays on the board traces. If you do enable the package



deskew option, the Quartus Prime software does not use the package delay numbers for timing analysis, and you must deskew the package delays with the board traces for the appropriate signals for your design.

#### 4.2.3.1. DQ/DQS/DM Deskew

To get the package delay information, follow these steps:

1. Select the **FPGA DQ/DQS Package Skews Deskewed on Board** checkbox on the **Board Settings** tab of the parameter editor.
2. Generate your IP.
3. Instantiate your IP in the project.
4. Run **Analysis and Synthesis** in the Quartus Prime software. (Skip this step if you are using an Arria 10 device.)
5. Run the `<core_name>.p0_pin_assignment.tcl` script. (Skip this step if you are using an Arria 10 device.)
6. Compile your design.
7. Refer to the **All Package Pins** compilation report, or find the pin delays displayed in the `<core_name>.pin` file.

#### 4.2.3.2. Address and Command Deskew

Deskew address and command delays as follows:

1. Select the **FPGA Address/Command Package Skews Deskewed on Board** checkbox on the **Board Settings** tab of the parameter editor.
2. Generate your IP.
3. Instantiate your IP in the project.
4. Run **Analysis and Synthesis** in the Quartus Prime software. (Skip this step if you are using an Arria 10 device.)
5. Run the `<core_name>.p0_pin_assignment.tcl` script. (Skip this step if you are using an Arria 10 device.)
6. Compile your design.
7. Refer to the **All Package Pins** compilation report, or find the pin delays displayed in the `<core_name>.pin` file.

#### 4.2.3.3. Package Deskew Recommendations for Arria 10 and Stratix 10 Devices

The following table shows package deskew recommendations for all protocols supported on Arria 10 devices.

As operating frequencies increase, it becomes increasingly critical to perform package deskew. The frequencies listed in the table are the *minimum* frequencies for which you must perform package deskew.

If you plan to use a listed protocol at the specified frequency or higher, you must perform package deskew. For example, you must perform package deskew if you plan to use dual-rank DDR4 at 800 MHz or above.



Protocol	Minimum Frequency (MHz) for Which to Perform Package Deskew		
	Single Rank	Dual Rank	Quad Rank
DDR4	933	800	667
DDR3	933	800	667
LPDDR3	667	533	Not required
QDR IV	933	Not applicable	Not applicable
RLDRAM 3	933	667	Not applicable
RLDRAM II	Not required	Not applicable	Not applicable
QDR II, II+, II+ Xtreme	Not required	Not applicable	Not applicable

#### 4.2.3.4. Deskew Example

Consider an example where you want to deskew an interface with 4 DQ pins, 1 DQS pin, and 1 DQSn pin.

Let's assume an operating frequency of 667 MHz, and the package lengths for the pins reported in the **.pin** file as follows:

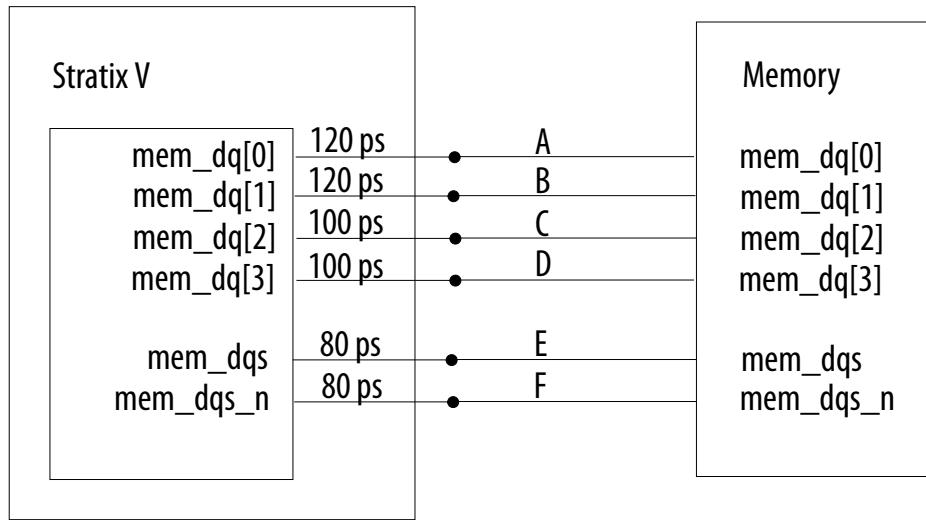
```

dq[ 0 ] = 120 ps
dq[ 1 ] = 120 ps
dq[ 2 ] = 100 ps
dq[ 3 ] = 100 ps
dqs     = 80 ps
dqs_n   = 80 ps

```

The following figure illustrates this example.

**Figure 50. Deskew Example**

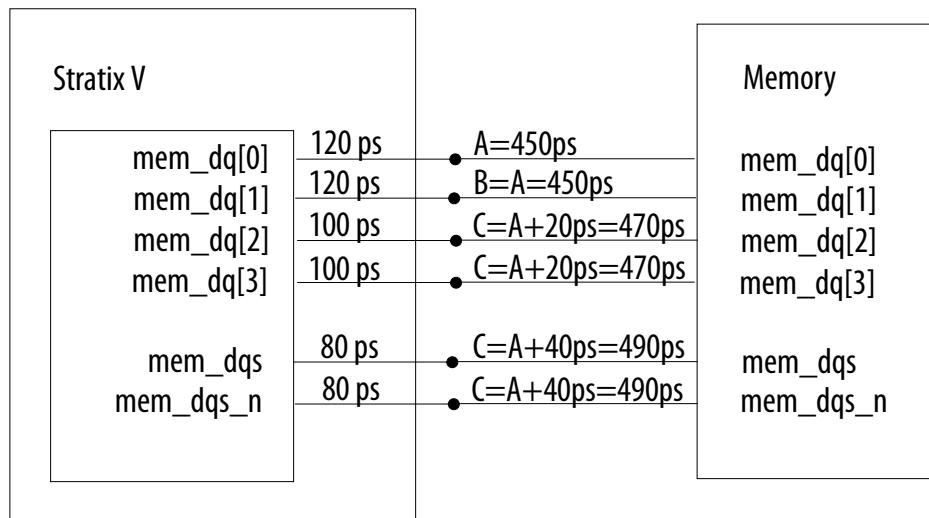


When you perform length matching for all the traces in the DQS group, you must take package delays into consideration. Because the package delays of traces A and B are 40 ps longer than the package delays of traces E and F, you would need to make the board traces for E and F 40 ps longer than the board traces for A and B.

A similar methodology would apply to traces C and D, which should be 20 ps longer than the lengths of traces A and B.

The following figure shows this scenario with the length of trace A at 450 ps.

**Figure 51. Deskew Example with Trace Delay Calculations**



When you enter the board skews into the Board Settings tab of the DDR3 parameter editor, you should calculate the board skew parameters as the sums of board delay and corresponding package delay. If a pin does not have a package delay (such as address and command pins), you should use the board delay only.

The example of the preceding figure shows an ideal case where board skews are perfectly matched. In reality, you should allow plus or minus 10 ps of skew mismatch within a DQS group (DQ/DQS/DM).

#### 4.2.3.5. Package Migration

Package delays can be different for the same pin in different packages. If you want to use multiple migratable packages in your system, you should compensate for package skew as described in this topic. The information in this topic applies to Arria 10, Stratix V, and Stratix 10 devices.

Assume two migratable packages, device A and device B, and that you want to compensate for the board trace lengths for device A. Follow these steps:

1. Compile your design for device A, with the Package Skew option enabled.
2. Note the skews in the `<core_name>.pin` file for device A. Deskew these package skews with board trace lengths as described in the preceding examples.
3. Recompile your design for device A.
4. For Device B open the parameter editor and deselect Package Deskeew option.
5. Calculate board skew parameters only taking into account the board traces for Device B and enter that value into the parameter editor for Device B.
6. Regenerate the IP and recompile the design for Device B.
7. Verify that timing requirements are met for both device A and device B.



## 4.3. Document Revision History

Date	Version	Changes
May 2017	2017.5.08	<ul style="list-style-type: none"> <li>Added Stratix 10 to <i>Package Deskew</i> and <i>Package Migration</i> sections.</li> <li>Rebranded as Intel.</li> </ul>
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	<ul style="list-style-type: none"> <li>Changed recommended value of skew mismatch in <i>Deskew Example</i> topic.</li> </ul>
November 2015	2015.11.02	<ul style="list-style-type: none"> <li>Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i>.</li> <li>Added content for LPDDR3.</li> </ul>
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	<ul style="list-style-type: none"> <li>Removed millimeter approximations from lengths expressed in picoseconds in <i>LPDDR2 Layout Guidelines</i> table.</li> <li>Minor formatting fixes in <i>LPDDR2 Layout Guidelines</i> table.</li> </ul>
December 2013	2013.12.16	Consolidated General Layout Guidelines.
November 2012	1.0	Initial release.

## 5. RLDRAM II and RLDRAM 3 Board Design Guidelines

---

The following topics provide layout guidelines for you to improve your system's signal integrity and to successfully implement an RLDRAM II or RLDRAM 3 interface.

The RLDRAM II Controller with UniPHY intellectual property (IP) enables you to implement common I/O (CIO) RLDRAM II interfaces with Arria® V, Stratix® III, Stratix IV, and Stratix V devices. The RLDRAM 3 UniPHY IP enables you to implement CIO RLDRAM 3 interfaces with Stratix V and Arria V GZ devices. You can implement separate I/O (SIO) RLDRAM II or RLDRAM 3 interfaces with the ALTDQ\_DQS or ALTDQ\_DQS2 IP cores.

The following topics focus on the following key factors that affect signal integrity:

- I/O standards
- RLDRAM II and RLDRAM 3 configurations
- Signal terminations
- Printed circuit board (PCB) layout guidelines

### I/O Standards

RLDRAM II interface signals use one of the following JEDEC I/O signalling standards:

- HSTL-15—provides the advantages of lower power and lower emissions.
- HSTL-18—provides increased noise immunity with slightly greater output voltage swings.

RLDRAM 3 interface signals use the following JEDEC I/O signalling standards: HSTL 1.2 V and SSTL-12.

To select the most appropriate standard for your interface, refer to the following:

- *Device Data Sheet for Arria II Devices* chapter in the *Arria II Device Handbook*
- *Device Data Sheet for Arria V Devices* chapter in the *Arria V Device Handbook*
- *Stratix III Device Data Sheet: DC and Switching Characteristics* chapter in the *Stratix III Device Handbook*
- *DC and Switching Characteristics for Stratix IV Devices* chapter in the *Stratix IV Device Handbook*
- *DC and Switching Characteristics for Stratix V Devices* chapter in the *Stratix V Device Handbook*

The RLDRAM II Controller with UniPHY IP defaults to HSTL 1.8 V Class I outputs and HSTL 1.8 V inputs. The RLDRAM 3 UniPHY IP defaults to HSTL 1.2 V Class I outputs and HSTL 1.2 V inputs.

**Note:** The default for RLDRAM 3 changes from Class I to Class II, supporting up to 933 MHz, with the release of the Quartus II software version 12.1 SP1.

Intel Corporation. All rights reserved. Intel, the Intel logo, Altera, Arria, Cyclone, Empirion, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.

ISO  
9001:2015  
Registered



### Related Information

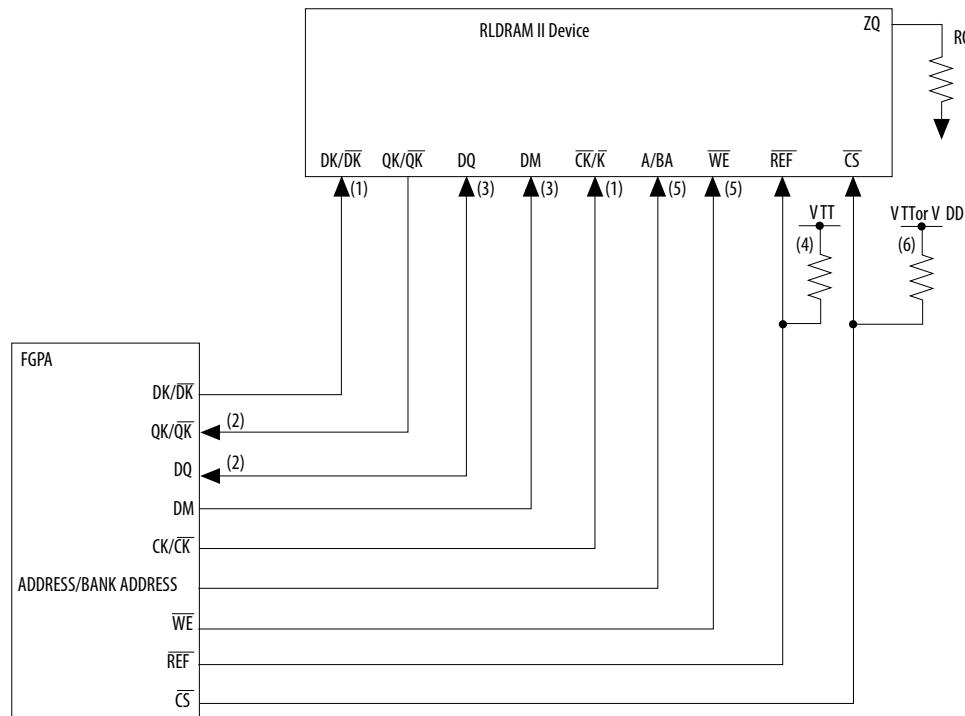
- Device Data Sheet for Arria II Devices
- Device Data Sheet for Arria V Devices
- Stratix III Device Data Sheet: DC and Switching Characteristics
- DC and Switching Characteristics for Stratix IV Devices
- DC and Switching Characteristics for Stratix V Devices

## 5.1. RLDRAM II Configurations

The RLDRAM II Controller with UniPHY IP supports CIO RLDRAM II interfaces with one or two devices. With two devices, the interface supports a width expansion configuration up to 72-bits. The termination and layout principles for SIO RLDRAM II interfaces are similar to CIO RLDRAM II, except that SIO RLDRAM II interfaces have unidirectional data buses.

The following figure shows the main signal connections between the FPGA and a single CIO RLDRAM II component.

**Figure 52. Configuration with a Single CIO RLDRAM II Component**



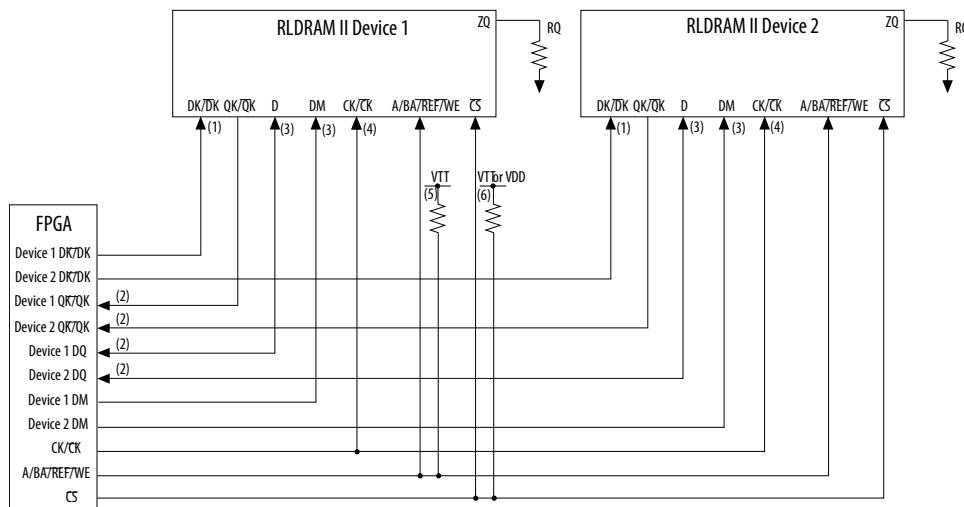
#### Notes to Figure:

1. Use external differential termination on DK/DK# and CK/CK#.
2. Use FPGA parallel on-chip termination (OCT) for terminating QK/QK# and DQ on reads.
3. Use RDRAM II component on-die termination (ODT) for terminating DQ and DM on writes.

4. Use external discrete termination with fly-by placement to avoid stubs.
5. Use external discrete termination for this signal, as shown for REF.
6. Use external discrete termination, as shown for REF, but you may require a pull-up resistor to VDD as an alternative option. Refer to the RLDRAM II device data sheet for more information about RLDRAM II power-up sequencing.

The following figure shows the main signal connections between the FPGA and two CIO RLDRAM II components in a width expansion configuration.

**Figure 53. Configuration with Two CIO RLDRAM II Components in a Width Expansion Configuration**



#### Notes to Figure:

1. Use external differential termination on DK/DK#.
2. Use FPGA parallel on-chip termination (OCT) for terminating QK/QK# and DQ on reads.
3. Use RLDRAM II component on-die termination (ODT) for terminating DQ and DM on writes.
4. Use external dual 200 Ω differential termination.
5. Use external discrete termination at the trace split of the balanced T or Y topology.
6. Use external discrete termination at the trace split of the balanced T or Y topology, but you may require a pull-up resistor to VDD as an alternative option. Refer to the RLDRAM II device data sheet for more information about RLDRAM II power-up sequencing.

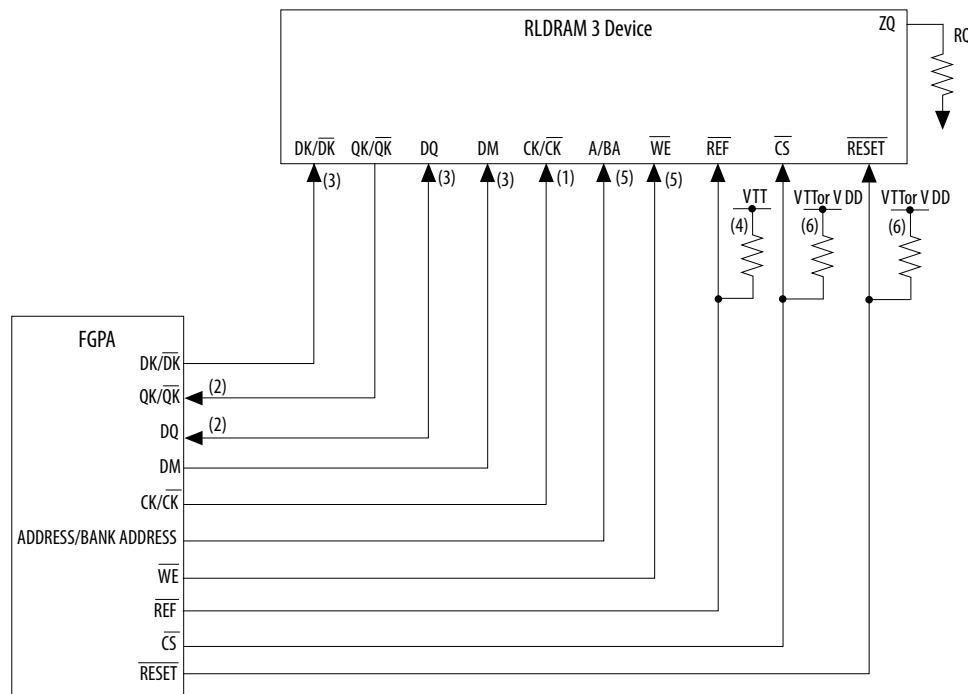
## 5.2. RLDRAM 3 Configurations

The RLDRAM 3 UniPHY IP supports interfaces for CIO RLDRAM 3 with one or two devices. With two devices, the interface supports a width expansion configuration up to 72-bits. The termination and layout principles for SIO RLDRAM 3 interfaces are similar to CIO RLDRAM 3, except that SIO RLDRAM 3 interfaces have unidirectional data buses.



The following figure shows the main signal connections between the FPGA and a single CIO RLDRAM 3 component.

**Figure 54. Configuration with a Single CIO RLDRAM 3 Component**

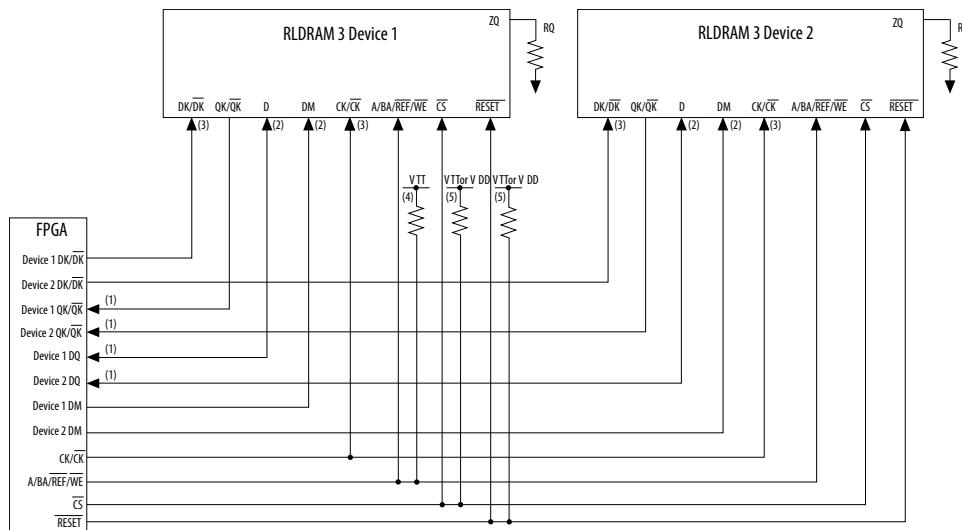


**Notes to Figure:**

1. Use external differential termination on CK/CK#.
2. Use FPGA parallel on-chip termination (OCT) for terminating QK/QK# and DQ on reads.
3. Use RLDRAM 3 component on-die termination (ODT) for terminating DQ, DM, and DK, DK# on writes.
4. Use external discrete termination with fly-by placement to avoid stubs.
5. Use external discrete termination for this signal, as shown for REF.
6. Use external discrete termination, as shown for REF, but you may require a pull-up resistor to VDD as an alternative option. Refer to the RLDRAM 3 device data sheet for more information about RLDRAM 3 power-up sequencing.

The following figure shows the main signal connections between the FPGA and two CIO RLDRAM 3 components in a width expansion configuration.

**Figure 55. Configuration with Two CIO RLDRAM 3 Components in a Width Expansion Configuration**



Notes to Figure:

1. Use FPGA parallel OCT for terminating QK/QK# and DQ on reads.
2. Use RLDRAM 3 component ODT for terminating DQ, DM, and DK on writes.
3. Use external dual 200  $\Omega$  differential termination.
4. Use external discrete termination at the trace split of the balanced T or Y topology.
5. Use external discrete termination at the trace split of the balanced T or Y topology, but you may require a pull-up resistor to VDD as an alternative option. Refer to the RLDRAM 3 device data sheet for more information about RLDRAM 3 power-up sequencing.

### 5.3. Signal Terminations

The following table lists the on-chip series termination ( $R_S$  OCT) and on-chip parallel termination ( $R_T$  OCT) schemes for supported devices.

**Note:** For RLDRAM 3, the default output termination resistance ( $R_S$ ) changes from 50 ohm to 25 ohm with the release of the Quartus II software version 12.1 SP1.

**Table 43. On-Chip Termination Schemes**

Termination Scheme	Class I Signal Standards	FPGA Device	
		Arria II GZ, Stratix III, and Stratix IV	Arria V and Stratix V
		Row/Column I/O	Row/Column I/O
$R_S$ OCT without Calibration	RLDRAM II - HSTL-15 and HSTL-18 RLDRAM 3 - HSTL 1.2 V	50	50
$R_S$ OCT with Calibration	RLDRAM II - HSTL-15 and HSTL-18	50	50 <sup>(1)</sup>

*continued...*



Termination Scheme	Class I Signal Standards	FPGA Device	
		Arria II GZ, Stratix III, and Stratix IV	Arria V and Stratix V
		Row/Column I/O	Row/Column I/O
	RLDRAM 3 - HSTL 1.2 V		
R <sub>T</sub> OCT with Calibration	RLDRAM II - HSTL-15 and HSTL-18 RLDRAM 3 - HSTL 1.2 V	50	50 <sup>(1)</sup>

Note to Table:

1. Although 50-ohms is the recommended option, Stratix V devices offer a wider range of calibrated termination impedances.

RLDRAM II and RLDRAM 3 CIO interfaces have bidirectional data paths. The UniPHY IP uses dynamic OCT on the datapath, which switches between series OCT for memory writes and parallel OCT for memory reads. The termination schemes also follow these characteristics:

- Although 50 -ohm. is the recommended option, Stratix V devices offer a wider range of calibrated termination impedances.
- R<sub>S</sub> OCT supports output buffers.
- R<sub>T</sub> OCT supports input buffers.
- R<sub>S</sub> OCT supports bidirectional buffers only when they are driving output signals.
- R<sub>T</sub> OCT bidirectional buffers only when they are input signals.

For Arria II GZ, Stratix III, and Stratix IV devices, the HSTL Class I I/O calibrated terminations are calibrated against 50-ohm 1% resistors connected to the R<sub>UP</sub> and R<sub>DN</sub> pins in an I/O bank with the same V<sub>CCIO</sub> as the RLDRAM II interface. For Arria V and Stratix V devices, the HSTL Class I I/O calibrated terminations are calibrated against 100-ohm 1% resistors connected to the R<sub>ZQ</sub> pins in an I/O bank with the same V<sub>CCIO</sub> as the RLDRAM II and RLDRAM 3 interfaces.

The calibration occurs at the end of the device configuration.

RLDRAM II and RLDRAM 3 memory components have a ZQ pin that connects through a resistor R<sub>Q</sub> to ground. Typically the RLDRAM II and RLDRAM 3 output signal impedance is a fraction of R<sub>Q</sub>. Refer to the RLDRAM II and RLDRAM 3 device data sheets for more information.

For information about OCT, refer to the following:

- *I/O Features in Arria II Devices* chapter in the *Arria II Device Handbook*
- *I/O Features in Arria V Devices* chapter in the *Arria V Device Handbook*
- *Stratix III Device I/O Features* chapter in the *Stratix III Device Handbook*
- *I/O Features in Stratix IV Devices* chapter in the *Stratix IV Device Handbook*
- *I/O Features in Stratix V Devices* chapter in the *Stratix V Device Handbook*

Intel strongly recommends signal terminations to optimize signal integrity and timing margins, and to minimize unwanted emissions, reflections, and crosstalk.

**Note:** Simulate your design to check your termination scheme.



### Related Information

- [I/O Features in Arria II Devices](#)
- [I/O Features in Arria V Devices](#)
- [Stratix III Device I/O Features](#)
- [I/O Features in Stratix IV Devices](#)
- [I/O Features in Stratix V Devices](#)

### 5.3.1. Input to the FPGA from the RLDRAM Components

The RLDRAM II or RLDRAM 3 component drives the following input signals into the FPGA:

- Read data (DQ on the bidirectional data signals for CIO RLDRAM II and CIO RLDRAM 3).
- Read clocks (QK/QK#).

Intel recommends that you use the FPGA parallel OCT to terminate the data on reads and read clocks.

### 5.3.2. Outputs from the FPGA to the RLDRAM II and RLDRAM 3 Components

The following output signals are from the FPGA to the RLDRAM II and RLDRAM 3 components:

- Write data (DQ on the bidirectional data signals for CIO RLDRAM II and RLDRAM 3)
- Data mask (DM)
- Address, bank address
- Command (CS, WE, and REF)
- Clocks (CK/CK# and DK/DK#)

For point-to-point single-ended signals requiring external termination, Intel recommends that you place a fly-by termination by terminating at the end of the transmission line after the receiver to avoid unterminated stubs. The guideline is to place the fly-by termination within 100 ps propagation delay of the receiver.

Although not recommended, you can place the termination before the receiver, which leaves an unterminated stub. The stub delay is critical because the stub between the termination and the receiver is effectively unterminated, causing additional ringing and reflections. Stub delays should be less than 50 ps.

Intel recommends that the differential clocks, CK, CK# and DK, DK# (RLDRAM II) and CK, CK# (RLDRAM 3), use a differential termination at the end of the trace at the external memory component. Alternatively, you can terminate each clock output with a parallel termination to VTT.



### 5.3.3. RLDRAM II Termination Schemes

The following table lists the recommended termination schemes for major CIO RLDRAM II memory interface signals. These signals include data (DQ), data mask (DM), clocks (CK, CK#, DK, DK#, QK, and QK#), address, bank address, and command (WE#, REF#, and CS#).

**Table 44. RLDRAM II Termination Recommendations for Arria II GZ, Arria V, Stratix III, Stratix IV, and Stratix V Devices**

Signal Type	HSTL 15/18 Standard <sup>(1) (2) (3) (4)</sup>	Memory End Termination
DK/DK# Clocks	Class I R50 NO CAL	100 -ohm□ Differential
QK/QK# Clocks	Class I P50 CAL	ZQ50
Data (Write)	Class I R50 CAL	ODT
Data (Read)	Class I P50 CAL	ZQ50
Data Mask	Class I R50 CAL	ODT
CK/CK# Clocks	Class I R50 NO CAL	$\times 1 = 100\text{-ohm Differential }^{(9)}$ $\times 2 = 200\text{-ohm Differential }^{(10)}$
Address/Bank Address <sup>(5) (6)</sup>	Class I Max Current	50 -ohm□ Parallel to V <sub>TT</sub>
Command (WE#, REF#) <sup>(5) (6)</sup>	Class I Max Current	50 -ohm□ Parallel to V <sub>TT</sub>
Command (CS#) <sup>(5) (6) (7)</sup>	Class I Max Current	50 -ohm□ Parallel to V <sub>TT</sub> or Pull-up to V <sub>DD</sub>
QVLD <sup>(8)</sup>	Class I P50 CAL	ZQ50

Notes to Table:

1. R is effective series output impedance.
2. P is effective parallel input impedance.
3. CAL is OCT with calibration.
4. NO CAL is OCT without calibration.
5. For width expansion configuration, the address and control signals are routed to 2 devices. Recommended termination is 50 -ohm parallel to V<sub>TT</sub> at the trace split of a balanced T or Y routing topology. Use a clamshell placement of the two RLDRAM II components to achieve minimal stub delays and optimum signal integrity. Clamshell placement is when two devices overlay each other by being placed on opposite sides of the PCB.
6. The UniPHY default IP setting for this output is Max Current. A Class I 50 -ohm output with calibration output is typically optimal in single load topologies.
7. Intel recommends that you use a 50 -ohm□parallel termination to V<sub>TT</sub> if your design meets the power sequencing requirements of the RLDRAM II component. Refer to the RLDRAM II data sheet for further information.
8. QVLD is not used in the RLDRAM II Controller with UniPHY implementations.
9.  $\times 1$  is a single-device load.
10.  $\times 2$  is a double-device load. An alternative option is to use a 100 -ohm differential termination at the trace split.

**Note:** Intel recommends that you simulate your specific design for your system to ensure good signal integrity.

### 5.3.4. RLDRAM 3 Termination Schemes

The following table lists the recommended termination schemes for major CIO RLDRAM 3 memory interface signals. These signals include data (DQ), data mask (DM), clocks (CK, CK#, DK, DK#, QK, and QK#), address, bank address, and command (WE#, REF#, and CS#).

**Table 45. RLDRAM 3 Termination Recommendations for Arria V GZ and Stratix V Devices**

Signal Type	Memory End Termination Option in the Chip (ODT)	Recommended On-Board Terminations
Data Read (DQ, QK)	40, 60 (series)	None
Data Write (DQ, DM, DK)	40, 60, 120 (parallel)	None
Address/Bank Address/ Command (WE#, RE# <sup>(1)</sup> , CS# <sup>(2)</sup> <sup>(3)</sup> )	None	50-ohm Parallel to V <sub>TT</sub>
CK/CK#	None	100 ohm Differential

Notes to Table:

1. For width expansion configuration, the address and control signals are routed to 2 devices. Recommended termination is 50-ohm parallel to V<sub>TT</sub> at the trace split of a balanced T or Y routing topology. Use a clamshell placement of the two RLDRAM 3 components to achieve minimal stub delays and optimum signal integrity. Clamshell placement is when two devices overlay each other by being placed on opposite sides of the PCB.
2. The UniPHY default IP setting for this output is Max Current. A Class I 50-ohm output with calibration output is typically optimal in single load topologies.
3. Intel recommends that you use a 50-ohm parallel termination to V<sub>TT</sub> if your design meets the power sequencing requirements of the RLDRAM 3 component. Refer to the RLDRAM 3 data sheet for further information.
4. QVLD is not used in the RLDRAM 3 Controller with UniPHY implementations.
5. For information on the I/O standards and on-chip termination (OCT) resistance values supported for RLDRAM 3, refer to the *I/O Features* chapter of the appropriate device handbook.

Intel recommends that you simulate your specific design for your system to ensure good signal integrity.

## 5.4. PCB Layout Guidelines

Intel recommends that you create your project in the Quartus® Prime software with a fully implemented RLDRAM II Controller with UniPHY interface, or RLDRAM 3 with UniPHY IP, and observe the interface timing margins to determine the actual margins for your design.

Although the recommendations in this chapter are based on simulations, you can apply the same general principles when determining the best termination scheme, drive strength setting, and loading style to any board designs. Intel recommends that you perform simulations, either using IBIS or HSPICE models, to determine the quality of signal integrity on your designs, and that you get accurate time base skew numbers when you simulate your specific implementation.

*Note:*

1. The following layout guidelines include several +/- length-based rules. These length-based guidelines are for first order timing approximations if you cannot simulate the actual delay characteristics of your PCB implementation. They do not include any margin for crosstalk.
2. To reliably close timing to and from the periphery of the device, signals to and from the periphery should be registered before any further logic is connected.

### Related Information

[Intel Power Distribution Network \(PDN\) Design Tool](#)



## 5.5. General Layout Guidelines

The following table lists general board design layout guidelines. These guidelines are Intel recommendations, and should not be considered as hard requirements. You should perform signal integrity simulation on all the traces to verify the signal integrity of the interface. You should extract the slew rate and propagation delay information, enter it into the IP and compile the design to ensure that timing requirements are met.

**Table 46. General Layout Guidelines**

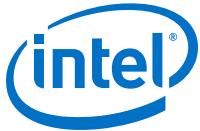
Parameter	Guidelines
Impedance	<ul style="list-style-type: none"><li>All unused via pads must be removed, because they cause unwanted capacitance.</li><li>Trace impedance plays an important role in the signal integrity. You must perform board level simulation to determine the best characteristic impedance for your PCB. For example, it is possible that for multi rank systems 40 ohms could yield better results than a traditional 50 ohm characteristic impedance.</li></ul>
Decoupling Parameter	<ul style="list-style-type: none"><li>Use 0.1 uF in 0402 size to minimize inductance</li><li>Make VTT voltage decoupling close to termination resistors</li><li>Connect decoupling caps between VTT and ground</li><li>Use a 0.1 uF cap for every other VTT pin and 0.01 uF cap for every VDD and VDDQ pin</li><li>Verify the capacitive decoupling using the Intel Power Distribution Network Design Tool</li></ul>
Power	<ul style="list-style-type: none"><li>Route GND and V<sub>CC</sub> as planes</li><li>Route VCCIO for memories in a single split plane with at least a 20-mil (0.020 inches, or 0.508 mm) gap of separation</li><li>Route VTT as islands or 250-mil (6.35-mm) power traces</li><li>Route oscillators and PLL power as islands or 100-mil (2.54-mm) power traces</li></ul>
General Routing	<p>All specified delay matching requirements include PCB trace delays, different layer propagation velocity variance, and crosstalk. To minimize PCB layer propagation variance, Intel recommends that signals from the same net group always be routed on the same layer.</p> <ul style="list-style-type: none"><li>Use 45° angles (<i>not</i> 90° corners)</li><li>Avoid T-Junctions for critical nets or clocks</li><li>Avoid T-junctions greater than 250 mils (6.35 mm)</li><li>Disallow signals across split planes</li><li>Restrict routing other signals close to system reset signals</li><li>Avoid routing memory signals closer than 0.025 inch (0.635 mm) to PCI or system clocks</li></ul>

### Related Information

[Power Distribution Network Design Tool](#)

## 5.6. RLDRAM II and RLDRAM 3 Layout Guidelines

The following table lists the RLDRAM II and RLDRAM 3 general routing layout guidelines. These guidelines apply to Arria V, Arria 10, Stratix V, and Stratix 10 devices.

**Table 47. RLDRAM II and RLDRAM 3 Layout Guidelines**

Parameter	Guidelines
General Routing	<ul style="list-style-type: none"><li>If you must route signals of the same net group on different layers with the same impedance characteristic, simulate your worst case PCB trace tolerances to ascertain actual propagation delay differences. Typical layer to layer trace delay variations are of 15 ps/inch order.</li><li>Avoid T-junctions greater than 150 ps.</li><li>Match all signals within a given DQ group with a maximum skew of <math>\pm 10</math> ps and route on the same layer.</li></ul>
Clock Routing	<ul style="list-style-type: none"><li>Route clocks on inner layers with outer-layer run lengths held to under 150 ps.</li><li>These signals should maintain a 10-mil (0.254 mm) spacing from other nets.</li><li>Clocks should maintain a length-matching between clock pairs of <math>\pm 5</math> ps.</li><li>Differential clocks should maintain a length-matching between P and N signals of <math>\pm 2</math> ps.</li><li>Space between different clock pairs should be at least three times the space between the traces of a differential pair.</li></ul>
Address and Command Routing	<ul style="list-style-type: none"><li>To minimize crosstalk, route address, bank address, and command signals on a different layer than the data and data mask signals.</li><li>Do not route the differential clock signals close to the address signals.</li><li>Keep the distance from the pin on the RLDRAM II or RLDRAM 3 component to the stub termination resistor (<math>V_{TT}</math>) to less than 50 ps for the address/command signal group.</li><li>Keep the distance from the pin on the RLDRAM II or RLDRAM 3 component to the fly-by termination resistor (<math>V_{TT}</math>) to less than 100 ps for the address/command signal group.</li></ul>
External Memory Routing Rules	<ul style="list-style-type: none"><li>Apply the following parallelism rules for the RLDRAM II or RLDRAM 3 data/address/command groups:<ul style="list-style-type: none"><li>4 mils for parallel runs &lt; 0.1 inch (approximately 1x spacing relative to plane distance).</li><li>5 mils for parallel runs &lt; 0.5 inch (approximately 1x spacing relative to plane distance).</li><li>10 mils for parallel runs between 0.5 and 1.0 inches (approximately 2x spacing relative to plane distance).</li><li>15 mils for parallel runs between 1.0 and 3.3 inch (approximately 3x spacing relative to plane distance).</li></ul></li></ul>
Maximum Trace Length	<ul style="list-style-type: none"><li>Keep the maximum trace length of all signals from the FPGA to the RLDRAM II or RLDRAM 3 components to 600 ps.</li></ul>
Trace Matching Guidance	<p>The following layout approach is recommended, based on the preceding guidelines:</p> <ol style="list-style-type: none"><li>If the RLDRAM II interface has multiple DQ groups (<math>\times 18</math> or <math>\times 36</math> RLDRAM II/RLDRAM 3 component or width expansion configuration), match all the DK/DK# and QK ,QK # clocks as tightly as possible to optimize the timing margins in your design.</li><li>Route the DK/DK# write clock and QK/QK# read clock associated with a DQ group on the same PCB layer. Match these clock pairs to within <math>\pm 5</math> ps.</li><li>Set the DK/DK# or QK/QK# clock as the target trace propagation delay for the associated data and data mask signals.</li><li>Route the data and data mask signals for the DQ group ideally on the same layer as the associated QK/QK# and DK/DK# clocks to within <math>\pm 10</math> ps skew of the target clock.</li><li>Route the CK/CK# clocks and set as the target trace propagation delays for the address/command signal group. Match the CK/CK# clock to within <math>\pm 50</math> ps of all the DK/DK# clocks.</li><li>Route the address/control signal group (address, bank address, CS, WE, and REF) ideally on the same layer as the CK/CK# clocks, to within <math>\pm 20</math> ps skew of the CK/CK# traces.</li></ol>

*continued...*



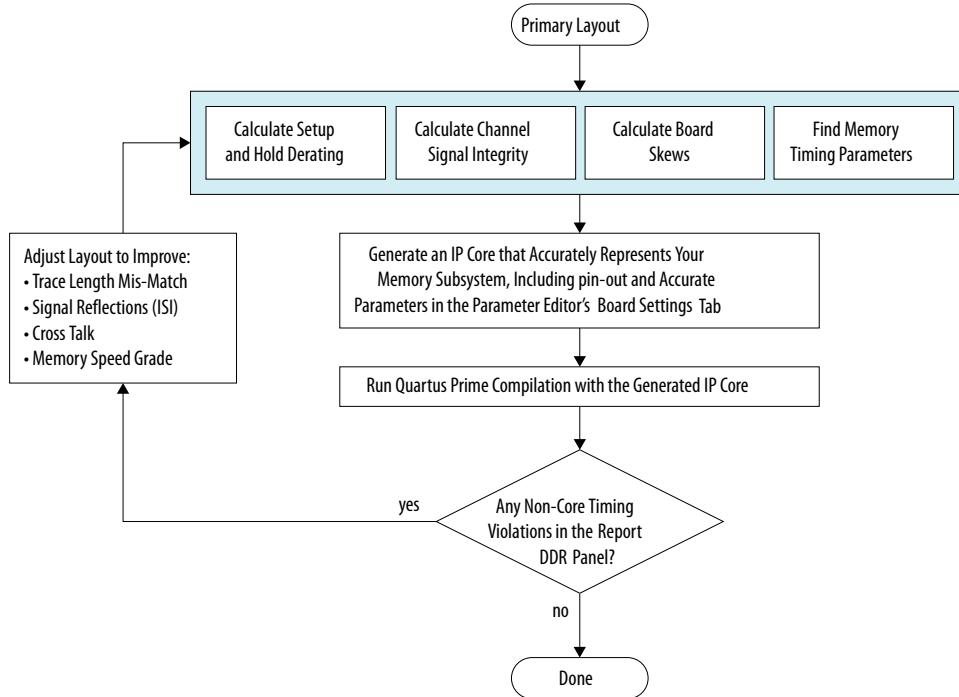
Parameter	Guidelines
	<p><b>Note:</b> It is important to match the delays of CK vs. DK, and CK vs. Addr-Cmd as much as possible.</p> <p>This layout approach provides a good starting point for a design requirement of the highest clock frequency supported for the RLDRAM II and RLDRAM 3 interfaces.</p>

## 5.7. Layout Approach

For all practical purposes, you can regard the TimeQuest timing analyzer's report on your memory interface as definitive for a given set of memory and board timing parameters.

You will find timing under **Report DDR** in TimeQuest and on the **Timing Analysis** tab in the parameter editor.

The following flowchart illustrates the recommended process to follow during the board design phase, to determine timing margin and make iterative improvements to your design.



### Board Skew

For information on calculating board skew parameters, refer to *Implementing and Parameterizing Memory IP*, in the *External Memory Interface Handbook*.

The Board Skew Parameter Tool is an interactive tool that can help you calculate board skew parameters if you know the absolute delay values for all the memory related traces.



### Memory Timing Parameters

For information on the memory timing parameters to be entered into the parameter editor, refer to the datasheet for your external memory device.

#### Related Information

[Board Skew Parameter Tool](#)

## 5.7.1. Arria V and Stratix V Board Setting Parameters

The following guidelines apply to the Board Setting parameters for Arria V and Stratix V devices.

#### Setup and Hold Derating

For information on calculating derating parameters, refer to *Implementing and Parameterizing Memory IP*, in the *External Memory Interface Handbook*.

#### Channel Signal Integrity

For information on determining channel signal integrity for Stratix V and earlier products, refer to the wiki page: [http://www.alterawiki.com/wiki/Measuring\\_Channel\\_Signal\\_Integrity](http://www.alterawiki.com/wiki/Measuring_Channel_Signal_Integrity) .

#### Board Skew

For information on calculating board skew parameters, refer to *Implementing and Parameterizing Memory IP*, in the *External Memory Interface Handbook*.

The Board Skew Parameter Tool is an interactive tool that can help you calculate board skew parameters if you know the absolute delay values for all the memory related traces.

#### Memory Timing Parameters

For information on the memory timing parameters to be entered into the parameter editor, refer to the datasheet for your external memory device.

## 5.7.2. Arria 10 Board Setting Parameters

For Board Setting and layout approach information for Arria 10 devices, refer to the wiki at the address below.

Arria 10 Board Setting and layout approach information: [http://www.alterawiki.com/wiki/Arria\\_10\\_EMIF\\_Simulation\\_Guidance](http://www.alterawiki.com/wiki/Arria_10_EMIF_Simulation_Guidance) .

## 5.8. Package Deskew for RLDRAM II and RLDRAM 3

You should follow Intel's package deskew guidance if you are using Arria 10, Stratix 10, or Stratix V devices.

For more information on package deskew, refer to *Package Deskew*.

#### Related Information

[Package Deskew](#)



## 5.9. Document Revision History

Date	Version	Changes
May 2017	2017.5.08	<ul style="list-style-type: none"> <li>Added Stratix 10 to <i>RLDRAM II and RLDRAM 3 Layout Guidelines</i> section.</li> <li>Rebranded as Intel.</li> </ul>
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	Maintenance release.
November 2015	2015.11.02	<ul style="list-style-type: none"> <li>Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i>.</li> </ul>
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	<ul style="list-style-type: none"> <li>Revised <i>RLDRAM 3 Termination Recommendations for Arria V GZ and Stratix V Devices</i> table.</li> <li>Removed millimeter approximations from lengths expressed in picoseconds in <i>RLDRAM II and RLDRAM 3 Layout Guidelines</i> table.</li> <li>Minor formatting fixes in <i>RLDRAM II and RLDRAM 3 Layout Guidelines</i> table.</li> <li>Added <i>Layout Approach</i> section.</li> </ul>
December 2013	2013.12.16	<ul style="list-style-type: none"> <li>Added note about byteenable support to <i>Signal Descriptions</i> section.</li> <li>Consolidated General Layout Guidelines.</li> </ul>
November 2012	3.2	Added content supporting RLDRAM 3 and updated RLDRAM II standards.
June 2012	3.1	Added Feedback icon.
November 2011	3.0	Added Arria V information.
June 2011	2.0	Added Stratix V information.
December 2010	1.0	Initial release.

## **6. QDR II and QDR-IV SRAM Board Design Guidelines**

The following topics provide guidelines for you to improve your system's signal integrity and layout guidelines to help successfully implement a QDR II or QDR II+ SRAM interface in your system.

The QDR II and QDR II+ SRAM Controller with UniPHY intellectual property (IP) enables you to implement QDR II and QDR II+ interfaces with Arria® II GX, Arria V, Stratix® III, Stratix IV, and Stratix V devices.

**Note:** In the following topics, QDR II SRAM refers to both QDR II and QDR II+ SRAM unless stated otherwise.

The following topics focus on the following key factors that affect signal integrity:

- I/O standards
- QDR II SRAM configurations
- Signal terminations
- Printed circuit board (PCB) layout guidelines

### **I/O Standards**

QDR II SRAM interface signals use one of the following JEDEC I/O signalling standards:

- HSTL-15—provides the advantages of lower power and lower emissions.
- HSTL-18—provides increased noise immunity with slightly greater output voltage swings.

To select the most appropriate standard for your interface, refer to the *Arria II GX Devices Data Sheet: Electrical Characteristics* chapter in the *Arria II Device Handbook*, *Stratix III Device Datasheet: DC and Switching Characteristics* chapter in the *Stratix III Device Handbook*, or the *Stratix IV Device Datasheet DC and Switching Characteristics* chapter in the *Stratix IV Device Handbook*.

Altera QDR II SRAM Controller with UniPHY IP defaults to HSTL 1.5 V Class I outputs and HSTL 1.5 V inputs.

### **Related Information**

- [Arria II GX Devices Data Sheet: Electrical Characteristics](#)
- [Stratix III Device Datasheet: DC and Switching Characteristics](#)
- [Stratix IV Device Datasheet DC and Switching Characteristics](#)

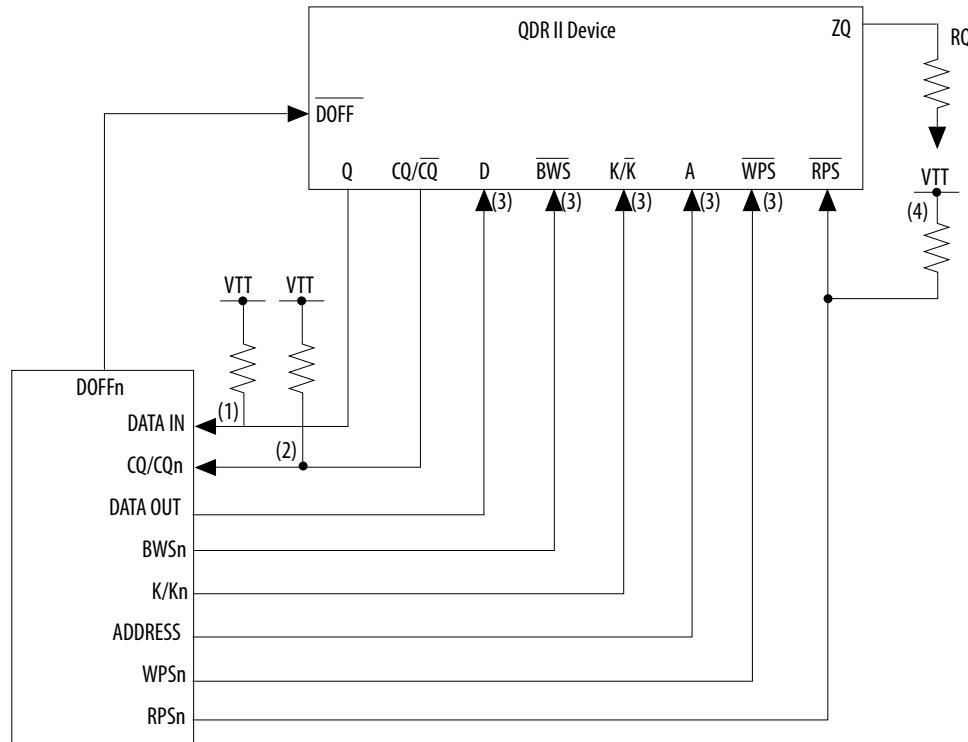


## 6.1. QDR II SRAM Configurations

The QDR II SRAM Controller with UniPHY IP supports interfaces with a single device, and two devices in a width expansion configuration up to maximum width of 72 bits.

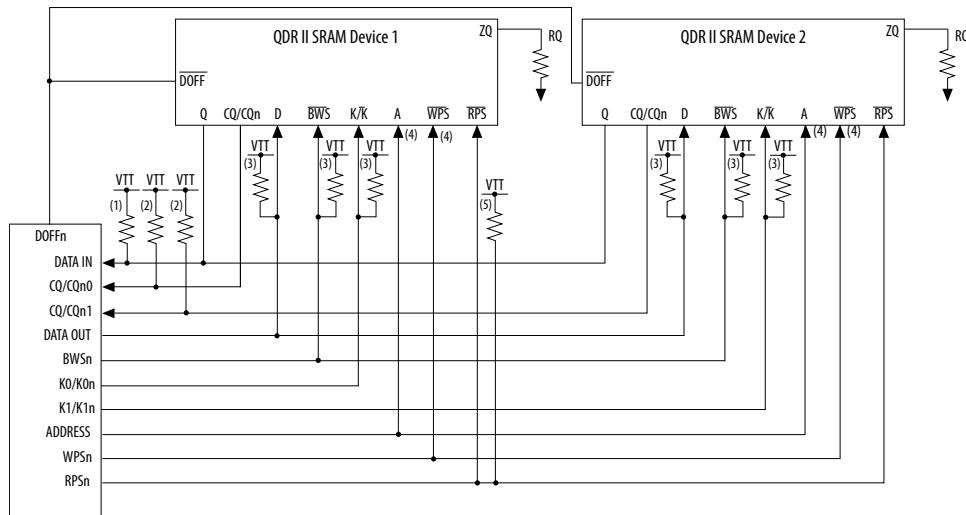
The following figure shows the main signal connections between the FPGA and a single QDR II SRAM component.

**Figure 56. Configuration With A Single QDR II SRAM Component**



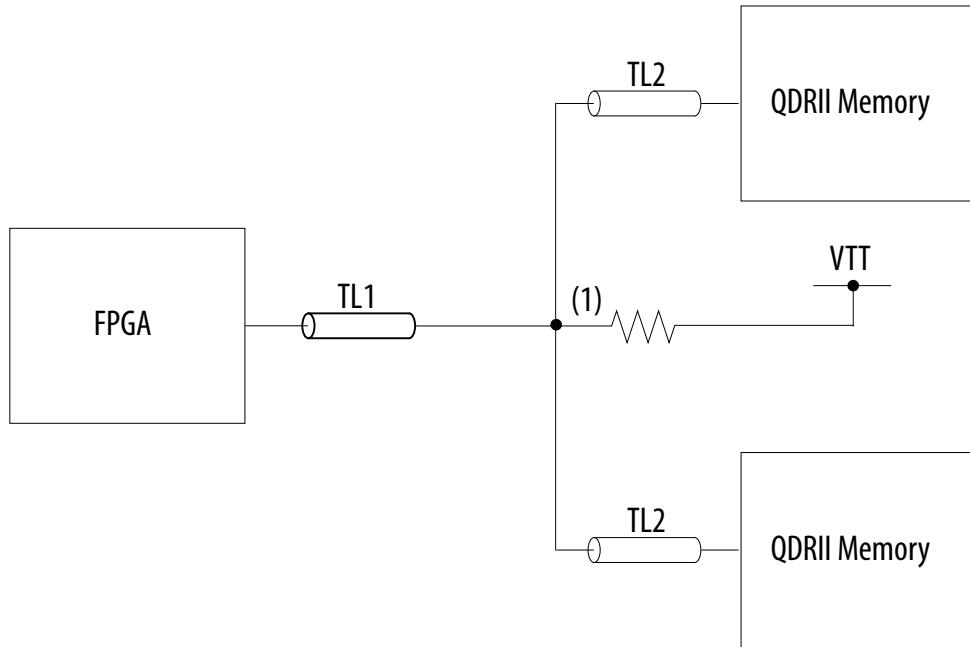
The following figure shows the main signal connections between the FPGA and two QDR II SRAM components in a width expansion configuration.

**Figure 57. Configuration With Two QDR II SRAM Components In A Width Expansion Configuration**



The following figure shows the detailed balanced topology recommended for the address and command signals in the width expansion configuration.

**Figure 58. External Parallel Termination for Balanced Topology**



## 6.2. Signal Terminations

Arria II GX, Stratix III and Stratix IV devices offer on-chip termination (OCT) technology.

The following table summarizes the extent of OCT support for each device.



**Table 48. On-Chip Termination Schemes (1)**

Termination Scheme	HSTL-15 and HSTL-18	FPGA Device					
		Arria II GX		Arria II GZ, Stratix III, and Stratix IV		Arria V and Stratix V	
		Column I/O	Row I/O	Column I/O	Row I/O	Column I/O	Row I/O
On-Chip Series Termination without Calibration	Class I	50	50	50	50	—	—
On-Chip Series Termination with Calibration	Class I	50	50	50	50	—	—
On-Chip Parallel Termination with Calibration	Class I	—	—	50	50	50	50

Note to Table:

- This table provides information about HSTL-15 and HSTL-18 standards because these are the supported I/O standards for QDR II SRAM memory interfaces by Intel FPGAs.

On-chip series ( $R_S$ ) termination is supported only on output and bidirectional buffers, while on-chip parallel ( $R_T$ ) termination is supported only on input and bidirectional buffers. Because QDR II SRAM interfaces have unidirectional data paths, dynamic OCT is not required.

For Arria II GX, Stratix III and Stratix IV devices, the HSTL Class I I/O calibrated terminations are calibrated against 50-ohm 1% resistors connected to the  $R_{UP}$  and  $R_{DN}$  pins in an I/O bank with the same VCCIO as the QDRII SRAM interface. The calibration occurs at the end of the device configuration.

QDR II SRAM controllers have a ZQ pin which is connected via a resistor  $R_Q$  to ground. Typically the QDR II SRAM output signal impedance is  $0.2 \times R_Q$ . Refer to the QDR II SRAM device data sheet for more information.

For information about OCT, refer to the *I/O Features in Arria II GX Devices* chapter in the *Arria II GX Device Handbook*, *I/O Features in Arria V Devices* chapter in the *Arria V Device Handbook*, *Stratix III Device I/O Features* chapter in the *Stratix III Device Handbook*, *I/O Features in Stratix IV Devices* chapter in the *Stratix IV Device Handbook*, and the *I/O Features in Stratix V Devices* chapter in the *Stratix V Device Handbook*.

### Related Information

- [I/O Features in Arria II GX Devices](#)
- [I/O Features in Arria V Devices](#)
- [Stratix III Device I/O Features](#)
- [I/O Features in Stratix IV Devices](#)
- [I/O Features in Stratix V Devices](#)



### 6.2.1. Output from the FPGA to the QDR II SRAM Component

The following output signals are from the FPGA to the QDR II SRAM component:

- write data
- byte write select ( $BWS_n$ )
- address
- control ( $WPS_n$  and  $RPS_n$ )
- clocks,  $K/K\#$

Intel recommends that you terminate the write clocks,  $K$  and  $K\#$ , with a single-ended fly-by 50-ohm parallel termination to  $V_{TT}$ . However, simulations show that you can consider a differential termination if the clock pair is well matched and routed differentially.

Intel strongly recommends signal terminations to optimize signal integrity and timing margins, and to minimize unwanted emissions, reflections, and crosstalk.

For point-to-point signals, Intel recommends that you place a fly-by termination by terminating at the end of the transmission line after the receiver to avoid unterminated stubs. The guideline is to place the fly-by termination within 100 ps propagation delay of the receiver.

Although not recommended, you can place the termination before the receiver, which leaves an unterminated stub. The stub delay is critical because the stub between the termination and the receiver is effectively unterminated, causing additional ringing and reflections. Stub delays should be less than 50 ps.

*Note:* Simulate your design to ensure correct functionality.

### 6.2.2. Input to the FPGA from the QDR II SRAM Component

The QDR II SRAM component drives the following input signals into the FPGA:

- read data
- echo clocks,  $CQ/CQ\#$

For point-to-point signals, Intel recommends that you use the FPGA parallel OCT wherever possible. For devices that do not support parallel OCT (Arria II GX), and for  $\times 36$  emulated configuration  $CQ/CQ\#$  termination, Intel recommends that you use a fly-by 50-ohm parallel termination to  $V_{TT}$ . Although not recommended, you can use parallel termination with a short stub of less than 50 ps propagation delay as an alternative option. The input echo clocks,  $CQ$  and  $CQ\#$  must not use a differential termination.

### 6.2.3. Termination Schemes

The following tables list the recommended termination schemes for major QDR II SRAM memory interface signals.

These signals include write data ( $D$ ), byte write select ( $BWS$ ), read data ( $Q$ ), clocks ( $K$ ,  $K\#$ ,  $CQ$ , and  $CQ\#$ ), address and command ( $WPS$  and  $RPS$ ).



**Table 49. Termination Recommendations for Arria II GX Devices**

Signal Type	HSTL 15/18 Standard (1) (2)	FPGA End Discrete Termination	Memory End Termination
K/K# Clocks	Class I R50 CAL	—	50-ohm Parallel to V <sub>TT</sub>
Write Data	Class I R50 CAL	—	50-ohm Parallel to V <sub>TT</sub>
BWS	Class I R50 CAL	—	50-ohm Parallel to V <sub>TT</sub>
Address (3) (4)	Class I Max Current	—	50-ohm Parallel to V <sub>TT</sub>
WPS, RPS (3) (4)	Class I Max Current	—	50-ohm Parallel to V <sub>TT</sub>
CQ/CQ#	Class I	50-ohm Parallel to V <sub>TT</sub>	ZQ50
CQ/CQ# ×36 emulated (5)	Class I	50-ohm Parallel to V <sub>TT</sub>	ZQ50
Read Data (Q)	Class I	50-ohm Parallel to V <sub>TT</sub>	ZQ50
QVLD (6)	—	—	ZQ50

Notes to Table:

1. R is effective series output impedance.
2. CAL is calibrated OCT.
3. For width expansion configuration, the address and control signals are routed to 2 devices. Recommended termination is 50 -ohm parallel to V<sub>TT</sub> at the trace split of a balanced T or Y routing topology. For 400 MHz burst length 2 configurations where the address signals are double data rate, it is recommended to use a clamshell placement of the two QDR II SRAM components to achieve minimal stub delays and optimum signal integrity. Clamshell placement is when two devices overlay each other by being placed on opposite sides of the PCB.
4. A Class I 50 -ohm output with calibration output is typically optimal in double load topologies.
5. For ×36 emulated mode, the recommended termination for the CQ/CQ# signals is a 50 -ohm parallel termination to V<sub>TT</sub> at the trace split. Intel recommends that you use this termination when ×36 DQ/DQS groups are not supported in the FPGA.
6. QVLD is not used in the QDR II or QDR II+ SRAM with UniPHY implementations.

**Table 50. Termination Recommendations for Arria V, Stratix III, Stratix IV, and Stratix V Devices**

Signal Type	HSTL 15/18 Standard (1) (2) (3)	FPGA End Discrete Termination	Memory End Termination
K/K# Clocks	DIFF Class I R50 NO CAL	—	Series 50 -ohm Without Calibration
Write Data	Class I R50 CAL	—	50 -ohm□Parallel to V <sub>TT</sub>
BWS	Class I R50 CAL	—	50 -ohm□Parallel to V <sub>TT</sub>
Address (4) (5)	Class I Max Current	—	50 -ohm□Parallel to V <sub>TT</sub>
WPS, RPS (4) (5)	Class I Max Current	—	50 -ohm□Parallel to V <sub>TT</sub>
CQ/CQ#	Class I P50 CAL	—	ZQ50
CQ/CQ# ×36 emulated (6)	—	50 -ohm Parallel to V <sub>TT</sub>	ZQ50
Read Data (Q)	Class I P50 CAL	—	ZQ50
QVLD (7)	Class I P50 CAL	—	ZQ50

Notes to Table:

1. R is effective series output impedance.
2. P is effective parallel input impedance.
3. CAL is calibrated OCT.

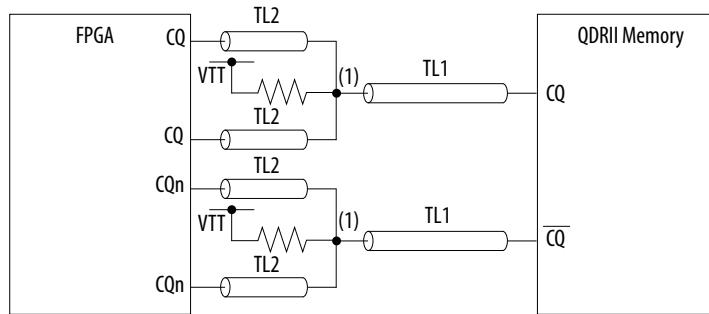
*continued...*

Signal Type	HSTL 15/18 Standard (1) (2) (3)	FPGA End Discrete Termination	Memory End Termination
<p>4. For width expansion configuration, the address and control signals are routed to 2 devices. Recommended termination is <math>50\Omega</math>-ohm parallel to <math>V_{TT}</math> at the trace split of a balanced T or Y routing topology. For 400 MHz burst length 2 configurations where the address signals are double data rate, it is recommended to use a "clam shell" placement of the two QDR II SRAM components to achieve minimal stub delays and optimum signal integrity. "Clam shell" placement is when two devices overlay each other by being placed on opposite sides of the PCB.</p> <p>5. The UniPHY default IP setting for this output is Max Current. A Class 1 <math>50\Omega</math>-ohm output with calibration output is typically optimal in single load topologies.</p> <p>6. For <math>\times 36</math> emulated mode, the recommended termination for the CQ/CQ# signals is a <math>50\Omega</math>-ohm parallel termination to <math>V_{TT}</math> at the trace split. Intel recommends that you use this termination when <math>\times 36</math> DQ/DQS groups are not supported in the FPGA.</p> <p>7. QVLD is not used in the QDR II or QDR II+ SRAM Controller with UniPHY implementations.</p>			

**Note:** Intel recommends that you simulate your specific design for your system to ensure good signal integrity.

For a  $\times 36$  QDR II SRAM interface that uses an emulated mode of two  $\times 18$  DQS groups in the FPGA, there are two CQ/CQ# connections at the FPGA and a single CQ/CQ# output from the QDR II SRAM device. Intel recommends that you use a balanced T topology with the trace split close to the FPGA and a parallel termination at the split, as shown in the following figure.

**Figure 59. Emulated  $\times 36$  Mode CQ/CQn Termination Topology**



For more information about  $\times 36$  emulated modes, refer to the "Exceptions for  $\times 36$  Emulated QDR II and QDR II+ SRAM Interfaces in Arria II GX, Stratix III, and Stratix IV Devices" section in the *Planning Pin and Resources* chapter.

#### Related Information

[Planning Pin and FPGA Resources](#) on page 9

### 6.3. General Layout Guidelines

The following table lists general board design layout guidelines. These guidelines are Intel recommendations, and should not be considered as hard requirements. You should perform signal integrity simulation on all the traces to verify the signal integrity of the interface. You should extract the slew rate and propagation delay information, enter it into the IP and compile the design to ensure that timing requirements are met.



**Table 51. General Layout Guidelines**

Parameter	Guidelines
Impedance	<ul style="list-style-type: none"> <li>All unused via pads must be removed, because they cause unwanted capacitance.</li> <li>Trace impedance plays an important role in the signal integrity. You must perform board level simulation to determine the best characteristic impedance for your PCB. For example, it is possible that for multi rank systems 40 ohms could yield better results than a traditional 50 ohm characteristic impedance.</li> </ul>
Decoupling Parameter	<ul style="list-style-type: none"> <li>Use 0.1 uF in 0402 size to minimize inductance</li> <li>Make VTT voltage decoupling close to termination resistors</li> <li>Connect decoupling caps between VTT and ground</li> <li>Use a 0.1 uF cap for every other VTT pin and 0.01 uF cap for every VDD and VDDQ pin</li> <li>Verify the capacitive decoupling using the Intel Power Distribution Network Design Tool</li> </ul>
Power	<ul style="list-style-type: none"> <li>Route GND and V<sub>CC</sub> as planes</li> <li>Route VCCIO for memories in a single split plane with at least a 20-mil (0.020 inches, or 0.508 mm) gap of separation</li> <li>Route VTT as islands or 250-mil (6.35-mm) power traces</li> <li>Route oscillators and PLL power as islands or 100-mil (2.54-mm) power traces</li> </ul>
General Routing	<p>All specified delay matching requirements include PCB trace delays, different layer propagation velocity variance, and crosstalk. To minimize PCB layer propagation variance, Intel recommends that signals from the same net group always be routed on the same layer.</p> <ul style="list-style-type: none"> <li>Use 45° angles (<i>not</i> 90° corners)</li> <li>Avoid T-Junctions for critical nets or clocks</li> <li>Avoid T-junctions greater than 250 mils (6.35 mm)</li> <li>Disallow signals across split planes</li> <li>Restrict routing other signals close to system reset signals</li> <li>Avoid routing memory signals closer than 0.025 inch (0.635 mm) to PCI or system clocks</li> </ul>

#### Related Information

[Power Distribution Network Design Tool](#)

## 6.4. QDR II Layout Guidelines

The following table summarizes QDR II and QDR II SRAM general routing layout guidelines.

*Note:*

1. The following layout guidelines include several +/- length based rules. These length based guidelines are for first order timing approximations if you cannot simulate the actual delay characteristics of your PCB implementation. They do not include any margin for crosstalk.
2. Intel recommends that you get accurate time base skew numbers when you simulate your specific implementation.
3. To reliably close timing to and from the periphery of the device, signals to and from the periphery should be registered before any further logic is connected.

**Table 52. QDR II and QDR II+ SRAM Layout Guidelines**

Parameter	Guidelines
General Routing	<ul style="list-style-type: none"> <li>If signals of the same net group must be routed on different layers with the same impedance characteristic, you must simulate your worst case PCB trace tolerances to ascertain actual propagation delay differences. Typical later to later trace delay variations are of 15 ps/inch order.</li> <li>Avoid T-junctions greater than 150 ps.</li> </ul>
Clock Routing	<ul style="list-style-type: none"> <li>Route clocks on inner layers with outer-layer run lengths held to under 150 ps.</li> <li>These signals should maintain a 10-mil (0.254 mm) spacing from other nets.</li> <li>Clocks should maintain a length-matching between clock pairs of <math>\pm 5</math> ps.</li> <li>Complementary clocks should maintain a length-matching between P and N signals of <math>\pm 2</math> ps.</li> <li>Keep the distance from the pin on the QDR II SRAM component to stub termination resistor (<math>V_{TT}</math>) to less than 50 ps for the K, K# clocks.</li> <li>Keep the distance from the pin on the QDR II SRAM component to fly-by termination resistor (<math>V_{TT}</math>) to less than 100 ps for the K, K# clocks.</li> <li>Keep the distance from the pin on the FPGA component to stub termination resistor (<math>V_{TT}</math>) to less than 50 ps for the echo clocks, CQ, CQ#, if they require an external discrete termination.</li> <li>Keep the distance from the pin on the FPGA component to fly-by termination resistor (<math>V_{TT}</math>) to less than 100 ps for the echo clocks, CQ, CQ#, if they require an external discrete termination.</li> </ul>
External Memory Routing Rules	<ul style="list-style-type: none"> <li>Keep the distance from the pin on the QDR II SRAM component to stub termination resistor (<math>V_{TT}</math>) to less than 50 ps for the write data, byte write select and address/command signal groups.</li> <li>Keep the distance from the pin on the QDR II SRAM component to fly-by termination resistor (<math>V_{TT}</math>) to less than 100 ps for the write data, byte write select and address/command signal groups.</li> <li>Keep the distance from the pin on the FPGA (Arria II GX) to stub termination resistor (<math>V_{TT}</math>) to less than 50 ps for the read data signal group.</li> <li>Keep the distance from the pin on the FPGA (Arria II GX) to fly-by termination resistor (<math>V_{TT}</math>) to less than 100 ps for the read data signal group.</li> <li>Parallelism rules for the QDR II SRAM data/address/command groups are as follows: <ul style="list-style-type: none"> <li>4 mils for parallel runs &lt; 0.1 inch (approximately 1x spacing relative to plane distance).</li> <li>5 mils for parallel runs &lt; 0.5 inch (approximately 1x spacing relative to plane distance).</li> <li>10 mils for parallel runs between 0.5 and 1.0 inches (approximately 2x spacing relative to plane distance).</li> <li>15 mils for parallel runs between 1.0 and 6.0 inch (approximately 3x spacing relative to plane distance).</li> </ul> </li> </ul>
Maximum Trace Length	<ul style="list-style-type: none"> <li>Keep the maximum trace length of all signals from the FPGA to the QDR II SRAM components to 6 inches.</li> </ul>

### Related Information

[Intel Power Distribution Network \(PDN\) Design tool](#)

## 6.5. QDR II SRAM Layout Approach

Using the layout guidelines in the above table, Intel recommends the following layout approach:



1. Route the K/K# clocks and set the clocks as the target trace propagation delays for the output signal group.
2. Route the write data output signal group (write data, byte write select), ideally on the same layer as the K/K# clocks, to within  $\pm 10$  ps skew of the K/K# traces.
3. Route the address/control output signal group (address, RPS, WPS), ideally on the same layer as the K/K# clocks, to within  $\pm 20$  ps skew of the K/K# traces.
4. Route the CQ/CQ# clocks and set the clocks as the target trace propagation delays for the input signal group.
5. Route the read data output signal group (read data), ideally on the same layer as the CQ/CQ# clocks, to within  $\pm 10$  ps skew of the CQ/CQ# traces.
6. The output and input groups do not need to have the same propagation delays, but they must have all the signals matched closely within the respective groups.

The following tables list the typical margins for QDR II and QDR II+ SRAM interfaces, with the assumption that there is zero skew between the signal groups.

**Table 53. Typical Worst Case Margins for QDR II SRAM Interfaces of Burst Length 2**

Device	Speed Grade	Frequency (MHz)	Typical Margin Address/Command (ps)	Typical Margin Write Data (ps)	Typical Margin Read Data (ps)
Arria II GX	I5	250	$\pm 240$	$\pm 80$	$\pm 170$
Arria II GX $\times 36$ emulated	I5	200	$\pm 480$	$\pm 340$	$\pm 460$
Stratix IV	—	350	—	—	—
Stratix IV $\times 36$ emulated	C2	300	$\pm 320$	$\pm 170$	$\pm 340$

**Table 54. Typical Worst Case Margins for QDR II+ SRAM Interfaces of Burst Length 4**

Device	Speed Grade	Frequency (MHz)	Typical Margin Address/Command (ps) (1)	Typical Margin Write Data (ps)	Typical Margin Read Data (ps)
Arria II GX	I5	250	$\pm 810$	$\pm 150$	$\pm 130$
Arria II GX $\times 36$ emulated	I5	200	$\pm 1260$	$\pm 410$	$\pm 420$
Stratix IV	C2	400	$\pm 550$	$\pm 10$	$\pm 80$
Stratix IV $\times 36$ emulated	C2	300	$\pm 860$	$\pm 180$	$\pm 300$

Note to Table:

1. The QDR II+ SRAM burst length of 4 designs have greater margins on the address signals because they are single data rate.

Other devices and speed grades typically show higher margins than the ones in the above tables.

**Note:** Intel recommends that you create your project with a fully implemented QDR II or QDR II+ SRAM Controller with UniPHY interface, and observe the interface timing margins to determine the actual margins for your design.

Although the recommendations in this chapter are based on simulations, you can apply the same general principles when determining the best termination scheme, drive strength setting, and loading style to any board designs. Even armed with this knowledge, it is still critical that you perform simulations, either using IBIS or HSPICE models, to determine the quality of signal integrity on your designs.

## 6.6. Package Deskeew for QDR II and QDR-IV

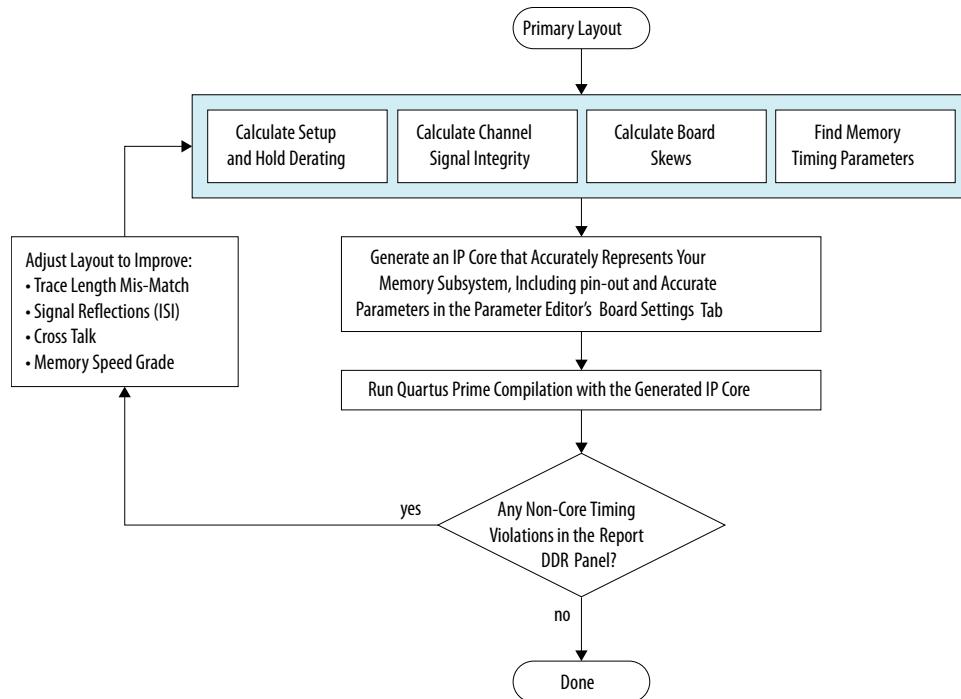
You should follow Intel's package deskeew guidance if you are using Stratix V or Arria 10 devices.

For more information on package deskeew, refer to *Package Deskeew*.

## 6.7. QDR-IV Layout Approach

For all practical purposes, you can regard the TimeQuest timing analyzer's report on your memory interface as definitive for a given set of memory and board timing parameters. You will find timing under Report DDR in TimeQuest and on the Timing Analysis tab in the parameter editor.

The following flowchart illustrates the recommended process to follow during the design phase, to determine timing margin and make iterative improvements to your design.



For more detailed simulation guidance for Arria 10, refer to the wiki: [http://www.alterawiki.com/wiki/Arria\\_10\\_EMIF\\_Simulation\\_Guidance](http://www.alterawiki.com/wiki/Arria_10_EMIF_Simulation_Guidance)



### Intersymbol Interference/Crosstalk

For information on intersymbol interference and crosstalk, refer to the wiki: [http://www.alterawiki.com/wiki/Arria\\_10\\_EMIF\\_Simulation\\_Guidance](http://www.alterawiki.com/wiki/Arria_10_EMIF_Simulation_Guidance)

### Board Skew

For information on calculating board skew parameters, refer to

If you know the absolute delays for all the memory related traces, the interactive [Board Skew Parameter Tool](#) can help you calculate the necessary parameters.

### Memory Timing Parameters

You can find the memory timing parameters to enter in the parameter editor, in your memory vendor's datasheet.

## 6.8. QDR-IV Layout Guidelines

Observe the following layout guidelines for your QDR-IV interface. These guidelines apply for all device families that support QDR-IV, including Arria 10 and Stratix 10.

Parameter	Guidelines
General Routing	<ul style="list-style-type: none"><li>If you must route signals of the same net group on different layers with the same impedance characteristic, simulate your worst case PCB trace tolerances to determine actual propagation delay differences. Typical layer-to-layer trace delay variations are on the order of 15 ps/inch.</li><li>Avoid T-junctions greater than 150 ps.</li><li>Match all signals within a given DQ group with a maximum skew of <math>\pm 10</math> ps and route on the same layer.</li></ul>
Clock Routing	<ul style="list-style-type: none"><li>Route clocks on inner layers with outer-layer run lengths held to less than 150 ps.</li><li>Clock signals should maintain a 10-mil (0.254 mm) spacing from other nets.</li><li>Clocks should maintain a length-matching between clock pairs of <math>\pm 5</math> ps.</li><li>Differential clocks should maintain a length-matching between P and N signals of <math>\pm 2</math> ps.</li><li>Space between different clock pairs should be at least three times the space between the traces of a differential pair.</li></ul>
Address and Command Routing	<ul style="list-style-type: none"><li>- To minimize crosstalk, route address, bank address, and command signals on a different layer than the data signals.</li><li>Do not route the differential clock signals close to the address signals.</li><li>Keep the distance from the pin on the QDR-IV component to the stub termination resistor (VTT) to less than 50 ps for the address/command signal group.</li><li>- Route the mem_ck (CK/CK#) clocks and set as the target trace propagation delays for the address/command signal group. Match the CK/CK# clock to within <math>\pm 50</math> ps of all the DK/DK# clocks for both ports.</li><li>- Route the address/control signal group ideally on the same layer as the mem_ck (CK/CK#) clocks, to within <math>\pm 20</math> ps skew of the mem_ck (CK/CK#) traces.</li></ul>

*continued...*



Parameter	Guidelines
Data Signals	<ul style="list-style-type: none"><li>For port B only: Swap the polarity of the QKB and QKB# signals with respect to the polarity of the differential buffer inputs on the FPGA. Connect the positive leg of the differential input buffer on the FPGA to QDR-IV QKB# (negative) pin and vice-versa. Note that the port names at the top-level of the IP already reflect this swap (that is, mem_qkb is assigned to the negative buffer leg, and mem_qkb_n is assigned to the positive buffer leg).</li><li>For each port, route the DK/DK# write clock and QK/QK# read clock associated with a DQ group on the same PCB layer. Match these clock pairs to within ±5 ps.</li><li>For each port, set the DK/DK# or QK/QK# clock as the target trace propagation delay for the associated data signals (DQ).</li><li>For each port, route the data (DQ) signals for the DQ group ideally on the same layer as the associated QK/QK# and DK/DK# clocks to within ±10 ps skew of the target clock.</li></ul>
Maximum Trace Length	<ul style="list-style-type: none"><li>Keep the maximum trace length of all signals from the FPGA to the QDR-IV components to 600 ps.</li></ul>
Spacing Guidelines	<ul style="list-style-type: none"><li>Avoid routing two signal layers next to each other. Always make sure that the signals related to memory interface are routed between appropriate GND or power layers.</li><li>For Data and Data Strobe traces: Maintain at least 3H spacing between the edges (air-gap) of these traces, where H is the vertical distance to the closest return path for that particular trace.</li><li>For Address/Command/Control traces: Maintain at least 3H spacing between the edges (air-gap) of these traces, where H is the vertical distance to the closest return path for that particular trace.</li><li>For Clock (mem_CK) traces: Maintain at least 5H spacing between two clock pair or a clock pair and any other memory interface trace, where H is the vertical distance to the closest return path for that particular trace.</li></ul>
Trace Matching Guidance	<p>The following layout approach is recommended, based on the preceding guidelines:</p> <ol style="list-style-type: none"><li>For port B only: Swap the polarity of the QKB and QKB# signals with respect to the polarity of the differential buffer inputs on the FPGA. Connect the positive leg of the differential input buffer on the FPGA to QDR-IV QKB# (negative) pin and vice-versa. Note that the port names at the top-level of the IP already reflect this swap (that is, mem_qkb is assigned to the negative buffer leg, and mem_qkb_n is assigned to the positive buffer leg).</li><li>For each port, set the DK/DK# or QK/QK# clock as the target trace propagation delay for the associated data signals (DQ).</li><li>For each port, route the data (DQ) signals for the DQ group ideally on the same layer as the associated QK/QK# and DK/DK# clocks to within ±10 ps skew of the target clock.</li><li>Route the mem_ck (CK/CK#) clocks and set as the target trace propagation delays for the address/command signal group. Match the CK/CK# clock to within ±50 ps of all the DK/DK# clocks for both ports.</li><li>Route the address/control signal group ideally on the same layer as the mem_ck (CK/CK#) clocks, to within ±10 ps skew of the mem_ck (CK/CK#) traces.</li></ol>

## 6.9. Document Revision History

Date	Version	Changes
May 2017	2017.5.08	<ul style="list-style-type: none"><li>Added Stratix 10 to QDR-IV Layout Guidelines section.</li><li>Rebranded as Intel.</li></ul>
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	Maintenance release.
November 2015	2015.11.02	Maintenance release.
May 2015	2015.05.04	In the first guideline of the <i>QDR-IV Layout Recommendations</i> and the <i>Data Signals</i> section of the <i>QDR-IV Layout Guidelines</i> , revised the information for Port B only.

*continued...*



Date	Version	Changes
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	<ul style="list-style-type: none"> <li>• Change to K/K# Clocks row in <i>Termination Recommendations for Arria V, Stratix III, Stratix IV, and Stratix V Devices</i> table.</li> <li>• Removed millimeter approximations from lengths expressed in picoseconds in <i>QDR II and QDR II+ SRAM Layout Guidelines</i> table.</li> <li>• Minor formatting fixes in <i>QDR II and QDR II+ SRAM Layout Guidelines</i> table.</li> </ul>
December 2013	2013.12.16	Consolidated General Layout Guidelines.
November 2012	4.2	Changed chapter number from 7 to 8.
June 2012	4.1	Added Feedback icon.
November 2011	4.0	Added Arria V information.
June 2011	3.0	Added Stratix V information.
December 2010	2.0	Maintenance release.
July 2010	1.0	Initial release.

## 7. Implementing and Parameterizing Memory IP

The following topics describe the general overview of the IP core design flow to help you quickly get started with any IP core.

The IP Library is installed as part of the Quartus® Prime installation process. You can select and parameterize any Intel® IP core from the library. Intel provides an integrated parameter editor that allows you to customize IP cores to support a wide variety of applications. The parameter editor guides you through the setting of parameter values and selection of optional ports. The following section describes the general design flow and use of Intel® IP cores.

**Note:** Information for Arria 10 External Memory Interface IP also applies to Arria 10 External Memory Interface for HPS IP unless stated otherwise.

### Related Information

#### Intel FPGA Design Store

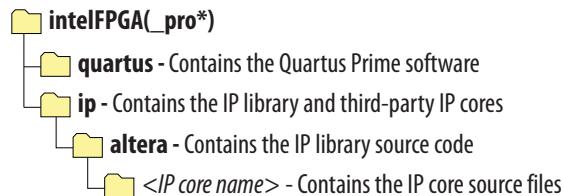
Design Example for Arria 10 DDR3 External Memory Interface is available in the Intel FPGA Design Store.

### 7.1. Installing and Licensing IP Cores

The Intel Quartus Prime software installation includes the Intel FPGA IP library. This library provides useful IP core functions for your production use without the need for an additional license. Some IP cores in the library require that you purchase a separate license for production use. The OpenCore® feature allows evaluation of any Intel FPGA IP core in simulation and compilation in the Quartus Prime software. Upon satisfaction with functionality and performance, visit the Self Service Licensing Center to obtain a license number for any Intel FPGA product.

The Quartus Prime software installs IP cores in the following locations by default:

**Figure 60. IP Core Installation Path**





**Table 55. IP Core Installation Locations**

Location	Software	Platform
<drive>:\intelFPGA_pro\quartus\ip\altera	Quartus Prime Pro Edition	Windows*
<drive>:\intelFPGA\quartus\ip\altera	Quartus Prime Standard Edition	Windows
<home directory>:/intelFPGA_pro/quartus/ip/altera	Quartus Prime Pro Edition	Linux*
<home directory>:/intelFPGA/quartus/ip/altera	Quartus Prime Standard Edition	Linux

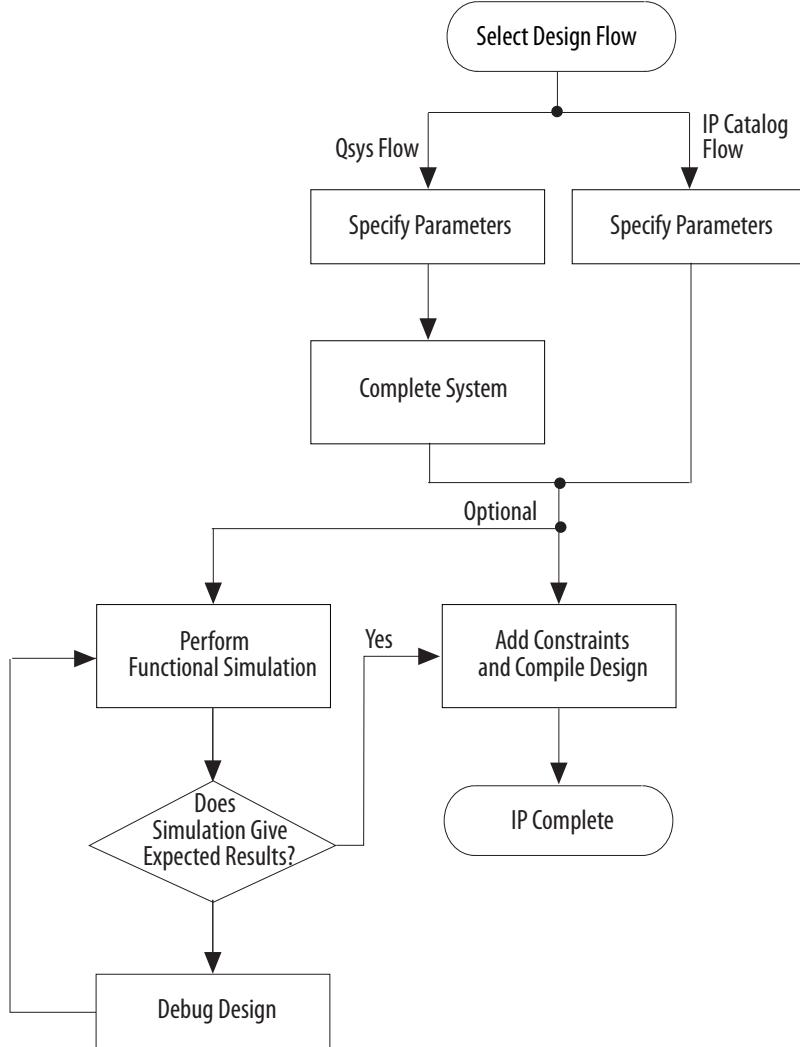
## 7.2. Design Flow

You can implement the external memory interface IP using the following flows:

- IP Catalog flow
- Qsys flow

The following figure shows the stages for creating a system in the Quartus Prime software using the available flows.

**Figure 61. Design Flows**



Note to Figure:

The IP Catalog design flow is suited for simple designs where you want to manually instantiate the external memory interface IP into a larger component. The Qsys design flow is recommended for more complex system designs where you want the tool to manage the instantiation process.

### 7.2.1. IP Catalog Design Flow

The IP Catalog design flow allows you to customize the external memory interface IP, and manually integrate the function into your design.

#### 7.2.1.1. IP Catalog and Parameter Editor

The IP Catalog displays the IP cores available for your project. Use the following features of the IP Catalog to locate and customize an IP core:

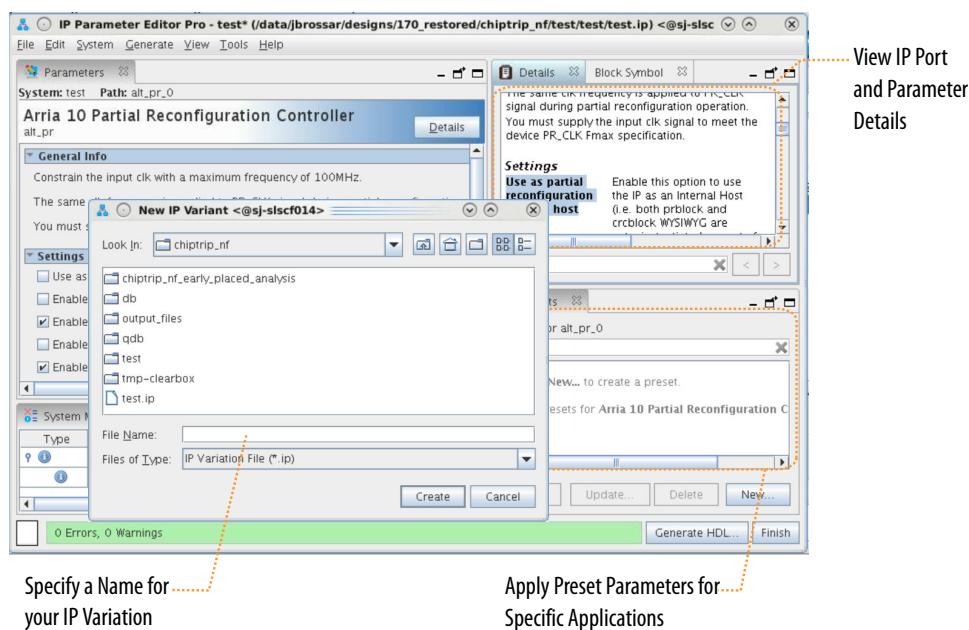


- Filter IP Catalog to **Show IP for active device family** or **Show IP for all device families**. If you have no project open, select the **Device Family** in IP Catalog.
- Type in the Search field to locate any full or partial IP core name in IP Catalog.
- Right-click an IP core name in IP Catalog to display details about supported devices, to open the IP core's installation folder, and for links to IP documentation.
- Click **Search for Partner IP** to access partner IP information on the web.

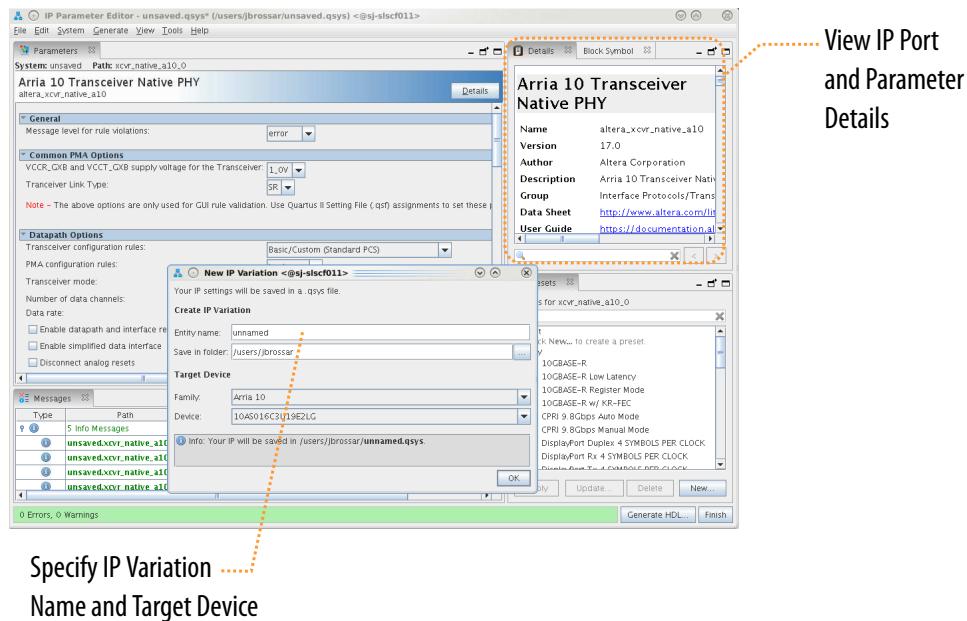
The parameter editor prompts you to specify an IP variation name, optional ports, and output file generation options. The parameter editor generates a top-level Quartus Prime IP file (.ip) for an IP variation in Quartus Prime Pro Edition projects.

The parameter editor generates a top-level Quartus IP file (.qip) for an IP variation in Quartus Prime Standard Edition projects. These files represent the IP variation in the project, and store parameterization information.

**Figure 62. IP Parameter Editor (Quartus Prime Pro Edition)**



**Figure 63. IP Parameter Editor (Quartus Prime Standard Edition)**



### 7.2.1.2. Specifying Parameters for the IP Catalog Flow

To specify parameters with the IP Catalog design flow, perform the following steps:

1. In the Quartus Prime software, create a Quartus Prime project using the **New Project Wizard** available from the File menu.
2. Launch the **IP Catalog** from the **Tools** menu.
3. Select an external memory interface IP from the **Memory Interfaces and Controllers** folder in the **Library** list.  
*Note:* The availability of external memory interface IP depends on the device family your design is using.
4. Depending on the window which appears, proceed as follows:
  - New IP Instance Window: Specify the Top-level Name and Device Settings, and click **Ok**.
  - Save IP Variation window: Specify the IP variation file name and IP variation file type, and click **Ok**.
5. In the **Presets** window, select the preset matching your design requirement, and click **Apply**.  
*Tip:* If none of the presets match your design requirements, you can apply the closest preset available and then change the parameters manually. This method may be faster than entering all the parameters manually, and reduces the chance of having incorrect settings.
6. Specify the parameters on all tabs.



- Note:**
- For detailed explanation of the parameters, refer to *Parameterizing Memory Controllers with UniPHY IP* and *Parameterizing Memory Controllers with Arria 10 External Memory Interface IP*.
  - Although you have applied presets, you may need to modify some of the preset parameters depending on the frequency of operation. A typical list of parameters which you might need to change includes the Memory CAS Latency setting, the Memory CAS Write Latency setting, and the tWTR, tFAW, tRRD, and tRTP settings.
- Tip:**
- As a good practice, review any warning messages displayed in the **Messages Window** and correct any errors before making further changes.
  - To simplify future work, you might want to store the current configuration by saving your own presets. To create, modify, or remove your own custom presets, click **New**, **Update**, or **Delete** at the bottom of the **Presets** list.
  - If you want to generate an example design for your current configuration, click **Example Design** at the top-right corner of the parameter editor, specify a path for the example design, and click **Ok**.
7. Depending on which external memory interface IP is selected, perform the following steps to complete the IP generation:



- For Arria 10 or Stratix 10 External Memory Interface IP:
    - a. Click **Finish**. Your configuration is saved as a .qsys file.
    - b. Click **Yes** when you are prompted to generate your IP.
    - c. Set **Create HDL design files for synthesis** to **Verilog** or **VHDL**.

*Tip:* If you want to do RTL simulation of your design, you should set **Create simulation model** to either **Verilog** or **VHDL**. Some RTL simulation-related files, including simulator-specific scripts, are generated only if you specify this parameter.

*Note:* For Arria 10 External Memory Interface IP, the synthesis and simulation model files are identical. However, there are some differences in file types when generating for VHDL. For synthesis files, only the top-level wrapper is generated in VHDL; the other files are generated in System Verilog. For simulation files, all the files are generated as a Mentor-tagged encrypted IP for VHDL-only simulator support.
    - d. Click **Generate**.
    - e. When generation has completed, click **Finish**.
  - For UniPHY-based IP:
    - a. Click the **Finish** button.

*Note:* The **Finish** button may be unavailable until you have corrected all parameterization errors listed in the **Messages** window.
    - b. If prompted, specify whether you want to generate an example design by checking or unchecking **Generate Example Design**, and then click **Generate**.

**Caution:** If you have already generated an example design, uncheck **Generate Example Design** to prevent your previously generated files from being overwritten.
    - c. When generation is completed, click **Exit**.
8. Click **Yes** if you are prompted to add the .qip to the current Quartus Prime project. You can also turn on **Automatically add Quartus Prime IP Files to all projects**.
- Tip:* Always read the generated `readme.txt` file, which contains information and guidelines specific to your configuration.
9. You can now integrate your custom IP core instance in your design, simulate, and compile. While integrating your IP core instance into your design, you must make appropriate pin assignments. You can create a virtual pin to avoid making specific pin assignments for top-level signals while you are simulating and not ready to map the design to hardware.

**Note:** For information about the Quartus Prime software, including virtual pins and the IP Catalog and Qsys, refer to Quartus Prime Help.

#### Related Information

- [Simulating Intel FPGA Designs](#)
- [Quartus Prime Help](#)



### 7.2.1.3. Using Example Designs

When you generate your IP, you can instruct the system to produce an example design consisting of an external memory interface IP of your configuration, together with a traffic generator.

For synthesis, the example design includes a project for which you can specify pin locations and a target device, compile in the Quartus Prime software, verify timing closure, and test on your board using the programming file generated by the Quartus Prime assembler. For simulation, the example design includes an example memory model with which you can run simulation and evaluate the result.

For a UniPHY-based external memory interface, click **Example Design** in the parameter editor, or enable **Generate Example Design**. The system produces an example design for synthesis in the `example_project` directory, and generation scripts for simulation in the `simulation` directory. To generate the complete example design for RTL simulation, follow the instructions in the `readme.txt` file in the `simulation` directory.

For Arria 10 External Memory Interface IP, click **Example Design** in the parameter editor. The system produces generation scripts in the directory path that you specify. To create a complete example design for synthesis or RTL simulation, follow the instructions in the generated `<variation_name>/altera_emif_arch_nf_140/<synth/sim>/<variation_name>_altera_emif_arch_nf_140_<unique ID>_readme.txt` file.

To compile an example design, open the `.qpf` file for the project and follow the standard design flow, including constraining the design prior to full compilation. If necessary, change the example project device to match the device in your project.

For more information about example designs, refer to *Functional Description—Example Top Level project* in Volume 3 of the *External Memory Interface Handbook*. For more information about simulating an example design, refer to *Simulating the Example Design* in the *Simulating Memory IP* chapter.

### 7.2.1.4. Constraining the Design

For Arria 10 External Memory Interface IP for HPS, pin location assignments are predefined in the Quartus Prime IP file (`.qip`). In UniPHY-based and non-HPS Arria 10 external memory interfaces, you must make your own location assignments.

**Note:** You should not overconstrain any EMIF IP-related registers unless you are advised to do so by Intel, or you fully understand the effect on the external memory interface operation. Also, ensure that any wildcards in your user logic do not accidentally target EMIF IP-related registers.

For more information about timing constraints and analysis, refer to *Analyzing Timing of Memory IP*.

#### 7.2.1.4.1. Adding Pins and DQ Group Assignments

The assignments defined in the `<variation_name>_pin_assignments.tcl` script (for UniPHY-based IP) and the Quartus Prime IP file (`.qip`) (for Arria 10 EMIF IP) help you to set up the I/O standards and the input/output termination for the external memory interface IP. These assignments also help to relate the DQ pin groups together for the Quartus Prime Fitter to place them correctly.



- For UniPHY-based external memory interfaces, run the `<variation_name>_pin_assignments.tcl` script to apply the input and output termination, I/O standards, and DQ group assignments to your design. To run the pin assignment script, follow these steps:
  - a. On the Processing menu, point to **Start**, and click **Start Analysis and Synthesis**. Allow **Analysis and Synthesis** to finish without errors before proceeding to step 2.
  - b. On the Tools menu click **Tcl Scripts**.
  - c. Specify the **pin\_assignments.tcl** and click **Run**.

The pin assignment script does not create a PLL reference clock for the design. You must create a clock for the design and provide pin assignments for the signals of both the example driver and testbench that the IP core variation generates.

**Note:** For some UniPHY-based IP configurations, the `afi_clk` clock does not have a global signal assignment constraint. In this case, you should add a suitable assignment for your design. For example, for a UniPHY-based DDR3 IP targeting a Stratix IV device, `if0|pll0|upll_memphy|auto_generated|clk[0]` does not have a global signal assignment and you should consider adding either a global clock or a dual regional clock assignment to your project for this clock.

- For Arria 10 External Memory Interface IP, the Quartus Prime software automatically reads assignments from the `.qip` file during compilation, so it is not necessary to apply assignments to your design manually.

**Note:**

- If you must overwrite the default assignments, ensure that you make your changes in the Quartus Prime Settings File (`.qsf`) and not the `.qip` file. Assignments in the `.qsf` file take precedence over assignments in the `.qip` file. Note also, that if you rerun the `<variation_name>_pin_assignments.tcl` file, it overwrites your changes.
- If the PLL input reference clock pin does not have the same I/O standard as the memory interface I/Os, a no-fit might occur because incompatible I/O standards cannot be placed in the same I/O bank.
- If you are upgrading your memory IP from an earlier Quartus Prime version, rerun the **pin\_assignments.tcl** script in the later Quartus Prime revision.
- If you encounter a shortage of clock resources, the AFI clock domain can be moved between regional, dual-regional, and global. Moving any other clock domain can result in fit errors or timing closure problems.

### 7.2.1.5. Compiling the Design

After constraining your design, compile your design in the Quartus Prime software to generate timing reports to verify whether timing has been met.

To compile the design, on the Processing menu, click **Start Compilation**.

After you have compiled the top-level file, you can perform RTL simulation or program your targeted Intel device to verify the top-level file in hardware.



**Note:** In UniPHY-based memory controllers, the `derive_pll_clocks` command can affect timing closure if it is called before the memory controller files are loaded. Ensure that the Quartus Prime IP File (.qip) appears in the file list before any Synopsys Design Constraint Files (.sdc) files that contain `derive_pll_clocks`.

For more information about simulating the memory IP, refer to *Simulating Memory IP*.

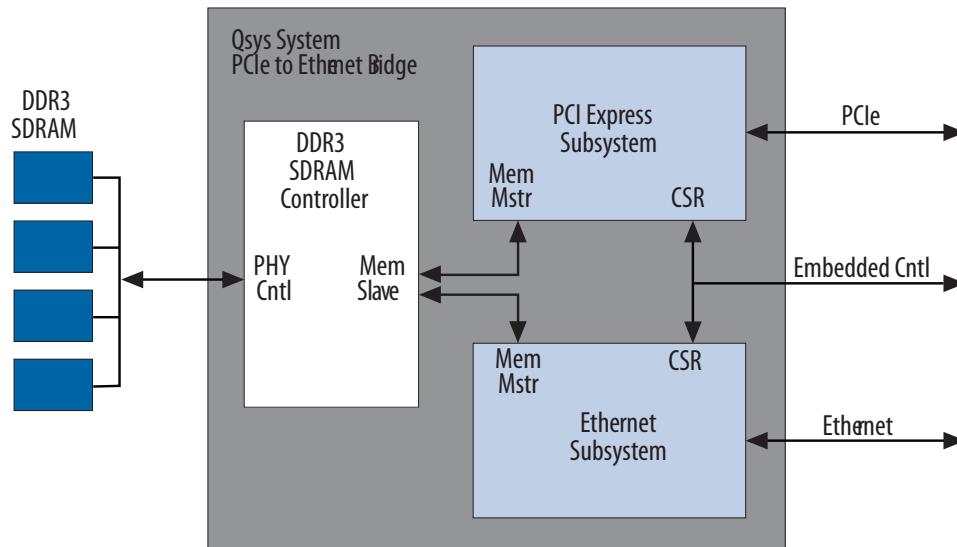
### 7.2.2. Qsys System Integration Tool Design Flow

You can use the Qsys system integration tool to build a system that includes your customized IP core.

You easily can add other components and quickly create a Qsys system. Qsys automatically generates HDL files that include all of the specified components and interconnections. In Qsys, you specify the connections you want. The HDL files are ready to be compiled by the Quartus Prime software to produce output files for programming an Intel device. Qsys generates Verilog HDL simulation models for the IP cores that comprise your system.

The following figure shows a high level block diagram of an example Qsys system.

**Figure 64. Example Qsys System**



For more information about the Qsys system interconnect, refer to the *Qsys Interconnect* chapter in volume 1 of the *Quartus Prime Handbook* and to the *Avalon Interface Specifications*.

For more information about the Qsys tool and the Quartus Prime software, refer to the *System Design with Qsys* section in volume 1 of the *Quartus Prime Handbook* and to Quartus Prime Help.

#### Related Information

- [Qsys Interconnect](#)
- [Avalon Interface Specifications](#)
- [System Design with Qsys](#)



### 7.2.2.1. Specify Parameters for the Qsys Flow

To specify parameters for your IP core using the Qsys flow, follow these steps:

1. In the Quartus Prime software, create a new Quartus Prime project using the **New Project Wizard** available from the **File** menu.
2. On the Tools menu, click **Qsys**.

*Note:* Qsys automatically sets device parameters based on your Quartus Prime project settings. To set device parameters manually, use the **Device Family** tab.

3. In the **IP Catalog**, select the available external memory interface IP from the **Memory Interfaces and Controllers** folder in the **Library** list. (For Arria 10 EMIF for HPS, select the external memory interface IP from the **Hard Processor Components** folder.) The relevant parameter editor appears.

*Note:* The availability of external memory interface IP depends on the device family your design is using. To use Arria 10 External Memory Interface for HPS IP, your design must target a device containing at least one HPS CPU core.

4. From the **Presets** list, select the preset matching your design requirement, and click **Apply**.

*Tip:* If none of the presets match your design requirements, you can apply the closest preset available and then change the inappropriate parameters manually. This method may be faster than entering all the parameters manually, and reduces the chance of having incorrect settings.

5. Specify the parameters on all tabs.

*Note:*

- For detailed explanation of the parameters, refer to *Parameterizing Memory Controllers with UniPHY IP* and *Parameterizing Memory Controllers with Arria 10 External Memory Interface IP*.
- Although you have applied presets, you may need to modify some of the preset parameters depending on the frequency of operation. A typical list of parameters which you might need to change includes the Memory CAS Latency setting, the Memory CAS Write Latency setting, and the tWTR, tFAW, tRRD, and tRTP settings.
- For UniPHY-based IP, turn on **Generate power-of-2 bus widths for Qsys or SOPC Builder** on the **Controller Settings** tab.

*Tip:*

- As a good practice, review any warning messages displayed in the **Messages Window** and correct any errors before making further changes.
- To simplify future work, you might want to store the current configuration by saving your own presets. To create, modify, or remove your own custom presets, click **New**, **Update**, or **Delete** at the bottom of the **Presets** list.
- If you want to generate an example design for your current configuration, click **Example Design** at the top-right corner of the parameter editor, specify a path for the example design, and click **Ok**.

6. Click **Finish** to complete the external memory interface IP instance and add it to the system.

*Note:* The **Finish** button may be unavailable until you have corrected all parameterization errors listed in the **Messages** window.



### 7.2.2.2. Completing the Qsys System

To complete the Qsys system, follow these steps:

1. Add and parameterize any additional components.
2. Connect the components using the Connection panel on the **System Contents** tab.
3. In the **Export** column, enter the name of any connections that should be a top-level Qsys system port.

*Note:* Ensure that the memory and oct interfaces are exported to the top-level Qsys system port. If these interfaces are already exported, take care not to accidentally rename or delete either of them in the **Export** column of the **System Contents** tab.

4. Click **Finish**.
5. Specify the **File Name** and click **Save**.
6. When you are prompted to generate now, click **Yes**.
7. Set **Create HDL design files for synthesis** to either Verilog or VHDL.

*Tip:* If you want to do RTL simulation of your design, you should set **Create simulation model** to either **Verilog** or **VHDL**. Some RTL simulation-related files, including simulator-specific scripts, are generated only if you specify this parameter.

*Note:* For Arria 10 External Memory Interface IP, the synthesis and simulation model files are identical. However, there are some differences in file types when generating for VHDL. For synthesis files, only the top-level wrapper is generated in VHDL; the other files are generated in System Verilog. For simulation files, all the files are generated as a Mentor-tagged encrypted IP for VHDL-only simulator support.

8. Click **Generate**.
9. When generation has completed, click **Finish**.
10. If you are prompted to add the .qip file to the current Quartus Prime project, click **Yes** (If you want, you can turn on **Automatically Add Quartus Prime IP Files to all projects**).

*Tip:* Always read the generated `readme.txt` file, because it contains information and guidelines specific to your configuration.

You can now simulate and compile your design. But before compilation, you must make appropriate pin assignments. You can create a virtual pin to avoid making specific pin assignments for top-level signals during simulation and not yet ready to map the design to hardware.

For information about the Quartus Prime software, including virtual pins and the IP Catalog and Qsys, refer to the Quartus Prime Help.

## 7.3. UniPHY-Based External Memory Interface IP

This section contains information about parameterizing UniPHY-based external memory interfaces.



### 7.3.1. Qsys Interfaces

The following tables list the signals available for each interface in Qsys, and provide a description and guidance on connecting those interfaces.

#### 7.3.1.1. DDR2 SDRAM Controller with UniPHY Interfaces

The following table lists the DDR2 SDRAM with UniPHY signals available for each interface in Qsys and provides a description and guidance on how to connect those interfaces.

**Table 56. DDR2 SDRAM Controller with UniPHY Interfaces**

Signals in Interface	Interface Type	Description/How to Connect
<b>pll_ref_clk interface</b>		
pll_ref_clk	Clock input	PLL reference clock input.
<b>global_reset interface</b>		
global_reset_n	Reset input	Asynchronous global reset for PLL and all logic in PHY.
<b>soft_reset interface</b>		
soft_reset_n	Reset input	Asynchronous reset input. Resets the PHY, but not the PLL that the PHY uses.
<b>afi_reset interface</b>		
afi_reset_n	Reset output (PLL master/no sharing)	When the interface is in PLL master or no sharing modes, this interface is an asynchronous reset output of the AFI interface. The controller asserts this interface when the PLL loses lock or the PHY is reset.
<b>afi_reset_export interface</b>		
afi_reset_export_n	Reset output (PLL master/no sharing)	This interface is a copy of the afi_reset interface. It is intended to be connected to PLL sharing slaves.
<b>afi_reset_in interface</b>		
afi_reset_n	Reset input (PLL slave)	When the interface is in PLL slave mode, this interface is a reset input that you must connect to the afi_reset_export_n output of an identically configured memory interface in PLL master mode.
<b>afi_clk interface</b>		
afi_clk	Clock output (PLL master/no sharing)	This AFI interface clock can be a full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_clk_in interface</b>		
afi_clk	Clock input (PLL slave)	This AFI interface clock can be a full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL slave mode, you must connect this afi_clk input to the afi_clk output of an identically configured memory interface in PLL master mode.
<b>afi_half_clk interface</b>		
<i>continued...</i>		



Signals in Interface	Interface Type	Description/How to Connect		
afi_half_clk	Clock output (PLL master/no sharing)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL master or no sharing modes, this interface is a clock output.		
<b>afi_half_clk_in interface</b>				
afi_half_clk	Clock input (PLL slave)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL slave mode, this is a clock input that you must connect to the afi_half_clk output of an identically configured memory interface in PLL master mode.		
<b>memory interface (DDR2 SDRAM)</b>				
mem_a	Conduit	Interface signals between the PHY and the memory device.		
mem_ba				
mem_ck				
mem_ck_n				
mem_cke				
mem_cs_n				
mem_dm				
mem_ras_n				
mem_cas_n				
mem_we_n				
mem_dq				
mem_dqs				
mem_dqs_n				
mem_odt				
mem_ac_parity				
mem_err_out_n				
mem_parity_error_n				
<b>memory interface (LPDDR2)</b>				
mem_ca	Conduit	Interface signals between the PHY and the memory device.		
mem_ck				
mem_ck_n				
mem_cke				
mem_cs_n				
mem_dm				
mem_dq				
mem_dqs				
mem_dqs_n				
<b>avl interface</b>				
<i>continued...</i>				



Signals in Interface	Interface Type	Description/How to Connect		
avl_ready	Avalon-MM Slave	Avalon-MM interface signals between the memory interface and user logic.		
avl_burst_begin				
avl_addr				
avl_rdata_valid				
avl_rdata				
avl_wdata				
avl_be				
avl_read_req				
avl_write_req				
avl_size				
<b>status interface</b>				
local_init_done	Conduit	Memory interface status signals.		
local_cal_success				
local_cal_fail				
<b>oct interface</b>				
rup (Stratix® III/IV, Arria® II GZ)	Conduit	OCT reference resistor pins for rup/rdn or rzqin.		
rdn (Stratix III/IV, Arria II GZ)				
rzq (Stratix V, Arria V, Cyclone V)				
<b>local_powerdown interface</b>				
local_powerdn_ack	Conduit	This powerdown interface for the controller is enabled only when you turn on <b>Enable Auto Powerdown</b> .		
<b>pll_sharing interface</b>				
pll_mem_clk	Conduit	Interface signals for PLL sharing, to connect PLL masters to PLL slaves. This interface is enabled only when you set <b>PLL sharing mode</b> to master or slave.		
pll_write_clk				
pll_addr_cmd_clk				
pll_locked				
pll_avl_clk				
pll_config_clk				
pll_hr_clk				
pll_p2c_read_clk				
pll_c2p_write_clk				
pll_dr_clk				
<b>dll_sharing interface</b>				
dll_delayctrl	Conduit	DLL sharing interface for connecting DLL masters to DLL slaves. This interface is enabled only when you set <b>DLL sharing mode</b> to master or slave.		
dll_pll_locked				
<b>oct_sharing interface</b>				
<i>continued...</i>				



Signals in Interface	Interface Type	Description/How to Connect
serieterminationcontrol	Conduit	OCT sharing interface for connecting OCT masters to OCT slaves. This interface is enabled only when you set <b>OCT sharing mode</b> to master or slave.
parallelterminationcontrol		
<b>autoprecharge_req interface</b>		
local_autopch_req	Conduit	Precharge interface for connection to a custom control block. This interface is enabled only when you turn on <b>Auto precharge Control</b> .
<b>user_refresh interface</b>		
local_refresh_req	Conduit	User refresh interface for connection to a custom control block. This interface is enabled only when you turn on <b>User Auto-Refresh Control</b> .
local_refresh_chip		
local_refresh_ack		
<b>self_refresh interface</b>		
local_self_rfsh_req	Conduit	Self refresh interface for connection to a custom control block. This interface is enabled only when you turn on <b>Self-refresh Control</b> .
local_self_rfsh_chip		
local_self_rfsh_ack		
<b>ecc_interrupt interface</b>		
ecc_interrupt	Conduit	ECC interrupt signal for connection to a custom control block. This interface is enabled only when you turn on <b>Error Detection and Correction Logic</b> .
<b>csr interface</b>		
csr_write_req	Avalon-MM Slave	Configuration and status register signals for the memory interface, for connection to an Avalon_MM master. This interface is enabled only when you turn on <b>Configuration and Status Register</b> .
csr_read_req		
csr_waitrequest		
csr_addr		
csr_be		
csr_wdata		
csr_rdata		
csr_rdata_valid		
<b>Hard Memory Controller MPFE FIFO Clock Interface</b>		
mp_cmd_clk	Conduit	When you enable the Hard Memory Interface, three FIFO buffers (command, read data, and write data) are created in the MPFE. Each FIFO buffer has its own clock and reset port. This interface is enabled when you turn on the Enable Hard Memory Interface.
mp_rfifo_clk		
mp_wfifo_clk		
mp_cmd_reset		
mp_rfifo_reset		
mp_wfifo_reset		
<b>Hard Memory Controller Bonding Interface</b>		
bonding_in_1	Conduit	Bonding interface to bond two controllers to expand the bandwidth. This interface is enabled when you turn on the Export bonding interface.
bonding_in_2		

*continued...*



Signals in Interface	Interface Type	Description/How to Connect
bonding_in_3		
bonding_out_1		
bonding_out_2		
bonding_out_3		
Note to Table:		
1.	Signals available only in DLL master mode.	

### 7.3.1.2. DDR3 SDRAM Controller with UniPHY Interfaces

The following table lists the DDR3 SDRAM with UniPHY signals available for each interface in Qsys and provides a description and guidance on how to connect those interfaces.

**Table 57. DDR3 SDRAM Controller with UniPHY Interfaces**

Signals in Interface	Interface Type	Description/How to Connect
<b>pll_ref_clk interface</b>		
pll_ref_clk	Clock input	PLL reference clock input.
<b>global_reset interface</b>		
global_reset_n	Reset input	Asynchronous global reset for PLL and all logic in PHY.
<b>soft_reset interface</b>		
soft_reset_n	Reset input	Asynchronous reset input. Resets the PHY, but not the PLL that the PHY uses.
<b>afi_reset interface</b>		
afi_reset_n	Reset output (PLL master/no sharing)	When the interface is in PLL master or no sharing modes, this interface is an asynchronous reset output of the AFI interface. This interface is asserted when the PLL loses lock or the PHY is reset.
<b>afi_reset_export interface</b>		
afi_reset_export_n	Reset output (PLL master/no sharing)	This interface is a copy of the afi_reset interface. It is intended to be connected to PLL sharing slaves.
<b>afi_reset_in interface</b>		
afi_reset_n	Reset input (PLL slave)	When the interface is in PLL slave mode, this interface is a reset input that you must connect to the afi_reset_export_n output of an identically configured memory interface in PLL master mode.
<b>afi_clk interface</b>		
afi_clk	Clock output (PLL master/no sharing)	This AFI interface clock can be full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_clk_in interface</b>		
afi_clk	Clock input (PLL slave)	This AFI interface clock can be full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL slave mode,

*continued...*



Signals in Interface	Interface Type	Description/How to Connect
		this is a clock input that you must connect to the afi_clk output of an identically configured memory interface in PLL master mode.
<b>afi_half_clk interface</b>		
afi_half_clk	Clock output (PLL master/no sharing)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_half_clk_in interface</b>		
afi_half_clk	Clock input (PLL slave)	The AFI half clock that is half the frequency of the afi_clk. When the interface is in PLL slave mode, you must connect this afi_half_clk input to the afi_half_clk output of an identically configured memory interface in PLL master mode.
<b>memory interface</b>		
mem_a	Conduit	Interface signals between the PHY and the memory device.
mem_ba		
mem_ck		
mem_ck_n		
mem_cke		
mem_cs_n		
mem_dm		
mem_ras_n		
mem_cas_n		
mem_we_n		
mem_dq		
mem_dqs		
mem_dqs_n		
mem_odt		
mem_reset_n		
mem_ac_parity		
mem_err_out_n		
mem_parity_error_n		
<b>avl interface</b>		
avl_ready	Avalon-MM Slave	Avalon-MM interface signals between the memory interface and user logic.
avl_burst_begin		
avl_addr		
avl_rdata_valid		
avl_rdata		
avl_wdata		

*continued...*



Signals in Interface	Interface Type	Description/How to Connect
avl_be		
avl_read_req		
avl_write_req		
avl_size		
<b>status interface</b>		
local_init_done	Conduit	Memory interface status signals.
local_cal_success		
local_cal_fail		
<b>oct interface</b>		
rup (Stratix III/IV, Arria II GZ)	Conduit	OCT reference resistor pins for rup/rdn or rzqin.
rdn (Stratix III/IV, Arria II GZ)		
rzq (Stratix V, Arria v, Cyclone V)		
<b>local_powerdown interface</b>		
local_powerdn_ack	Conduit	This powerdown interface for the controller is enabled only when you turn on <b>Enable Auto Power Down</b> .
<b>pll_sharing interface</b>		
pll_mem_clk	Conduit	Interface signals for PLL sharing, to connect PLL masters to PLL slaves. This interface is enabled only when you set <b>PLL sharing mode</b> to master or slave.
pll_write_clk		
pll_addr_cmd_clk		
pll_locked		
pll_avl_clk		
pll_config_clk		
pll_hr_clk		
pll_p2c_read_clk		
pll_c2p_write_clk		
pll_dr_clk		
<b>dll_sharing interface</b>		
dll_delayctrl	Conduit	DLL sharing interface for connecting DLL masters to DLL slaves. This interface is enabled only when you set <b>DLL sharing mode</b> to master or slave.
dll_pll_locked		
<b>oct_sharing interface</b>		
seriesterminationcontrol	Conduit	OCT sharing interface for connecting OCT masters to OCT slaves. This interface is enabled only when you set <b>OCT sharing mode</b> to master or slave.
parallelterminationcontrol		
<b>autoprecharge_req interface</b>		
local_autopch_req	Conduit	Precharge interface for connection to a custom control block. This interface is enabled only when you turn on <b>Auto-precharge Control</b> .
<b>user_refresh interface</b>		
<i>continued...</i>		



Signals in Interface	Interface Type	Description/How to Connect
local_refresh_req	Conduit	User refresh interface for connection to a custom control block. This interface is enabled only when you turn on <b>User Auto Refresh Control</b> .
local_refresh_chip		
local_refresh_ack		
<b>self_refresh interface</b>		
local_self_rfsh_req	Conduit	Self refresh interface for connection to a custom control block. This interface is enabled only when you turn on <b>Self-refresh Control</b> .
local_self_rfsh_chip		
local_self_rfsh_ack		
<b>ecc_interrupt interface</b>		
ecc_interrupt	Conduit	ECC interrupt signal for connection to a custom control block. This interface is enabled only when you turn on <b>Error Detection and Correction Logic</b> .
<b>csr interface</b>		
csr_write_req	Avalon-MM Slave	Configuration and status register signals for the memory interface, for connection to an Avalon_MM master. This interface is enabled only when you turn on <b>Configuration and Status Register</b> .
csr_read_req		
csr_waitrequest		
csr_addr		
csr_be		
csr_wdata		
csr_rdata		
csr_rdata_valid		
<b>Hard Memory Controller MPFE FIFO Clock Interface</b>		
mp_cmd_clk	Conduit	When you enable the Hard Memory Interface, three FIFO buffers (command, read data, and write data) are created in the MPFE. Each FIFO buffer has its own clock and reset port. This interface is enabled when you turn on the Enable Hard Memory Interface.
mp_rfifo_clk		
mp_wfifo_clk		
mp_cmd_reset_n		
mp_rfifo_reset_n		
mp_wfifo_reset_n		
<b>Hard Memory Controller Bonding Interface</b>		
bonding_in_1	Conduit	Use bonding interface to bond two controllers to expand the bandwidth. This interface is enabled when you turn on the Export bonding interface.
bonding_in_2		
bonding_in_3		
bonding_out_1		
bonding_out_2		
bonding_out_3		
Note to Table: 1. Signals available only in DLL master mode.		



### 7.3.1.3. LPDDR2 SDRAM Controller with UniPHY Interfaces

The following table lists the LPDDR2 SDRAM signals available for each interface in Qsys and provides a description and guidance on how to connect those interfaces.

**Table 58. LPDDR2 SDRAM Controller with UniPHY Interfaces**

Signals in Interface	Interface Type	Description/How to Connect
<b>pll_ref_clk interface</b>		
pll_ref_clk	Clock input	PLL reference clock input.
<b>global_reset interface</b>		
global_reset_n	Reset input	Asynchronous global reset for PLL and all logic in PHY.
<b>soft_reset interface</b>		
soft_reset_n	Reset input	Asynchronous reset input. Resets the PHY, but not the PLL that the PHY uses.
<b>afi_reset interface</b>		
afi_reset_n	Reset output (PLL master/no sharing)	When the interface is in PLL master or no sharing modes, this interface is an asynchronous reset output of the AFI interface. The controller asserts this interface when the PLL loses lock or the PHY is reset.
<b>afi_reset_export interface</b>		
afi_reset_export_n	Reset output (PLL master/no sharing)	This interface is a copy of the afi_reset interface. It is intended to be connected to PLL sharing slaves.
<b>afi_reset_in interface</b>		
afi_reset_n	Reset input (PLL slave)	When the interface is in PLL slave mode, this interface is a reset input that you must connect to the afi_reset_export_n output of an identically configured memory interface in PLL master mode.
<b>afi_clk interface</b>		
afi_clk	Clock output (PLL master/no sharing)	This AFI interface clock can be a full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_clk_in interface</b>		
afi_clk	Clock input (PLL slave)	This AFI interface clock can be a full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL slave mode, you must connect this afi_clk input to the afi_clk output of an identically configured memory interface in PLL master mode.
<b>afi_half_clk interface</b>		
afi_half_clk	Clock output (PLL master/no sharing)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_half_clk_in interface</b>		
afi_half_clk	Clock input (PLL slave)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL slave mode, this is a clock input that you must connect to the afi_half_clk output of an identically configured memory interface in PLL master mode.
<b>Memory interface</b>		
<i>continued...</i>		



Signals in Interface	Interface Type	Description/How to Connect
mem_ca	Conduit	Interface signals between the PHY and the memory device.
mem_ck		
mem_ck_n		
mem_cke		
mem_cs_n		
mem_dm		
mem_dq		
mem_dqs		
mem_dqs_n		
<b>avl interface</b>		
avl_ready	Avalon-MM Slave	Avalon-MM interface signals between the memory interface and user logic.
avl_burst_begin		
avl_addr		
avl_rdata_valid		
avl_rdata		
avl_wdata		
avl_be		
avl_read_req		
avl_write_req		
avl_size		
<b>status interface</b>		
local_init_done	Conduit	Memory interface status signals.
local_cal_success		
local_cal_fail		
<b>oct interface</b>		
rzq	Conduit	OCT reference resistor pins for rzqin.
<b>local_powerdn interface</b>		
local_powerdn_ack	Conduit	This powerdown interface for the controller is enabled only when you turn on Enable Auto Powerdown.
<b>local_deep_powerdn interface</b>		
local_deep_powerdn_ack	Conduit	Deep power down interface for the controller to enable deep power down. This interface is enable when turn on Enable Deep Power-Down Controls.
local_deep_powerdn_chip		
local_deep_powerdn_req		
<b>pll_sharing interface</b>		
pll_mem_clk	Conduit	Interface signals for PLL sharing, to connect PLL masters to PLL slaves. This interface is enabled only when you set PLL sharing mode to master or slave.
pll_write_clk		

*continued...*



Signals in Interface	Interface Type	Description/How to Connect
pll_addr_cmd_clk		
pll_locked		
pll_avl_clk		
pll_config_clk		
pll_mem_phy_clk		
afi_phy_clk		
pll_write_clk_pre_phy_clk		
<b>dll_sharing interface</b>		
dll_delayctrl	Conduit	DLL sharing interface for connecting DLL masters to DLL slaves. This interface is enabled only when you set DLL sharing mode to master or slave.
dll_pll_locked		
<b>oct_sharing interface</b>		
seriesterminationcontrol	Conduit	OCT sharing interface for connecting OCT masters to OCT slaves. This interface is enabled only when you set OCT sharing mode to master or slave.
parallelterminationcontrol		
<b>autoprecharge_req interface</b>		
local_autopch_req	Conduit	Precharge interface for connection to a custom control block. This interface is enabled only when you turn on Auto-precharge Control.
<b>user_refresh interface</b>		
local_refresh_req	Conduit	User refresh interface for connection to a custom control block. This interface is enabled only when you turn on User Auto-Refresh Control.
local_refresh_chip		
local_refresh_ack		
<b>self_refresh interface</b>		
local_self_rfsh_req	Conduit	Self refresh interface for connection to a custom control block. This interface is enabled only when you turn on Self-refresh Control.
local_self_rfsh_chip		
local_self_rfsh_ack		
<b>ecc_interrupt interface</b>		
ecc_interrupt	Conduit	ECC interrupt signal for connection to a custom control block. This interface is enabled only when you turn on Error Detection and Correction Logic.
<b>csr interface</b>		
csr_write_req	Avalon-MM Slave	Configuration and status register signals for the memory interface, for connection to an Avalon_MM master. This interface is enabled only when you turn on Configuration and Status Register.
csr_read_req		
csr_waitrequest		
csr_addr		
csr_be		
csr_wdata		
csr_rdata		
csr_rdata_valid		

*continued...*



Signals in Interface	Interface Type	Description/How to Connect
<b>Local_rdata_error interface</b>		
Local_rdata_error	Conduit	Indicates read data error when Error Detection and Correction logic is enabled.
<b>Hard Memory Controller MPFE FIFO Clock Interface</b>		
mp_cmd_clk	Conduit	When you enable the Hard Memory Interface, three FIFO buffers (command, read data, and write data) are created in the MPFE. Each FIFO buffer has its own clock and reset port. This interface is enabled when you turn on the Enable Hard Memory Interface.
mp_rfifo_clk		
mp_wfifo_clk		
mp_cmd_reset_n		
mp_rfifo_reset_n		
mp_wfifo_reset_n		
<b>Hard Memory Controller Bonding Interface</b>		
bonding_in_1	Conduit	Bonding interface to bond two controllers to expand the bandwidth. This interface is enabled when you turn on the Export bonding interface.
bonding_in_2		
bonding_in_3		
bonding_out_1		
bonding_out_2		
bonding_out_3		

#### 7.3.1.4. QDR II and QDR II+ SRAM Controller with UniPHY Interfaces

The following table lists the QDR II and QDR II+ SRAM signals available for each interface in Qsys and provides a description and guidance on how to connect those interfaces.

**Table 59. QDR II and QDR II+ SRAM Controller with UniPHY Interfaces**

Signals in Interface	Interface Type	Description/How to Connect
<b>pll_ref_clk interface</b>		
pll_ref_clk	Clock input	PLL reference clock input.
<b>global_reset interface</b>		
global_reset_n	Reset input	Asynchronous global reset for PLL and all logic in PHY.
<b>soft_reset interface</b>		
soft_reset_n	Reset input	Asynchronous reset input. Resets the PHY, but not the PLL that the PHY uses.
<b>afi_reset interface</b>		
afi_reset_n	Reset output (PLL master/no sharing)	When the interface is in PLL master or no sharing modes, this interface is an asynchronous reset output of the AFI interface. This interface is asserted when the PLL loses lock or the PHY is reset.
<b>afi_reset_export interface</b>		
<i>continued...</i>		



Signals in Interface	Interface Type	Description/How to Connect
afi_reset_export_n	Reset output (PLL master/no sharing)	This interface is a copy of the afi_reset interface. It is intended to be connected to PLL sharing slaves.
<b>afi_reset_in interface</b>		
afi_reset_n	Reset input (PLL slave)	When the interface is in PLL slave mode, this interface is a reset input that you must connect to the afi_reset_export_n output of an identically configured memory interface in PLL master mode.
<b>afi_clk interface</b>		
afi_clk	Clock output (PLL master/no sharing)	This AFI interface clock can be full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_clk_in interface</b>		
afi_clk	Clock input (PLL slave)	This AFI interface clock can be full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL slave mode, this is a clock input that you must connect to the afi_clk output of an identically configured memory interface in PLL master mode.
<b>afi_half_clk interface</b>		
afi_half_clk	Clock output (PLL master/no sharing)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_half_clk_in interface</b>		
afi_half_clk	Clock input (PLL slave)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL slave mode, you must connect this afi_half_clk input to the afi_half_clk output of an identically configured memory interface in PLL master mode.
<b>memory interface</b>		
mem_a	Conduit	Interface signals between the PHY and the memory device.
mem_cqn		The sequencer holds mem_doff_n low during initialization to ensure that internal PLL and DLL circuits in the memory device do not lock until clock signals have stabilized.
mem_bws_n		
mem_cq		
mem_d		
mem_k		
mem_k_n		
mem_q		
mem_wps_n		
mem_rps_n		
mem_doff_n		
<b>avl_r interface</b>		
<i>continued...</i>		



Signals in Interface	Interface Type	Description/How to Connect
avl_r_read_req	Avalon-MM Slave	Avalon-MM interface between memory interface and user logic for read requests.
avl_r_ready		
avl_r_addr		
avl_r_size		
avl_r_rdata_valid		
avl_r_rdata		
<b>avl_w interface</b>		
avl_w_write_req	Avalon-MM Slave	Avalon-MM interface between memory interface and user logic for write requests.
avl_w_ready		
avl_w_addr		
avl_w_size		
avl_w_wdata		
avl_w_be		
<b>status interface</b>		
local_init_done	Conduit	Memory interface status signals.
local_cal_success		
local_cal_fail		
<b>oct interface</b>		
rup (Stratix III/IV, Arria II GZ, Arria II GX)	Conduit	OCT reference resistor pins for rup/rdn or rzqin.
rdn (Stratix III/IV, Arria II GZ, Arria II GX)		
rzq (Stratix V, Arria V, Cyclone V)		
<b>pll_sharing interface</b>		
pll_mem_clk	Conduit	Interface signals for PLL sharing, to connect PLL masters to PLL slaves. This interface is enabled only when you set <b>PLL sharing mode</b> to master or slave.
pll_write_clk		
pll_addr_cmd_clk		
pll_locked		
pll_avl_clk		
pll_config_clk		
pll_hr_clk		
pll_p2c_read_clk		
pll_c2p_write_clk		
pll_dr_clk		
<b>dll_sharing interface</b>		
dll_delayctrl	Conduit	DLL sharing interface for connecting DLL masters to DLL slaves. This interface is enabled only when you set <b>DLL sharing mode</b> to master or slave.
dll_pll_locked		

*continued...*



Signals in Interface	Interface Type	Description/How to Connect
<b>oct_sharing interface</b>		
serieterminationcontrol(Stratix III/IV/V, Arria II GZ, Arria V, Cyclone V)	Conduit	OCT sharing interface for connecting OCT masters to OCT slaves. This interface is enabled only when you set <b>OCT sharing mode</b> to master or slave.
parallelterminationcontrol (Stratix III/IV/V, Arria II GZ, Arria V, Cyclone V)		
terminationcontrol (Arria II GX)		
Note to Table: 1. Signals available only in DLL master mode.		

### 7.3.1.5. RLDRAM II Controller with UniPHY Interfaces

The following table lists the RLDRAM II signals available for each interface in Qsys and provides a description and guidance on how to connect those interfaces.

**Table 60. RLDRAM II Controller with UniPHY Interfaces**

Interface Name	Interface Type	Description
<b>pll_ref_clk interface</b>		
pll_ref_clk	Clock input.	PLL reference clock input.
<b>global_reset interface</b>		
global_reset_n	Reset input	Asynchronous global reset for PLL and all logic in PHY.
<b>soft_reset interface</b>		
soft_reset_n	Reset input	Asynchronous reset input. Resets the PHY, but not the PLL that the PHY uses.
<b>afi_reset interface</b>		
afi_reset_n	Reset output (PLL master/no sharing)	When the interface is in PLL master or no sharing modes, this interface is an asynchronous reset output of the AFI interface. This interface is asserted when the PLL loses lock or the PHY is reset.
<b>afi_reset_export interface</b>		
afi_reset_export_n	Reset output (PLL master/no sharing)	This interface is a copy of the afi_reset interface. It is intended to be connected to PLL sharing slaves.
<b>afi_reset_in interface</b>		
afi_reset_n	Reset input (PLL slave)	When the interface is in PLL slave mode, this interface is a reset input that you must connect to the afi_reset_export_n output of an identically configured memory interface in PLL master mode.
<b>afi_clk interface</b>		
afi_clk	Clock output (PLL master/no sharing)	This AFI interface clock can be full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_clk_in interface</b>		
afi_clk	Clock input (PLL slave)	This AFI interface clock can be full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL slave mode,
		<b>continued...</b>



Interface Name	Interface Type	Description
		you must connect this afi_clk input to the afi_clk output of an identically configured memory interface in PLL master mode.
<b>afi_half_clk interface</b>		
afi_half_clk	Clock output (PLL master/no sharing)	The AFI half clock that is half the frequency of the afi_clk. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_half_clk_in interface</b>		
afi_half_clk	Clock input (PLL slave)	The AFI half clock that is half the frequency of the afi_clk. When the interface is in PLL slave mode, you must connect this afi_half_clk input to the afi_half_clk output of an identically configured memory interface in PLL master mode.
<b>memory interface</b>		
mem_a	Conduit	Interface signals between the PHY and the memory device.
mem_ba		
mem_ck		
mem_ck_n		
mem_cs_n		
mem_dk		
mem_dk_n		
mem_dm		
mem_dq		
mem_qk		
mem_qk_n		
mem_ref_n		
mem_we_n		
<b>avl interface</b>		
avl_size	Avalon-MM Slave	Avalon-MM interface between memory interface and user logic.
avl_wdata		
avl_rdata_valid		
avl_rdata		
avl_ready		
avl_write_req		
avl_read_req		
avl_addr		
<b>status interface</b>		
local_init_done	Conduit	Memory interface status signals.
local_cal_success		

*continued...*



Interface Name	Interface Type	Description
local_cal_fail		
<b>oct interface</b>		
rup (Stratix III/IV, Arria II GZ)	Conduit	OCT reference resistor pins for rup/rdn or rzqin.
rdn (Stratix III/IV, Arria II GZ)		
rzq (Stratix V)		
<b>pll_sharing interface</b>		
pll_mem_clk	Conduit	Interface signals for PLL sharing, to connect PLL masters to PLL slaves. This interface is enabled only when you set <b>PLL sharing mode</b> to master or slave.
pll_write_clk		
pll_addr_cmd_clk		
pll_locked		
pll_avl_clk		
pll_config_clk		
pll_hr_clk		
pll_p2c_read_clk		
pll_c2p_write_clk		
pll_dr_clk		
<b>dll_sharing interface</b>		
dll_delayctrl	Conduit	DLL sharing interface for connecting DLL masters to DLL slaves. This interface is enabled only when you set <b>DLL sharing mode</b> to master or slave.
<b>oct_sharing interface</b>		
seriesterminationcontrol	Conduit	OCT sharing interface for connecting OCT masters to OCT slaves. This interface is enabled only when you set <b>OCT sharing mode</b> to master or slave.
parallelterminationcontrol		
<b>parity_error_interrupt interface</b>		
parity_error	Conduit	Parity error interrupt conduit for connection to custom control block. This interface is enabled only if you turn on <b>Enable Error Detection Parity</b> .
<b>user_refresh interface</b>		
ref_req	Conduit	User refresh interface for connection to custom control block. This interface is enabled only if you turn on <b>Enable User Refresh</b> .
ref_ba		
ref_ack		
<b>reserved interface</b>		
reserved	Conduit	Reserved interface required for certain pin configurations when you select the Nios® II-based sequencer.
Note to Table: 1. Signals available only in DLL master mode.		

### 7.3.1.6. RLDRAM 3 UniPHY Interface

The following table lists the RLDRAM 3 signals available for each interface in Qsys and provides a description and guidance on how to connect those interfaces.



**Table 61. RLDRAM 3 UniPHY Interface**

Signals in Interface	Interface Type	Description/How to Connect
<b>pll_ref_clk interface</b>		
pll_ref_clk	Clock input	PLL reference clock input.
<b>global_reset interface</b>		
global_reset_n	Reset input	Asynchronous global reset for PLL and all logic in PHY.
<b>soft_reset interface</b>		
soft_reset_n	Reset input	Asynchronous reset input. Resets the PHY, but not the PLL that the PHY uses.
<b>afi_reset interface</b>		
afi_reset_n	Reset output (PLL master/no sharing)	When the interface is in PLL master or no sharing modes, this interface is an asynchronous reset output of the AFI interface. The controller asserts this interface when the PLL loses lock or the PHY is reset.
<b>afi_reset_export interface</b>		
afi_reset_export_n	Reset output (PLL master/no sharing)	This interface is a copy of the afi_reset interface. It is intended to be connected to PLL sharing slaves.
<b>afi_reset_in interface</b>		
afi_reset_n	Reset input (PLL slave)	When the interface is in PLL slave mode, this interface is a reset input that you must connect to the afi_reset_export_n output of an identically configured memory interface in PLL master mode.
<b>afi_clk interface</b>		
afi_clk	Clock output (PLL master/no sharing)	This AFI interface clock can be a full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_clk_in interface</b>		
afi_clk	Clock input (PLL slave)	This AFI interface clock can be a full-rate or half-rate memory clock frequency based on the memory interface parameterization. When the interface is in PLL slave mode, you must connect this afi_clk input to the afi_clk output of an identically configured memory interface in PLL master mode.
<b>afi_half_clk interface</b>		
afi_half_clk	Clock output (PLL master/no sharing)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL master or no sharing modes, this interface is a clock output.
<b>afi_half_clk_in interface</b>		
afi_half_clk	Clock input (PLL slave)	The AFI half clock that is half the frequency of afi_clk. When the interface is in PLL slave mode, this is a clock input that you must connect to the afi_half_clk output of an identically configured memory interface in PLL master mode.
<b>memory interface</b>		
mem_a	Conduit	Interface signals between the PHY and the memory device.
mem_ba		
mem_ck		

*continued...*



Signals in Interface	Interface Type	Description/How to Connect
mem_ck_n		
mem_cs_n		
mem_dk		
mem_dk_n		
mem_dm		
mem_dq		
mem_qk		
mem_qk_n		
mem_ref_n		
mem_we_n		
mem_reset_n		
<b>afi interface</b>		
afi_addr	Avalon-MM Slave	Altera PHY interface (AFI) signals between the PHY and controller.
afi_ba		
afi_cs_n		
afi_we_n		
afi_ref_n		
afi_wdata_valid		
afi_wdata		
afi_dm		
afi_rdata		
afi_rdata_en		
afi_rdata_en_full		
afi_rdata_valid		
afi_RST_n		
afi_cal_success		
afi_cal_fail		
afi_wlat		
afi_rlat		
<b>oct interface</b>		
oct_rzqin	Conduit	OCT reference resistor pins for rzqin.
<b>pll_sharing interface</b>		
pll_mem_clk	Conduit	Interface signals for PLL sharing, to connect PLL masters to PLL slaves. This interface is enabled only when you set PLL sharing mode to master or slave.
pll_write_clk		
pll_addr_cmd_clk		
pll_locked		

*continued...*



Signals in Interface	Interface Type	Description/How to Connect
pll_avl_clk		
pll_config_clk		
pll_mem_phy_clk		
afi_phy_clk		
pll_write_clk_pre_phy_clk		
pll_p2c_read_clk		
pll_c2p_write_clk		
<b>dll_sharing interface</b>		
dll_delayctrl	Conduit	DLL sharing interface for connecting DLL masters to DLL slaves. This interface is enabled only when you set DLL sharing mode to master or slave.
dll_pll_locked		
<b>oct_sharing interface</b>		
serieterminationcontrol	Conduit	OCT sharing interface for connecting OCT masters to OCT slaves. This interface is enabled only when you set OCT sharing mode to master or slave.
parallelterminationcontrol		

### 7.3.2. Generated Files for Memory Controllers with the UniPHY IP

When you complete the IP generation flow, there are generated files created in your project directory. The directory structure created varies somewhat, depending on the tool used to parameterize and generate the IP.

**Note:** The PLL parameters are statically defined in the `<variation_name>_parameters.tcl` at generation time. To ensure timing constraints and timing reports are correct, when you edit the PLL parameters, apply those changes to the PLL parameters in this file.

The following table lists the generated directory structure and key files created with the IP Catalog and Qsys.

**Table 62. Generated Directory Structure and Key Files for the IP Catalog Synthesis Files**

Directory	File Name	Description
<code>&lt;working_dir&gt;/</code>	<code>&lt;variation_name&gt;.qip</code>	Quartus Prime IP file which refers to all generated files in the synthesis fileset. Include this file in your Quartus Prime project.
<code>&lt;working_dir&gt;/</code>	<code>&lt;variation_name&gt;.v</code> or <code>&lt;variation_name&gt;.vhdl</code>	Top-level wrapper synthesis files. <code>.v</code> is IEEE Encrypted Verilog. <code>.vhdl</code> is generated VHDL.
<code>&lt;working_dir&gt;/&lt;variation_name&gt;/</code>	<code>&lt;variation_name&gt;_0002.v</code>	UniPHY top-level wrapper.
<code>&lt;working_dir&gt;/&lt;variation_name&gt;/</code>	<code>*.v, *.sv, *.tcl, *.sdc, *.ppf</code>	RTL and constraints files for synthesis.
<code>&lt;working_dir&gt;/&lt;variation_name&gt;/</code>	<code>&lt;variation_name&gt;_p0_pin_assignments.tcl</code>	Pin constraints script to be run after synthesis.

**Table 63. Generated Directory Structure and Key Files for the IP Catalog Simulation Files**

Directory	File Name	Description
<working_dir>/<variation_name>_sim/	<variation_name>.v	Top-level wrapper simulation files for both Verilog and VHDL.
<working_dir>/<variation_name>_sim/<subcomponent_module>/	*.v, *.sv, *.vhdl, *.vho, *hex, *.mif	RTL and constraints files for simulation. .v and .sv files are IEEE Encrypted Verilog. .vhdl and .vho are generated VHDL.

**Table 64. Generated Directory Structure and Key Files for the IP Catalog—Example Design Fileset Synthesis Files**

Directory	File Name	Description
<variation_name>_example_design/example_project/	<variation_name>_example.qip	Quartus Prime IP file that refers to all generated files in the synthesizable project.
<variation_name>_example_design/example_project/	<variation_name>_example.qpf	Quartus Prime project for synthesis flow.
<variation_name>_example_design/example_project/	<variation_name>_example.qsf	Quartus Prime project for synthesis flow.
<variation_name>_example_design/example_project/<variation_name>_example/	<variation_name>_example.v	Top-level wrapper.
<variation_name>_example_design/example_project/<variation_name>_example/submodules/	*.v, *.sv, *.tcl, *.sdc, *.ppf	RTL and constraints files.
<variation_name>_example_design/example_project/<variation_name>_example/submodules/	<variation_name>_example_if0_p0_pin_assignments.tcl	Pin constraints script to be run after synthesis. _if0 and _p0 are instance names.

**Table 65. Generated Directory Structure and Key Files for the IP Catalog—Example Design Fileset Simulation Files**

Directory	File Name	Description
<variation_name>_example_design/simulation/	generate_sim_verilog_example_design.tcl	Run this file to generate the Verilog simulation example design.
<variation_name>_example_design/simulation/	generate_sim_vhdl_example_design.tcl	Run this file to generate the VHDL simulation example design.
<variation_name>_example_design/simulation/	README.txt	A text file with instructions about how to generate and run the simulation example design.
<variation_name>_example_design/simulation/verilog/mentor	run.do	ModelSim* script to simulate the generated Verilog example design.
<variation_name>_example_design/simulation/vhdl/mentor	run.do	ModelSim script to simulate the generated VHDL example design.
<variation_name>_example_design/simulation/verilog/<variation_name>_sim/	<variation_name>_example_sim.v	Top-level wrapper (Testbench) for Verilog.

*continued...*



Directory	File Name	Description
<variation_name>_example_design /simulation/vhdl/<variation_name>_sim/	<variation_name>_example_sim.vhd	Top-level wrapper (Testbench) for VHDL.
<variation_name>_example_design /simulation/<variation_name>_sim/verilog/submodules/	*.v, *.sv, *.hex, *.mif	RTL and ROM data for Verilog.
<variation_name>_example_design /simulation/<variation_name>_sim/vhdl/submodules/	*.vhd, *.vho, *.hex, *.mif	RTL and ROM data for VHDL.

**Table 66. Generated Directory Structure and Key Files for Qsys**

Directory	File Name	Description
<working_dir>/<system_name>/synthesis/	<system_name>.qip	Quartus Prime IP file that refers to all the generated files in the synthesis fileset.
<working_dir>/<system_name>/synthesis/	<system_name>.v	System top-level RTL for synthesis.
<working_dir>/<system_name>/simulation/	<system_name>.v or <variation_name>.vhd	System top-level RTL for simulation. .v file is IEEE Encrypted Verilog. .vhd file is generated VHDL.
<working_dir>/<system_name>/synthesis/ submodules/	*.v, *.sv, *.tcl, *.sdc, *.ppf	RTL and constraints files for synthesis.
<working_dir>/<system_name>/simulation/ submodules/	*.v, *.sv, *.hex, *.mif	RTL and ROM data for simulation.

The following table lists the prefixes or instance names of submodule files within the memory interface IP. These instances are concatenated to form unique synthesis and simulation filenames.

**Table 67. Prefixes of Submodule Files**

Prefixes	Description
_c0	Specifies the controller.
_d0	Specifies the driver or traffic generator.
_dll0	Specifies the DLL.
_e0	Specifies the example design.
_if0	Specifies the memory Interface.
_m0	Specifies the AFI mux.
_oct0	Specifies the OCT.
_p0	Specifies the PHY.
_pll0	Specifies the PLL.
_s0	Specifies the sequencer.
_t0	Specifies the traffic generator status checker.



### 7.3.3. Parameterizing Memory Controllers

This section describes the parameters you can set for various UniPHY-based memory controllers.

#### Parameterizing Memory Controllers with UniPHY IP

The **Parameter Settings** page in the parameter editor allows you to parameterize the following settings for the LPDDR2, DDR2, DDR3 SDRAM, QDR II, QDR II+ SRAM, RLDRAM II, and RLDRAM 3 memory controllers with the UniPHY IP:

- PHY Settings
- Memory Parameters
- Memory Timing
- Board Settings
- Controller Settings
- Diagnostics

The messages window at the bottom of the parameter editor displays information about the memory interface, warnings, and errors if you are trying to create something that is not supported.

#### Enabling the Hard Memory Interface

For Arria V and Cyclone V devices, enable the hard memory interface by turning on **Interface Type > Enable Hard Memory Interface** in the parameter editor. The hard memory interface uses the hard memory controller and hard memory PHY blocks in the Arria V and Cyclone V devices.

The half-rate bridge option is available only as an SOPC Builder component, **Avalon-MM DDR Memory Half-Rate Bridge**, for use in a Qsys project.

#### 7.3.3.1. PHY Settings for UniPHY IP

The following table lists the PHY parameters for UniPHY-based EMIF IP.

**Table 68. PHY Parameters**

Parameter	Description
<b>General Settings</b>	
<b>Speed Grade</b>	Specifies the speed grade of the targeted FPGA device that affects the generated timing constraints and timing reporting.
<b>Generate PHY only</b>	Turn on this option to generate the UniPHY core without a memory controller. When you turn on this option, the AFI interface is exported so that you can easily connect your own memory controller. Not applicable to RLDRAM 3 UniPHY as no controller support for RLDRAM 3 UniPHY.
<b>Clocks</b>	
<b>Memory clock frequency</b>	The frequency of the clock that drives the memory device. Use up to 4 decimal places of precision. To obtain the maximum supported frequency for your target memory configuration, refer to the External Memory Interface Spec Estimator page on <a href="http://www.altera.com">www.altera.com</a> .

*continued...*



Parameter	Description
<b>Achieved memory clock frequency</b>	The actual frequency the PLL generates to drive the external memory interface (memory clock).
<b>PLL reference clock frequency</b>	The frequency of the input clock that feeds the PLL. Use up to 4 decimal places of precision.
<b>Rate on Avalon-MM interface</b>	<p>The width of data bus on the Avalon-MM interface. <b>Full</b> results in a width of <math>2 \times</math> the memory data width. <b>Half</b> results in a width of <math>4 \times</math> the memory data width. <b>Quarter</b> results in a width of <math>8 \times</math> the memory data width. Use <b>Quarter</b> for memory frequency 533 MHz and above.</p> <p>To determine the Avalon-MM interface rate selection for other memories, refer to the local interface clock rate for your target device in the External Memory Interface Spec Estimator page on <a href="http://www.altera.com">www.altera.com</a>.</p> <p><i>Note:</i> MAX 10 devices support only half-rate Avalon-MM interface.</p>
<b>Achieved local clock frequency</b>	The actual frequency the PLL generates to drive the local interface for the memory controller (AFI clock).
<b>Enable AFI half rate clock</b>	Export the afi_half_rate clock which is running half of the AFI clock rate to the top level.
<b>Advanced PHY Settings</b>	
<b>Advanced clock phase control</b>	<p>Enables access to clock phases. Default value should suffice for most DIMMs and board layouts, but can be modified if necessary to compensate for larger address and command versus clock skews.</p> <p>This option is available for DDR, DDR2 and DDR3 SDRAM only.</p> <p><i>Note:</i> This parameter is not available for MAX 10 devices.</p>
<b>Additional address and command clock phase</b>	<p>Allows you to increase or decrease the amount of phase shift on the address and command clock. The base phase shift center aligns the address and command clock at the memory device, which may not be the optimal setting under all circumstances. Increasing or decreasing the amount of phase shift can improve timing. The default value is 0 degrees.</p> <p>In DDR, DDR2, DDR3 SDRAM, and LPDDR2 SDRAM, you can set this value from -360 to 360 degrees. In QDRII/II+ SRAM and RLDRAM II, the available settings are -45, -22.5, 22.5, and 45.</p> <p>To achieve the optimum setting, adjust the value based on the address and command timing analysis results.</p> <p><i>Note:</i> This parameter is not available for MAX 10 devices.</p>
<b>Additional phase for core-to-periphery transfer</b>	<p>Allows you to phase shift the latching clock of the core-to-periphery transfers. By delaying the latch clock, a positive phase shift value improves setup timing for transfers between registers in the core and the half-rate DDIO_OUT blocks in the periphery, respectively. Adjust this setting according to the core timing analysis. The default value is 0 degrees. You can set this value from -179 to 179 degrees.</p> <p><i>Note:</i> This parameter is not available for MAX 10 devices.</p>
<b>Additional CK/CK# phase</b>	<p>Allows you to increase or decrease the amount of phase shift on the CK/CK# clock. The base phase shift center aligns the address and command clock at the memory device, which may not be the optimal setting under all circumstances. Increasing or decreasing the amount of phase shift can improve timing. Increasing or decreasing the phase shift on CK/CK# also impacts the read, write, and leveling transfers, which increasing or decreasing the phase shift on the address and command clocks does not.</p> <p>To achieve the optimum setting, adjust the value based on the address and command timing analysis results. Ensure that the read, write, and write leveling timings are met after adjusting the clock phase. Adjust this value when there is a core timing failure after adjusting <b>Additional address and command clock phase</b>.</p> <p>The default value is 0 degrees. You can set this value from -360 to 360 degrees. This option is available for LPDDR2, DDR, DDR2, and DDR3 SDRAM only.</p> <p><i>Note:</i> This parameter is not available for MAX 10 devices.</p>
<b>Supply voltage</b>	The supply voltage and sub-family type of memory.

*continued...*



Parameter	Description
	This option is available for DDR3 SDRAM only.
<b>I/O standard</b>	The I/O standard voltage. Set the I/O standard according to your design's memory standard.
<b>PLL sharing mode</b>	When you select <b>No sharing</b> , the parameter editor instantiates a PLL block without exporting the PLL signals. When you select <b>Master</b> , the parameter editor instantiates a PLL block and exports the signals. When you select <b>Slave</b> , the parameter editor exposes a PLL interface and you must connect an external PLL master to drive the PLL slave interface signals. Select <b>No sharing</b> if you are not sharing PLLs, otherwise select <b>Master</b> or <b>Slave</b> . For more information about resource sharing, refer to "The DLL and PLL Sharing Interface" section in the <i>Functional Description—UniPHY</i> chapter of the <i>External Memory Interface Handbook</i> . <i>Note:</i> This parameter is not available for MAX 10 devices.
<b>Number of PLL sharing interfaces</b>	This option allows you to specify the number of PLL sharing interfaces to create, facilitating creation of many one-to-one connections in Qsys flow. In Megawizard, you can select one sharing interface and manually connect the master to all the slaves. This option is enabled when you set <b>PLL sharing mode</b> to <b>Master</b> . <i>Note:</i> This parameter is not available for MAX 10 devices.
<b>DLL sharing mode</b>	When you select <b>No sharing</b> , the parameter editor instantiates a DLL block without exporting the DLL signals. When you select <b>Master</b> , the parameter editor instantiates a DLL block and exports the signals. When you select <b>Slave</b> , the parameter editor exposes a DLL interface and you must connect an external DLL master to drive the DLL slave signals. Select <b>No sharing</b> if you are not sharing DLLs, otherwise select <b>Master</b> or <b>Slave</b> . For more information about resource sharing, refer to "The DLL and PLL Sharing Interface" section in the <i>Functional Description—UniPHY</i> chapter of the <i>External Memory Interface Handbook</i> . <i>Note:</i> This parameter is not available for MAX 10 devices.
<b>Number of DLL sharing interfaces</b>	This option allows you to specify the number of DLL sharing interfaces to create, facilitating creation of many one-to-one connections in Qsys flow. In Megawizard, you can select one sharing interface and manually connect the master to all the slaves. This option is enabled when you set <b>PLL sharing mode</b> to <b>Master</b> . <i>Note:</i> This parameter is not available for MAX 10 devices.
<b>OCT sharing mode</b>	When you select <b>No sharing</b> , the parameter editor instantiates an OCT block without exporting the OCT signals. When you select <b>Master</b> , the parameter editor instantiates an OCT block and exports the signals. When you select <b>Slave</b> , the parameter editor exposes an OCT interface and you must connect an external OCT control block to drive the OCT slave signals. Select <b>No sharing</b> if you are not sharing OCT blocks, otherwise select <b>Master</b> or <b>Slave</b> . For more information about resource sharing, refer to "The OCT Sharing Interface" section in the <i>Functional Description—UniPHY</i> chapter of the <i>External Memory Interface Handbook</i> . <i>Note:</i> This parameter is not available for MAX 10 devices.
<b>Number of OCT sharing interfaces</b>	This option allows you to specify the number of OCT sharing interfaces to create, facilitating creation of many one-to-one connections in Qsys flow. In Megawizard, you can select one sharing interface and manually connect the master to all the slaves. This option is enabled when you set <b>PLL sharing mode</b> to <b>Master</b> .

*continued...*



Parameter	Description
	<i>Note:</i> This parameter is not available for MAX 10 devices.
<b>Reconfigurable PLL location</b>	When you set the PLL used in the UniPHY memory interface to be reconfigurable at run time, you must specify the location of the PLL. This assignment generates a PLL that can only be placed in the given sides.
<b>Sequencer optimization</b>	Select <b>Performance</b> to enable the Nios II-based sequencer, or <b>Area</b> to enable the RTL-based sequencer. Intel recommends that you enable the Nios-based sequencer for memory clock frequencies greater than 400 MHz and enable the RTL-based sequencer if you want to reduce resource utilization. This option is available for QDRII and QDR II+ SRAM, and RLDRAM II only. <i>Note:</i> This parameter is not available for MAX 10 devices.

#### Related Information

- External Memory Interface Spec Estimator
- Functional Description—UniPHY

### 7.3.3.2. Memory Parameters for LPDDR2, DDR2 and DDR3 SDRAM for UniPHY IP

The following table lists the memory parameters for LPDDR2, DDR2 and DDR3 SDRAM.

Use the **Memory Parameters** tab to apply the memory parameters from your memory manufacturer's data sheet.

**Table 69. Memory Parameters for LPDDR2, DDR2, and DDR3 SDRAM**

Parameter	Description
<b>Memory vendor</b>	The vendor of the memory device. Select the memory vendor according to the memory vendor you use. For memory vendors that are not listed in the setting, select JEDEC with the nearest memory parameters and edit the parameter values according to the values of the memory vendor that you use. However, if you select a configuration from the list of memory presets, the default memory vendor for that preset setting is automatically selected.
<b>Memory format</b>	The format of the memory device. Select <b>Discrete</b> if you are using just the memory device. Select <b>Unbuffered</b> or <b>Registered</b> for DIMM format. Use the DIMM format to turn on levelling circuitry for LPDDR2 support device only. DDR2 supports DIMM also.
<b>Number of clock enables per device/DIMM</b>	The number of clock enable pins per device or DIMM. This value also determines the number of ODT signals. (This parameter is available only when the selected memory format is <b>Registered</b> .) <i>Note:</i> This parameter is not available for MAX 10 devices.

*continued...*



Parameter	Description
<b>Number of chip selects per device/DIMM</b>	The number of chip selects per device or DIMM. This value is not necessarily the same as the number of ranks for RDIMMs or LRDIMMs. This value must be 2 or greater for RDIMMs or LRDIMMs. (This parameter is available only when the selected memory format is <b>Registered</b> .) <i>Note:</i> This parameter is not available for MAX 10 devices.
<b>Number of ranks per slot</b>	The number of ranks per DIMM slot. (This parameter is available only when the selected memory format is <b>Registered</b> .) <i>Note:</i> This parameter is not available for MAX 10 devices.
<b>Number of slots</b>	The number of DIMM slots. (This parameter is available only when the selected memory format is <b>Registered</b> .) <i>Note:</i> This parameter is not available for MAX 10 devices.
<b>Memory device speed grade</b>	The maximum frequency at which the memory device can run.
<b>Total interface width</b>	The total number of DQ pins of the memory device. Limited to 144 bits for DDR2 and DDR3 SDRAM (with or without leveling). The total interface is depending on the rate on Avalon-MM interface because the maximum Avalon data width is 1024. If you select 144 bit for total interface width with Quarter-rate, the avalon data width is 1152 exceeding maximum avalon data width.
<b>DQ/DQS group size</b>	The number of DQ bits per DQS group.
<b>Number of DQS groups</b>	The number of DQS groups is calculated automatically from the Total interface width and the DQ/DQS group size parameters.
<b>Number of chip selects (DDR2 and DDR3 SDRAM device only)</b>	The number of chip-selects the IP core uses for the current device configuration. Specify the total number of chip-selects according to the number of memory device.
<b>Number of clocks</b>	The width of the clock bus on the memory interface.
<b>Row address width</b>	The width of the row address on the memory interface.
<b>Column address width</b>	The width of the column address on the memory interface.

*continued...*



Parameter	Description
<b>Bank-address width</b>	The width of the bank address bus on the memory interface.
<b>Enable DM pins</b>	Specifies whether the DM pins of the memory device are driven by the FPGA. You can turn off this option to avoid overusing FPGA device pins when using x4 mode memory devices. When you are using x4 mode memory devices, turn off this option for DDR3 SDRAM. You must turn on this option if you are using Avalon byte enable.
<b>DQS# Enable (DDR2)</b>	Turn on differential DQS signaling to improve signal integrity and system performance. This option is available for DDR2 SDRAM only.

### 7.3.3.2.1. Memory Initialization Options for DDR2

Memory Initialization Options—DDR2		
<b>Address and command parity</b>		Enables address/command parity checking. This is required for Registered DIMM.
<b>Mode Register 0</b>	<b>Burst length</b>	Specifies the burst length.
	<b>Read burst type</b>	Specifies accesses within a given burst in sequential or interleaved order. Specify sequential ordering for use with the Intel memory controller. Specify interleaved ordering only for use with an interleaved-capable custom controller, when the <b>Generate PHY only</b> parameter is enabled on the PHY Settings tab.
	<b>DLL precharge power down</b>	Determines whether the DLL in the memory device is in slow exit mode or in fast exit mode during precharge power down. For more information, refer to memory vendor data sheet.
	<b>Memory CAS latency setting</b>	Determines the number of clock cycles between the READ command and the availability of the first bit of output data at the memory device. For more information, refer to memory vendor data sheet speed bin table. Set this parameter according to the target memory speed grade and memory clock frequency.
<b>Mode Register 1</b>	<b>Output drive strength setting</b>	Determines the output driver impedance setting at the memory device. To obtain the optimum signal integrity performance, select the optimum setting based on the board simulation results.

*continued...*



Memory Initialization Options—DDR2		
	<b>Memory additive CAS latency setting</b>	Determines the posted CAS additive latency of the memory device. Enable this feature to improve command and bus efficiency, and increase system bandwidth. For more information, refer to the <i>Optimizing the Controller</i> chapter.
	<b>Memory on-die termination (ODT) setting</b>	Determines the on-die termination resistance at the memory device. To obtain the optimum signal integrity performance, select the optimum setting based on the board simulation results.
<b>Mode Register 2</b>	<b>SRT Enable</b>	Determines the selfrefresh temperature (SRT). Select <b>1x refresh rate</b> for normal temperature (0-85C) or select <b>2x refresh rate</b> for high-temperature (>85C).

### 7.3.3.2.2. Memory Initialization Options for DDR3

Memory Initialization Options—DDR3		
<b>Mirror Addressing: 1 per chip select</b>		Specifies the mirror addressing for multiple rank DIMMs. Refer to memory vendor data sheet for more information. Enter ranks with mirrored addresses in this field. For example, for four chip selects, enter 1101 to mirror the address on chip select #3, #2, and #0.
<b>Address and command parity</b>		Enables address/command parity checking to detect errors in data transmission. This is required for registered DIMM (RDIMM).
<b>Mode Register 0</b>	<b>Read burst type</b>	Specifies accesses within a given burst in sequential or interleaved order. Specify sequential ordering for use with the Intel memory controller. Specify interleaved ordering only for use with an interleaved-capable custom controller, when the <b>Generate PHY only</b> parameter is enabled on the PHY Settings tab.
	<b>DLL precharge power down</b>	Specifies whether the DLL in the memory device is off or on during precharge power-down.
	<b>Memory CAS latency setting</b>	The number of clock cycles between the read command and the availability of the first bit of output data at the memory device and also interface frequency. Refer to memory vendor data sheet speed bin table. Set this parameter according to the target memory speed grade and memory clock frequency.
<b>Mode Register 1</b>	<b>Output drive strength setting</b>	The output driver impedance setting at the memory device. To obtain the optimum signal integrity performance, select the optimum setting based on the board simulation results.
	<b>Memory additive CAS latency setting</b>	The posted CAS additive latency of the memory device.

*continued...*



Memory Initialization Options—DDR3		
		Enable this feature to improve command and bus efficiency, and increase system bandwidth. For more information, refer to the <i>Optimizing the Controller</i> chapter.
	<b>ODT Rtt nominal value</b>	The on-die termination resistance at the memory device. To obtain the optimum signal integrity performance, select the optimum setting based on the board simulation results.
<b>Mode Register 2</b>	<b>Auto selfrefresh method</b>	Disable or enable auto selfrefresh.
	<b>Selfrefresh temperature</b>	Specifies the selfrefresh temperature as <b>Normal</b> or <b>Extended</b> .
	<b>Memory write CAS latency setting</b>	The number of clock cycles from the releasing of the internal write to the latching of the first data in, at the memory device and also interface frequency. Refer to memory vendor data sheet speed bin table and set according to the target memory speed grade and memory clock frequency.
	<b>Dynamic ODT (Rtt_WR) value</b>	The mode of the dynamic ODT feature of the memory device. This is used for multi-rank configurations. Refer to <i>DDR2 and DDR3 SDRAM Board Layout Guidelines</i> . To obtain the optimum signal integrity performance, select the optimum setting based on the board simulation results.
	<b>DDR3 RDIMM/LRDIMM Control Words</b>	The memory device features a set of control words of SSTE32882 registers. These 4-bit control words of serial presence-detect (SPD) information allow the controller to optimize device properties to match different DIMM net topologies. You can obtain the control words from the memory manufacturer's data sheet. You enter each word in hexadecimal, starting with RC15 on the left and ending with RC0 on the right. <i>Note:</i> This parameter is not available for MAX 10 devices.
	<b>LRDIMM Additional Control Words</b>	The memory device features a set of control words of SSTE32882 registers. These 4-bit control words of serial presence-detect (SPD) information allow the controller to optimize device properties to match different DIMM net topologies. You can obtain the control words from the memory manufacturer's data sheet. You enter each word in hexadecimal, starting with SPD(77-72) or SPD(83-78) on the left and ending with SPD(71-69) on the right. <i>Note:</i> This parameter is not available for MAX 10 devices.

### 7.3.3.2.3. Memory Initialization Options for LPDDR2



Memory Initialization Options—LPDDR2		
<b>Mode Register 1</b>	<b>Burst Length</b>	Specifies the burst length.
	<b>Read Burst Type</b>	Specifies accesses within a given burst in sequential or interleaved order. Specify sequential ordering for use with the Intel memory controller. Specify interleaved ordering only for use with an interleaved-capable custom controller, when the <b>Generate PHY only</b> parameter is enabled on the PHY Settings tab.
<b>Mode Register 2</b>	<b>Memory CAS latency setting</b>	Determines the number of clock cycles between the READ command and the availability of the first bit of output data at the memory device. Set this parameter according to the target memory interface frequency. Refer to memory data sheet and also target memory speed grade.
<b>Mode Register 3</b>	<b>Output drive strength settings</b>	Determines the output driver impedance setting at the memory device. To obtain the optimum signal integrity performance, select the optimum setting based on the board simulation results.

### 7.3.3.3. Memory Parameters for QDR II and QDR II+ SRAM for UniPHY IP

The following table describes the memory parameters for QDR II and QDR II+ SRAM for UniPHY IP.

Use the **Memory Parameters** tab to apply the memory parameters from your memory manufacturer's data sheet.

**Table 70. Memory Parameters for QDR II and QDR II+ SRAM**

Parameter	Description
<b>Address width</b>	The width of the address bus on the memory device.
<b>Data width</b>	The width of the data bus on the memory device.
<b>Data-mask width</b>	The width of the data-mask on the memory device,
<b>CQ width</b>	The width of the CQ (read strobe) bus on the memory device.
<b>K width</b>	The width of the K (write strobe) bus on the memory device.
<b>Burst length</b>	The burst length supported by the memory device. For more information, refer to memory vendor data sheet.
<b>Topology</b>	
<b>x36 emulated mode</b>	Emulates a larger memory-width interface using smaller memory-width interfaces on the FPGA. Turn on this option when the target FPGA do not support x36 DQ/DQS group. This option allows two x18 DQ/DQS groups to emulate 1 x36 read data group.
<b>Emulated write groups</b>	Number of write groups to use to form the x36 memory interface on the FPGA. Select <b>2</b> to use 2 x18 DQ/DQS group to form x36 write data group. Select <b>4</b> to use 4 x9 DQ/DQS group to form x36 write data group.
<b>Device width</b>	Specifies the number of memory devices used for width expansion.



### 7.3.3.4. Memory Parameters for RLDRAM II for UniPHY IP

The following table describes the memory parameters for RLDRAM II.

Use the **Memory Parameters** tab to apply the memory parameters from your memory manufacturer's data sheet.

**Table 71. Memory Parameters for RLDRAM II**

Parameter	Description
<b>Address width</b>	The width of the address bus on the memory device.
<b>Data width</b>	The width of the data bus on the memory device.
<b>Bank-address width</b>	The width of the bank-address bus on the memory device.
<b>Data-mask width</b>	The width of the data-mask on the memory device,
<b>QK width</b>	The width of the QK (read strobe) bus on the memory device. Select <b>1</b> when data width is set to 9. Select <b>2</b> when data width is set to 18 or 36.
<b>DK width</b>	The width of the DK (write strobe) bus on the memory device. Select <b>1</b> when data width is set to 9 or 18. Select <b>2</b> when data width is set to 36.
<b>Burst length</b>	The burst length supported by the memory device. For more information, refer to memory vendor data sheet.
<b>Memory mode register configuration</b>	Configuration bits that set the memory mode. Select the option according to the interface frequency.
<b>Device impedance</b>	Select External (ZQ) to adjust the driver impedance using the external impedance resistor (RQ). The output impedance range is 25-60 $\Omega$ . You must connect the RQ resistor between ZQ pin and ground. The value of RQ must be 5 times the output impedance. For example, 60 $\Omega$ output impedance requires 300 $\Omega$ RQ. Set the value according to the board simulation.
<b>On-Die Termination</b>	Turn on this option to enable ODT in the memory to terminate the DQs and DM pins to Vtt. Dynamically switch off during read operation and switch on during write operation. Refer to memory vendor data sheet for more information.
<b>Topology</b>	
<b>Device width</b>	Specifies the number of memory devices used for width expansion.

### 7.3.3.5. Memory Timing Parameters for DDR2, DDR3, and LPDDR2 SDRAM for UniPHY IP

The following table lists the memory timing parameters for DDR2, DDR3, and LPDDR2 SDRAM.

Use the **Memory Timing** tab to apply the memory timings from your memory manufacturer's data sheet.

**Table 72. Parameter Description**

Parameter	Protocol	Description
<b>tIS (base)</b>	DDR2, DDR3, LPDDR2	Address and control setup to CK clock rise. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tIH (base)</b>	DDR2, DDR3, LPDDR2	Address and control hold after CK clock rise. Set according to the memory speed grade and refer to the memory vendor data sheet.
		<i>continued...</i>



Parameter	Protocol	Description
<b>tDS (base)</b>	DDR2, DDR3, LPDDR2	Data setup to clock (DQS) rise. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDH (base)</b>	DDR2, DDR3, LPDDR2	Data hold after clock (DQS) rise. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDQSQ</b>	DDR2, DDR3, LPDDR2	DQS, DQS# to DQ skew, per access. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tQHS</b>	DDR2, LPDDR2	DQ output hold time from DQS, DQS# (absolute time value)
<b>tQH</b>	DDR3	DQ output hold time from DQS, DQS# (percentage of tCK). Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDQSCK</b>	DDR2, DDR3, LPDDR2	DQS output access time from CK/CK#. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDQSCK Delta Short</b>	LPDDR2	Absolute difference between any two tDQSCK measurements (within a byte lane) within a contiguous sequence of bursts in a 160ns rolling window. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDQSCK Delta Medium</b>	LPDDR2	Absolute difference between any two tDQSCK measurements (within a byte lane) within a contiguous sequence of bursts in a 1.6us rolling window. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDQSCK Delta Long</b>	LPDDR2	Absolute difference between any two tDQSCK measurements (within a byte lane) within a contiguous sequence of bursts in a 32ms rolling window. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDQSS</b>	DDR2, DDR3, LPDDR2	First latching edge of DQS to associated clock edge (percentage of tCK). Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDQSH</b>	DDR2, LPDDR2	DQS Differential High Pulse Width (percentage of tCK). Specifies the minimum high time of the DQS signal received by the memory. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tQSH</b>	DDR3	
<b>tDSH</b>	DDR2, DDR3, LPDDR2	DQS falling edge hold time from CK (percentage of tCK). Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tDSS</b>	DDR2, DDR3, LPDDR2	DQS falling edge to CK setup time (percentage of tCK). Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tINIT</b>	DDR2, DDR3, LPDDR2	Memory initialization time at power-up. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tMRD</b>	DDR2, DDR3	Load mode register command period. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tMRW</b>	LPDDR2	

*continued...*



Parameter	Protocol	Description
<b>tRAS</b>	DDR2, DDR3, LPDDR2	Active to precharge time. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tRCD</b>	DDR2, DDR3, LPDDR2	Active to read or write time. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tRP</b>	DDR2, DDR3, LPDDR2	Precharge command period. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tREFI</b>	DDR2, DDR3	Refresh command interval. (All banks only for LPDDR2.) Set according to the memory speed grade and temperature range. Refer to the memory vendor data sheet.
<b>tREFICab</b>	LPDDR2	
<b>tRFC</b>	DDR2, DDR3	Auto-refresh command interval. (All banks only for LPDDR2.) Set according to the memory device capacity. Refer to the memory vendor data sheet.
<b>tRFCab</b>	LPDDR2	
<b>tWR</b>	DDR2, DDR3, LPDDR2	Write recovery time. Set according to the memory speed grade and refer to the memory vendor data sheet.
<b>tWTR</b>	DDR2, DDR3, LPDDR2	Write to read period. Set according to the memory speed grade and memory clock frequency. Refer to the memory vendor data sheet. Calculate the value based on the memory clock frequency.
<b>tFAW</b>	DDR2, DDR3, LPDDR2	Four active window time. Set according to the memory speed grade and page size. Refer to the memory vendor data sheet.
<b>tRRD</b>	DDR2, DDR3, LPDDR2	RAS to RAS delay time. Set according to the memory speed grade, page size and memory clock frequency. Refer to the memory vendor data sheet. Calculate the value based on the memory interface frequency and memory clock frequency.
<b>tRTP</b>	DDR2, DDR3, LPDDR2	Read to precharge time. Set according to memory speed grade. Refer to the memory vendor data sheet. Calculate the value based on the memory interface frequency and memory clock frequency.

### 7.3.3.6. Memory Timing Parameters for QDR II and QDR II+ SRAM for UniPHY IP

The following table lists the memory timing parameters for QDR II and QDR II+ SRAM.

Use the **Memory Timing** tab to apply the memory timings from your memory manufacturer's data sheet.

**Table 73. Parameter Description**

Parameter	Description
<b>QDR II and QDR II+ SRAM</b>	
<b>tWL (cycles)</b>	The write latency. Set write latency 0 for burst length of 2, and set write latency to 1 for burst length of 4.
<b>tRL (cycles)</b>	The read latency. Set according to memory protocol. Refer to memory data sheet.
<b>tSA</b>	The address and control setup to K clock rise. Set according to memory protocol. Refer to memory data sheet.

*continued...*



Parameter	Description
<b>tHA</b>	The address and control hold after K clock rise. Set according to memory protocol. Refer to memory data sheet.
<b>tSD</b>	The data setup to clock (K/K#) rise. Set according to memory protocol. Refer to memory data sheet.
<b>tHD</b>	The data hold after clock (K/K#) rise. Set according to memory protocol. Refer to memory data sheet.
<b>tCQD</b>	Echo clock high to data valid. Set according to memory protocol. Refer to memory data sheet.
<b>tCQDOH</b>	Echo clock high to data invalid. Set according to memory protocol. Refer to memory data sheet.
<b>Internal jitter</b>	The QDRII/II+ internal jitter. Refer to memory data sheet.
<b>TCQHCQnH</b>	The CQ clock rise to CQn clock rise (rising edge to rising edge). Set according to memory speed grade. Refer to memory data sheet.
<b>TKHKnH</b>	The K clock rise to Kn clock rise (rising edge to rising edge). Set according to memory speed grade. Refer to memory data sheet.

### 7.3.3.7. Memory Timing Parameters for RLDRAM II for UniPHY IP

The following table lists the memory timing parameters for RLDRAM II.

Use the **Memory Timing** tab to apply the memory timings from your memory manufacturer's data sheet.

**Table 74. Memory Timing Parameters**

Parameter	Description
<b>RLDRAM II</b>	
<b>Maximum memory clock frequency</b>	The maximum frequency at which the memory device can run. Set according to memory speed grade. Refer to memory data sheet.
<b>Refresh interval</b>	The refresh interval. Set according to memory speed grade. Refer to memory data sheet.
<b>tCKH (%)</b>	The input clock (CK/CK#) high expressed as a percentage of the full clock period. Set according to memory speed grade. Refer to memory data sheet.
<b>tQKH (%)</b>	The read clock (QK/QK#) high expressed as a percentage of tCKH. Set according to memory speed grade. Refer to memory data sheet.
<b>tAS</b>	Address and control setup to CK clock rise. Set according to memory speed grade. Refer to memory data sheet.
<b>tAH</b>	Address and control hold after CK clock rise. Set according to memory speed grade. Refer to memory data sheet.
<b>tDS</b>	Data setup to clock (CK/CK#) rise. Set according to memory speed grade. Refer to memory data sheet.
<b>tDH</b>	Data hold after clock (CK/CK#) rise. Set according to memory speed grade. Refer to memory data sheet.
<b>tQKQ_max</b>	QK clock edge to DQ data edge (in same group). Set according to memory speed grade. Refer to memory data sheet.

*continued...*



Parameter	Description
<b>tQKQ_min</b>	QK clock edge to DQ data edge (in same group). Set according to memory speed grade. Refer to memory data sheet.
<b>tCKDK_max</b>	Clock to input data clock (max). Set according to memory speed grade. Refer to memory data sheet.
<b>tCKDK_min</b>	Clock to input data clock (min). Set according to memory speed grade. Refer to memory data sheet.

### 7.3.3.8. Memory Parameters for RLDRAM 3 for UniPHY IP

The following tables list the memory parameters for RLDRAM 3 for UniPHY IP.

Use the **Memory Timing** tab to apply the memory timings from your memory manufacturer's data sheet.

**Table 75. Memory Parameters for RLDRAM 3 for UniPHY**

Parameter	Description
<b>Enable data-mask pins</b>	Specifies whether the DM pins of the memory device are driven by the FPGA.
<b>Data-mask width</b>	The width of the data-mask on the memory device.
<b>Data width</b>	The width of the data bus on the memory device.
<b>QK width</b>	The width of the QK (read strobe) bus on the memory device. Select 2 when data width is set to 18. Select 4 when data width is set to 36.
<b>DK width</b>	The width of the DK (write strobe) bus on the memory device. For x36 device, DQ[8:0] and DQ[26:18] are referenced to DK0/DK0#, and DQ[17:9] and DQ[35:27] are referenced to DK1/DK1#.
<b>Address width</b>	The width of the address bus on the memory device.
<b>Bank-address width</b>	The width of the bank-address bus on the memory device.
<b>Burst length</b>	The burst length supported by the memory device. Refer to memory vendor data sheet.
<b>tRC</b>	Mode register bits that set the tRC. Set the tRC according to the memory speed grade and data latency. Refer to the tRC table in the memory vendor data sheet.
<b>Data Latency</b>	Mode register bits that set the latency. Set latency according to the interface frequency and memory speed grade. Refer to speed bin table in the memory data sheet.
<b>Output Drive</b>	Mode register bits that set the output drive impedance setting. Set the value according to the board simulation.
<b>ODT</b>	Mode register bits that set the ODT setting. Set the value according to the board simulation.
<b>AREF Protocol</b>	Mode register setting for refreshing memory content of a bank. Select <b>Multibank</b> to allow refresh 4 bank simultaneously. Select Bank Address Control to refresh a particular bank by setting the bank address.
<b>Write Protocol</b>	Mode register setting for write protocol. When multiple bank (dual bank or quad bank) is selected, identical data is written to multiple banks.
<b>Topology</b>	
<b>Device width</b>	Specifies the number of memory devices used for width expansion.

**Table 76. Memory Timing Parameters for RLDRAM 3 for UniPHY**

Parameter	Description
<b>Memory Device Timing</b>	
<b>Maximum memory clock frequency</b>	The maximum frequency at which the memory device can run.
<b>tDS (base)</b>	Base specification for data setup to DK/DK#. Set according to memory speed grade. Refer to memory data sheet.
<b>tDH (base)</b>	Base specification for data hold from DK/DK#. Set according to memory speed grade. Refer to memory data sheet.
<b>tQKQ_max</b>	QK/QK# clock edge to DQ data edge (in same group). Set according to memory speed grade. Refer to memory data sheet.
<b>tQH (% of CK)</b>	DQ output hold time from QK/QK#. Set according to memory speed grade. Refer to memory data sheet.
<b>tCKDK_max(% of CK)</b>	Clock to input data clock (max). Set according to memory speed grade. Refer to memory data sheet.
<b>tCKDK_min (% of CK)</b>	Clock to input data clock (min). Set according to memory speed grade. Refer to memory data sheet.
<b>tCKQK_max</b>	QK edge to clock edge skew (max). Set according to memory speed grade. Refer to memory data sheet.
<b>tIS (base)</b>	Base specification for address and control setup to CK. Set according to memory speed grade. Refer to memory data sheet.
<b>tIH (base)</b>	Base specification for address and control hold from CK. Set according to memory speed grade. Refer to memory data sheet.
<b>Controller Timing</b>	
<b>Read-to-Write NOP commands (min)</b>	Minimum number of no operation commands following a read command and before a write command. The value must be at least ((Burst Length/2) + RL - WL + 2). The value, along with other delay/skew parameters, are used by the "Bus Turnaround" timing analysis to determine if bus contention is an issue. Set according to the controller specification.
<b>Write-to-Read NOP commands (min)</b>	Minimum number of no operation commands following a write command and before a read command. The value must be at least ((Burst Length/2) + WL - RL + 1). The value, along with other delay/skew parameters, are used by the "Bus Turnaround" timing analysis to determine if bus contention is an issue. Set according to the controller specification.
<b>RLDRAM 3 Board Derate</b>	
<b>CK/CK# slew rate (differential)</b>	CK/CK# slew rate (differential).
<b>Address/Command slew rate</b>	Address and command slew rate.
<b>DK/DK# slew rate (Differential)</b>	DK/DK# slew rate (differential).
<b>DQ slew rate</b>	DQ slew rate.
<b>tIS</b>	Address/command setup time to CK.

### 7.3.4. Board Settings

Use the **Board Settings** tab to model the board-level effects in the timing analysis.



The **Board Settings** tab allows you to specify the following settings:

- Setup and hold derating (For LPDDR2/DDR2/DDR3 SDRAM, RLDRAM 3 and RLDRAM II for UniPHY IP)
- Channel Signal Integrity
- Board skews (For UniPHY IP.)

**Note:** For accurate timing results, you must enter board settings parameters that are correct for your PCB.

The IP core supports single and multiple chip-select configurations. Intel has determined the effects on the output signaling of single-rank configurations for certain Intel boards, and included the channel uncertainties in the Quartus Prime timing models.

Because the Quartus Prime timing models hold channel uncertainties that are representative of specific Intel boards, you must determine the board-level effects of your board, including any additional channel uncertainty relative to Intel's reference design, and enter those values into the Board Settings panel in the parameter editor. You can use HyperLynx or a similar simulator to obtain values that are representative of your board.

For more information about how to include your board simulation results in the Quartus Prime software, refer to the following sections. For more information about how to assign pins using pin planners, refer to the design flow tutorials and design examples on the List of Designs Using Intel External Memory IP page of [www.alterawiki.com](http://www.alterawiki.com).

For more general information about timing deration methodology, refer to the *Timing Deration Methodology for Multiple Chip Select DDR2 and DDR3 SDRAM Designs* section in the *Analyzing Timing of Memory IP* chapter.

#### Related Information

- [Analyzing Timing of Memory IP](#)
- [List of Designs using Intel External Memory IP](#)

##### 7.3.4.1. Setup and Hold Derating for UniPHY IP

The slew rate of the output signals affects the setup and hold times of the memory device, and thus the write margin. You can specify the slew rate of the output signals to see their effect on the setup and hold times of both the address and command signals and the DQ signals, or alternatively, you may want to specify the setup and hold times directly.

For RDIMMs, the slew rate is defined at the register on the RDIMM, instead of at the memory component. For LRDIMMs, the slew rate is defined at the buffer on the LRDIMM, instead of at the memory component.

**Note:** You should enter information derived during your PCB development process of prelayout (line) and postlayout (board) simulation.

The following table lists the setup and hold derating parameters.

**Table 77. Setup and Hold Derating Parameters**

Parameter	Description
<b>LPDDR2/DDR2/DDR3 SDRAM/RDRAM 3</b>	
<b>Derating method</b>	Derating method. The default settings are based on Intel internal board simulation data. To obtain accurate timing analysis according to the condition of your board, Intel recommends that you perform board simulation and enter the slew rate in the Quartus Prime software to calculate the derated setup and hold time automatically or enter the derated setup and hold time directly. For more information, refer to the "Timing Deration Methodology for Multiple Chip Select DDR2 and DDR3 SDRAM Designs" section in the <i>Analyzing Timing of Memory IP</i> chapter.
<b>CK/CK# slew rate (differential)</b>	CK/CK# slew rate (differential).
<b>Address/Command slew rate</b>	Address and command slew rate.
<b>DQS/DQS# slew rate (Differential)</b>	DQS and DQS# slew rate (differential).
<b>DQ slew rate</b>	DQ slew rate.
<b>tIS</b>	Address/command setup time to CK.
<b>tIH</b>	Address/command hold time from CK.
<b>tDS</b>	Data setup time to DQS.
<b>tDH</b>	Data hold time from DQS.
<b>RDRAM II</b>	
<b>tAS Vref to CK/CK# Crossing</b>	For a given address/command and CK/CK# slew rate, the memory device data sheet provides a corresponding "tAS Vref to CK/CK# Crossing" value that can be used to determine the derated address/command setup time.
<b>tAS VIH MIN to CK/CK# Crossing</b>	For a given address/command and CK/CK# slew rate, the memory device data sheet provides a corresponding "tAS VIH MIN to CK/CK# Crossing" value that can be used to determine the derated address/command setup time.
<b>tAH CK/CK# Crossing to Vref</b>	For a given address/command and CK/CK# slew rate, the memory device data sheet provides a corresponding "tAH CK/CK# Crossing to Vref" value that can be used to determine the derated address/command hold time.
<b>tAH CK/CK# Crossing to VIH MIN</b>	For a given address/command and CK/CK# slew rate, the memory device data sheet provides a corresponding "tAH CK/CK# Crossing to VIH MIN" value that can be used to determine the derated address/command hold time.
<b>tDS Vref to CK/CK# Crossing</b>	For a given data and DK/DK# slew rate, the memory device data sheet provides a corresponding "tDS Vref to CK/CK# Crossing" value that can be used to determine the derated data setup time.
<b>tDS VIH MIN to CK/CK# Crossing</b>	For a given data and DK/DK# slew rate, the memory device data sheet provides a corresponding "tDS VIH MIN to CK/CK# Crossing" value that can be used to determine the derated data setup time.
<b>tDH CK/CK# Crossing to Vref</b>	For a given data and DK/DK# slew rate, the memory device data sheet provides a corresponding "tDH CK/CK# Crossing to Vref" value that can be used to determine the derated data hold time.
<b>tDH CK/CK# Crossing to VIH MIN</b>	For a given data and DK/DK# slew rate, the memory device data sheet provides a corresponding "tDH CK/CK# Crossing to VIH MIN" value that can be used to determine the derated data hold time.
<b>Derated tAS</b>	The derated address/command setup time is calculated automatically from the "tAS", the "tAS Vref to CK/CK# Crossing", and the "tAS VIH MIN to CK/CK# Crossing" parameters.

*continued...*



Parameter	Description
<b>Derated tAH</b>	The derated address/command hold time is calculated automatically from the "tAH", the "tAH CK/CK# Crossing to Vref", and the "tAH CK/CK# Crossing to VIH MIN" parameters.
<b>Derated tDS</b>	The derated data setup time is calculated automatically from the "tDS", the "tDS Vref to CK/CK# Crossing", and the "tDS VIH MIN to CK/CK# Crossing" parameters.
<b>Derated tDH</b>	The derated data hold time is calculated automatically from the "tDH", the "tDH CK/CK# Crossing to Vref", and the "tDH CK/CK# Crossing to VIH MIN" parameters.

### 7.3.4.2. Intersymbol Interference Channel Signal Integrity for UniPHY IP

Channel signal integrity is a measure of the distortion of the eye due to intersymbol interference or crosstalk or other effects.

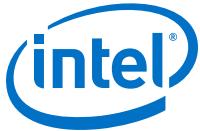
Typically, when going from a single-rank configuration to a multi-rank configuration there is an increase in the channel loss, because there are multiple stubs causing reflections. Although the Quartus Prime timing models include some channel uncertainty, you must perform your own channel signal integrity simulations and enter the additional channel uncertainty, relative to the reference eye, into the parameter editor GUI.

For details about measuring channel loss parameters and entering channel signal integrity information into the parameter editor GUI, refer to the Wiki: [http://www.alterawiki.com/wiki/Measuring\\_Channel\\_Signal\\_Integrity](http://www.alterawiki.com/wiki/Measuring_Channel_Signal_Integrity).

The following table lists intersymbol interference parameters.

**Table 78. ISI Parameters**

Parameter	Description
<b>Derating method</b>	Choose between default Intel settings (with specific Intel boards) or manually enter board simulation numbers obtained for your specific board. This option is supported in LPDDR2/DDR2/DDR3 SDRAM only.
<b>Address and command eye reduction (setup)</b>	The reduction in the eye diagram on the setup side (or left side of the eye) due to ISI on the address and command signals compared to a case when there is no ISI. (For single rank designs, ISI can be zero; in multirank designs, ISI is necessary for accurate timing analysis.) For more information about how to measure the ISI value for the address and command signals, refer to the "Measuring Eye Reduction for Address/Command, DQ, and DQS Setup and Hold Time" section in <i>Analyzing Timing of Memory IP</i> .
<b>Address and command eye reduction (hold)</b>	The reduction in the eye diagram on the hold side (or right side of the eye) due to ISI on the address and command signals compared to a case when there is no ISI. For more information about how to measure the ISI value for the address and command signals, refer to "Measuring Eye Reduction for Address/Command, DQ, and DQS Setup and Hold Time" section in <i>Analyzing Timing of Memory IP</i> .
<b>DQ/ D eye reduction</b>	The total reduction in the eye diagram due to ISI on DQ signals compared to a case when there is no ISI. Intel assumes that the ISI reduces the eye width symmetrically on the left and right side of the eye. For more information about how to measure the ISI value for the address and command signals, refer to "Measuring Eye Reduction for Address/Command, DQ, and DQS Setup and Hold Time" section in <i>Analyzing Timing of Memory IP</i> .
<b>Delta DQS/Delta K/ Delta DK arrival time</b>	The increase in variation on the range of arrival times of DQS compared to a case when there is no ISI. Intel assumes that the ISI causes DQS to further vary symmetrically to the left and to the right. For more information about how to measure the ISI value for the address and command signals, refer to "Measuring Eye Reduction for Address/Command, DQ, and DQS Setup and Hold Time" section in <i>Analyzing Timing of Memory IP</i> .



### 7.3.4.3. Board Skews for UniPHY IP

PCB traces can have skews between them that can reduce timing margins. Furthermore, skews between different chip selects can further reduce the timing margin in multiple chip-select topologies.

The **Board Skews** section of the parameter editor allows you to enter parameters to compensate for these variations.

**Note:** You must ensure the timing margin reported in TimeQuest Report DDR is positive when the board skew parameters are correct for the PCB.

The following tables list the board skew parameters. For parameter equations containing delay values, delays should be measured as follows:

- Non-fly-by topology (Balanced Tree)
  - For discrete devices—all the delay (CK, Addr/Cmd, DQ and DQS) from the FPGA are right to every memory device
  - For UDIMMs—all the delay (CK, Addr/Cmd, DQ and DQS) from the FPGA to UDIMM connector for every memory device on the UDIMM. If UDIMM delay information is available, calculate delays to every memory device on the UDIMM.
  - For RDIMMs—the Addr/Cmd and CK delay are from the FPGA to the register on the RDIMM. The DQ and DQS delay are from FPGA to RDIMM connector for every memory device on the RDIMM.
  - For LRDIMMS—the delay from the FPGA to the register on the LRDIMM.
- Fly-by topology
  - For discrete devices—the Addr/Cmd and CK delay are from the FPGA to the first memory device. The DQ and DQS delay are from FPGA to every memory device.
  - For UDIMMs—the Addr/Cmd and CK delay are from the FPGA to the UDIMM connector. The DQ and DQS delay are from the FPGA to UDIMM connector for every memory device on the UDIMM.
  - For RDIMMs—the Addr/Cmd and CK delay are from the FPGA to the register on the RDIMM. The DQ and DQS delay are from FPGA to RDIMM connector for every memory device on the RDIMM.
  - For LRDIMMS—the delay from the FPGA to the buffer on the LRDIMM.

Equations apply to any given memory device, except when marked by the board or group qualifiers (*\_b* or *\_g*), where they apply to the particular device or group being iterated over.

Use the Board Skew Parameter Tool to help you calculate the board skews.

#### Related Information

[Board Skew Parameter Tool](#)

### 7.3.4.3.1. Board Skew Parameters for LPDDR2/DDR2/DDR3 SDRAM

The following table lists board skew parameters for LPDDR2, DDR2, and DDR3 interfaces.



**Table 79. Parameter Descriptions**

Parameter	Description
<b>FPGA DQ/DQS Package Skews Deskewed on Board</b>	<p>Enable this parameter if you will deskew the FPGA package with your board traces on the DQ and DQS pins. This option increases the read capture and write margins. Enable this option when memory clock frequency is larger than 800 MHz. Enabling this option improves the read capture and write timing margin. You can also rely on the read capture and write timing margin in timing report to enable this option.</p> <p>When this option is enabled, package skews are output on the DQ and DQS pins in the Pin-Out File (.pin) and package skew is not included in timing analysis. All of the other board delay and skew parameters related to DQ or DQS must consider the package and the board together. For more information, refer to <i>DDR2 and DDR3 Board Layout Guidelines</i>.</p>
<b>Address/Command Package Deskew</b>	<p>Enable this parameter if you will deskew the FPGA package with your board traces on the address and command pins. This option increases the address and command margins. Enable this option when memory clock frequency is larger than 800 MHz. Enabling this option will improve the address and command timing margin. You can also rely on the address and command margin in timing report to enable this option.</p> <p>When this option is enabled, package skews are output on the address and command pins in the Pin-Out File (.pin) and package skew is not included in timing analysis. All of the other board delay and skew parameters related to Address and Command must consider the package and the board together. For more information, refer to <i>DDR2 and DDR3 Board Layout Guidelines</i>.</p>
<b>Maximum CK delay to DIMM/device</b>	<p>The delay of the longest CK trace from the FPGA to the memory device, whether on a DIMM or the same PCB as the device, is expressed by the following equation:</p> $\max_r[\max_n(CK_{n\_r} PathDelay)]$ <p>Where <math>n</math> is the number of memory clock and <math>r</math> is number rank of DIMM/device. For example in dual-rank DIMM implementation, if there are 2 pairs of memory clocks in each rank DIMM, the maximum CK delay is expressed by the following equation:  <math>\max(CK_1 PathDelay \text{ rank 1}, CK_2 Path Delay \text{ rank 1}, CK_1 Path Delay \text{ rank 2}, CK_2 Path Delay \text{ rank 2})</math></p>
<b>Maximum DQS delay to DIMM/device</b>	<p>The delay of the longest DQS trace from the FPGA to the memory device, whether on a DIMM or the same PCB as the device, is expressed by the following equation:</p> $\max_r[\max_n(DQS_{n\_r} Path Delay)]$ <p>Where <math>n</math> is the number of DQS and <math>r</math> is number of rank of DIMM/device. For example in dual-rank DIMM implementation, if there are 2 DQS in each rank DIMM, the maximum DQS delay is expressed by the following equation:  <math>\max(DQS_1 PathDelay \text{ rank 1}, DQS_2 Path Delay \text{ rank 1}, DQS_1 Path Delay \text{ rank 2}, DQS_2 Path Delay \text{ rank 2})</math></p>
<b>Minimum delay difference between CK and DQS</b>	<p>The minimum skew or smallest positive skew (or largest negative skew) between the CK signal and any DQS signal arriving at the same DIMM/device over all DIMMs/devices is expressed by the following equation:</p> $\min_r[\min_{n, m}\{(CK_{n\_r} Delay - DQS_{m\_r} Delay)\}]$ <p>Where <math>n</math> is the number of memory clock, <math>m</math> is the number of DQS, and <math>r</math> is the number of rank of DIMM/device. For example in dual-rank DIMM implementation, if there are 2 pairs of memory clock and 4 DQS signals (two for each clock pair), for each rank DIMM, the minimum delay difference between CK and DQS is expressed by the following equation:</p> $\min\left\{\begin{array}{l} (CK_{1\_1} Delay - DQS_{1\_1} Delay), (CK_{1\_1} Delay - DQS_{2\_1} Delay), (CK_{2\_1} Delay - DQS_{3\_1} Delay), (CK_{2\_1} Delay - DQS_{4\_1} Delay) \\ (CK_{1\_2} Delay - DQS_{1\_2} Delay), (CK_{1\_2} Delay - DQS_{2\_2} Delay), (CK_{2\_2} Delay - DQS_{3\_2} Delay), (CK_{2\_2} Delay - DQS_{4\_2} Delay) \end{array}\right\}$ <p>This parameter value affects the write leveling margin for DDR3 interfaces with leveling in multi-rank configurations. This parameter value also applies to non-leveling configurations of any number of ranks with the requirement that DQS must have positive margins in TimeQuest Report DDR.</p>

*continues*



Parameter	Description
	<p>For multiple boards, the minimum skew between the CK signal and any DQS signal when arriving at the same DIMM over all DIMMs is expressed by the following equation, if you want to use the same design for several different boards:</p> $\min_b^{\text{boards}} \left[ \min_g^{\text{groups}} [CK_{g-b} - DQS_{g-b}] \right]$ <p><i>Note:</i> If you are using a clamshell topology in a multirank/multi chip-select design with either DIMM or discrete devices or using dual-die devices, the above calculations do not apply; you may use the default values in the GUI.</p>
<b>Maximum delay difference between CK and DQS</b>	<p>The maximum skew or smallest negative skew (or largest positive skew) between the CK signal and any DQS signal when arriving at the same DIMM/device over all DIMMs/devices is expressed by the following equation:</p> $\max_r^{\text{max}} \left[ \max_{n,m} \{ (CK_{n\_r} \text{Delay} - DQS_{m\_r} \text{Delay}) \} \right]$ <p>Where <math>n</math> is the number of memory clock, <math>m</math> is the number of DQS, and <math>r</math> is the number of rank of DIMM/device. For example in dual-rank DIMM implementation, if there are 2 pairs of memory clock and 4 DQS signals (two for each clock) for each rank DIMM, the maximum delay difference between CK and DQS is expressed by the following equation:</p> $\max \left\{ \begin{array}{l} (CK_{1\_1} \text{Delay} - DQS_{1\_1} \text{Delay}), (CK_{1\_1} \text{Delay} - DQS_{2\_1} \text{Delay}), (CK_{2\_1} \text{Delay} - DQS_{3\_1} \text{Delay}), (CK_{2\_1} \text{Delay} - DQS_{4\_1} \text{Delay}) \\ (CK_{1\_2} \text{Delay} - DQS_{1\_2} \text{Delay}), (CK_{1\_2} \text{Delay} - DQS_{2\_2} \text{Delay}), (CK_{2\_2} \text{Delay} - DQS_{3\_2} \text{Delay}), (CK_{2\_1} \text{Delay} - DQS_{4\_2} \text{Delay}) \end{array} \right\}$ <p>This value affects the write Leveling margin for DDR3 interfaces with leveling in multi-rank configurations. This parameter also applies to non-leveling configurations of any number of ranks with the requirement that DQS must have positive margins in TimeQuest Report DDR.</p> <p>For multiple boards, the maximum skew (or largest positive skew) between the CK signal and any DQS signal when arriving at the same DIMM over all DIMMs is expressed by the following equation, if you want to use the same design for several different boards:</p> $\max_b^{\text{boards}} \left[ \max_g^{\text{groups}} [CK_{g-b} - DQS_{g-b}] \right]$ <p><i>Note:</i> If you are using a clamshell topology in a multirank/multi chip-select design with either DIMM or discrete devices or using dual-die devices, the above calculations do not apply; you may use the default values in the GUI.</p>
<b>Maximum skew within DQS group</b>	<p>The largest skew among DQ and DM signals in a DQS group. This value affects the read capture and write margins for DDR2 and DDR3 SDRAM interfaces in all configurations (single or multiple chip-select, DIMM or component). For multiple boards, the largest skew between DQ and DM signals in a DQS group is expressed by the following equation:</p> $\max_b^{\text{boards}} \left[ \max_g^{\text{groups}} [maxDQ_{g-b} - minDQ_{g-b}] \right]$
<b>Maximum skew between DQS groups</b>	<p>The largest skew between DQS signals in different DQS groups. This value affects the resynchronization margin in memory interfaces without leveling such as DDR2 SDRAM and discrete-device DDR3 SDRAM in both single- or multiple-chip-select configurations. For protocols or families that do not have read resynchronization analysis, this parameter has no effect.</p> <p>For multiple boards, the largest skew between DQS signals in different DQS groups is expressed by the following equation, if you want to use the same design for several different boards:</p> $\max_b^{\text{boards}} \left[ \max_g^{\text{groups}} [DQS_{g-b}] - \min_b^{\text{boards}} \left[ \min_g^{\text{groups}} [DQS_{g-b}] \right] \right]$

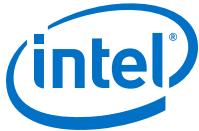
continued



Parameter	Description
<b>Average delay difference between DQ and DQS</b>	<p>The average delay difference between each DQ signal and the DQS signal, calculated by averaging the longest and smallest DQ signal delay values minus the delay of DQS. The average delay difference between DQ and DQS is expressed by the following equation:</p> $\sum_{n=1}^{n=n} \left[ \left( \frac{\text{Longest DQ Path Delay in } DQS_n \text{ group} + \text{Shortest DQ Path Delay in } DQS_n \text{ group}}{2} \right) - DQS_n \text{ PathDelay} \right]$ <p>where n is the number of DQS groups. For multi-rank or multiple CS configuration, the equation is:</p> $\sum_{r=1}^{r=r} \left[ \text{Average delay differnt between DQ and DQS in rank r} \right]$
<b>Maximum skew within address and command bus</b>	<p>The largest skew between the address and command signals for a single board is expressed by the following equation:</p> $0.5[(MaxACdelay - MinCKdelay) - (MinACdelay - MaxCKdelay)]$ <p>For multiple boards, the largest skew between the address and command signals is expressed by the following equation if you want to use the same design for several different boards:</p> $\frac{\sum_{b=1}^{boards} [(MaxAC_b - MinCK_b) - Min_b (MaxAC_b - MinCK_b)]}{2}$
<b>Average delay difference between address and command and CK</b>	<p>A value equal to the average of the longest and smallest address and command signal delay values, minus the delay of the CK signal. The value can be positive or negative. Positive values represent address and command signals that are longer than CK signals; negative values represent address and command signals that are shorter than CK signals. The average delay difference between address and command and CK is expressed by the following equation:</p> $\sum_{n=1}^{n=n} \left[ \left( \frac{\text{Longest AC Path Delay} + \text{Shortest AC Path Delay}}{2} \right) - CK_n \text{ PathDelay} \right]$ <p>where n is the number of memory clocks. For multi-rank or multiple CS configuration, the equation is:</p> $\sum_{r=1}^{r=r} \left[ \text{Average delay differnt between AC and CK in rank r} \right]$ <p>The Quartus Prime software uses this skew to optimize the delay of the address and command signals to have appropriate setup and hold margins for DDR2 and DDR3 SDRAM interfaces. You should derive this value through board simulation.</p> <p>For multiple boards, the average delay difference between address and command and CK is expressed by the following equation, if you want to use the same design for several different boards:</p> $Avg_b^{boards} \left[ \left( \frac{MaxAC_b + MinAC_b}{2} \right) - \left( \frac{MaxCK_b + MinCK_b}{2} \right) \right]$

### 7.3.4.3.2. Board Skew Parameters for QDR II and QDR II+

The following table lists board skew parameters for QDR II and QDR II+ interfaces.

**Table 80. Parameter Descriptions**

Parameter	Description
<b>Maximum delay difference between devices</b>	The maximum delay difference of data signals between devices is expressed by the following equation: $\text{Abs} \left[ \left( \frac{\text{Longest device 1 delay} - \text{Shortest device 2 delay}}{2} \right) - \left( \frac{\text{Longest device 2 delay} - \text{Shortest device 1 delay}}{2} \right) \right]$ For example, in a two-device configuration there is greater propagation delay for data signals going to and returning from the furthest device relative to the nearest device. This parameter is applicable for depth expansion. Set the value to 0 for non-depth expansion design.
<b>Maximum skew within write data group (ie, K group)</b>	The maximum skew between D and BWS signals referenced by a common K signal.
<b>Maximum skew within read data group (ie, CQ group)</b>	The maximum skew between Q signals referenced by a common CQ signal.
<b>Maximum skew between CQ groups</b>	The maximum skew between CQ signals of different read data groups. Set the value to 0 for non-depth expansion designs.
<b>Maximum skew within address/command bus</b>	The maximum skew between the address/command signals. $0.5[(\text{MaxACdelay} - \text{MinCKdelay}) - (\text{MinACdelay} - \text{MaxCKdelay})]$
<b>Average delay difference between address/command and K</b>	A value equal to the average of the longest and smallest address/command signal delay values, minus the delay of the K signal. The value can be positive or negative. The average delay difference between the address and command and K is expressed by the following equation: $\sum_{n=1}^n \left[ \left( \frac{\text{Longest AC Path Delay} + \text{Shortest AC Path Delay}}{2} \right) - K_n \text{PathDelay} \right]$ where n is the number of K clocks.
<b>Average delay difference between write data signals and K</b>	A value equal to the average of the longest and smallest write data signal delay values, minus the delay of the K signal. Write data signals include the D and BWS signals. The value can be positive or negative. The average delay difference between D and K is expressed by the following equation: $\sum_{n=1}^n \left[ \left( \frac{\text{Longest D Path Delay in } K_n \text{ group} + \text{Shortest D Path Delay in } K_n \text{ group}}{2} \right) - K_n \text{PathDelay} \right]$ where n is the number of DQS groups.
<b>Average delay difference between read data signals and CQ</b>	A value equal to the average of the longest and smallest read data signal delay values, minus the delay of the CQ signal. The value can be positive or negative. The average delay difference between Q and CQ is expressed by the following equation: $\sum_{n=1}^n \left[ \left( \frac{\text{Longest Q Path Delay in } CQ_n \text{ group} + \text{Shortest Q Path Delay in } CQ_n \text{ group}}{2} \right) - CQ_n \text{PathDelay} \right]$ where n is the number of CQ groups.

#### 7.3.4.3.3. Board Skew parameters for RLDRAM II and RLDRAM 3

The following table lists board skew parameters for RLDRAM II and RLDRAM 3 interfaces.



**Table 81. Parameter Descriptions**

Parameter	Description
<b>Maximum CK delay to device</b>	<p>The delay of the longest CK trace from the FPGA to any device/DIMM is expressed by the following equation:</p> $\max_n(CK_n PathDelay)$ <p>where n is the number of memory clocks. For example, the maximum CK delay for two pairs of memory clocks is expressed by the following equation:</p> $\max_2(CK_1 PathDelay, CK_2 PathDelay)$
<b>Maximum DK delay to device</b>	<p>The delay of the longest DK trace from the FPGA to any device/DIMM is expressed by the following equation:</p> $\max_n(DK_n PathDelay)$ <p>where n is the number of DK. For example, the maximum DK delay for two DK is expressed by the following equation:</p> $\max_2(DK_1 PathDelay, DK_2 PathDelay)$
<b>Minimum delay difference between CK and DK</b>	<p>The minimum delay difference between the CK signal and any DK signal when arriving at the memory device(s). The value is equal to the minimum delay of the CK signal minus the maximum delay of the DK signal. The value can be positive or negative.</p> <p>The minimum delay difference between CK and DK is expressed by the following equations:</p> $\min_{n,m}(CK_n PathDelay - DK_m PathDelay)$ <p>where n is the number of memory clocks and m is the number of DK. For example, the minimum delay difference between CK and DK for two pairs of memory clocks and four DK signals (two DK signals for each clock) is expressed by the following equation:</p> $\min_{2,2}\{(Ck_1 Delay - DK_1 Delay), (Ck_1 Delay - DK_2 Delay), (Ck_2 Delay - DK_3 Delay), (Ck_2 Delay - DK_4 Delay)\}$
<b>Maximum delay difference between CK and DK</b>	<p>The maximum delay difference between the CK signal and any DK signal when arriving at the memory device(s). The value is equal to the maximum delay of the CK signal minus the minimum delay of the DK signal. The value can be positive or negative.</p> <p>The maximum delay difference between CK and DK is expressed by the following equations:</p> $\max_{n,m}(CK_n PathDelay - DK_m PathDelay)$ <p>where n is the number of memory clocks and m is the number of DK. For example, the maximum delay difference between CK and DK for two pairs of memory clocks and four DK signals (two DK signals for each clock) is expressed by the following equation:</p>

*continued...*



Parameter	Description
$\max_{2,2} \{(Ck_1 Delay - DK_1 Delay), (Ck_1 Delay - DK_2 Delay), (Ck_2 Delay - DK_3 Delay), (Ck_2 Delay - DK_4 Delay)\}$	
<b>Maximum delay difference between devices</b>	The maximum delay difference of data signals between devices is expressed by the following equation: $Abs\left[\left(\frac{\text{Longest device 1 delay} - \text{Shortest device 2 delay}}{2}\right) - \left(\frac{\text{Longest device 2 delay} - \text{Shortest device 1 delay}}{2}\right)\right]$ For example, in a two-device configuration there is greater propagation delay for data signals going to and returning from the furthest device relative to the nearest device. This parameter is applicable for depth expansion. Set the value to 0 for non-depth expansion design.
<b>Maximum skew within QK group</b>	The maximum skew between the DQ signals referenced by a common QK signal.
<b>Maximum skew between QK groups</b>	The maximum skew between QK signals of different data groups.
<b>Maximum skew within address/command bus</b>	The maximum skew between the address/command signals. $0.5[(MaxACdelay - MinCKdelay) - (MinACdelay - MaxCKdelay)]$
<b>Average delay difference between address/command and CK</b>	A value equal to the average of the longest and smallest address/command signal delay values, minus the delay of the CK signal. The value can be positive or negative. The average delay difference between the address and command and CK is expressed by the following equation: $\sum_{n=1}^n \left[ \left( \frac{\text{Longest AC Path Delay} + \text{Shortest AC Path Delay}}{2} \right) - CK_n PathDelay \right] n$ where n is the number of memory clocks.
<b>Average delay difference between write data signals and DK</b>	A value equal to the average of the longest and smallest write data signal delay values, minus the delay of the DK signal. Write data signals include the DQ and DM signals. The value can be positive or negative. The average delay difference between DQ and DK is expressed by the following equation: $\sum_{n=1}^n \left[ \left( \frac{\text{Longest DQ Path Delay in } DK_n \text{ group} + \text{Shortest DQ Path Delay in } DK_n \text{ group}}{2} \right) - DK_n PathDelay \right] n$ where n is the number of DK groups.
<b>Average delay difference between read data signals and QK</b>	A value equal to the average of the longest and smallest read data signal delay values, minus the delay of the QK signal. The value can be positive or negative. The average delay difference between DQ and QK is expressed by the following equation: $\sum_{n=1}^n \left[ \left( \frac{\text{Longest DQ Path Delay in } QK_n \text{ group} + \text{Shortest DQ Path Delay in } QK_n \text{ group}}{2} \right) - QK_n PathDelay \right] n$ where n is the number of QK groups.



### 7.3.5. Controller Settings for UniPHY IP

Use the **Controller Settings** tab to apply the controller settings suitable for your design.

**Note:** This section describes parameters for the High Performance Controller II (HPC II) with advanced features first introduced in version 11.0 for designs generated in version 11.0 or later. Designs created in earlier versions and regenerated in version 11.0 or later do not inherit the new advanced features; for information on parameters for HPC II without the advanced features, refer to the External Memory Interface Handbook for Quartus II version 10.1, available on the *Literature: External Memory Interfaces* page of [www.altera.com](http://www.altera.com).

**Table 82. Controller Settings for LPDDR2/DDR2/DDR3 SDRAM**

Parameter	Description
<b>Avalon Interface</b>	<b>Generate power-of-2 bus widths for SOPC Builder</b>  Rounds down the Avalon-MM side data bus to the nearest power of 2. You must enable this option for Qsys systems.  If this option is enabled, the Avalon data buses are truncated to 256 bits wide. One Avalon read-write transaction of 256 bit width maps to four memory beat transactions, each of 72 bits (8 MSB bits are zero, while 64 LSB bits carry useful content). The four memory beats may comprise an entire burst length-of-4 transaction, or part of a burst-length-of-8 transaction.
	<b>Generate SOPC Builder compatible resets</b>  This option is not required when using the IP Catalog or Qsys.
	<b>Maximum Avalon-MM burst length</b>  Specifies the maximum burst length on the Avalon-MM bus. Affects the AVL_SIZE_WIDTH parameter.
	<b>Enable Avalon-MM byte-enable signal</b>  When you turn on this option, the controller adds the byte enable signal (avl_be) for the Avalon-MM bus to control the data mask (mem_dm) pins going to the memory interface. You must also turn on <b>Enable DM pins</b> if you are turning on this option.  When you turn off this option, the byte enable signal (avl_be) is not enabled for the Avalon-MM bus, and by default all bytes are enabled. However, if you turn on <b>Enable DM pins</b> with this option turned off, all write words are written.
	<b>Avalon interface address width</b>  The address width on the Avalon-MM interface.
	<b>Avalon interface data width</b>  The data width on the Avalon-MM interface.
<b>Low Power Mode</b>	<b>Enable self-refresh controls</b>  Enables the self-refresh signals on the controller top-level design. These controls allow you to control when the memory is placed into self-refresh mode.

*continued...*



Parameter	Description
	<b>Enable Deep Power-Down Controls</b> Enables the Deep Power-Down signals on the controller top level. These signals control when the memory is placed in Deep Power-Down mode. This parameter is available only for LPDDR2 SDRAM.
	<b>Enable auto-power down</b> Allows the controller to automatically place the memory into (Precharge) power-down mode after a specified number of idle cycles. Specifies the number of idle cycles after which the controller powers down the memory in the auto-power down cycles parameter.
	<b>Auto power-down cycles</b> The number of idle controller clock cycles after which the controller automatically powers down the memory. The legal range is from 1 to 65,535 controller clock cycles.
<b>Efficiency</b>	<b>Enable user auto refresh controls</b> Enables the user auto-refresh control signals on the controller top level. These controller signals allow you to control when the controller issues memory autorefresh commands.
	<b>Enable auto precharge control</b> Enables the autoprecharge control on the controller top level. Asserting the autoprecharge control signal while requesting a read or write burst allows you to specify whether the controller should close (autoprecharge) the currently open page at the end of the read or write burst.
	<b>Local-to-memory address mapping</b> Allows you to control the mapping between the address bits on the Avalon-MM interface and the chip, row, bank, and column bits on the memory. Select <b>Chip-Row-Bank-Col</b> to improve efficiency with sequential traffic. Select <b>Chip-Bank-Row-Col</b> to improve efficiency with random traffic. Select <b>Row-Chip-Bank-Col</b> to improve efficiency with multiple chip select and sequential traffic.
	<b>Command queue look ahead depth</b> Selects a look-ahead depth value to control how many read or writes requests the look-ahead bank management logic examines. Larger numbers are likely to increase the efficiency of the bank management, but at the cost of higher resource usage. Smaller values may be less efficient, but also use fewer resources. The valid range is from 1 to 16.
	<b>Enable reordering</b> Allows the controller to perform command and data reordering that reduces bus turnaround time and row/bank switching time to improve controller efficiency.

*continued...*



Parameter	Description
	<b>Starvation limit for each command</b> Specifies the number of commands that can be served before a waiting command is served. The valid range is from 1 to 63.
<b>Configuration, Status, and Error Handling</b>	<b>Enable Configuration and Status Register Interface</b> Enables run-time configuration and status interface for the memory controller. This option adds an additional Avalon-MM slave port to the memory controller top level, which you can use to change or read out the memory timing parameters, memory address sizes, mode register settings and controller status. If Error Detection and Correction Logic is enabled, the same slave port also allows you to control and retrieve the status of this logic.
	<b>CSR port host interface</b> Specifies the type of connection to the CSR port. The port can be exported, internally connected to a JTAG Avalon Master, or both. Select <b>Internal (JTAG)</b> to connect the CSR port to a JTAG Avalon Master. Select <b>Avalon-MM Slave</b> to export the CSR port. Select <b>Shared</b> to export and connect the CSR port to a JTAG Avalon Master.
	<b>Enable error detection and correction logic</b> Enables ECC for single-bit error correction and double-bit error detection. Your memory interface must be a multiple of 16, 24, 40, or 72 bits wide to use ECC.
	<b>Enable auto error correction</b> Allows the controller to perform auto correction when a single-bit error is detected by the ECC logic.
<b>Multiple Port Front End</b>	<b>Export bonding port</b> Turn on this option to export bonding interface for wider avalon data width with two controllers. Bonding ports are exported to the top level.
	<b>Number of ports</b> Specifies the number of Avalon-MM Slave ports to be exported. The number of ports depends on the width and the type of port you selected. There are four 64-bit read FIFOs and four 64-bit write FIFOs in the multi-port front-end (MPFE) component. For example, If you select 256 bits width and bidirectional slave port, all the FIFOs are fully utilized, therefore you can only select one port. <i>Note:</i> This parameter is not available for MAX 10 devices.
	<b>Width</b> Specifies the local data width for each Avalon-MM Slave port. The width depends on the type of slave port and also the number of ports selected. This is due to the limitation of the FIFO counts in the MPFE. There are four 64-bit read FIFOs and four 64-bit write FIFOs in the MPFE. For example, if you

*continued...*



Parameter	Description
	select one bidirectional slave port, you can select up to 256 bits to utilize all the read and write FIFOs. As a general guideline to choosing an optimum port width for your half-rate or quarter-rate design, apply the following equation: $\text{port width} = 2 \times \text{DQ width} \times \text{Interface width multiplier}$ where the interface width multiplier is 2 for half-rate interfaces and 4 for quarter-rate interfaces.
<b>Priority</b>	Specifies the absolute priority for each Avalon-MM Slave port. Any transaction from the port with higher priority number will be served before transactions from the port with lower priority number.
<b>Weight</b>	Specifies the relative priority for each Avalon-MM Slave port. When there are two or more ports having the same absolute priority, the transaction from the port with higher (bigger number) relative weight will be served first. You can set the weight from a range of 0 to 32.
<b>Type</b>	Specifies the type of Avalon MM slave port to either a bidirectional port, read only port or write only port.

**Table 83. Controller Settings for QDR II/QDR II+ SRAM and RLDRAM II**

Parameter	Description
<b>Generate power-of-2 data bus widths for SOPC Builder</b>	Rounds down the Avalon-MM side data bus to the nearest power of 2. You must enable this option for Qsys systems.
<b>Generate SOPC Builder compatible resets</b>	This option is not required when using the IP Catalog or Qsys.
<b>Maximum Avalon-MM burst length</b>	Specifies the maximum burst length on the Avalon-MM bus.
<b>Enable Avalon-MM byte-enable signal</b>	When you turn on this option, the controller adds a byte-enable signal ( <code>avl_be_w</code> ) for the Avalon-MM bus, in which controls the <code>bws_n</code> signal on the memory side to mask bytes during write operations. When you turn off this option, the <code>avl_be_w</code> signal is not available and the controller will always drive the memory <code>bws_n</code> signal so as to not mask any bytes during write operations.
<b>Avalon interface address width</b>	Specifies the address width on the Avalon-MM interface.
<b>Avalon interface data width</b>	Specifies the data width on the Avalon-MM interface.

*continued...*



Parameter	Description
<b>Reduce controller latency by</b>	Specifies the number of clock cycles by which to reduce controller latency. Lower controller latency results in lower resource usage and $f_{MAX}$ while higher latency results in higher resource usage and $f_{MAX}$ .
<b>Enable user refresh</b>	Enables user-controlled refresh. Refresh signals will have priority over read/write requests. This option is available for RLDRAM II only.
<b>Enable error detection parity</b>	Enables per-byte parity protection. This option is available for RLDRAM II only

#### Related Information

Literature: External Memory Interfaces

### 7.3.6. Diagnostics for UniPHY IP

The **Diagnostics** tab allows you to set parameters for certain diagnostic functions.

The following table describes parameters for simulation.

**Table 84. Simulation Options**

Parameter	Description
<b>Simulation Options</b>	
<b>Auto-calibration mode</b>	<p>Specifies whether you want to improve simulation performance by reducing calibration. There is no change to the generated RTL. The following autocalibration modes are available:</p> <ul style="list-style-type: none"> <li>• <b>Skip calibration</b>—provides the fastest simulation. It loads the settings calculated from the memory configuration and enters user mode.</li> <li>• <b>Quick calibration</b>—calibrates (without centering) one bit per group before entering user mode.</li> <li>• <b>Full calibration</b>—calibrates the same as in hardware, and includes all phases, delay sweeps, and centering on every data bit. You can use timing annotated memory models. Be aware that full calibration can take hours or days to complete.</li> </ul> <p>To perform proper PHY simulation, select <b>Quick calibration</b> or <b>Full calibration</b>. For more information, refer to the “Simulation Options” section in the <i>Simulating Memory IP</i> chapter.</p> <p>For QDR II, QDR II+ SRAM, and RLDRAM II, the Nios II-based sequencer must be selected to enable the auto calibration modes selection.</p> <p><i>Note:</i> This parameter is not available for MAX 10 devices.</p>
<b>Skip memory initialization delays</b>	<p>When you turn on this option, required delays between specific memory initialization commands are skipped to speed up simulation.</p> <p><i>Note:</i> This parameter is not available for MAX 10 devices.</p>
<b>Enable verbose memory model output</b>	<p>Turn on this option to display more detailed information about each memory access during simulation.</p> <p><i>Note:</i> This parameter is not available for MAX 10 devices.</p>
<b>Enable support for Nios II ModelSim flow in Eclipse</b>	<p>Initializes the memory interface for use with the <b>Run as Nios II ModelSim</b> flow with Eclipse.</p> <p>This parameter is not available for QDR II and QDR II+ SRAM.</p>

*continued...*



Parameter	Description
<i>Note:</i> This parameter is not available for MAX 10 devices.	
<b>Debug Options</b>	
<b>Debug level</b>	Specifies the debug level of the memory interface.
<b>Efficiency Monitor and Protocol Checker Settings</b>	
<b>Enable the Efficiency Monitor and Protocol Checker on the Controller Avalon Interface</b>	Enables efficiency monitor and protocol checker block on the controller Avalon interface. This option is not available for QDR II and QDR II+ SRAM, or for the MAX 10 device family, or for Arria V or Cyclone V designs using the Hard Memory Controller.

## 7.4. Intel Arria 10 External Memory Interface IP

This section contains information about parameterizing Intel Arria 10 External Memory Interface IP.

### 7.4.1. Qsys Interfaces

The interfaces in the Arria 10 External Memory Interface IP each have signals that can be connected in Qsys. The following tables list the signals available for each interface and provide a description and guidance on how to connect those interfaces.

Listed interfaces and signals are available in all configurations unless stated otherwise in the description column. For Arria 10 External Memory Interface for HPS, the `global_reset_reset_sink`, `pll_ref_clk_clock_sink`, `hps_emif_conduit_end`, `oct_conduit_end` and `mem_conduit_end` interfaces are the only available interfaces regardless of your configuration.

#### Arria 10 External Memory Interface IP Interfaces

**Table 85. Interface: afi\_clk\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
afi_clk	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3, RLDRAM 3, QDR IV</li><li>Hard PHY only</li></ul>	The Altera PHY Interface (AFI) clock output signal. The clock frequency in relation to the memory clock frequency depends on the <b>Clock rate of user logic</b> value set in the parameter editor. Connect this interface to the (clock input) conduit of the custom AFI-based memory controller connected to the <code>afi_conduit_end</code> or any user logic block that requires the generated clock frequency.



**Table 86. Interface: afi\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
afi_cal_success	Output	<ul style="list-style-type: none"> <li>DDR3, DDR4, LPDDR3, RLDRAM 3, QDR IV</li> <li>Hard PHY only</li> </ul>	<p>The Altera PHY Interface (AFI) signals between the external memory interface IP and the custom AFI-based memory controller.</p> <p>Connect this interface to the AFI conduit of the custom AFI-based memory controller.</p>
afi_cal_fail	Output		
afi_cal_req	Input		
afi_rlat	Output		
afi_wlat	Output		
afi_addr	Input		
afi_RST_n	Input		
afi_wdata_valid	Input		
afi_wdata	Input		
afi_rdata_en_full	Input		
afi_rdata	Output		
afi_rdata_valid	Output		
afi_rrank	Input		
afi_wrrank	Input		
afi_ba	Input	<ul style="list-style-type: none"> <li>DDR3, DDR4, RLDRAM 3</li> <li>Hard PHY only</li> </ul>	
afi_CS_n	Input	<ul style="list-style-type: none"> <li>DDR3, DDR4, LPDDR3, RLDRAM 3</li> <li>Hard PHY only</li> </ul>	
afi_cke	Input	<ul style="list-style-type: none"> <li>DDR3, DDR4, LPDDR3</li> <li>Hard PHY only</li> </ul>	
afi_odt	Input		
afi_dqs_burst	Input		
afi_ap	Input	<ul style="list-style-type: none"> <li>QDR IV</li> <li>Hard PHY only</li> </ul>	
afi_pe_n	Output		
afi_ainv	Input		
afi_ld_n	Input		
afi_rw_n	Input		
afi_cfg_n	Input		
afi_lbk0_n	Input		
afi_lbk1_n	Input		
afi_rdata_dinv	Output	<ul style="list-style-type: none"> <li>QDR IV</li> <li>Hard PHY only</li> </ul>	<p>The Altera PHY Interface (AFI) signals between the external memory interface IP and the custom AFI-based memory controller.</p> <p>Connect this interface to the AFI conduit of the custom AFI-based memory controller.</p>
afi_wdata_dinv	Input		

*continued...*



Signals in Interface	Direction	Availability	Description
afi_we_n	Input	<ul style="list-style-type: none"><li>DDR3, RLDRAM 3</li><li>Hard PHY only</li></ul>	The Altera PHY Interface (AFI) signals between the external memory interface IP and the custom AFI-based memory controller. Connect this interface to the AFI conduit of the custom AFI-based memory controller. For more information, refer to the <a href="#">AFI 4.0 Specification</a> .
afi_dm	Input	<ul style="list-style-type: none"><li>DDR3, LPDDR3, RLDRAM 3</li><li>Hard PHY only</li><li><b>Enable DM pins=True</b></li></ul>	
afi_ras_n	Input	<ul style="list-style-type: none"><li>DDR3</li><li>Hard PHY only</li></ul>	
afi_cas_n	Input		
afi_rm	Input	<ul style="list-style-type: none"><li>DDR3</li><li>Hard PHY only</li><li>LRDIMM with <b>Number of rank multiplication pins &gt; 0</b></li></ul>	
afi_par	Input	<ul style="list-style-type: none"><li>DDR3</li><li>Hard PHY only</li><li>RDIMM/LRDIMM</li><li>DDR4</li><li>Hard PHY only</li><li><b>Enable alert_n/par pins = True</b></li></ul>	
afi_bg	Input	<ul style="list-style-type: none"><li>DDR4</li><li>Hard PHY only</li></ul>	
afi_act_n	Input		
afi_dm_n	Input	<ul style="list-style-type: none"><li>DDR4</li><li>Hard PHY only</li><li><b>Enable DM pins=True</b></li></ul>	
afi_ref_n	Input	<ul style="list-style-type: none"><li>RLDRAM 3</li><li>Hard PHY only</li></ul>	

**Table 87. Interface: afi\_half\_clk\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
afi_half_clk	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3, RLDRAM 3, QDR IV</li><li>Hard PHY only</li></ul>	The Altera PHY Interface (AFI) half clock output signal. The clock runs at half the frequency of the AFI clock (afi_clk_clock). Connect this interface to the clock input conduit of the user logic block that needs to be clocked at the generated clock frequency.

**Table 88. Interface: afi\_reset\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
afi_reset_n	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3, RLDRAM 3, QDR IV</li><li>Hard PHY only</li></ul>	The Altera PHY Interface (AFI) reset output signal. Asserted when the PLL becomes unlocked or when the PHY is reset. Asynchronous assertion and synchronous deassertion. Connect this interface to the reset input conduit of the custom AFI-based memory controller



Signals in Interface	Direction	Availability	Description
			connected to the afi_conduit_end and all the user logic blocks that are under the AFI clock domain afi_clk or afi_half_clk clock).

**Table 89. Interface: cal\_debug\_avalon\_slave**

Interface type: Avalon Memory-Mapped Slave

Signals in Interface	Direction	Availability	Description
cal_debug_waitrequest	Output	<ul style="list-style-type: none"> <li>• EMIF Debug Toolkit</li> <li>• <b>On-Chip Debug Port=Export</b></li> </ul>	<p>The Avalon-MM signals between the external memory interface IP and the external memory interface Debug Component. Connect this interface to the (to_ioaux) Avalon-MM master of the <b>Arria 10 EMIF Debug Component</b> IP or to (cal_debug_out_avalon_master) Avalon-MM master of the other external memory interface IP that has exported the interface. If you are not using the Altera EMIF Debug Toolkit, connect this interface to the Avalon-MM master of the custom debug logic.</p> <p>When in daisy-chaining mode, ensure one of the connected Avalon masters is either the <b>Arria 10 EMIF Debug Component</b> IP or the external memory interface IP with <b>EMIF Debug Toolkit/On-Chip Debug Port</b> set to <b>Add EMIF Debug Interface</b>.</p>
cal_debug_read	Input		
cal_debug_write	Input		
cal_debug_addr	Input		
cal_debug_read_data	Output		
cal_debug_write_data	Input		
cal_debug_bytenable	Input		
cal_debug_read_data_valid	Output		

**Table 90. Interface: cal\_debug\_clk\_clock\_sink**

Interface type: Clock Input

Signals in Interface	Direction	Availability	Description
cal_debug_clk	Input	<ul style="list-style-type: none"> <li>• <b>EMIF Debug Toolkit / On-Chip Debug Port=Export</b></li> </ul>	<p>The calibration debug clock input signal. Connect this interface to the (avl_clk_out) clock output of the <b>Arria 10 EMIF Debug Component</b> IP or to (cal_debug_out_clk_clock_source) clock input of the other external memory interface IP, depending on which IP the cal_debug_avalon_slave interface is connecting to. If you are not using the Altera EMIF Debug Toolkit, connect this interface to the clock output of the custom debug logic.</p>

**Table 91. Interface: cal\_debug\_out\_avalon\_master**

Interface type: Avalon Memory-Mapped Master

Signals in Interface	Direction	Availability	Description
cal_debug_out_waitrequest	Input	<ul style="list-style-type: none"><li>• <b>EMIF Debug Toolkit / On-Chip Debug Port=Export</b></li><li>• Add EMIF Debug Interface with <b>Enable Daisy-Chaining for EMIF Debug Toolkit/ On-Chip Debug Port=True</b></li></ul>	The Avalon-MM signals between the external memory interface IP and the other external memory interface IP. Connect this interface to the (cal_debug_avalon_slave) Avalon-MM Master of the external memory interface IP that has exported the interface .
cal_debug_out_read	Output		
cal_debug_out_write	Output		
cal_debug_out_addr	Output		
cal_debug_out_read_data	Input		
cal_debug_out_write_data	Output		
cal_debug_out_bytewable	Output		
cal_debug_out_read_data_valid	Input		

**Table 92. Interface: cal\_debug\_out\_clk\_clock\_source**

Interface type: Clock Output

Signals in Interface	Direction	Availability	Description
cal_debug_out_clk	Output	<ul style="list-style-type: none"><li>• <b>EMIF Debug Toolkit / On-Chip Debug Port=Export</b></li><li>• Add EMIF Debug Interface with <b>Enable Daisy-Chaining for EMIF Debug Toolkit/ On-Chip Debug Port=True</b></li></ul>	The calibration debug clock output signal. For <b>EMIF Debug Toolkit/On-Chip Debug Port=Export</b> with <b>Enable Daisy-Chaining for EMIF Debug Toolkit/ On-Chip Debug Port=True</b> , the clock frequency follows the <i>cal_debug_clk</i> frequency. Otherwise, the clock frequency in relation to the memory clock frequency depends on the <b>Clock rate of user logic</b> value set in the parameter editor. Connect this interface to the (cal_debug_out_reset_reset_source) clock input of the other external memory interface IP where the cal_debug_avalon_master interface is being connected to or to any user logic block that needs to be clocked at the generated clock frequency.

**Table 93. Interface: cal\_debug\_out\_reset\_reset\_source**

Interface type: Reset Output

Signals in Interface	Direction	Availability	Description
cal_debug_out_reset_n	Output	<ul style="list-style-type: none"><li>• <b>EMIF Debug Toolkit / On-Chip Debug Port=Export</b></li><li>• Add EMIF Debug Interface with <b>Enable Daisy-Chaining for EMIF Debug Toolkit/ On-Chip Debug Port=True</b></li></ul>	The calibration debug reset output signal. Asynchronous assertion and synchronous deassertion. Connect this interface to the (cal_debug_reset_reset_sink) reset input of the other external memory interface IP where the



Signals in Interface	Direction	Availability	Description
			cal_debug_avalon_master interface being connected to and all the user logic blocks that are under the calibration debug clock domain (cal_debug_out_clk clock reset). If you are not using the Altera EMIF Debug Toolkit, connect this interface to the reset output of the custom debug logic.

**Table 94. Interface: cal\_debug\_reset\_reset\_sink**

Interface type: Reset Input

Signals in Interface	Direction	Availability	Description
cal_debug_reset_n	Input	• <b>EMIF Debug Toolkit / On-Chip Debug Port=Export</b>	The calibration debug reset input signal. Require asynchronous assertion and synchronous deassertion. Connect this interface to the (avl_rst_out) reset output of the <b>Arria 10 EMIF Debug Component</b> IP or to (cal_debug_out_reset_reset_source) clock input of the other external memory interface IP, depending on which IP the cal_debug_avalon_slave interface is being connected to.

**Table 95. Interface: clks\_sharing\_master\_out\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
clks_sharing_master_out	Input	• <b>Core clocks sharing=Master</b>	The core clock output signals. Connect this interface to the (clks_sharing_slave_in_conduit_end) conduit of the other external memory interface IP with the Core clock sharing set to slave or other PLL Slave.

**Table 96. Interface: clks\_sharing\_slave\_in\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
clks_sharing_slave_in	Input	• <b>Core clocks sharing=Slave</b>	The core clock input signals. Connect this interface to the (clks_sharing_master_out_conduit_end) conduit of the other external memory interface IP with the Core clock sharing set to Master or other PLL Master.

**Table 97. Interface: ctrl\_amm\_avalon\_slave**

Interface type: Avalon Memory-Mapped Slave

Signals in Interface	Direction	Availability	Description
amm_ready	Output	<ul style="list-style-type: none"><li>DDR3, DDR4 with Hard PHY &amp; Hard Controller</li><li>QDR II/II+/II+ Xtreme, QDR IV</li></ul>	The Avalon-MM signals between the external memory interface IP and the user logic. Connect this interface to the Avalon-MM Master of the user logic that needs to access the external memory device. For QDR II/II+/II+ Xtreme, connect the <code>ctrl_amm_avalon_slave_0</code> to the user logic for read request and connect the <code>ctrl_amm_avalon_slave_1</code> to the user logic for write request. In Ping Pong PHY mode, each interface controls only one memory device. Connect <code>ctrl_amm_avalon_slave_0</code> to the user logic that will access the first memory device, and connect <code>ctrl_amm_avalon_slave_1</code> to the user logic that will access the secondary memory device.
amm_read	Input		
amm_write	Input		
amm_address	Input		
amm_readdata	Output		
amm_writedata	Input		
amm_burstcount	Input		
amm_readdatavalid	Output		
amm_bytelenable	Input	<ul style="list-style-type: none"><li>DDR3, DDR4 with Hard PHY &amp; <b>Hard Controller and Enable DM pins=True</b></li><li>QDR II/II+/II+ Xtreme with <b>Enable BWS# pins=True</b></li></ul>	

**Table 98. Interface: ctrl\_auto\_prelude\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
ctrl_auto_prelude_req	Input	<ul style="list-style-type: none"><li>DDR3, DDR4 with Hard PHY &amp; Hard Controller and <b>Enable Auto-Prelude Control=True</b></li></ul>	The auto-precharge control input signal. Asserting the <code>ctrl_auto_prelude_req</code> signal while issuing a read or write burst instructs the external memory interface IP to issue read or write with auto-precharge to the external memory device. This precharges the row immediately after the command currently accessing it finishes, potentially speeding up a future access to a different row of the same bank. Connect this interface to the conduit of the user logic block that controls when the external memory interface IP needs to issue read or write with auto-precharge to the external memory device.

**Table 99. Interface: ctrl\_ecc\_user\_interrupt\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
ctrl_ecc_user_interrupt	Output	<ul style="list-style-type: none"><li>DDR3, DDR4 with Hard PHY &amp; Hard Controller and <b>Enable Error Detection and Correction Logic = True</b></li></ul>	Controller ECC user interrupt interface for connection to a custom control block that must be notified when ECC errors occur.



**Table 100. Interface: ctrl\_mmr\_avalon\_slave**

Interface type: Avalon Memory-Mapped Slave

Signals in Interface	Direction	Availability	Description
mmr_waitrequest	Output	• DDR3, DDR4, LPDDR3 with Hard PHY & Hard Controller and <b>Enable Memory-Mapped Configuration and Status Register (MMR)=True</b>	The Avalon-MM signals between the external memory interface IP and the user logic. Connect this interface to the Avalon-MM master of the user logic that needs to access the Memory-Mapped Configuration and Status Register (MMR) in the external memory interface IP.
mmr_read	Input		
mmr_write	Input		
mmr_address	Input		
mmr_readdata	Output		
mmr_writedata	Input		
mmr_burstcount	Input		
mmr_bytelenable	Input		
mmr_beginbursttransfer	Input		
mmr_readdatavalid	Output		

**Table 101. Interface: ctrl\_power\_down\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
ctrl_power_down_ack	Output	• DDR3, DDR4, LPDDR3 with Hard PHY & Hard Controller and <b>Enable Auto Power Down=True</b>	The auto power-down acknowledgment signals. When the <code>ctrl_power_down_ack</code> signal is asserted, it indicates that the external memory interface IP is placing the external memory device into power-down mode. Connect this interface to the conduit of the user logic block that requires the auto power-down status, or leave it unconnected.

**Table 102. Interface: ctrl\_user\_priority\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
ctrl_user_priority_hi	Input	• DDR3, DDR4, LPDDR3 with Hard PHY & Hard Controller • Avalon Memory-Mapped and <b>Enable Command Priority Control=true</b>	The command priority control input signal. Asserting the <code>ctrl_user_priority_hi</code> signal while issuing a read or write request instructs the external memory interface to treat it as a high-priority command. The external memory interface attempts to execute high-priority commands sooner, to reduce latency. Connect this interface to the conduit of the user logic block that determines when the external memory interface IP treats the read or write request as a high-priority command.

**Table 103. Interface: emif\_usr\_clk\_clock\_source**

Interface type: Clock Output

Signals in Interface	Direction	Availability	Description
emif_usr_clk	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3, with Hard PHY &amp; Hard Controller</li><li>QDR II/II+/II+ Xtreme</li><li>QDR IV</li></ul>	The user clock output signal. The clock frequency in relation to the memory clock frequency depends on the <b>Clock rate of user logic</b> value set in the parameter editor. Connect this interface to the clock input of the respective user logic connected to the <code>ctrl_amm_avalon_slave_0</code> interface, or to any user logic block that must be clocked at the generated clock frequency.

**Table 104. Interface: emif\_usr\_reset\_reset\_source**

Interface type: Reset Output

Signals in Interface	Direction	Availability	Description
emif_usr_reset_n	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3 with Hard PHY &amp; Hard Controller</li><li>QDR II/II+/II+ Xtreme</li><li>QDR IV</li></ul>	The user reset output signal. Asserted when the PLL becomes unlocked or the PHY is reset. Asynchronous assertion and synchronous deassertion. Connect this interface to the clock input of the respective user logic connected to the <code>ctrl_amm_avalon_slave_0</code> interface, or to any user logic block that must be clocked at the generated clock frequency.

**Table 105. Interface: emif\_usr\_clk\_sec\_clock\_source**

Interface type: Clock Output

Signals in Interface	Direction	Availability	Description
emif_usr_clk_sec	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, with Ping Pong PHY</li></ul>	The user clock output signal. The clock frequency in relation to the memory clock frequency depends on the <b>Clock rate of user logic</b> value set in the parameter editor. Connect this interface to the clock input of the respective user logic connected to the <code>ctrl_amm_avalon_slave_1</code> interface, or to any user logic block that must be clocked at the generated clock frequency.

**Table 106. Interface: emif\_usr\_reset\_sec\_reset\_source**

Interface type: Reset Output

Signals in Interface	Direction	Availability	Description
emif_usr_reset_n_sec	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, with Ping Pong PHY</li></ul>	The user reset output signal. Asserted when the PLL becomes unlocked or the PHY is reset. Asynchronous assertion and synchronous deassertion.



Signals in Interface	Direction	Availability	Description
			Connect this interface to the clock input of the respective user logic connected to the <code>ctrl_amm_avalon_slave_1</code> interface, or to any user logic block that must be clocked at the generated clock frequency.

**Table 107. Interface: global\_reset\_reset\_sink**

Interface type: Reset Input

Signals in Interface	Direction	Availability	Description
global_reset_n	Input	<ul style="list-style-type: none"> <li><b>Core Clock Sharing</b>=No Sharing / Master</li> </ul>	The global reset input signal. Asserting the <code>global_reset_n</code> signal causes the external memory interface IP to be reset and recalibrated. Connect this interface to the reset output of the asynchronous or synchronous reset source that controls when the external memory interface IP needs to be reset and recalibrated.

**Table 108. Interface: hps\_emif\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
hps_to_emif	Input	<ul style="list-style-type: none"> <li>Arria 10 EMIF for HPS IP</li> </ul>	The user interface signals between the external memory interface IP and the Hard Processor System (HPS). Connect this interface to the EMIF conduit of the Arria 10 Hard Processor System.
emif_to_hps	Output		

**Table 109. Interface: mem\_conduit\_end**

Interface type: Conduit

The memory interface signals between the external memory interface IP and the external memory device.

Export this interface to the top level for I/O assignments. Typically `mem_rm[0]` and `mem_rm[1]` connect to CS2# and CS3# of the memory buffer of all LRDIMM slots.

Signals in Interface	Direction	Availability
mem_ck	Output	Always available
mem_ck_n	Output	
mem_reset_n	Output	
mem_a	Output	
mem_k_n	Output	<ul style="list-style-type: none"> <li>QDR II</li> </ul>
mem_ras_n	Output	<ul style="list-style-type: none"> <li>DDR3</li> </ul>

*continued...*



Signals in Interface	Direction	Availability
mem_cas_n	Output	
mem_odt	Output	• DDR3, DDR4, LPDDR3
mem_dqs	Bidirectional	
mem_dqs_n	Bidirectional	
mem_ba	Output	• DDR3, DDR4, RLDRAM 3
mem_cs_n	Output	• DDR3, DDR4, LPDDR3, RLDRAM 3
mem_dq	Bidirectional	
mem_we_n	Output	• DDR3, RLDRAM 3
mem_dm	Output	• DDR3, LPDDR3, RLDRAM 3 with <b>Enable DM pins=True</b>
mem_rm	Output	• DDR3, RLDRAM 3 with <b>Memory format=LRDIMM</b> and <b>Number of rank multiplication pins &gt; 0</b>
mem_par	Output	• DDR3 with <b>Memory format=RDIMM / LRDIMM</b> • DDR4 with <b>Enable alert_n/par pins=True</b>
mem_alert_n	Input	
mem_cke	Output	• DDR3, DDR4, LPDDR3
mem_bg	Output	• DDR4
mem_act_n	Output	
mem_db1_n	Bidirectional	• DDR4 with <b>Enable DM pins=True</b> or <b>Write DBI=True</b> or <b>Read DBI=True</b>
mem_k	Output	• QDR II/II+/II+ Xtreme
mem_wps_n	Output	
mem_rps_n	Output	
mem_doff_n	Output	
mem_d	Output	
mem_q	Input	
mem_cq	Input	
mem_cq_n	Input	
mem_bws_n	Output	
mem_dk	Output	
mem_dk_n	Output	
mem_ref_n	Output	
mem_qk	Input	• QDR II/II+/II+ Xtreme with <b>Enable BWS# pins=True</b>
mem_qk_n	Input	• RLDRAM 3
mem_ap	Output	• QDR IV with <b>Use Address Parity Bit=True</b>
mem_pe_n	Input	• QDR IV with <b>Use Address Parity Bit=True</b>
mem_ainv	Output	• QDR IV with <b>Address Bus Inversion=True</b>
mem_lda_n	Output	• QDR IV

continued...



Signals in Interface	Direction	Availability
mem_lda_b	Output	• QDR IV
mem_rwa_n	Output	• QDR IV
mem_rwb_n	Output	• QDR IV
mem_cfg_n	Output	• QDR IV
mem_lbk0_n	Output	• QDR IV
mem_lbk1_n	Output	• QDR IV
mem_dka	Output	• QDR IV
mem_dka_n	Output	• QDR IV
mem_dkb	Output	• QDR IV
mem_dkb_n	Output	• QDR IV
mem_qka	Input	• QDR IV
mem_qka_n	Input	• QDR IV
mem_qkb	Input	• QDR IV
mem_qkb_n	Input	• QDR IV
mem_dqa	Bidirectional	• QDR IV
mem_dqb	Bidirectional	• QDR IV
mem_dinva	Bidirectional	• QDR IV with <b>Data Bus Inversion=True</b>
mem_dinvb	Bidirectional	• QDR IV with <b>Data Bus Inversion=True</b>

**Table 110. Interface: oct\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
oct_rzqin	Input	Always available	The On-Chip Termination (OCT) RZQ reference resistor input signal. Export this interface to the top level for I/O assignments.

**Table 111. Interface: pll\_ref\_clk\_clock\_sink**

Signals in Interface	Interface Type	Direction	Availability	Description
pll_ref_clk	Clock Input	Input	• <b>Core clock sharing=No Sharing / Master</b>	The PLL reference clock input signal. Connect this interface to the clock output of the clock source that matches the <b>PLL reference clock frequency</b> value set in the parameter editor.

**Table 112. Interface: status\_conduit\_end**

Signals in Interface	Interface Type	Direction	Availability	Description
local_cal_success	Conduit	Output	Always available	The PHY calibration status output signals. When the local_cal_success signal is asserted, it indicates that the PHY calibration was successful. Otherwise, if local_cal_fail signal is asserted, it indicates that PHY calibration has failed.
local_cal_fail				Connect this interface to the conduit of the user logic block that requires the calibration status information, or leave it unconnected.

**Related Information**

[http://www.alterawiki.com/wiki/Measuring\\_Channel\\_Signal\\_Integrity](http://www.alterawiki.com/wiki/Measuring_Channel_Signal_Integrity)

#### 7.4.2. Generated Files for Arria 10 External Memory Interface IP

When you complete the IP generation flow, there are generated files created in your project directory. The directory structure created varies somewhat, depending on the tool used to parameterize and generate the IP.

**Note:** The PLL parameters are statically defined in the **<variation\_name>\_parameters.tcl** at generation time. To ensure timing constraints and timing reports are correct, when you edit the PLL parameters, apply those changes to the PLL parameters in this file.

The following table lists the generated directory structure and key files created when generating the IP.

**Table 113. Generated Directory Structure and Key Files for Synthesis**

Directory	File Name	Description
working_dir/	working_dir/<Top-level Name>/	The Qsys files for your IP component or system based on your configuration.
working_dir/<Top-level Name>/	*.ppf	Pin Planner File for use with the Pin Planner.
working_dir/<Top-level Name>/synth/	<Top-level Name>.v or <Top-level Name>.vhdl	Qsys generated top-level wrapper for synthesis.
working_dir/<Top-level Name>/altera_emif_<acds version>/synth/	*.v or (*.v and *.vhdl)	Arria 10 EMIF (non-HPS) top-level dynamic wrapper files for synthesis. This wrapper instantiates the EMIF ECC and EMIF Debug Interface IP core.

**continued...**



Directory	File Name	Description
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_a10_hps_&lt;acds version&gt;/synth/</code>	<code>*.v or (*.v and *.vhdl)</code>	Arria 10 EMIF for HPS top-level dynamic wrapper files for synthesis.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_a10_hps_&lt;acds version&gt;/synth/</code>	<code>*.sv, *.sdc, *.tcl and *.hex and *_readme.txt</code>	Arria 10 EMIF Core RTL, constraints files, ROM content files and information files for synthesis. Whether the file type is set to Verilog or VHDL, all the Arria 10 EMIF Core RTL files will be generated as a SystemVerilog file. The <code>readme.txt</code> file contains information and guidelines specific to your configuration.
<code>working_dir/&lt;Top-level Name&gt;/&lt;other components&gt;_&lt;acds version&gt;/synth/</code>	<code>*</code>	Other EMIF ECC, EMIF Debug Interface IP or Merlin Interconnect component files for synthesis.

**Table 114. Generated Directory Structure and Key Files for Simulation**

Directory	File Name	Description
<code>working_dir/&lt;Top-level Name&gt;/sim/</code>	<code>&lt;Top-level Name&gt;.v or &lt;Top-level Name&gt;.vhdl</code>	Qsys generated top-level wrapper for simulation.
<code>working_dir/&lt;Top-level Name&gt;/sim/&lt;simulator vendor&gt;/</code>	<code>*.tcl, *.cds.lib, *.lib, *.var, *.sh, *.setup</code>	Simulator-specific simulation scripts.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_&lt;acds version&gt;/sim/</code>	<code>*.v or *.vhdl</code>	Arria 10 EMIF (non-HPS) top-level dynamic wrapper files for simulation. This wrapper instantiates the EMIF ECC and EMIF Debug Interface IP cores.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_a10_hps_&lt;acds version&gt;/sim/</code>	<code>*.v or *.vhdl</code>	Arria 10 EMIF for HPS top-level dynamic wrapper files for simulation.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_arch_nf_&lt;acds version&gt;/sim/</code>	<code>*.sv or (*.sv and *.vhdl), *.hex and *_readme.txt</code>	Arria 10 EMIF RTL, ROM content files, and information files for simulation. For SystemVerilog / Mix language simulator, you may directly use the files from this folder. For VHDL-only simulator, other than the ROM content files, you have to use files in <code>&lt;current folder&gt;/mentor</code> directory instead. The <code>readme.txt</code> file contains information and guidelines specific to your configuration.
<code>working_dir/&lt;Top-level Name&gt;/&lt;other components&gt;_&lt;acds version&gt;/sim/</code>		Other EMIF ECC, EMIF Debug Interface IP, or Merlin Interconnect component files for simulation

**Table 115. Generated Directory Structure and Key Files for Qsys-Generated Testbench System**

Directory	File Name	Description
<code>working_dir/&lt;Top-level Name&gt;_tb/</code>	<code>*.qsys</code>	The Qsys files for the QSYS generated testbench system.
<code>working_dir/&lt;Top-level Name&gt;_tb/sim/</code>	<code>&lt;Top-level Name&gt;.v or &lt;Top-level Name&gt;.vhdl</code>	Qsys generated testbench file for simulation. This wrapper instantiates BFM components. For Arria 10 EMIF IP, this module should instantiate the memory model for the memory conduit being exported from your created system.
<code>working_dir/&lt;Top-level Name&gt;_tb/&lt;Top-level Name&gt;_&lt;id&gt;/sim/</code>	<code>&lt;Top-level Name&gt;.v or &lt;Top-level Name&gt;.vhdl</code>	Qsys generated top-level wrapper for simulation.
<code>working_dir/&lt;Top-level Name&gt;_tb/sim/&lt;simulator vendor&gt;/</code>	<code>*.tcl, *cds.lib, *.lib, *.var, *.sh, *.setup</code>	Simulator-specific simulation scripts.
<code>working_dir/&lt;Top-level Name&gt;_tb/sim/&lt;simulator vendor&gt;/</code>	<code>*.v or *.vhdl</code>	Arria 10 EMIF (non-HPS) top-level dynamic wrapper files for simulation. This wrapper instantiates the EMIF ECC and EMIF Debug Interface IP cores.
<code>working_dir/&lt;Top-level Name&gt;_tb/altera_emif_a10_hps_&lt;acds version&gt;/sim/</code>	<code>*.v or *.vhdl</code>	Arria 10 EMIF for HPS top-level dynamic wrapper files for simulation.
<code>working_dir/&lt;Top-level Name&gt;_tb/altera_emif_arch_nf_&lt;acds version&gt;/sim/</code>	<code>*sv or (*.sv and *.vhdl), *.hex and *_readme.txt</code>	Arria 10 EMIF Core RTL, ROM content files and information files for simulation. For SystemVerilog / Mix language simulator, you may use the files from this folder. For VHDL-only simulator, other than the ROM content files, you must use files in the <current folder>/mentor directory instead. The <code>readme.txt</code> file contains information and guidelines specific to your configuration.
<code>working_dir/&lt;Top-level Name&gt;_tb/sim/altera_emif_arch_nf_&lt;acds version&gt;/sim/mentor/</code>	<code>*.sv and *.vhdl</code>	Arria 10 EMIF Core RTL for simulation. Only available when you create a VHDL simulation model. All .sv files are Mentor-tagged encrypted IP (IEEE Encrypted Verilog) for VHDL-only simulator support.
<code>working_dir/&lt;Top-level Name&gt;_tb/&lt;other components&gt;_&lt;acds version&gt;/sim/</code>	<code>*</code>	Other EMIF ECC, EMIF Debug Interface IP or Merlin Interconnect component files for simulation.
<code>working_dir/&lt;Top-level Name&gt;_tb/&lt;other components&gt;_&lt;acds version&gt;/sim/mentor/</code>	<code>*</code>	Other EMIF ECC, EMIF Debug Interface IP or Merlin Interconnect component files for simulation. Only available depending on individual component simulation model support and when creating a VHDL simulation model. All files in this folder are

***continued...***



Directory	File Name	Description
		Mentor-tagged encrypted IP (IEEE Encrypted Verilog) for VHDL-only simulator support.

**Table 116. Generated Directory Structure and Key Files for Example Simulation Designs**

Directory	File Name	Description
<i>working_dir/_example_design*/</i>	*.qsys, *.tcl and readme.txt	Qsys files, generation scripts, and information for generating the Arria 10 EMIF IP example design. These files are available only when you generate an example design. You may open the .qsys file in Qsys to add more components to the example design.
<i>working_dir/_example_design*/sim/ed_sim/sim/</i>	*.tcl, *cds.lib, *.lib, *.var, *.sh, *.setup	Simulator-specific simulation scripts.
<i>working_dir/_example_design*/sim/ed_sim/&lt;simulator vendor&gt;/</i>	*.v or *.vhd	Qsys-generated top-level wrapper for simulation.
<i>working_dir/_example_design*/sim/ip/ed_sim/ed_sim_emif_0/altera_emif_&lt;acds_version&gt;/simip/ed_sim/ed_sim_emif_0/altera_emif_</i>	*.v or *.vhd)	Arria 10 EMIF (non-HPS) top-level dynamic wrapper files for simulation. This wrapper instantiates the EMIF ECC and EMIF Debug Interface IP cores.
<i>working_dir/_example_design*/sim/ip/ed_sim/ed_sim_emif_0/altera_emif_arch_nf_&lt;acds_version&gt;/sim/</i>	*sv or (*.sv and *.vhd), *.hex and *_readme.txt	Arria 10 EMIF RTL, ROM content files, and information files for simulation. For SystemVerilog / Mix language simulator, you may directly use the files from this folder. For VHDL-only simulator, other than the ROM content files, you have to use files in <current folder>/mentor directory instead. The readme.txt file contains information and guidelines specific to your configuration.
<i>working_dir/_example_design*/sim/ed_sim/&lt;other components&gt;_&lt;acds_version&gt;/sim/and</i> <i>working_dir/_example_design*/sim/ip/ed_sim/&lt;other components&gt;/sim/and</i> <i>working_dir/_example_design*/sim/ip/ed_sim/&lt;other components&gt;/&lt;other components&gt;_&lt;acds_version&gt;</i>	*	Other EMIF ECC, EMIF Debug Interface IP, or Merlin Interconnect component files for simulation

**Table 117. Generated Directory Structure and Key Files for Example Synthesis Designs**

Directory	File Name	Description
<code>working_dir/_example_design*/</code>	<code>*.qsys, *.tcl and readme.txt</code>	Qsys files, generation scripts, and information for generating the Arria 10 EMIF IP example design. These files are available only when you generate an example design. You may open the .qsys file in Qsys to add more components to the example design.
<code>working_dir/_example_design*/qii/ed_synth/synth</code>	<code>*.v or (*.v and *.vhd)</code>	Qsys-generated top-level wrapper for synthesis.
<code>working_dir/_example_design*/qii/ip/ed_synth/ed_synth_emif_0/altera_emif_&lt;acds_version&gt;/synth</code>	<code>*.v or (*.v and *.vhd)</code>	Arria 10 EMIF (non HPS) top-level dynamic wrapper files for synthesis. This wrapper instantiates the EMIF ECC and EMIF debug interface core IP.
<code>working_dir/_example_design*/qii/ip/ed_synth/ed_synth_emif_0/altera_emif_arch_nf_&lt;acds_version&gt;/synth/</code>	<code>*.sv, *.sdc, *.tcl, *.hex, and *_readme.txt</code>	Arria 10 EMIF Core RTL, constraints files, ROM content files and information files for synthesis. Whether the file type is set to Verilog or VHDL, all the Arria 10 EMIF Core RTL files are generated as a SystemVerilog file. The <code>readme.txt</code> file contains information and guidelines specific to your configuration.
<code>working_dir/_example_design*/sim/ed_synth/&lt;other_components&gt;_&lt;acds_version&gt;/synth</code> and <code>working_dir/_example_design*/sim/ip/ed_synth/&lt;other_components&gt;/synth/</code> and <code>working_dir/_example_design*/sim/ip/ed_synth/&lt;other_components&gt;/&lt;other_components&gt;_&lt;acds_version&gt;/synth</code>	*	Other EMIF ECC, EMIF debug interface IP, or Merlin interconnect component files for synthesis.

### 7.4.3. Arria 10 EMIF IP DDR4 Parameters

The Arria 10 EMIF IP parameter editor allows you to parameterize settings for the Arria 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Arria 10 EMIF IP.



#### 7.4.3.1. Arria 10 EMIF IP DDR4 Parameters: General

**Table 118. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 119. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 120. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Use recommended PLL reference clock frequency</b>	PHY_DDR4_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.

*continued...*



Display Name	Identifier	Description
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 121. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

#### 7.4.3.2. Arria 10 EMIF IP DDR4 Parameters: Memory

**Table 122. Group: Memory / Topology**

Display Name	Identifier	Description
<b>DQS group of ALERT#</b>	MEM_DDR4_ALERT_N_DQS_GROUP	Select the DQS group with which the ALERT# pin is placed.
<b>ALERT# pin placement</b>	MEM_DDR4_ALERT_N_PLACEMENT_ENUM	Specifies placement for the mem_alert_n signal. If you select "I/O Lane with Address/Command Pins", you can pick the I/O lane and pin index in the add/cmd bank with the subsequent drop down menus. If you select "I/O Lane with DQS Group", you can specify the DQS group with which to place the mem_alert_n pin. If you select "Automatically select a location", the IP automatically selects a pin for the mem_alert_n signal. If you select this option, no additional location constraints can be applied to the mem_alert_n pin,

*continued...*



Display Name	Identifier	Description
		or a fitter error will result during compilation. For optimum signal integrity, you should choose "I/O Lane with Address/Command Pins". For interfaces containing multiple memory devices, it is recommended to connect the ALERT# pins together to the ALERT#pin on the FPGA.
<b>Enable ALERT#/PAR pins</b>	MEM_DDR4_ALERT_PAR_EN	Allows address/command calibration, which may provide better margins on the address/command bus. The alert_n signal is not accessible in the AFI or Avalon domains. This means there is no way to know whether a parity error has occurred during user mode. The parity pin is a dedicated pin in the address/command bank, but the alert_n pin can be placed in any bank that spans the memory interface. You should explicitly choose the location of the alert_n pin and place it in the address/command bank.
<b>Bank address width</b>	MEM_DDR4_BANK_ADDRESS_WIDTH	Specifies the number of bank address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of bank address pins needed for access to all available banks.
<b>Bank group width</b>	MEM_DDR4_BANK_GROUP_WIDTH	Specifies the number of bank group pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of bank group pins needed for access to all available bank groups.
<b>Chip ID width</b>	MEM_DDR4_CHIP_ID_WIDTH	Specifies the number of chip ID pins. Only applicable to registered and load-reduced DIMMs that use 3DS/TSV memory devices.
<b>Number of clocks</b>	MEM_DDR4_CK_WIDTH	Specifies the number of CK/CK# clock pairs exposed by the memory interface. Usually more than 1 pair is required for RDIMM/LRDIMM formats. The value of this parameter depends on the memory device selected; refer to the data sheet for your memory device.
<b>Column address width</b>	MEM_DDR4_COL_ADDRESS_WIDTH	Specifies the number of column address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of address pins needed for access to all available columns.
<b>Number of chip selects per DIMM</b>	MEM_DDR4_CS_PER_DIMM	Specifies the number of chip selects per DIMM.
<b>Number of chip selects</b>	MEM_DDR4_DISCRETE_CS_WIDTH	Specifies the total number of chip selects in the interface, up to a maximum of 4. This parameter applies to discrete components only.
<b>Data mask</b>	MEM_DDR4_DM_EN	Indicates whether the interface uses data mask (DM) pins. This feature allows specified portions of the data bus to be written to memory (not available in x4 mode). One DM pin exists per DQS group.
<b>Number of DQS groups</b>	MEM_DDR4_DQS_WIDTH	Specifies the total number of DQS groups in the interface. This value is automatically calculated as the DQ width divided by the number of DQ pins per DQS group.
<b>DQ pins per DQS group</b>	MEM_DDR4_DQ_PER_DQS	Specifies the total number of DQ pins per DQS group.
<b>DQ width</b>	MEM_DDR4_DQ_WIDTH	Specifies the total number of data pins in the interface. The maximum supported width is 144, or 72 in Ping Pong PHY mode.
<b>Memory format</b>	MEM_DDR4_FORMAT_ENUM	Specifies the format of the external memory device. The following formats are supported: Component - a Discrete memory device; UDIMM - Unregistered/Unbuffered DIMM where address/control, clock, and data are unbuffered; RDIMM - Registered DIMM where address/control and clock

*continued...*



Display Name	Identifier	Description
		are buffered; LRDIMM - Load Reduction DIMM where address/control, clock, and data are buffered. LRDIMM reduces the load to increase memory speed and supports higher densities than RDIMM; SODIMM - Small Outline DIMM is similar to UDIMM but smaller in size and is typically used for systems with limited space. Some memory protocols may not be available in all formats.
<b>Number of DIMMs</b>	MEM_DDR4_NUM_OF_DIMMS	Total number of DIMMs.
<b>Number of physical ranks per DIMM</b>	MEM_DDR4_RANKS_PER_DIMM	Number of ranks per DIMM. For LRDIMM, this represents the number of physical ranks on the DIMM behind the memory buffer
<b>Read DBI</b>	MEM_DDR4_READ_DB_I	Specifies whether the interface uses read data bus inversion (DBI). Enable this feature for better signal integrity and read margin. This feature is not available in x4 configurations.
<b>Row address width</b>	MEM_DDR4_ROW_ADDRESS_WIDTH	Specifies the number of row address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of address pins needed for access to all available rows.
<b>Write DBI</b>	MEM_DDR4_WRITE_DB_I	Indicates whether the interface uses write data bus inversion (DBI). This feature provides better signal integrity and write margin. This feature is unavailable if Data Mask is enabled or in x4 mode.

**Table 123. Group: Memory / Latency and Burst**

Display Name	Identifier	Description
<b>Addr/CMD parity latency</b>	MEM_DDR4_AC_PARITY_LATENCY	Additional latency incurred by enabling address/command parity check. Select a value to enable address/command parity with the latency associated with the selected value. Select Disable to disable address/command parity.
<b>Memory additive CAS latency setting</b>	MEM_DDR4_ATCL_ENUM	Determines the posted CAS additive latency of the memory device. Enable this feature to improve command and bus efficiency, and increase system bandwidth.
<b>Burst Length</b>	MEM_DDR4_BL_ENUM	Specifies the DRAM burst length which determines how many consecutive addresses should be accessed for a given read/write command.
<b>Read Burst Type</b>	MEM_DDR4_BT_ENUM	Indicates whether accesses within a given burst are in sequential or interleaved order. Select sequential if you are using the Intel-provided memory controller.
<b>Memory CAS latency setting</b>	MEM_DDR4_TCL	Specifies the number of clock cycles between the read command and the availability of the first bit of output data at the memory device. Overall read latency equals the additive latency (AL) + the CAS latency (CL). Overall read latency depends on the memory device selected; refer to the datasheet for your device.
<b>Memory write CAS latency setting</b>	MEM_DDR4_WTCL	Specifies the number of clock cycles from the release of internal write to the latching of the first data in at the memory device. This value depends on the memory device selected; refer to the datasheet for your device.



**Table 124. Group: Memory / Mode Register Settings**

Display Name	Identifier	Description
<b>Auto self-refresh method</b>	MEM_DDR4_ASR_ENUM	Indicates whether to enable or disable auto self-refresh. Auto self-refresh allows the controller to issue self-refresh requests, rather than manually issuing self-refresh in order for memory to retain data.
<b>Fine granularity refresh</b>	MEM_DDR4_FINE_GRANULARITY_REFRESH	Increased frequency of refresh in exchange for shorter refresh. Shorter tRFC and increased cycle time can produce higher bandwidth.
<b>Internal VrefDQ monitor</b>	MEM_DDR4_INTERNAL_VREFDQ_MONITOR	Indicates whether to enable the internal VrefDQ monitor.
<b>ODT input buffer during powerdown mode</b>	MEM_DDR4_ODT_IN_POWERDOWN	Indicates whether to enable on-die termination (ODT) input buffer during powerdown mode.
<b>Read preamble</b>	MEM_DDR4_READ_PREAMBLE	Number of read preamble cycles. This mode register setting determines the number of cycles DQS (read) will go low before starting to toggle.
<b>Self refresh abort</b>	MEM_DDR4_SELF_REFRESH_ABORT	Self refresh abort for latency reduction.
<b>Temperature controlled refresh enable</b>	MEM_DDR4_TEMP_CONTROLLED_RFSH_ENA	Indicates whether to enable temperature controlled refresh, which allows the device to adjust the internal refresh period to be longer than tREFI of the normal temperature range by skipping external refresh commands.
<b>Temperature controlled refresh range</b>	MEM_DDR4_TEMP_CO_NTROLLED_RFSH_RANGE	Indicates temperature controlled refresh range where normal temperature mode covers 0C to 85C and extended mode covers 0C to 95C.
<b>Write preamble</b>	MEM_DDR4_WRITE_PREAMBLE	Write preamble cycles.

#### 7.4.3.3. Arria 10 EMIF IP DDR4 Parameters: Mem I/O

**Table 125. Group: Mem I/O / Memory I/O Settings**

Display Name	Identifier	Description
<b>DB Host Interface DQ Driver</b>	MEM_DDR4_DB_DQ_DRV_ENUM	Specifies the driver impedance setting for the host interface of the data buffer. This parameter determines the value of the control word BC03 of the data buffer. Perform board simulation to obtain the optimal value for this setting.
<b>DB Host Interface DQ RTT_NOM</b>	MEM_DDR4_DB_RTT_NOM_ENUM	Specifies the RTT_NOM setting for the host interface of the data buffer. Only "RTT_NOM disabled" is supported. This parameter determines the value of the control word BC00 of the data buffer.
<b>DB Host Interface DQ RTT_PARK</b>	MEM_DDR4_DB_RTT_PARK_ENUM	Specifies the RTT_PARK setting for the host interface of the data buffer. This parameter determines the value of control word BC02 of the data buffer. Perform board simulation to obtain the optimal value for this setting.
<b>DB Host Interface DQ RTT_WR</b>	MEM_DDR4_DB_RTT_WR_ENUM	Specifies the RTT_WR setting of the host interface of the data buffer. This parameter determines the value of the control word BC01 of the data buffer. Perform board simulation to obtain the optimal value for this setting.
<b>Use recommended initial VrefDQ value</b>	MEM_DDR4_DEFAULT_VREFOUT	Specifies to use the recommended initial VrefDQ value. This value is used as a starting point and may change after calibration.

*continued...*



Display Name	Identifier	Description
<b>Output drive strength setting</b>	MEM_DDR4_DRV_STR_ENUM	Specifies the output driver impedance setting at the memory device. To obtain optimum signal integrity performance, select option based on board simulation results.
<b>RCD CA Input Bus Termination</b>	MEM_DDR4_RCD_CA_IBT_ENUM	Specifies the input termination setting for the following pins of the registering clock driver: DA0..DA17, DBA0..DBA1, DBG0..DBG1, DACT_n, DC2, DPAR. This parameter determines the value of bits DA[1:0] of control word RC7x of the registering clock driver. Perform board simulation to obtain the optimal value for this setting.
<b>RCD DCKE Input Bus Termination</b>	MEM_DDR4_RCD_CKE_IBT_ENUM	Specifies the input termination setting for the following pins of the registering clock driver: DCKE0, DCKE1. This parameter determines the value of bits DA[5:4] of control word RC7x of the registering clock driver. Perform board simulation to obtain the optimal value for this setting.
<b>RCD DCS[3:0]_n Input Bus Termination</b>	MEM_DDR4_RCD_CS_IBT_ENUM	Specifies the input termination setting for the following pins of the registering clock driver: DCS[3:0]_n. This parameter determines the value of bits DA[3:2] of control word RC7x of the registering clock driver. Perform board simulation to obtain the optimal value for this setting.
<b>RCD DDT Input Bus Termination</b>	MEM_DDR4_RCD_ODT_IBT_ENUM	Specifies the input termination setting for the following pins of the registering clock driver: DDT0, DDT1. This parameter determines the value of bits DA[7:6] of control word RC7x of the registering clock driver. Perform board simulation to obtain the optimal value for this setting.
<b>ODT Rtt nominal value</b>	MEM_DDR4_RTT_NOM_ENUM	Determines the nominal on-die termination value applied to the DRAM. The termination is applied any time that ODT is asserted. If you specify a different value for RTT_WR, that value takes precedence over the values mentioned here. For optimum signal integrity performance, select your option based on board simulation results.
<b>RTT PARK</b>	MEM_DDR4_RTT_PAR_K	If set, the value is applied when the DRAM is not being written AND ODT is not asserted HIGH.
<b>Dynamic ODT (Rtt_WR) value</b>	MEM_DDR4_RTT_WR_ENUM	Specifies the mode of the dynamic on-die termination (ODT) during writes to the memory device (used for multi-rank configurations). For optimum signal integrity performance, select this option based on board simulation results.
<b>RCD and DB Manufacturer (LSB)</b>	MEM_DDR4_SPD_133_RCD_DB_VENDOR_LSB	Specifies the LSB of the ID code of the registering clock driver and data buffer manufacturer. The value must come from Byte 133 of the SPD from the DIMM vendor.
<b>RCD and DB Manufacturer (MSB)</b>	MEM_DDR4_SPD_134_RCD_DB_VENDOR_MSB	Specifies the MSB of the ID code of the registering clock driver and data buffer manufacturer. The value must come from Byte 134 of the SPD from the DIMM vendor.
<b>RCD Revision Number</b>	MEM_DDR4_SPD_135_RCD_REV	Specifies the die revision of the registering clock driver. The value must come from Byte 135 of the SPD from the DIMM vendor.
<b>SPD Byte 137 - RCD Drive Strength for Command/Address</b>	MEM_DDR4_SPD_137_RCD_CA_DRV	Specifies the drive strength of the registering clock driver's control and command/address outputs to the DRAM. The value must come from Byte 137 of the SPD from the DIMM vendor.
<b>SPD Byte 138 - RCD Drive Strength for CK</b>	MEM_DDR4_SPD_138_RCD_CK_DRV	Specifies the drive strength of the registering clock driver's clock outputs to the DRAM. The value must come from Byte 138 of the SPD from the DIMM vendor.
<b>DB Revision Number</b>	MEM_DDR4_SPD_139_DB_REV	Specifies the die revision of the data buffer. The value must come from Byte 139 of the SPD from the DIMM vendor.

*continued...*



Display Name	Identifier	Description
<b>SPD Byte 140 - DRAM VrefDQ for Package Rank 0</b>	MEM_DDR4_SPD_140_DRAM_VREFDQ_R0	Specifies the VrefDQ setting for package rank 0 of an LRDIMM. The value must come from Byte 140 of the SPD from the DIMM vendor.
<b>SPD Byte 141 - DRAM VrefDQ for Package Rank 1</b>	MEM_DDR4_SPD_141_DRAM_VREFDQ_R1	Specifies the VrefDQ setting for package rank 1 of an LRDIMM. The value must come from Byte 141 of the SPD from the DIMM vendor.
<b>SPD Byte 142 - DRAM VrefDQ for Package Rank 2</b>	MEM_DDR4_SPD_142_DRAM_VREFDQ_R2	Specifies the VrefDQ setting for package rank 2 (if it exists) of an LRDIMM. The value must come from Byte 142 of the SPD from the DIMM vendor.
<b>SPD Byte 143 - DRAM VrefDQ for Package Rank 3</b>	MEM_DDR4_SPD_143_DRAM_VREFDQ_R3	Specifies the VrefDQ setting for package rank 3 (if it exists) of an LRDIMM. The value must come from Byte 143 of the SPD from the DIMM vendor.
<b>SPD Byte 144 - DB VrefDQ for DRAM Interface</b>	MEM_DDR4_SPD_144_DB_VREFDQ	Specifies the VrefDQ setting of the data buffer's DRAM interface. The value must come from Byte 144 of the SPD from the DIMM vendor.
<b>SPD Byte 145-147 - DB MDQ Drive Strength and RTT</b>	MEM_DDR4_SPD_145_DB_MDQ_DRV	Specifies the drive strength of the MDQ pins of the data buffer's DRAM interface. The value must come from either Byte 145 (data rate = 1866), 146 (1866 data rate = 2400), or 147 (2400 data rate = 3200) of the SPD from the DIMM vendor.
<b>SPD Byte 148 - DRAM Drive Strength</b>	MEM_DDR4_SPD_148_DRAM_DRV	Specifies the drive strength of the DRAM. The value must come from Byte 148 of the SPD from the DIMM vendor.
<b>SPD Byte 149-151 - DRAM ODT (RTT_WR and RTT_NOM)</b>	MEM_DDR4_SPD_149_DRAM_RTTRT_NOM	Specifies the RTT_WR and RTT_NOM setting of the DRAM. The value must come from either Byte 149 (data rate = 1866), 150 (1866 data rate = 2400), or 151 (2400 data rate = 3200) of the SPD from the DIMM vendor.
<b>SPD Byte 152-154 - DRAM ODT (RTT_PARK)</b>	MEM_DDR4_SPD_152_DRAM_RTTPARK	Specifies the RTT_PARK setting of the DRAM. The value must come from either Byte 152 (data rate = 1866), 153 (1866 data rate = 2400), or 154 (2400 data rate = 3200) of the SPD from the DIMM vendor.
<b>VrefDQ training range</b>	MEM_DDR4_VREFDQ_TRAINING_RANGE	VrefDQ training range.
<b>VrefDQ training value</b>	MEM_DDR4_VREFDQ_TRAINING_VALUE	VrefDQ training value.

**Table 126. Group: Mem I/O / ODT Activation**

Display Name	Identifier	Description
<b>Use Default ODT Assertion Tables</b>	MEM_DDR4_USE_DEFODT	Enables the default ODT assertion pattern as determined from vendor guidelines. These settings are provided as a default only; you should simulate your memory interface to determine the optimal ODT settings and assertion patterns.

#### 7.4.3.4. Arria 10 EMIF IP DDR4 Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 127. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_DDR4_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_DDR4_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 128. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_DDR4_USER_AC_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR4_USER_AC_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_DDR4_USER_AC_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 129. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_DDR4_USER_CK_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR4_USER_CK_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_DDR4_USER_CK_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.



**Table 130. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_DDR4_USER_AU_TO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_DDR4_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_DDR4_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR4_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_DDR4_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 131. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_DDR4_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_DDR4_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.4.3.5. Arria 10 EMIF IP DDR4 Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 132. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_DDR4_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>TdiVW_total</b>	MEM_DDR4_TDIVW_TOTAL_UI	TdiVW_total describes the minimum horizontal width of the DQ eye opening required by the receiver (memory device/ DIMM). It is measured in UI (1UI = half the memory clock period).

*continued...*



Display Name	Identifier	Description
<b>tDQSK</b>	MEM_DDR4_TDQSK_PS	tDQSK describes the skew between the memory clock (CK) and the input data strobes (DQS) used for reads. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDQSQ</b>	MEM_DDR4_TDQSQ_U_I	tDQSQ describes the latest valid transition of the associated DQ pins for a READ. tDQSQ specifically refers to the DQS, DQS# to DQ skew. It is the length of time between the DQS, DQS# crossing to the last valid transition of the slowest DQ pin in the DQ group associated with that DQS strobe.
<b>tDQSS</b>	MEM_DDR4_TDQSS_C_YC	tDQSS describes the skew between the memory clock (CK) and the output data strobes used for writes. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDSH</b>	MEM_DDR4_TDSH_CY_C	tDSH specifies the write DQS hold time. This is the time difference between the rising CK edge and the falling edge of DQS, measured as a percentage of tCK.
<b>tDSS</b>	MEM_DDR4_TDSS_CY_C	tDSS describes the time between the falling edge of DQS to the rising edge of the next CK transition.
<b>tIH (base) DC level</b>	MEM_DDR4_TIH_DC_MV	tIH (base) DC level refers to the voltage level which the address/command signal must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tIH (base)</b>	MEM_DDR4_TIH_PS	tIH (base) refers to the hold time for the Address/Command (A) bus after the rising edge of CK. Depending on what AC level the user has chosen for a design, the hold margin can vary (this variance will be automatically determined when the user chooses the "tIH (base) AC level").
<b>tINIT</b>	MEM_DDR4_TINIT_US	tINIT describes the time duration of the memory initialization after a device power-up. After RESET_n is de-asserted, wait for another 500us until CKE becomes active. During this time, the DRAM will start internal initialization; this will be done independently of external clocks.
<b>tIS (base) AC level</b>	MEM_DDR4_TIS_AC_MV	tIS (base) AC level refers to the voltage level which the address/command signal must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tIS (base)</b>	MEM_DDR4_TIS_PS	tIS (base) refers to the setup time for the Address/Command/Control (A) bus to the rising edge of CK.
<b>tMRD</b>	MEM_DDR4_TMRD_CK_CYC	The mode register set command cycle time, tMRD is the minimum time period required between two MRS commands.
<b>tQH</b>	MEM_DDR4_TQH_UI	tQH specifies the output hold time for the DQ in relation to DQS, DQS#. It is the length of time between the DQS, DQS# crossing to the earliest invalid transition of the fastest DQ pin in the DQ group associated with that DQS strobe.
<b>tQSH</b>	MEM_DDR4_TQSH_CY_C	tQSH refers to the differential High Pulse Width, which is measured as a percentage of tCK. It is the time during which the DQS is high for a read.

*continued...*



Display Name	Identifier	Description
<b>tRAS</b>	MEM_DDR4_TRAS_NS	tRAS describes the activate to precharge duration. A row cannot be deactivated until the tRAS time has been met. Therefore tRAS determines how long the memory has to wait after a activate command before a precharge command can be issued to close the row.
<b>tRCD</b>	MEM_DDR4_TRCD_NS	tRCD, row command delay, describes the amount of delay between the activation of a row through the RAS command and the access to the data through the CAS command.
<b>tRP</b>	MEM_DDR4_TRP_NS	tRP refers to the Precharge (PRE) command period. It describes how long it takes for the memory to disable access to a row by precharging and before it is ready to activate a different row.
<b>tWLH</b>	MEM_DDR4_TWHL_PS	tWLH describes the write leveling hold time from the rising edge of DQS to the rising edge of CK.
<b>tWLS</b>	MEM_DDR4_TWLS_PS	tWLS describes the write leveling setup time. It is measured from the rising edge of CK to the rising edge of DQS.
<b>tWR</b>	MEM_DDR4_TWR_NS	tWR refers to the Write Recovery time. It specifies the amount of clock cycles needed to complete a write before a precharge command can be issued.
<b>VdiVW_total</b>	MEM_DDR4_VDIVW_TOTAL	VdiVW_total describes the Rx Mask voltage, or the minimum vertical width of the DQ eye opening required by the receiver (memory device/DIMM). It is measured mV.

**Table 133. Group: Mem Timing / Parameters dependent on Speed Bin, Operating Frequency, and Page Size**

Display Name	Identifier	Description
<b>tCCD_L</b>	MEM_DDR4_TCCD_L_CYC	tCCD_L refers to the CAS_n-to-CAS_n delay (long). It is the minimum time interval between two read/write (CAS) commands to the same bank group.
<b>tCCD_S</b>	MEM_DDR4_TCCD_S_CYC	tCCD_S refers to the CAS_n-to-CAS_n delay (short). It is the minimum time interval between two read/write (CAS) commands to different bank groups.
<b>tFAW_dlr</b>	MEM_DDR4_TFAW_DL_R_CYC	tFAW_dlr refers to the four activate window to different logical ranks. It describes the period of time during which only four banks can be active across all logical ranks within a 3DS DDR4 device.
<b>tFAW</b>	MEM_DDR4_TFAW_NS	tFAW refers to the four activate window time. It describes the period of time during which only four banks can be active.
<b>tRRD_dlr</b>	MEM_DDR4_TRRD_DL_R_CYC	tRRD_dlr refers to the Activate to Activate Command Period to Different Logical Ranks. It is the minimum time interval (measured in memory clock cycles) between two activate commands to different logical ranks within a 3DS DDR4 device.
<b>tRRD_L</b>	MEM_DDR4_TRRD_L_CYC	tRRD_L refers to the Activate to Activate Command Period (long). It is the minimum time interval (measured in memory clock cycles) between two activate commands to the same bank group.
<b>tRRD_S</b>	MEM_DDR4_TRRD_S_CYC	tRRD_S refers to the Activate to Activate Command Period (short). It is the minimum time interval between two activate commands to the different bank groups.

*continued...*



Display Name	Identifier	Description
tRTP	MEM_DDR4_TRTP_CYC	tRTP refers to the internal READ Command to PRECHARGE Command delay. It is the number of memory clock cycles that is needed between a read command and a precharge command to the same rank.
tWTR_L	MEM_DDR4_TWTR_L_CYC	tWTR_L or Write Timing Parameter describes the delay from start of internal write transaction to internal read command, for accesses to the same bank group. The delay is measured from the first rising memory clock edge after the last write data is received to the rising memory clock edge when a read command is received.
tWTR_S	MEM_DDR4_TWTR_S_CYC	tWTR_S or Write Timing Parameter describes the delay from start of internal write transaction to internal read command, for accesses to the different bank group. The delay is measured from the first rising memory clock edge after the last write data is received to the rising memory clock edge when a read command is received.

**Table 134. Group: Mem Timing / Parameters dependent on Density and Temperature**

Display Name	Identifier	Description
tREFI	MEM_DDR4_TREFI_US	tREFI refers to the average periodic refresh interval. It is the maximum amount of time the memory can tolerate in between each refresh command
tRFC_dlr	MEM_DDR4_TRFC_DL_R_NS	tRFC_dlr refers to the Refresh Cycle Time to different logical rank. It is the amount of delay after a refresh command to one logical rank before an activate command can be accepted by another logical rank within a 3DS DDR4 device. This parameter is dependent on the memory density and is necessary for proper hardware functionality.
tRFC	MEM_DDR4_TRFC_NS	tRFC refers to the Refresh Cycle Time. It is the amount of delay after a refresh command before an activate command can be accepted by the memory. This parameter is dependent on the memory density and is necessary for proper hardware functionality.

#### 7.4.3.6. Arria 10 EMIF IP DDR4 Parameters: Board

**Table 135. Group: Board / Intersymbol Interference/Crosstalk**

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_DDR4_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total loss of margin on both the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQS/DQS# ISI/crosstalk</b>	BOARD_DDR4_USER_RCLK_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQS/DQS# signal when driven by the memory device during a read. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQ ISI/crosstalk</b>	BOARD_DDR4_USER_RDATA_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQ signal when driven by the memory device during a read. The number to be entered is

*continued...*



Display Name	Identifier	Description
		the total loss of margin on the setup and hold side (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQS/DQS# ISI/crosstalk</b>	BOARD_DDR4_USER_WCLK_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQS/DQS# signal when driven by the FPGA during a write. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_DDR4_USER_WDATA_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_DDR4_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.

**Table 136. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_DDR4_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Maximum board skew within address/command bus</b>	BOARD_DDR4_BRD_SKEW_WITHIN_AC_NS	The largest skew between the address and command signals.
<b>Maximum board skew within DQS group</b>	BOARD_DDR4_BRD_SKEW_WITHIN_DQS_NS	The largest skew between all DQ and DM pins in a DQS group. This value affects the read capture and write margins.
<b>Average delay difference between DQS and CK</b>	BOARD_DDR4_DQS_TO_CK_SKEW_NS	The average delay difference between the DQS signals and the CK signal, calculated by averaging the longest and smallest DQS trace delay minus the CK trace delay. Positive values represent DQS signals that are longer than CK signals and negative values represent DQS signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_DDR4_IS_SKEW_WITHIN_AC_DESKREWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (DQS group)</b>	BOARD_DDR4_IS_SKEW_WITHIN_DQS_DESKREWED	Enable this parameter if you are compensating for package skew on the DQ, DQS, and DM buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to DIMM/device</b>	BOARD_DDR4_MAX_CK_DELAY_NS	The delay of the longest CK trace from the FPGA to any DIMM/device.

*continued...*



Display Name	Identifier	Description
<b>Maximum DQS delay to DIMM/device</b>	BOARD_DDR4_MAX_DQS_DELAY_NS	The delay of the longest DQS trace from the FPGA to any DIMM/device
<b>Maximum delay difference between DIMMs/devices</b>	BOARD_DDR4_SKEW_BETWEEN_DIMMS_NS	The largest propagation delay on DQ signals between ranks (applicable only when there is more than one rank). For example: when you configure two ranks using one DIMM there is a short distance between the ranks for the same DQ pin; when you implement two ranks using two DIMMs the distance is larger.
<b>Maximum skew between DQS groups</b>	BOARD_DDR4_SKEW_BETWEEN_DQS_NS	The largest skew between DQS signals.

#### 7.4.3.7. Arria 10 EMIF IP DDR4 Parameters: Controller

**Table 137. Group: Controller / Low Power Mode**

Display Name	Identifier	Description
<b>Auto Power-Down Cycles</b>	CTRL_DDR4_AUTO_POWER_DOWN_CYCS	Specifies the number of idle controller cycles after which the memory device is placed into power-down mode. You can configure the idle waiting time. The supported range for number of cycles is from 1 to 65534.
<b>Enable Auto Power-Down</b>	CTRL_DDR4_AUTO_POWER_DOWN_EN	Enable this parameter to have the controller automatically place the memory device into power-down mode after a specified number of idle controller clock cycles. The idle wait time is configurable. All ranks must be idle to enter auto power-down.

**Table 138. Group: Controller / Efficiency**

Display Name	Identifier	Description
<b>Address Ordering</b>	CTRL_DDR4_ADDR_ORDER_ENUM	Controls the mapping between Avalon addresses and memory device addresses. By changing the value of this parameter, you can change the mappings between the Avalon-MM address and the DRAM address. (CS = chip select, CID = chip ID in 3DS/TSV devices, BG = bank group address, Bank = bank address, Row = row address, Col = column address)
<b>Enable Auto-Precharge Control</b>	CTRL_DDR4_AUTO_PRECHARGE_EN	Select this parameter to enable the auto-precharge control on the controller top level. If you assert the auto-precharge control signal while requesting a read or write burst, you can specify whether the controller should close (auto-precharge) the currently open page at the end of the read or write burst, potentially making a future access to a different page of the same bank faster.
<b>Enable Reordering</b>	CTRL_DDR4_REORDER_EN	Enable this parameter to allow the controller to perform command and data reordering. Reordering can improve efficiency by reducing bus turnaround time and row/bank switching time. Data reordering allows the single-port memory controller to change the order of read and write commands to achieve highest efficiency. Command reordering allows the controller to issue bank management commands early based on incoming patterns, so that the

*continued...*



Display Name	Identifier	Description
		desired row in memory is already open when the command reaches the memory interface. For more information, refer to the Data Reordering topic in the EMIF Handbook.
<b>Starvation limit for each command</b>	CTRL_DDR4_STARVE_LIMIT	Specifies the number of commands that can be served before a waiting command is served. The controller employs a counter to ensure that all requests are served after a pre-defined interval -- this ensures that low priority requests are not ignored, when doing data reordering for efficiency. The valid range for this parameter is from 1 to 63. For more information, refer to the Starvation Control topic in the EMIF Handbook.
<b>Enable Command Priority Control</b>	CTRL_DDR4_USER_PRIORITY_EN	Select this parameter to enable user-requested command priority control on the controller top level. This parameter instructs the controller to treat a read or write request as high-priority. The controller attempts to fill high-priority requests sooner, to reduce latency. Connect this interface to the conduit of your logic block that determines when the external memory interface IP treats the read or write request as a high-priority command.

**Table 139. Group: Controller / Configuration, Status, and Error Handling**

Display Name	Identifier	Description
<b>Enable Auto Error Correction</b>	CTRL_DDR4_ECC_AUTO_CORRECTION_EN	Specifies that the controller perform auto correction when a single-bit error is detected by the ECC logic.
<b>Enable Error Detection and Correction Logic with ECC</b>	CTRL_DDR4_ECC_EN	Enables error-correction code (ECC) for single-bit error correction and double-bit error detection. Your memory interface must have a width of 16, 24, 40, or 72 bits to use ECC. ECC is implemented as soft logic.
<b>Enable Memory-Mapped Configuration and Status Register (MMR) Interface</b>	CTRL_DDR4_MMR_EN	Enable this parameter to change or read memory timing parameters, memory address size, mode register settings, controller status, and request sideband operations.

**Table 140. Group: Controller / Data Bus Turnaround Time**

Display Name	Identifier	Description
<b>Additional read-to-read turnaround time (different ranks)</b>	CTRL_DDR4_RD_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a read of another logical rank. This can resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (different ranks)</b>	CTRL_DDR4_RD_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (same rank)</b>	CTRL_DDR4_RD_TO_WR_SAME_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read to a write within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.

*continued...*



Display Name	Identifier	Description
<b>Additional write-to-read turnaround time (different ranks)</b>	CTRL_DDR4_WR_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a read of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (same rank)</b>	CTRL_DDR4_WR_TO_RD SAME_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write to a read within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-write turnaround time (different ranks)</b>	CTRL_DDR4_WR_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.

#### 7.4.3.8. Arria 10 EMIF IP DDR4 Parameters: Diagnostics

**Table 141. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_DDR4_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 142. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Skip address/command deskew calibration</b>	DIAG_DDR4_SKIP_CA_DESKEW	Specifies to skip the address/command deskew calibration stage. Address/command deskew performs per-bit deskew for the address and command pins.
<b>Skip address/command leveling calibration</b>	DIAG_DDR4_SKIP_CA_LEVEL	Specifies to skip the address/command leveling stage during calibration. Address/command leveling attempts to center the memory clock edge against CS# by adjusting delay elements inside the PHY, and then applying the same delay offset to the rest of the address and command pins.

*continued...*



Display Name	Identifier	Description
<b>Skip VREF calibration</b>	DIAG_DDR4_SKIP_VR EF_CAL	Specifies to skip the VREF stage of calibration. Enable this parameter for debug purposes only; generally, you should include the VREF calibration stage during normal operation.
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_A VALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_A VALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface", an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLite cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_M ODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.

**Table 143. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX_DESIGN_IS SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX_DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 144. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 145. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 146. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

#### 7.4.3.9. Arria 10 EMIF IP DDR4 Parameters: Example Designs



**Table 147. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_DDR_4_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 148. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_DDR_4_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_DDR_4_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 149. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_DDR_4_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.

**Table 150. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_DDR_4_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.



#### 7.4.3.10. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPOLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

#### 7.4.3.11. x4 Mode for Arria 10 External Memory Interface

Non-HPS Arria 10 external memory interfaces support DQ pins-per-DQS group-of-4 (x4 mode) for DDR3 and DDR4 memory protocols.

The following restrictions apply to the use of x4 mode:

- The total interface width is limited to 72 bits.
- You must disable the **Enable DM pins** option.
- For DDR4, you must disable the **DBI** option.

*Note:* x4 mode is not available for Arria 10 EMIF IP for HPS.

#### 7.4.3.12. Additional Notes About Parameterizing Arria 10 EMIF IP for HPS

Although Arria 10 EMIF IP and Arria 10 EMIF IP for HPS are similar components, there are some additional requirements necessary in the HPS case.

The following rules and restrictions apply to Arria 10 EMIF IP for HPS:

- Supported memory protocols are limited to DDR3 and DDR4.
- The only supported configuration is the hard PHY with the hard memory controller.
- The maximum memory clock frequency for Arria 10 EMIF IP for HPS may be different than for regular Arria 10 EMIF IP. Refer to the External Memory Interface Spec Estimator for details.
- Only half-rate interfaces are supported.
- Sharing of clocks is not supported.



- The total interface width is limited to a multiple of 16, 24, 40 or 72 bits (with ECC enabled), or a positive value divisible by the number of DQ pins per DQS group (with ECC not enabled). For devices other than 10ASXXXXX40, the total interface width is further limited to a maximum of 40 bits with ECC enabled and 32 bits with ECC not enabled.
- Only x8 data groups are supported; that is, DQ pins-per-DQS group must be 8.
- DM pins must be enabled.
- The EMIF debug toolkit is not supported.
- Ping Pong PHY is not supported.

- The interface to and from the HPS is a fixed-width conduit.
- A maximum of 3 address/command I/O lanes are supported. For example:
  - DDR3
    - For component format, maximum number of chip selects is 2.
    - For UDIMM or SODIMM format:
      - Maximum number of DIMMs is 2, when the number of physical ranks per DIMM is 1.
      - Maximum number of DIMMs is 1, when the number of physical ranks per DIMM is 2.
      - Maximum number of physical ranks per DIMMs is 2, when the number of DIMMs is 1.
    - For RDIMM format:
      - Maximum number of clocks is 1.
      - Maximum number of DIMMs is 1.
      - Maximum number of physical ranks per DIMM is 2.
    - LRDIMM memory format is not supported.
  - DDR4
    - For component format:
      - Maximum number of clocks is 1.
      - Maximum number of chip selects is 2
    - For UDIMM or RDIMM format:
      - Maximum number of clocks is 1.
      - Maximum number of DIMMs is 2, when the number of physical ranks per DIMM is 1.
      - Maximum number of DIMMs is 1, when the number of physical ranks per DIMM is 2.
      - Maximum number of physical ranks per DIMM is 2, when the number of DIMMs is 1.
    - For SODIMM format:
      - Maximum number of clocks is 1.
      - Maximum number of DIMMs is 1.
      - Maximum number of physical ranks per DIMM is 1.
  - Arria 10 EMIF IP for HPS also has specific pin-out requirements. For information, refer to *Planning Pin and FPGA Resources*.

#### 7.4.4. Arria 10 EMIF IP DDR3 Parameters

The Arria 10 EMIF IP parameter editor allows you to parameterize settings for the Arria 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.



**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Arria 10 EMIF IP.

#### 7.4.4.1. Arria 10 EMIF IP DDR3 Parameters: General

**Table 151. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 152. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 153. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the

*continued...*



Display Name	Identifier	Description
		same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Use recommended PLL reference clock frequency</b>	PHY_DDR3_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 154. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

#### 7.4.4.2. Arria 10 EMIF IP DDR3 Parameters: Memory



**Table 155. Group: Memory / Topology**

Display Name	Identifier	Description
<b>DQS group of ALERT#</b>	MEM_DDR3_ALERT_N_DQS_GROUP	Select the DQS group with which the ALERT# pin is placed.
<b>ALERT# pin placement</b>	MEM_DDR3_ALERT_N_PLACEMENT_ENUM	Specifies placement for the mem_alert_n signal. If you select "I/O Lane with Address/Command Pins", you can pick the I/O lane and pin index in the add/cmd bank with the subsequent drop down menus. If you select "I/O Lane with DQS Group", you can specify the DQS group with which to place the mem_alert_n pin. If you select "Automatically select a location", the IP automatically selects a pin for the mem_alert_n signal. If you select this option, no additional location constraints can be applied to the mem_alert_n pin, or a fitter error will result during compilation. For optimum signal integrity, you should choose "I/O Lane with Address/Command Pins". For interfaces containing multiple memory devices, it is recommended to connect the ALERT# pin on the FPGA.
<b>Bank address width</b>	MEM_DDR3_BANK_ADDRESS_WIDTH	Specifies the number of bank address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of bank address pins needed for access to all available banks.
<b>Number of clocks</b>	MEM_DDR3_CK_WIDTH	Specifies the number of CK/CK# clock pairs exposed by the memory interface. Usually more than 1 pair is required for RDIMM/LRDIMM formats. The value of this parameter depends on the memory device selected; refer to the data sheet for your memory device.
<b>Column address width</b>	MEM_DDR3_COL_ADDRESS_WIDTH	Specifies the number of column address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of address pins needed for access to all available columns.
<b>Number of chip selects per DIMM</b>	MEM_DDR3_CS_PER_DIMM	Specifies the number of chip selects per DIMM.
<b>Number of chip selects</b>	MEM_DDR3_DISCRETE_CS_WIDTH	Specifies the total number of chip selects in the interface, up to a maximum of 4. This parameter applies to discreet components only.
<b>Enable DM pins</b>	MEM_DDR3_DM_EN	Indicates whether the interface uses data mask (DM) pins. This feature allows specified portions of the data bus to be written to memory (not available in x4 mode). One DM pin exists per DQS group
<b>Number of DQS groups</b>	MEM_DDR3_DQS_WIDTH	Specifies the total number of DQS groups in the interface. This value is automatically calculated as the DQ width divided by the number of DQ pins per DQS group.
<b>DQ pins per DQS group</b>	MEM_DDR3_DQ_PER_DQS	Specifies the total number of DQ pins per DQS group.
<b>DQ width</b>	MEM_DDR3_DQ_WIDTH	Specifies the total number of data pins in the interface. The maximum supported width is 144, or 72 in Ping Pong PHY mode.
<b>Memory format</b>	MEM_DDR3_FORMAT_ENUM	Specifies the format of the external memory device. The following formats are supported: Component - a Discrete memory device; UDIMM - Unregistered/Unbuffered DIMM where address/control, clock, and data are unbuffered; RDIMM - Registered DIMM where address/control and clock are buffered; LRDIMM - Load Reduction DIMM where address/control, clock, and data are buffered. LRDIMM reduces the load to increase memory speed and supports higher densities than RDIMM; SODIMM - Small Outline

*continued...*



Display Name	Identifier	Description
		DIMM is similar to UDIMM but smaller in size and is typically used for systems with limited space. Some memory protocols may not be available in all formats.
<b>Number of DIMMs</b>	MEM_DDR3_NUM_OF_DIMMS	Total number of DIMMs.
<b>Number of physical ranks per DIMM</b>	MEM_DDR3_RANKS_PER_DIMM	Number of ranks per DIMM. For LRDIMM, this represents the number of physical ranks on the DIMM behind the memory buffer
<b>Number of rank multiplication pins</b>	MEM_DDR3_RM_WIDTH	Number of rank multiplication pins used to access all physical ranks on an LRDIMM. Rank multiplication is a ratio between the number of physical ranks for an LRDIMM and the number of logical ranks for the controller. These pins should be connected to CS#[2] and/or CS#[3] of all LRDIMMs in the system
<b>Row address width</b>	MEM_DDR3_ROW_ADDRESS_WIDTH	Specifies the number of row address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of address pins needed for access to all available rows.

**Table 156. Group: Memory / Latency and Burst**

Display Name	Identifier	Description
<b>Memory additive CAS latency setting</b>	MEM_DDR3_ATCL_ENUM	Determines the posted CAS additive latency of the memory device. Enable this feature to improve command and bus efficiency, and increase system bandwidth.
<b>Burst Length</b>	MEM_DDR3_BL_ENUM	Specifies the DRAM burst length which determines how many consecutive addresses should be accessed for a given read/write command.
<b>Read Burst Type</b>	MEM_DDR3_BT_ENUM	Indicates whether accesses within a given burst are in sequential or interleaved order. Select sequential if you are using the Intel-provided memory controller.
<b>Memory CAS latency setting</b>	MEM_DDR3_TCL	Specifies the number of clock cycles between the read command and the availability of the first bit of output data at the memory device. Overall read latency equals the additive latency (AL) + the CAS latency (CL). Overall read latency depends on the memory device selected; refer to the datasheet for your device.
<b>Memory write CAS latency setting</b>	MEM_DDR3_WTCL	Specifies the number of clock cycles from the release of internal write to the latching of the first data in at the memory device. This value depends on the memory device selected; refer to the datasheet for your device.

**Table 157. Group: Memory / Mode Register Settings**

Display Name	Identifier	Description
<b>Auto self-refresh method</b>	MEM_DDR3_ASR_ENUM	Indicates whether to enable or disable auto self-refresh. Auto self-refresh allows the controller to issue self-refresh requests, rather than manually issuing self-refresh in order for memory to retain data.
<b>DDR3 LRDIMM additional control words</b>	MEM_DDR3_LRDIMM_EXTENDED_CONFIG	Each 4-bit setting can be obtained from the manufacturer's data sheet and should be entered in hexadecimal, starting with BC0F on the left and ending with BC00 on the right

*continued...*



Display Name	Identifier	Description
<b>DLL precharge power down</b>	MEM_DDR3_PD_ENUM	Specifies whether the DLL in the memory device is off or on during precharge power-down
<b>DDR3 RDIMM/LRDIMM control words</b>	MEM_DDR3_RDIMM_CONFIG	Each 4-bit/8-bit setting can be obtained from the manufacturer's data sheet and should be entered in hexadecimal, starting with the 8-bit setting RCBx on the left and continuing to RC1x followed by the 4-bit setting RCOF and ending with RC00 on the right
<b>Self-refresh temperature</b>	MEM_DDR3_SRT_ENUM	Specifies the self-refresh temperature as "Normal" or "Extended" mode. More information on Normal and Extended temperature modes can be found in the memory device datasheet.

#### 7.4.4.3. Arria 10 EMIF IP DDR3 Parameters: Mem I/O

**Table 158. Group: Mem I/O / Memory I/O Settings**

Display Name	Identifier	Description
<b>Output drive strength setting</b>	MEM_DDR3_DRV_STR_ENUM	Specifies the output driver impedance setting at the memory device. To obtain optimum signal integrity performance, select option based on board simulation results.
<b>ODT Rtt nominal value</b>	MEM_DDR3_RTT_NOM_ENUM	Determines the nominal on-die termination value applied to the DRAM. The termination is applied any time that ODT is asserted. If you specify a different value for RTT_WR, that value takes precedence over the values mentioned here. For optimum signal integrity performance, select your option based on board simulation results.
<b>Dynamic ODT (Rtt_WR) value</b>	MEM_DDR3_RTT_WR_ENUM	Specifies the mode of the dynamic on-die termination (ODT) during writes to the memory device (used for multi-rank configurations). For optimum signal integrity performance, select this option based on board simulation results.

**Table 159. Group: Mem I/O / ODT Activation**

Display Name	Identifier	Description
<b>Use Default ODT Assertion Tables</b>	MEM_DDR3_USE_DEFAULT_ODT	Enables the default ODT assertion pattern as determined from vendor guidelines. These settings are provided as a default only; you should simulate your memory interface to determine the optimal ODT settings and assertion patterns.

#### 7.4.4.4. Arria 10 EMIF IP DDR3 Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 160. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_DDR3_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal

*continued...*



Display Name	Identifier	Description
		integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_DDR3_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 161. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_DDR3_USER_AC_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR3_USER_AC_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_DDR3_USER_AC_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 162. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_DDR3_USER_CK_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR3_USER_CK_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_DDR3_USER_CK_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.



**Table 163. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_DDR3_USER_AU_TO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_DDR3_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_DDR3_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR3_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_DDR3_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 164. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_DDR3_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_DDR3_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.4.4.5. Arria 10 EMIF IP DDR3 Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 165. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_DDR3_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tDH (base) DC level</b>	MEM_DDR3_TDH_DC_MV	tDH (base) DC level refers to the voltage level which the data bus must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tDH (base)</b>	MEM_DDR3_TDH_PS	tDH (base) refers to the hold time for the Data (DQ) bus after the rising edge of CK.

*continued...*



Display Name	Identifier	Description
<b>tDQSK</b>	MEM_DDR3_TDQSK_PS	tDQSK describes the skew between the memory clock (CK) and the input data strobes (DQS) used for reads. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDQSQ</b>	MEM_DDR3_TDQSQ_PS	tDQSQ describes the latest valid transition of the associated DQ pins for a READ. tDQSQ specifically refers to the DQS, DQS# to DQ skew. It is the length of time between the DQS, DQS# crossing to the last valid transition of the slowest DQ pin in the DQ group associated with that DQS strobe.
<b>tDQSS</b>	MEM_DDR3_TDQSS_CYC	tDQSS describes the skew between the memory clock (CK) and the output data strobes used for writes. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDSH</b>	MEM_DDR3_TDSH_CYC	tDSH specifies the write DQS hold time. This is the time difference between the rising CK edge and the falling edge of DQS, measured as a percentage of tCK.
<b>tDSS</b>	MEM_DDR3_TDSS_CYC	tDSS describes the time between the falling edge of DQS to the rising edge of the next CK transition.
<b>tDS (base) AC level</b>	MEM_DDR3_TDS_AC_MV	tDS (base) AC level refers to the voltage level which the data bus must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tDS (base)</b>	MEM_DDR3_TDS_PS	tDS(base) refers to the setup time for the Data (DQ) bus before the rising edge of the DQS strobe.
<b>tIH (base) DC level</b>	MEM_DDR3_TIH_DC_MV	tIH (base) DC level refers to the voltage level which the address/command signal must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tIH (base)</b>	MEM_DDR3_TIH_PS	tIH (base) refers to the hold time for the Address/Command (A) bus after the rising edge of CK. Depending on what AC level the user has chosen for a design, the hold margin can vary (this variance will be automatically determined when the user chooses the "tIH (base) AC level").
<b>tINIT</b>	MEM_DDR3_TINIT_US	tINIT describes the time duration of the memory initialization after a device power-up. After RESET_n is de-asserted, wait for another 500us until CKE becomes active. During this time, the DRAM starts internal initialization; this happens independently of external clocks.
<b>tIS (base) AC level</b>	MEM_DDR3_TIS_AC_MV	tIS (base) AC level refers to the voltage level which the address/command signal must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tIS (base)</b>	MEM_DDR3_TIS_PS	tIS (base) refers to the setup time for the Address/Command/Control (A) bus to the rising edge of CK.
<b>tMRD</b>	MEM_DDR3_TMRD_CK_CYC	The mode register set command cycle time, tMRD is the minimum time period required between two MRS commands.

*continued...*



Display Name	Identifier	Description
<b>tQH</b>	MEM_DDR3_TQH_CYC	tQH specifies the output hold time for the DQ in relation to DQS, DQS#. It is the length of time between the DQS, DQS# crossing to the earliest invalid transition of the fastest DQ pin in the DQ group associated with that DQS strobe.
<b>tQSH</b>	MEM_DDR3_TQSH_CYC	tQSH refers to the differential High Pulse Width, which is measured as a percentage of tCK. It is the time during which the DQS is high for a read.
<b>tRAS</b>	MEM_DDR3_TRAS_NS	tRAS describes the activate to precharge duration. A row cannot be deactivated until the tRAS time has been met. Therefore tRAS determines how long the memory has to wait after a activate command before a precharge command can be issued to close the row.
<b>tRCD</b>	MEM_DDR3_TRCD_NS	tRCD, row command delay, describes the amount of delay between the activation of a row through the RAS command and the access to the data through the CAS command.
<b>tRP</b>	MEM_DDR3_TRP_NS	tRP refers to the Precharge (PRE) command period. It describes how long it takes for the memory to disable access to a row by precharging and before it is ready to activate a different row.
<b>tWLH</b>	MEM_DDR3_TWHL_PS	tWLH describes the write leveling hold time from the rising edge of DQS to the rising edge of CK.
<b>tWLS</b>	MEM_DDR3_TWLS_PS	tWLS describes the write leveling setup time. It is measured from the rising edge of CK to the rising edge of DQS.
<b>tWR</b>	MEM_DDR3_TWR_NS	tWR refers to the Write Recovery time. It specifies the amount of clock cycles needed to complete a write before a precharge command can be issued.

**Table 166. Group: Mem Timing / Parameters dependent on Speed Bin, Operating Frequency, and Page Size**

Display Name	Identifier	Description
<b>tFAW</b>	MEM_DDR3_TFAW_NS	tFAW refers to the four activate window time. It describes the period of time during which only four banks can be active.
<b>tRRD</b>	MEM_DDR3_TRRD_CYC	tRRD refers to the Row Active to Row Active Delay. It is the minimum time interval (measured in memory clock cycles) between two activate commands to rows in different banks in the same rank
<b>tRTP</b>	MEM_DDR3_TRTP_CYC	tRTP refers to the internal READ Command to PRECHARGE Command delay. It is the number of memory clock cycles that is needed between a read command and a precharge command to the same rank.
<b>tWTR</b>	MEM_DDR3_TWTR_CYC	tWTR or Write Timing Parameter describes the delay from start of internal write transaction to internal read command, for accesses to the same bank. The delay is measured from the first rising memory clock edge after the last write data is received to the rising memory clock edge when a read command is received.

**Table 167. Group: Mem Timing / Parameters dependent on Density and Temperature**

Display Name	Identifier	Description
tREFI	MEM_DDR3_TREFI_US	tREFI refers to the average periodic refresh interval. It is the maximum amount of time the memory can tolerate in between each refresh command
tRFC	MEM_DDR3_TRFC_NS	tRFC refers to the Refresh Cycle Time. It is the amount of delay after a refresh command before an activate command can be accepted by the memory. This parameter is dependent on the memory density and is necessary for proper hardware functionality.

#### 7.4.4.6. Arria 10 EMIF IP DDR3 Parameters: Board

**Table 168. Group: Board / Intersymbol Interference/Crosstalk**

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_DDR3_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total loss of margin on both the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQS/DQS# ISI/crosstalk</b>	BOARD_DDR3_USER_RCLK_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQS/DQS# signal when driven by the memory device during a read. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQ ISI/crosstalk</b>	BOARD_DDR3_USER_RDATA_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQ signal when driven by the memory device during a read. The number to be entered is the total loss of margin on the setup and hold side (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQS/DQS# ISI/crosstalk</b>	BOARD_DDR3_USER_WCLK_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQS/DQS# signal when driven by the FPGA during a write. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_DDR3_USER_WDATA_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_DDR3_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.



**Table 169. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_DDR3_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Maximum board skew within DQS group</b>	BOARD_DDR3_BRD_SKew_WITHIN_DQS_NS	The largest skew between all DQ and DM pins in a DQS group. This value affects the read capture and write margins.
<b>Average delay difference between DQS and CK</b>	BOARD_DDR3_DQS_TO_CK_SKEW_NS	The average delay difference between the DQS signals and the CK signal, calculated by averaging the longest and smallest DQS trace delay minus the CK trace delay. Positive values represent DQS signals that are longer than CK signals and negative values represent DQS signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_DDR3_IS_SKew_WITHIN_AC_DESKewed	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (DQS group)</b>	BOARD_DDR3_IS_SKew_WITHIN_DQS_DESKewed	Enable this parameter if you are compensating for package skew on the DQ, DQS, and DM buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to DIMM/device</b>	BOARD_DDR3_MAX_CK_DELAY_NS	The delay of the longest CK trace from the FPGA to any DIMM/device.
<b>Maximum DQS delay to DIMM/device</b>	BOARD_DDR3_MAX_DQS_DELAY_NS	The delay of the longest DQS trace from the FPGA to any DIMM/device
<b>Maximum system skew within address/command bus</b>	BOARD_DDR3_PKG_BRD_SKew_WITHIN_AC_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.
<b>Maximum delay difference between DIMMs/devices</b>	BOARD_DDR3_SKew_BETWEEN_DIMMS_NS	The largest propagation delay on DQ signals between ranks (applicable only when there is more than one rank). For example: when you configure two ranks using one DIMM there is a short distance between the ranks for the same DQ pin; when you implement two ranks using two DIMMs the distance is larger.
<b>Maximum skew between DQS groups</b>	BOARD_DDR3_SKew_BETWEEN_DQS_NS	The largest skew between DQS signals.

#### 7.4.4.7. Arria 10 EMIF IP DDR3 Parameters: Controller

**Table 170. Group: Controller / Low Power Mode**

Display Name	Identifier	Description
<b>Auto Power-Down Cycles</b>	CTRL_DDR3_AUTO_POWER_DOWN_CYCS	Specifies the number of idle controller cycles after which the memory device is placed into power-down mode. You can configure the idle waiting time. The supported range for number of cycles is from 1 to 65534.
<b>Enable Auto Power-Down</b>	CTRL_DDR3_AUTO_POWER_DOWN_EN	Enable this parameter to have the controller automatically place the memory device into power-down mode after a specified number of idle controller clock cycles. The idle wait time is configurable. All ranks must be idle to enter auto power-down.

**Table 171. Group: Controller / Efficiency**

Display Name	Identifier	Description
<b>Address Ordering</b>	CTRL_DDR3_ADDR_ORDER_ENUM	Controls the mapping between Avalon addresses and memory device addresses. By changing the value of this parameter, you can change the mappings between the Avalon-MM address and the DRAM address.
<b>Enable Auto-Precharge Control</b>	CTRL_DDR3_AUTO_PRECHARGE_EN	Select this parameter to enable the auto-precharge control on the controller top level. If you assert the auto-precharge control signal while requesting a read or write burst, you can specify whether the controller should close (auto-precharge) the currently open page at the end of the read or write burst, potentially making a future access to a different page of the same bank faster.
<b>Enable Reordering</b>	CTRL_DDR3_REORDER_EN	Enable this parameter to allow the controller to perform command and data reordering. Reordering can improve efficiency by reducing bus turnaround time and row/bank switching time. Data reordering allows the single-port memory controller to change the order of read and write commands to achieve highest efficiency. Command reordering allows the controller to issue bank management commands early based on incoming patterns, so that the desired row in memory is already open when the command reaches the memory interface. For more information, refer to the Data Reordering topic in the EMIF Handbook.
<b>Starvation limit for each command</b>	CTRL_DDR3_STARVE_LIMIT	Specifies the number of commands that can be served before a waiting command is served. The controller employs a counter to ensure that all requests are served after a pre-defined interval -- this ensures that low priority requests are not ignored, when doing data reordering for efficiency. The valid range for this parameter is from 1 to 63. For more information, refer to the Starvation Control topic in the EMIF Handbook.
<b>Enable Command Priority Control</b>	CTRL_DDR3_USER_PRIORITY_EN	Select this parameter to enable user-requested command priority control on the controller top level. This parameter instructs the controller to treat a read or write request as high-priority. The controller attempts to fill high-priority requests sooner, to reduce latency. Connect this interface to the conduit of your logic block that determines when the external memory interface IP treats the read or write request as a high-priority command.



**Table 172. Group: Controller / Configuration, Status, and Error Handling**

Display Name	Identifier	Description
<b>Enable Auto Error Correction</b>	CTRL_DDR3_ECC_AUTO_CORRECTION_EN	Specifies that the controller perform auto correction when a single-bit error is detected by the ECC logic.
<b>Enable Error Detection and Correction Logic with ECC</b>	CTRL_DDR3_ECC_EN	Enables error-correction code (ECC) for single-bit error correction and double-bit error detection. Your memory interface must have a width of 16, 24, 40, or 72 bits to use ECC. ECC is implemented as soft logic.
<b>Enable Memory-Mapped Configuration and Status Register (MMR) Interface</b>	CTRL_DDR3_MMR_EN	Enable this parameter to change or read memory timing parameters, memory address size, mode register settings, controller status, and request sideband operations.

**Table 173. Group: Controller / Data Bus Turnaround Time**

Display Name	Identifier	Description
<b>Additional read-to-read turnaround time (different ranks)</b>	CTRL_DDR3_RD_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a read of another logical rank. This can resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (different ranks)</b>	CTRL_DDR3_RD_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (same rank)</b>	CTRL_DDR3_RD_TO_WR_SAME_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read to a write within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (different ranks)</b>	CTRL_DDR3_WR_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a read of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (same rank)</b>	CTRL_DDR3_WR_TO_RD_SAME_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write to a read within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-write turnaround time (different ranks)</b>	CTRL_DDR3_WR_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.



#### 7.4.4.8. Arria 10 EMIF IP DDR3 Parameters: Diagnostics

**Table 174. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_DDR3_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 175. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface," an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLite cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.



**Table 176. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX_DESIGN_IS_SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX_DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 177. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 178. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 179. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

#### 7.4.4.9. Arria 10 EMIF IP DDR3 Parameters: Example Designs

**Table 180. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_DDR3_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 181. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_DDR3_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_DDR3_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 182. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_DDR3_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.



**Table 183. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_DDR3_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

#### 7.4.4.10. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

#### 7.4.4.11. x4 Mode for Arria 10 External Memory Interface

Non-HPS Arria 10 external memory interfaces support DQ pins-per-DQS group-of-4 (x4 mode) for DDR3 and DDR4 memory protocols.

The following restrictions apply to the use of x4 mode:

- The total interface width is limited to 72 bits.
- You must disable the **Enable DM pins** option.
- For DDR4, you must disable the **DBI** option.



*Note:* x4 mode is not available for Arria 10 EMIF IP for HPS.

#### 7.4.4.12. Additional Notes About Parameterizing Arria 10 EMIF IP for HPS

Although Arria 10 EMIF IP and Arria 10 EMIF IP for HPS are similar components, there are some additional requirements necessary in the HPS case.

The following rules and restrictions apply to Arria 10 EMIF IP for HPS:

- Supported memory protocols are limited to DDR3 and DDR4.
- The only supported configuration is the hard PHY with the hard memory controller.
- The maximum memory clock frequency for Arria 10 EMIF IP for HPS may be different than for regular Arria 10 EMIF IP. Refer to the External Memory Interface Spec Estimator for details.
- Only half-rate interfaces are supported.
- Sharing of clocks is not supported.
- The total interface width is limited to a multiple of 16, 24, 40 or 72 bits (with ECC enabled), or a positive value divisible by the number of DQ pins per DQS group (with ECC not enabled). For devices other than 10ASXXXXX40, the total interface width is further limited to a maximum of 40 bits with ECC enabled and 32 bits with ECC not enabled.
- Only x8 data groups are supported; that is, DQ pins-per-DQS group must be 8.
- DM pins must be enabled.
- The EMIF debug toolkit is not supported.
- Ping Pong PHY is not supported.



- The interface to and from the HPS is a fixed-width conduit.
- A maximum of 3 address/command I/O lanes are supported. For example:
  - DDR3
    - For component format, maximum number of chip selects is 2.
    - For UDIMM or SODIMM format:
      - Maximum number of DIMMs is 2, when the number of physical ranks per DIMM is 1.
      - Maximum number of DIMMs is 1, when the number of physical ranks per DIMM is 2.
      - Maximum number of physical ranks per DIMMs is 2, when the number of DIMMs is 1.
    - For RDIMM format:
      - Maximum number of clocks is 1.
      - Maximum number of DIMMs is 1.
      - Maximum number of physical ranks per DIMM is 2.
    - LRDIMM memory format is not supported.
  - DDR4
    - For component format:
      - Maximum number of clocks is 1.
      - Maximum number of chip selects is 2
    - For UDIMM or RDIMM format:
      - Maximum number of clocks is 1.
      - Maximum number of DIMMs is 2, when the number of physical ranks per DIMM is 1.
      - Maximum number of DIMMs is 1, when the number of physical ranks per DIMM is 2.
      - Maximum number of physical ranks per DIMM is 2, when the number of DIMMs is 1.
    - For SODIMM format:
      - Maximum number of clocks is 1.
      - Maximum number of DIMMs is 1.
      - Maximum number of physical ranks per DIMM is 1.
  - Arria 10 EMIF IP for HPS also has specific pin-out requirements. For information, refer to *Planning Pin and FPGA Resources*.

#### 7.4.5. Arria 10 EMIF IP LPDDR3 Parameters

The Arria 10 EMIF IP parameter editor allows you to parameterize settings for the Arria 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.



**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Arria 10 EMIF IP.

#### 7.4.5.1. Arria 10 EMIF IP LPDDR3 Parameters: General

**Table 184. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 185. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 186. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the

*continued...*



Display Name	Identifier	Description
		same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Use recommended PLL reference clock frequency</b>	PHY_LPDDR3_DEFUALT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 187. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

#### 7.4.5.2. Arria 10 EMIF IP LPDDR3 Parameters: Memory

**Table 188. Group: Memory / Topology**

Display Name	Identifier	Description
<b>Bank address width</b>	MEM_LPDDR3_BANK_ADDR_WIDTH	The number of bank address bits.
<b>Number of clocks</b>	MEM_LPDDR3_CK_WIDTH	Number of CK/CK# clock pairs exposed by the memory interface.
<b>Column address width</b>	MEM_LPDDR3_COL_A_DDR_WIDTH	The number of column address bits.
<b>Number of chip selects</b>	MEM_LPDDR3_DISCRE_CS_WIDTH	Total number of chip selects in the interface.
<b>Enable DM pins</b>	MEM_LPDDR3_DM_EN	Indicates whether interface uses data mask (DM) pins. This feature allows specified portions of the data bus to be written to memory (not available in x4 mode). One DM pin exists per DQS group
<b>Number of DQS groups</b>	MEM_LPDDR3_DQS_WIDTH	Specifies the total number of DQS groups in the interface. This value is automatically calculated as the DQ width divided by the number of DQ pins per DQS group.
<b>DQ width</b>	MEM_LPDDR3_DQ_WIDHT	Total number of DQ pins in the interface.
<b>Row address width</b>	MEM_LPDDR3_ROW_A_DDR_WIDTH	The number of row address bits.

**Table 189. Group: Memory / Latency and Burst**

Display Name	Identifier	Description
<b>Burst length</b>	MEM_LPDDR3_BL	Burst length of the memory device.
<b>Data latency</b>	MEM_LPDDR3_DATA_LATENCY	Determines the mode register setting that controls the data latency. Sets both READ and WRITE latency (RL and WL).
<b>DQ ODT</b>	MEM_LPDDR3_DQODT	The ODT setting for the DQ pins during writes.
<b>Power down ODT</b>	MEM_LPDDR3_PDODT	Turn on turn off ODT during power down.
<b>WL set</b>	MEM_LPDDR3_WLSELECT	The set of the currently selected write latency. Only certain memory devices support WL Set B. Refer to the WRITE Latency table in the memory vendor data sheet.

#### 7.4.5.3. Arria 10 EMIF IP LPDDR3 Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 190. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_LPDDR3_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal

*continued...*



Display Name	Identifier	Description
		integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_LPDDR3_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 191. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_LPDDR3_USER_A_C_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_LPDDR3_USER_A_C_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_LPDDR3_USER_A_C_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 192. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_LPDDR3_USER_C_K_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_LPDDR3_USER_C_K_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_LPDDR3_USER_C_K_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 193. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_LPDDR3_USER_AUTO_STARTING_VREFIN_IN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_LPDDR3_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_LPDDR3_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_LPDDR3_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_LPDDR3_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 194. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_LPDDR3_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_LPDDR3_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.4.5.4. Arria 10 EMIF IP LPDDR3 Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 195. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_LPDDR3_SPEED_BIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tDH (base) DC level</b>	MEM_LPDDR3_TDH_DC_MV	tDH (base) DC level refers to the voltage level which the data bus must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tDH (base)</b>	MEM_LPDDR3_TDH_PS	tDH (base) refers to the hold time for the Data (DQ) bus after the rising edge of CK.

*continued...*



Display Name	Identifier	Description
<b>tDQSKC</b>	MEM_LPDDR3_TDQSC_K_PS	tDQSKC describes the skew between the memory clock (CK) and the input data strobes (DQS) used for reads. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDQSQ</b>	MEM_LPDDR3_TDQSQ_PS	tDQSQ describes the latest valid transition of the associated DQ pins for a READ. tDQSQ specifically refers to the DQS, DQS# to DQ skew. It is the length of time between the DQS, DQS# crossing to the last valid transition of the slowest DQ pin in the DQ group associated with that DQS strobe.
<b>tDSH</b>	MEM_LPDDR3_TDSH_CYC	tDSH specifies the write DQS hold time. This is the time difference between the rising CK edge and the falling edge of DQS, measured as a percentage of tCK.
<b>tDSS</b>	MEM_LPDDR3_TDSS_CYC	tDSS describes the time between the falling edge of DQS to the rising edge of the next CK transition.
<b>tDS (base) AC level</b>	MEM_LPDDR3_TDS_A_C_MV	tDS (base) AC level refers to the voltage level which the data bus must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tDS (base)</b>	MEM_LPDDR3_TDS_P_S	tDS(base) refers to the setup time for the Data (DQ) bus before the rising edge of the DQS strobe.
<b>tIHCA (base) DC level</b>	MEM_LPDDR3_TIH_D_C_MV	DC level of tIH (base) for derating purpose
<b>tIHCA (base)</b>	MEM_LPDDR3_TIH_PS	Address and control hold after CK clock rise
<b>tINIT</b>	MEM_LPDDR3_TINIT_US	tINIT describes the time duration of the memory initialization after a device power-up. After RESET_n is de-asserted, wait for another 500us until CKE becomes active. During this time, the DRAM will start internal initialization; this will be done independently of external clocks.
<b>tISCA (base) AC level</b>	MEM_LPDDR3_TIS_AC_MV	AC level of tIS (base) for derating purpose
<b>tISCA (base)</b>	MEM_LPDDR3_TIS_PS	Address and control setup to CK clock rise
<b>tMRR</b>	MEM_LPDDR3_TMRR_CK_CYC	tMRR describes the minimum MODE REGISTER READ command period.
<b>tMRW</b>	MEM_LPDDR3_TMRW_CK_CYC	tMRW describes the minimum MODE REGISTER WRITE command period.
<b>tQH</b>	MEM_LPDDR3_TQH_CYC	tQH specifies the output hold time for the DQ in relation to DQS, DQS#. It is the length of time between the DQS, DQS# crossing to the earliest invalid transition of the fastest DQ pin in the DQ group associated with that DQS strobe.
<b>tQSH</b>	MEM_LPDDR3_TQSH_CYC	tQSH refers to the differential High Pulse Width, which is measured as a percentage of tCK. It is the time during which the DQS is high for a read.
<b>tRAS</b>	MEM_LPDDR3_TRAS_NS	tRAS describes the activate to precharge duration. A row cannot be deactivated until the tRAS time has been met. Therefore tRAS determines how long the memory has to wait after a activate command before a precharge command can be issued to close the row.
<b>tRCD</b>	MEM_LPDDR3_TRCD_NS	tRCD, row command delay, describes the amount of delay between the activation of a row through the RAS command and the access to the data through the CAS command.

*continued...*



Display Name	Identifier	Description
tWLH	MEM_LPDDR3_TWHL_PS	tWLH describes the write leveling hold time from the rising edge of DQS to the rising edge of CK.
tWLS	MEM_LPDDR3_TWLS_PS	tWLS describes the write leveling setup time. It is measured from the rising edge of CK to the rising edge of DQS.
tWR	MEM_LPDDR3_TWR_NS	tWR refers to the Write Recovery time. It specifies the amount of clock cycles needed to complete a write before a precharge command can be issued.

**Table 196. Group: Mem Timing / Parameters dependent on Speed Bin, Operating Frequency, and Page Size**

Display Name	Identifier	Description
tFAW	MEM_LPDDR3_TFAW_NS	tFAW refers to the four activate window time. It describes the period of time during which only four banks can be active.
tRRD	MEM_LPDDR3_TRRD_CYC	tRRD refers to the Row Active to Row Active Delay. It is the minimum time interval (measured in memory clock cycles) between two activate commands to rows in different banks in the same rank
tRTP	MEM_LPDDR3_TRTP_CYC	tRTP refers to the internal READ Command to PRECHARGE Command delay. It is the number of memory clock cycles that is needed between a read command and a precharge command to the same rank.
tWTR	MEM_LPDDR3_TWTR_CYC	tWTR or Write Timing Parameter describes the delay from start of internal write transaction to internal read command, for accesses to the same bank. The delay is measured from the first rising memory clock edge after the last write data is received to the rising memory clock edge when a read command is received.

**Table 197. Group: Mem Timing / Parameters dependent on Density and Temperature**

Display Name	Identifier	Description
tREFI	MEM_LPDDR3_TREFI_US	tREFI refers to the average periodic refresh interval. It is the maximum amount of time the memory can tolerate in between each refresh command
tRFCab	MEM_LPDDR3_TRFC_NS	Auto-refresh command interval (all banks)

#### 7.4.5.5. Arria 10 EMIF IP LPDDR3 Parameters: Board

**Table 198. Group: Board / Intersymbol Interference/Crosstalk**

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_AC_ISI_NS	The address and command window reduction due to intersymbol interference and crosstalk effects. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQS/DQS# ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_RCLK_ISI_NS	The reduction of the read data window due to intersymbol interference and crosstalk effects on the DQS/DQS# signal when driven by the memory device during a read. The

*continued...*

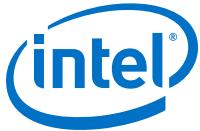


Display Name	Identifier	Description
		number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQ ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_RDATA_ISI_NS	The reduction of the read data window due to intersymbol interference and crosstalk effects on the DQ signal when driven by the memory device during a read. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQS/DQS# ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_WCLK_ISI_NS	The reduction of the write data window due to intersymbol interference and crosstalk effects on the DQS/DQS# signal when driven by the FPGA during a write. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_WDATA_ISI_NS	The reduction of the write data window due to intersymbol interference and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_LPDDR3_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx)*, and manually enter values based on your simulation results, instead of using the default values.

**Table 199. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_LPDDR3_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Maximum board skew within DQS group</b>	BOARD_LPDDR3_BRD_SKEW_WITHIN_DQS_NS	The largest skew between all DQ and DM pins in a DQS group. This value affects the read capture and write margins.
<b>Average delay difference between DQS and CK</b>	BOARD_LPDDR3_DQS_TO_CK_SKEW_NS	The average delay difference between the DQS signals and the CK signal, calculated by averaging the longest and smallest DQS trace delay minus the CK trace delay. Positive values represent DQS signals that are longer than CK signals and negative values represent DQS signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_LPDDR3_IS_SKEWED_WITHIN_AC_DE_SKEWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.

*continued...*



Display Name	Identifier	Description
<b>Package deskewed with board layout (DQS group)</b>	BOARD_LPDDR3_IS_SKew_WITHIN_DQS_ESKEWED	Enable this parameter if you are compensating for package skew on the DQ, DQS, and DM buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to device</b>	BOARD_LPDDR3_MAX_CK_DELAY_NS	The maximum CK delay to device refers to the delay of the longest CK trace from the FPGA to any device.
<b>Maximum DQS delay to device</b>	BOARD_LPDDR3_MAX_DQS_DELAY_NS	The maximum DQS delay to device refers to the delay of the longest DQS trace from the FPGA to any device
<b>Maximum system skew within address/command bus</b>	BOARD_LPDDR3_PKG_BRD_SKEW_WITHIN_AC_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.
<b>Maximum delay difference between devices</b>	BOARD_LPDDR3_SKEW_BETWEEN_DIMMS_NS	This parameter describes the largest propagation delay on the DQ signals between ranks. For example, in a two-rank configuration where devices are placed in series, there is an extra propagation delay for DQ signals going to and coming back from the furthest device compared to the nearest device. This parameter is only applicable when there is more than one rank.
<b>Maximum skew between DQS groups</b>	BOARD_LPDDR3_SKEW_BETWEEN_DQS_NS	The largest skew between DQS signals.

#### 7.4.5.6. Arria 10 EMIF IP LPDDR3 Parameters: Controller

**Table 200. Group: Controller / Low Power Mode**

Display Name	Identifier	Description
<b>Auto Power-Down Cycles</b>	CTRL_LPDDR3_AUTO_POWER_DOWN_CYCS	Specifies the number of idle controller cycles after which the memory device is placed into power-down mode. You can configure the idle waiting time. The supported range for number of cycles is from 1 to 65534.
<b>Enable Auto Power-Down</b>	CTRL_LPDDR3_AUTO_POWER_DOWN_EN	Enable this parameter to have the controller automatically place the memory device into power-down mode after a specified number of idle controller clock cycles. The idle wait time is configurable. All ranks must be idle to enter auto power-down.

**Table 201. Group: Controller / Efficiency**

Display Name	Identifier	Description
<b>Address Ordering</b>	CTRL_LPDDR3_ADDR_ORDER_ENUM	Controls the mapping between Avalon addresses and memory device addresses. By changing the value of this parameter, you can change the mappings between the Avalon-MM address and the DRAM address.
<b>Enable Auto-Precharge Control</b>	CTRL_LPDDR3_AUTO_PRECHARGE_EN	Select this parameter to enable the auto-precharge control on the controller top level. If you assert the auto-precharge control signal while requesting a read or write burst, you can specify whether the controller should close (auto-precharge) the currently open page at the end of the read or write burst, potentially making a future access to a different page of the same bank faster.
<b>Enable Reordering</b>	CTRL_LPDDR3_REORDER_EN	Enable this parameter to allow the controller to perform command and data reordering. Reordering can improve efficiency by reducing bus turnaround time and row/bank

*continued...*



Display Name	Identifier	Description
		switching time. Data reordering allows the single-port memory controller to change the order of read and write commands to achieve highest efficiency. Command reordering allows the controller to issue bank management commands early based on incoming patterns, so that the desired row in memory is already open when the command reaches the memory interface. For more information, refer to the Data Reordering topic in the EMIF Handbook.
<b>Starvation limit for each command</b>	CTRL_LPDDR3_STARVE_LIMIT	Specifies the number of commands that can be served before a waiting command is served. The controller employs a counter to ensure that all requests are served after a pre-defined interval -- this ensures that low priority requests are not ignored, when doing data reordering for efficiency. The valid range for this parameter is from 1 to 63. For more information, refer to the Starvation Control topic in the EMIF Handbook.
<b>Enable Command Priority Control</b>	CTRL_LPDDR3_USER_PRIORITY_EN	Select this parameter to enable user-requested command priority control on the controller top level. This parameter instructs the controller to treat a read or write request as high-priority. The controller attempts to fill high-priority requests sooner, to reduce latency. Connect this interface to the conduit of your logic block that determines when the external memory interface IP treats the read or write request as a high-priority command.

**Table 202. Group: Controller / Configuration, Status, and Error Handling**

Display Name	Identifier	Description
<b>Enable Memory-Mapped Configuration and Status Register (MMR) Interface</b>	CTRL_LPDDR3_MMR_EN	Enable this parameter to change or read memory timing parameters, memory address size, mode register settings, controller status, and request sideband operations.

**Table 203. Group: Controller / Data Bus Turnaround Time**

Display Name	Identifier	Description
<b>Additional read-to-read turnaround time (different ranks)</b>	CTRL_LPDDR3_RD_TO_RD_DIFF_CHIP_DELT_A_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a read of another logical rank. This can resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (different ranks)</b>	CTRL_LPDDR3_RD_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (same rank)</b>	CTRL_LPDDR3_RD_TO_WR_SAME_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read to a write within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (different ranks)</b>	CTRL_LPDDR3_WR_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a read of another logical rank. This can help resolve bus contention problems specific to your board

*continued...*



Display Name	Identifier	Description
		topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (same rank)</b>	CTRL_LPDDR3_WR_TO_RD_SAME_CHIP_DELAY_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write to a read within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-write turnaround time (different ranks)</b>	CTRL_LPDDR3_WR_TO_WR_DIFF_CHIP_DELAY_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.

#### 7.4.5.7. Arria 10 EMIF IP LPDDR3 Parameters: Diagnostics

**Table 204. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_LPDDR3_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 205. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug

*continued...*



Display Name	Identifier	Description
		interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface", an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLITE cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Skip address/command deskew calibration</b>	DIAG_LPDDR3_SKIP_CA_DESKEW	Specifies to skip the address/command deskew calibration stage. Address/command deskew performs per-bit deskew for the address and command pins.
<b>Skip address/command leveling calibration</b>	DIAG_LPDDR3_SKIP_CA_LEVEL	Specifies to skip the address/command leveling stage during calibration. Address/command leveling attempts to center the memory clock edge against CS# by adjusting delay elements inside the PHY, and then applying the same delay offset to the rest of the address and command pins.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.

**Table 206. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX DESIGN_IS SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 207. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic

*continued...*



Display Name	Identifier	Description
		generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 208. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 209. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

#### 7.4.5.8. Arria 10 EMIF IP LPDDR3 Parameters: Example Designs

**Table 210. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX DESIGN GUI LPDDR3_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.



**Table 211. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_LPD DR3_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_LPD DR3_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 212. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_LPD DR3_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.

**Table 213. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_LPD DR3_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

#### 7.4.5.9. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPOLOGY CAPACITY (INTERNAL-ORGANIZATION)



For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

#### 7.4.6. Arria 10 EMIF IP QDR-IV Parameters

The Arria 10 EMIF IP parameter editor allows you to parameterize settings for the Arria 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Arria 10 EMIF IP.

##### 7.4.6.1. Arria 10 EMIF IP QDR-IV Parameters: General

**Table 214. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 215. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft

*continued...*



Display Name	Identifier	Description
		Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 216. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Use recommended PLL reference clock frequency</b>	PHY_QDR4_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the

*continued...*



Display Name	Identifier	Description
		clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 217. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

#### 7.4.6.2. Arria 10 EMIF IP QDR-IV Parameters: Memory

**Table 218. Group: Memory / Topology**

Display Name	Identifier	Description
<b>Address width</b>	MEM_QDR4_ADDR_WIDTH	Number of address pins.
<b>DINVA / DINVB width</b>	MEM_QDR4_DINV_PER_PORT_WIDTH	Number of DINV pins for port A or B of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled. Two memory input pins without expansion and four pins with width expansion.
<b>DKA / DKB width</b>	MEM_QDR4_DK_PER_PORT_WIDTH	Number of DK clock pairs for port A or B of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled. Two memory input pins without expansion and four pins with width expansion.
<b>DQ width per device</b>	MEM_QDR4_DQ_PER_PORT_PER_DEVICE	Specifies number of DQ pins per RLDRAM3 device and number of DQ pins per port per QDR IV device. Available widths for DQ are x18 and x36.

*continued...*



Display Name	Identifier	Description
<b>DQA / DQB width</b>	MEM_QDR4_DQ_PER_PORT_WIDTH	Number of DQ pins for port A or B of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled. The interface supports a width expansion configuration up to 72-bits.
<b>QKA / QKB width</b>	MEM_QDR4_QK_PER_PORT_WIDTH	Number of QK clock pairs for port A or B of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled. Two memory input pins without expansion and four pins with width expansion.
<b>Enable width expansion</b>	MEM_QDR4_WIDTH_EXPANDED	Indicates whether to combine two memory devices to double the data bus width. With two devices, the interface supports a width expansion configuration up to 72-bits. For width expansion configuration, the address and control signals are routed to 2 devices.

**Table 219. Group: Memory / Configuration Register Settings**

Display Name	Identifier	Description
<b>ODT (Address/Command)</b>	MEM_QDR4_AC_ODT_MODE_ENUM	Determines the configuration register setting that controls the address/command ODT setting.
<b>Address bus inversion</b>	MEM_QDR4_ADDR_IN_V_ENA	Enable address bus inversion. AINV are all active high at memory device.
<b>ODT (Clock)</b>	MEM_QDR4_CK_ODT_MODE_ENUM	Determines the configuration register setting that controls the clock ODT setting.
<b>Data bus inversion</b>	MEM_QDR4_DATA_IN_V_ENA	Enable data bus inversion for DQ pins. DINVA[1:0] and DINVB[1:0] are all active high. When set to 1, the corresponding bus is inverted at memory device. If the data inversion feature is programmed to be OFF, then the DINVA/DINVB output bits will always be driven to 0.
<b>ODT (Data)</b>	MEM_QDR4_DATA_ODT_MODE_ENUM	Determines the configuration register setting that controls the data ODT setting.
<b>Output drive (pull-down)</b>	MEM_QDR4_PD_OUTPUT_DRIVE_MODE_ENUM	Determines the configuration register setting that controls the pull-down output drive setting.
<b>Output drive (pull-up)</b>	MEM_QDR4_PU_OUTPUT_DRIVE_MODE_ENUM	Determines the configuration register setting that controls the pull-up output drive setting.

#### 7.4.6.3. Arria 10 EMIF IP QDR-IV Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 220. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_QDR4_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal

*continued...*



Display Name	Identifier	Description
		integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_QDR4_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 221. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_QDR4_USER_AC_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR4_USER_AC_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_QDR4_USER_AC_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 222. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_QDR4_USER_CK_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR4_USER_CK_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_QDR4_USER_CK_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.



**Table 223. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_QDR4_USER_AU_TO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_QDR4_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_QDR4_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR4_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_QDR4_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 224. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_QDR4_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_QDR4_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.4.6.4. Arria 10 EMIF IP QDR-IV Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 225. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_QDR4_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tASH</b>	MEM_QDR4_TASH_PS	tASH provides the setup/hold window requirement for the address bus in relation to the CK clock. Because the individual signals in the address bus may not be perfectly aligned with each other, this parameter describes the intersection window for all the individual address signals setup/hold margins.

*continued...*



Display Name	Identifier	Description
tCKDK_max	MEM_QDR4_TCKDK_M AX_PS	tCKDK_max refers to the maximum skew from the memory clock (CK) to the write strobe (DK).
tCKDK_min	MEM_QDR4_TCKDK_M IN_PS	tCKDK_min refers to the minimum skew from the memory clock (CK) to the write strobe (DK).
tCKQK_max	MEM_QDR4_TCKQK_M AX_PS	tCKQK_max refers to the maximum skew from the memory clock (CK) to the read strobe (QK).
tCSH	MEM_QDR4_TCSH_PS	tCSH provides the setup/hold window requirement for the control bus (LD#, RW#) in relation to the CK clock. Because the individual signals in the control bus may not be perfectly aligned with each other, this parameter describes the intersection window for all the individual control signals setup/hold margins.
tISH	MEM_QDR4_TISH_PS	tISH provides the setup/hold window requirement for the entire data bus (DK or DINV) in all the data groups with respect to the DK clock. After deskew calibration, this parameter describes the intersection window for all the individual data bus signals setup/hold margins.
tQH	MEM_QDR4_TQH_CYC	tQH specifies the output hold time for the DQ/DINV in relation to QK.
tQKQ_max	MEM_QDR4_TQKQ_M AX_PS	tQKQ_max describes the maximum skew between the read strobe (QK) clock edge to the data bus (DQ/DINV) edge.

#### 7.4.6.5. Arria 10 EMIF IP QDR-IV Parameters: Board

Table 226. Group: Board / Intersymbol Interference/Crosstalk

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_QDR4_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>QK/QK# ISI/crosstalk</b>	BOARD_QDR4_USER_RCLK_ISI_NS	QK/QK# ISI/crosstalk describes the reduction of the read data window due to intersymbol interference and crosstalk effects on the QK/QK# signal when driven by the memory device during a read. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQ ISI/crosstalk</b>	BOARD_QDR4_USER_RDATA_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQ signal when driven by the memory device during a read. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>DK/DK# ISI/crosstalk</b>	BOARD_QDR4_USER_WCLK_ISI_NS	DK/DK# ISI/crosstalk describes the reduction of the write data window due to intersymbol interference and crosstalk effects on the DK/DK# signal when driven by the FPGA during a write. The number to be entered in the Quartus Prime software is the total of the measured loss of margin

continued...



Display Name	Identifier	Description
		on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_QDR4_USER_WDATA_ISI_NS	The reduction of the write data window due to intersymbol interference and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_QDR4_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.

**Table 227. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_QDR4_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Average delay difference between DK and CK</b>	BOARD_QDR4_DK_TO_CK_SKEW_NS	This parameter describes the average delay difference between the DK signals and the CK signal, calculated by averaging the longest and smallest DK trace delay minus the CK trace delay. Positive values represent DK signals that are longer than CK signals and negative values represent DK signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_QDR4_IS_SKew_WITHIN_AC_DESKEWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (QK group)</b>	BOARD_QDR4_IS_SKew_WITHIN_QK_DESKEWED	If you are compensating for package skew on the QK bus in the board layout (hence checking the box here), please include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to device</b>	BOARD_QDR4_MAX_CK_DELAY_NS	The maximum CK delay to device refers to the delay of the longest CK trace from the FPGA to any device.
<b>Maximum DK delay to device</b>	BOARD_QDR4_MAX_DK_DELAY_NS	The maximum DK delay to device refers to the delay of the longest DK trace from the FPGA to any device.
<b>Maximum system skew within address/command bus</b>	BOARD_QDR4_PKG_BRD_SKEW_WITHIN_AC_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.
<b>Maximum system skew within QK group</b>	BOARD_QDR4_PKG_BRD_SKEW_WITHIN_QK_NS	Maximum system skew within QK group refers to the largest skew between all DQ and DM pins in a QK group. This value can affect the read capture and write margins.
<b>Maximum delay difference between devices</b>	BOARD_QDR4_SKew_BETWEEN_DIMMS_NS	This parameter describes the largest propagation delay on the DQ signals between ranks. For example, in a two-rank configuration where devices are placed in series, there is an extra propagation delay for DQ signals going to and coming

*continued...*



Display Name	Identifier	Description
		back from the furthest device compared to the nearest device. This parameter is only applicable when there is more than one rank.
<b>Maximum skew between DK groups</b>	BOARD_QDR4_SKEW_BETWEEN_DK_NS	This parameter describes the largest skew between DK signals in different DK groups.

#### 7.4.6.6. Arria 10 EMIF IP QDR-IV Parameters: Controller

**Table 228. Group: Controller**

Display Name	Identifier	Description
<b>Generate power-of-2 data bus widths for Qsys</b>	CTRL_QDR4_AVL_ENA_BLE_POWER_OF_TWO_BUS	If enabled, the Avalon data bus width is rounded down to the nearest power-of-2. The width of the symbols within the data bus is also rounded down to the nearest power-of-2. You should only enable this option if you know you will be connecting the memory interface to Qsys interconnect components that require the data bus and symbol width to be a power-of-2. If this option is enabled, you cannot utilize the full density of the memory device. For example, in x36 data width upon selecting this parameter, will define the Avalon data bus to 256-bit. This will ignore the upper 4-bit of data width.
<b>Maximum Avalon-MM burst length</b>	CTRL_QDR4_AVL_MAX_BURST_COUNT	Specifies the maximum burst length on the Avalon-MM bus. This will be used to configure the FIFOs to be able to manage the maximum data burst. More core logic will require for increase in FIFO length.

#### 7.4.6.7. Arria 10 EMIF IP QDR-IV Parameters: Diagnostics

**Table 229. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_QDR4_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.



**Table 230. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_A_VALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_A_VALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface", an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLITE cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Skip VREF_in calibration</b>	DIAG_QDR4_SKIP_VREF_CAL	Users can check this option to skip the VREF stage of calibration. Users should enable this option for debug purpose only; it is recommended to keep enabled during normal operation.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.

**Table 231. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX_DESIGN_ISP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX_DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 232. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 233. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 234. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

#### 7.4.6.8. Arria 10 EMIF IP QDR-IV Parameters: Example Designs



**Table 235. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_QDR_4_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 236. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_QDR_4_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_QDR_4_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 237. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_QDR_4_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.

**Table 238. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_QDR_4_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.



#### 7.4.6.9. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPOLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

#### 7.4.7. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters

The Arria 10 EMIF IP parameter editor allows you to parameterize settings for the Arria 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Arria 10 EMIF IP.

##### 7.4.7.1. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: General

**Table 239. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.



**Table 240. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 241. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Use recommended PLL reference clock frequency</b>	PHY_QDR2_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the

*continued...*

Display Name	Identifier	Description
		clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 242. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

#### 7.4.7.2. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Memory

**Table 243. Group: Memory / Topology**

Display Name	Identifier	Description
<b>Address width</b>	MEM_QDR2_ADDR_WIDTH	Number of address pins.
<b>Burst length</b>	MEM_QDR2_BL	Burst length of the memory device.
<b>Enable BWS# pins</b>	MEM_QDR2_BWS_EN	Indicates whether the interface uses the BWS#( Byte Write Select) pins. If enabled, 1 BWS# pin for every 9 D pins will be added.
<b>BWS# width</b>	MEM_QDR2_BWS_N_WIDTH	Number of BWS# (Byte Write Select) pins of the memory interface. Automatically calculated based on the data width per device and whether width expansion is enabled. BWS# pins are used to select which byte is written into the device during the current portion of the write operations. Bytes not written remain unaltered.
<b>CQ width</b>	MEM_QDR2_CQ_WIDTH	Width of the CQ (read strobe) clock on the memory device.
<b>Data width per device</b>	MEM_QDR2_DATA_PER_DEVICE	Number of D and Q pins per QDR II device.

*continued...*



Display Name	Identifier	Description
<b>Data width</b>	MEM_QDR2_DATA_WIDTH	Number of D and Q pins of the memory interface. Automatically calculated based on the D and Q width per device and whether width expansion is enabled.
<b>K width</b>	MEM_QDR2_K_WIDTH	Width of the K (address, command and write strobe) clock on the memory device.
<b>Enable width expansion</b>	MEM_QDR2_WIDTH_EXPANDED	Indicates whether to combine two memory devices to double the data bus width. With two devices, the interface supports a width expansion configuration up to 72-bits. For width expansion configuration, the address and control signals are routed to 2 devices.

#### 7.4.7.3. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 244. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_QDR2_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_QDR2_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 245. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_QDR2_USER_ADDRESS_COMMAND_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR2_USER_ADDRESS_COMMAND_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_QDR2_USER_ADDRESS_COMMAND_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 246. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_QDR2_USER_CK_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR2_USER_CK_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_QDR2_USER_CK_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 247. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_QDR2_USER_AU_TO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_QDR2_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_QDR2_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR2_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_QDR2_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 248. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_QDR2_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_QDR2_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.



#### 7.4.7.4. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 249. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Internal Jitter</b>	MEM_QDR2_INTERNAL_JITTER_NS	QDRII internal jitter.
<b>Speed bin</b>	MEM_QDR2_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tCCQO</b>	MEM_QDR2_TCCQO_NS	tCCQO describes the skew between the rising edge of the C clock to the rising edge of the echo clock (CQ) in QDRII memory devices.
<b>tCQDOH</b>	MEM_QDR2_TCQDOH_NS	tCQDOH refers to the minimum time expected between the echo clock (CQ or CQ#) edge and the last of the valid Read data (Q).
<b>tCQD</b>	MEM_QDR2_TCQD_NS	tCQD refers to the maximum time expected between an echo clock edge and valid data on the Read Data bus (Q).
<b>tCQH</b>	MEM_QDR2_TCQH_NS	tCQH describes the time period during which the echo clock (CQ, #CQ) is considered logically high.
<b>tHA</b>	MEM_QDR2_THA_NS	tHA refers to the hold time after the rising edge of the clock (K) to the address and command control bus (A). The address and command control bus must remain stable for at least tHA after the rising edge of K.
<b>tHD</b>	MEM_QDR2_THD_NS	tHD refers to the hold time after the rising edge of the clock (K) to the data bus (D). The data bus must remain stable for at least tHD after the rising edge of K.
<b>tRL</b>	MEM_QDR2TRL_CYC	tRL refers to the QDR memory specific read latency. This parameter describes the length of time after a Read command has been registered on the rising edge of the Write Clock (K) at the QDR memory before the first piece of read data (Q) can be expected at the output of the memory. It is measured in Write Clock (K) cycles. The Read Latency is specific to a QDR memory device and cannot be modified to a different value. The Read Latency (tRL) can have the following values: 1.5, 2, 2.5 clk cycles.
<b>tSA</b>	MEM_QDR2_TSA_NS	tSA refers to the setup time for the address and command bus (A) before the rising edge of the clock (K). The address and command bus must be stable for at least tSA before the rising edge of K.
<b>tSD</b>	MEM_QDR2_TSD_NS	tSD refers to the setup time for the data bus (D) before the rising edge of the clock (K). The data bus must be stable for at least tSD before the rising edge of K.
<b>tWL</b>	MEM_QDR2_TWLCYC	tWL refers to the write latency requirement at the QDR memory. This parameter describes the length of time after a Write command has been registered at the memory on the rising edge of the Write clock (K) before the memory expects the Write Data (D). It is measured in (K) clock cycles and is usually 1.

#### 7.4.7.5. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Board

**Table 250. Group: Board / Intersymbol Interference/Crosstalk**

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_QDR2_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>CQ/CQ# ISI/crosstalk</b>	BOARD_QDR2_USER_RCLK_ISI_NS	CQ/CQ# ISI/crosstalk describes the reduction of the read data window due to intersymbol interference and crosstalk effects on the CQ/CQ# signal when driven by the memory device during a read. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read Q ISI/crosstalk</b>	BOARD_QDR2_USER_RDATA_ISI_NS	Read Q ISI/crosstalk describes the reduction of the read data window due to intersymbol interference and crosstalk effects on the CQ/CQ# signal when driven by the memory device during a read. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>K/K# ISI/crosstalk</b>	BOARD_QDR2_USER_WCLK_ISI_NS	K/K# ISI/crosstalk describes the reduction of the write data window due to intersymbol interference and crosstalk effects on the K/K# signal when driven by the FPGA during a write. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write D ISI/crosstalk</b>	BOARD_QDR2_USER_WDATA_ISI_NS	Write D ISI/crosstalk describes the reduction of the write data window due to intersymbol interference and crosstalk effects on the signal when driven by driven by the FPGA during a write. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_QDR2_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.

**Table 251. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and K</b>	BOARD_QDR2_AC_TO_K_SKEW_NS	This parameter refers to the average delay difference between the Address/Command signals and the K signal, calculated by averaging the longest and smallest Address/Command trace delay minus the maximum K trace delay. Positive values represent address and command signals that

*continued...*



Display Name	Identifier	Description
		are longer than K signals and negative values represent address and command signals that are shorter than K signals.
<b>Maximum board skew within D group</b>	BOARD_QDR2_BRD_SKEW_WITHIN_D_NS	This parameter refers to the largest skew between all D and BWS# signals in a D group. D pins are used for driving data signals to the memory device during a write operation. BWS# pins are used as Byte Write Select signals to control which byte(s) are written to the memory during a write operation. Users should enter their board skew only. Package skew will be calculated automatically, based on the memory interface configuration, and added to this value. This value affects the read capture and write margins.a
<b>Maximum board skew within Q group</b>	BOARD_QDR2_BRD_SKEW_WITHIN_Q_NS	This parameter describes the largest skew between all Q signals in a Q group. Q pins drive the data signals from the memory to the FPGA when the read operation is active. Users should enter their board skew only. Package skew will be calculated automatically, based on the memory interface configuration, and added to this value. This value affects the read capture and write margins.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_QDR2_IS_SKEW_WITHIN_AC_DESK_EWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (D group)</b>	BOARD_QDR2_IS_SKEW_WITHIN_D_DESK_EWED	If you are compensating for package skew on the D and BWS# signals in the board layout (hence checking the box here), please include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (Q group)</b>	BOARD_QDR2_IS_SKEW_WITHIN_Q_DESK_EWED	If you are compensating for package skew on the Q bus in the board layout (hence checking the box here), please include package skew in calculating the following board skew parameters.
<b>Maximum K delay to device</b>	BOARD_QDR2_MAX_K_DELAY_NS	The maximum K delay to device refers to the delay of the longest K trace from the FPGA to any device
<b>Maximum system skew within address/command bus</b>	BOARD_QDR2_PKG_BUS_SKew_WITHIN_ADDRESS_COMMAND_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.

#### 7.4.7.6. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Controller

**Table 252. Group: Controller**

Display Name	Identifier	Description
<b>Generate power-of-2 data bus widths for Qsys</b>	CTRL_QDR2_AVL_ENA_BLE_POWER_OF_TWO_BUS	If enabled, the Avalon data bus width is rounded down to the nearest power-of-2. The width of the symbols within the data bus is also rounded down to the nearest power-of-2. You should only enable this option if you know you will be connecting the memory interface to Qsys interconnect components that require the data bus and symbol width to be a power-of-2. If this option is enabled, you cannot utilize the full density of the memory device. For example, in x36 data width upon selecting this parameter, will define the Avalon data bus to 256-bit. This will ignore the upper 4-bit of data width.
<b>Maximum Avalon-MM burst length</b>	CTRL_QDR2_AVL_MAX_BURST_COUNT	Specifies the maximum burst length on the Avalon-MM bus. This will be used to configure the FIFOs to be able to manage the maximum data burst. More core logic will require for increase in FIFO length.



#### 7.4.7.7. Arria 10 EMIF IP QDR II/II+/III+ Xtreme Parameters: Diagnostics

**Table 253. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_QDR2_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 254. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface," an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLite cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.



**Table 255. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX_DESIGN_IS_SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX_DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 256. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 257. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 258. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

#### 7.4.7.8. Arria 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Example Designs

**Table 259. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX DESIGN GUI QDR 2_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 260. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX DESIGN GUI QDR 2_GEN_SIM	Specifies that the 'Generate Example Design' button create all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX DESIGN GUI QDR 2_GEN_SYNTH	Specifies that the 'Generate Example Design' button create all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 261. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX DESIGN GUI QDR 2_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.



**Table 262. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
Select board	EX_DESIGN_GUI_QDR2_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

#### 7.4.7.9. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

#### 7.4.8. Arria 10 EMIF IP RLDRAM 3 Parameters

The Arria 10 EMIF IP parameter editor allows you to parameterize settings for the Arria 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.



The following tables describe the parameterization settings available in the parameter editor for the Arria 10 EMIF IP.

#### 7.4.8.1. Arria 10 EMIF IP RLDRAM 3 Parameters: General

**Table 263. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 264. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 265. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.

*continued...*



Display Name	Identifier	Description
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Use recommended PLL reference clock frequency</b>	PHY_RLD3_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 266. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

#### 7.4.8.2. Arria 10 EMIF IP RLDRAM 3 Parameters: Memory

**Table 267. Group: Memory / Topology**

Display Name	Identifier	Description
<b>Address width</b>	MEM_RLD3_ADDR_WIDTH	Number of address pins.
<b>CS# width</b>	MEM_RLD3_CS_WIDTH	Number of chip selects of the memory interface.
<b>Enable depth expansion using twin die package</b>	MEM_RLD3_DEPTH_EXPANDED	Indicates whether to combine two RLDRAM3 devices to double the address space, resulting in more density.
<b>DK width</b>	MEM_RLD3_DK_WIDTH	Number of DK clock pairs of the memory interface. This is equal to the number of write data groups, and is automatically calculated based on the DQ width per device and whether width expansion is enabled.

*continued...*



Display Name	Identifier	Description
<b>DQ width per device</b>	MEM_RLD3_DQ_PER_DEVICE	Specifies number of DQ pins per RLDRAM3 device and number of DQ pins per port per QDR IV device. Available widths for DQ are x18 and x36.
<b>DQ width</b>	MEM_RLD3_DQ_WIDTH	Number of data pins of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled.
<b>QK width</b>	MEM_RLD3_QK_WIDTH	Number of QK output clock pairs of the memory interface. This is equal to the number of read data groups, and is automatically calculated based on the DQ width per device and whether width expansion is enabled.
<b>Enable width expansion</b>	MEM_RLD3_WIDTH_EXPANDED	Indicates whether to combine two memory devices to double the data bus width. With two devices, the interface supports a width expansion configuration up to 72-bits. For width expansion configuration, the address and control signals are routed to 2 devices.

**Table 268. Group: Memory / Mode Register Settings**

Display Name	Identifier	Description
<b>AREF protocol</b>	MEM_RLD3_AREF_PROTOCOL_ENUM	Determines the mode register setting that controls the AREF protocol setting. The AUTO REFRESH (AREF) protocol is selected by setting mode register 1. There are two ways in which AREF commands can be issued to the RLDRAM, the memory controller can either issue bank address-controlled or multibank AREF commands. Multibank refresh protocol allows for the simultaneous refreshing of a row in up to four banks
<b>Data Latency</b>	MEM_RLD3_DATA_LATENCY_MODE_ENUM	Determines the mode register setting that controls the data latency. Sets both READ and WRITE latency (RL and WL).
<b>ODT</b>	MEM_RLD3_ODT_MODE_ENUM	Determines the mode register setting that controls the ODT setting.
<b>Output drive</b>	MEM_RLD3_OUTPUT_DRIVE_MODE_ENUM	Determines the mode register setting that controls the output drive setting.
<b>tRC</b>	MEM_RLD3_T_RC_MODE_ENUM	Determines the mode register setting that controls the tRC(activate to activate timing parameter). Refer to the tRC table in the memory vendor data sheet. Set the tRC according to the memory speed grade and data latency. Full name of tRC
<b>Write protocol</b>	MEM_RLD3_WRITE_PROTOCOL_ENUM	Determines the mode register setting that controls the write protocol setting. When multiple bank (dual bank or quad bank) is selected, identical data is written to multiple banks.

#### 7.4.8.3. Arria 10 EMIF IP RLDRAM 3 Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 269. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_RLD3_DEFAULT_I_O	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal <b>continued...</b>



Display Name	Identifier	Description
		integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_RLD3_IO_VOLTA GE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC _OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 270. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_RLD3_USER_AC_ IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_RLD3_USER_AC_ MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_RLD3_USER_AC_ SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 271. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_RLD3_USER_CK_ IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_RLD3_USER_CK_ MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_RLD3_USER_CK_ SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 272. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_RLD3_USER_AUTO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_RLD3_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_RLD3_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_RLD3_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_RLD3_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 273. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_RLD3_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_RLD3_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.4.8.4. Arria 10 EMIF IP RLDRAM 3 Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 274. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_RLD3_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tCKDK_max</b>	MEM_RLD3_TCKDK_MAX_CYC	tCKDK_max refers to the maximum skew from the memory clock (CK) to the write strobe (DK).
<b>tCKDK_min</b>	MEM_RLD3_TCKDK_MIN_CYC	tCKDK_min refers to the minimum skew from the memory clock (CK) to the write strobe (DK).
<b>tCKQK_max</b>	MEM_RLD3_TCKQK_MAX_PS	tCKQK_max refers to the maximum skew from the memory clock (CK) to the read strobe (QK).

*continued...*



Display Name	Identifier	Description
<b>tDH (base) DC level</b>	MEM_RLD3_TD�_DC_MV	tDH (base) DC level refers to the voltage level which the data bus must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tDH (base)</b>	MEM_RLD3_TD�_PS	tDH (base) refers to the hold time for the Data (DQ) bus after the rising edge of CK.
<b>tDS (base) AC level</b>	MEM_RLD3_TDS_AC_MV	tDS (base) AC level refers to the voltage level which the data bus must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tDS (base)</b>	MEM_RLD3_TDS_PS	tDS(base) refers to the setup time for the Data (DQ) bus before the rising edge of the DQS strobe.
<b>tIH (base) DC level</b>	MEM_RLD3_TIH_DC_MV	tIH (base) DC level refers to the voltage level which the address/command signal must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tIH (base)</b>	MEM_RLD3_TIH_PS	tIH (base) refers to the hold time for the Address/Command (A) bus after the rising edge of CK. Depending on what AC level the user has chosen for a design, the hold margin can vary (this variance will be automatically determined when the user chooses the "tIH (base) AC level").
<b>tIS (base) AC level</b>	MEM_RLD3_TIS_AC_MV	tIS (base) AC level refers to the voltage level which the address/command signal must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tIS (base)</b>	MEM_RLD3_TIS_PS	tIS (base) refers to the setup time for the Address/Command/Control (A) bus to the rising edge of CK.
<b>tQH</b>	MEM_RLD3_TQH_CYC	tQH specifies the output hold time for the DQ/DINV in relation to QK.
<b>tQKQ_max</b>	MEM_RLD3_TQKQ_MAX_PS	tQKQ_max describes the maximum skew between the read strobe (QK) clock edge to the data bus (DQ/DINV) edge.

#### 7.4.8.5. Arria 10 EMIF IP RLDRAM 3 Parameters: Board

Table 275. Group: Board / Intersymbol Interference/Crosstalk

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_RLD3_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total loss of margin on both the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>QK/QK# ISI/crosstalk</b>	BOARD_RLD3_USER_RCLK_ISI_NS	QK/QK# ISI/crosstalk describes the reduction of the read data window due to intersymbol interference and crosstalk effects on the QK/QK# signal when driven by the memory device during a read. The number to be entered in the Quartus Prime software is the total of the measured loss of

*continued...*



Display Name	Identifier	Description
		margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>DK/DK# ISI/crosstalk</b>	BOARD_RLD3_USER_WCLK_ISI_NS	DK/DK# ISI/crosstalk describes the reduction of the write data window due to intersymbol interference and crosstalk effects on the DK/DK# signal when driven by the FPGA during a write. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_RLD3_USER_WDATA_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_RLD3_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.

**Table 276. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_RLD3_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Maximum board skew within QK group</b>	BOARD_RLD3_BRD_SKEW_WITHIN_QK_NS	Maximum board skew within QK group refers to the largest skew between all DQ and DM pins in a QK group. This value can affect the read capture and write margins.
<b>Average delay difference between DK and CK</b>	BOARD_RLD3_DK_TO_CK_SKEW_NS	This parameter describes the average delay difference between the DK signals and the CK signal, calculated by averaging the longest and smallest DK trace delay minus the CK trace delay. Positive values represent DK signals that are longer than CK signals and negative values represent DK signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_RLD3_IS_SKEW_WITHIN_AC_DESK_EWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (QK group)</b>	BOARD_RLD3_IS_SKEW_WITHIN_QK_DESK_EWED	If you are compensating for package skew on the QK bus in the board layout (hence checking the box here), please include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to device</b>	BOARD_RLD3_MAX_CK_DELAY_NS	The maximum CK delay to device refers to the delay of the longest CK trace from the FPGA to any device.
<b>Maximum DK delay to device</b>	BOARD_RLD3_MAX_DK_DELAY_NS	The maximum DK delay to device refers to the delay of the longest DK trace from the FPGA to any device.

*continued...*



Display Name	Identifier	Description
<b>Maximum system skew within address/command bus</b>	BOARD_RLD3_PKG_BRD_SKEW_WITHIN_A_C_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.
<b>Maximum delay difference between devices</b>	BOARD_RLD3_SKEW_BETWEEN_DIMMS_NS	This parameter describes the largest propagation delay on the DQ signals between ranks. For example, in a two-rank configuration where devices are placed in series, there is an extra propagation delay for DQ signals going to and coming back from the furthest device compared to the nearest device. This parameter is only applicable when there is more than one rank.
<b>Maximum skew between DK groups</b>	BOARD_RLD3_SKEW_BETWEEN_DK_NS	This parameter describes the largest skew between DK signals in different DK groups.

#### 7.4.8.6. Arria 10 EMIF IP RLDRAM 3 Parameters: Diagnostics

**Table 277. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_RLD3_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 278. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface", an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug

*continued...*



Display Name	Identifier	Description
		Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLite cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.

**Table 279. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX DESIGN_IS SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 280. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.

*continued...*



Display Name	Identifier	Description
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 281. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG_EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 282. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

#### 7.4.8.7. Arria 10 EMIF IP RLDRAM 3 Parameters: Example Designs

**Table 283. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_RLD3_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 284. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_RLD3_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl

*continued...*



Display Name	Identifier	Description
		from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_RLD_3_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 285. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_RLD_3_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.

**Table 286. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_RLD_3_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

#### 7.4.8.8. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPOLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.



When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

#### 7.4.9. Equations for Arria 10 EMIF IP Board Skew Parameters

The following topics illustrate the underlying equations for the board skew parameters for each supported memory protocol.

##### 7.4.9.1. Equations for DDR3/DDR4/LPDDR3 Board Skew Parameters

**Table 287. Parameter Equations**

Parameter	Description/Equation
Maximum CK delay to DIMM/device	<p>The delay of the longest CK trace from the FPGA to any DIMM/device.</p> $\max_r[\max_n(CK_{n,r}PathDelay)]$ <p>Where <math>n</math> is the number of memory clock and <math>r</math> is the number rank of DIMM/device. For example in dual-rank DIMM implementation, if there are 2 pairs of memory clocks in each rank DIMM, the maximum CK delay is expressed by the following equation:</p> $\max(CK_1PathDelay \text{ rank 1}, CK_2Path Delay \text{ rank 1}, CK_1Path Delay \text{ rank 2}, CK_2Path Delay \text{ rank 2})$
Maximum DQS delay to DIMM/device	<p>The delay of the longest DQS trace from the FPGA to the DIMM/device.</p> $\max_r[\max_n(DQS_{n,r}Path Delay)]$ <p>Where <math>n</math> is the number of DQS and <math>r</math> is the number of rank of DIMM/device. For example in dual-rank DIMM implementation, if there are 2 DQS in each rank DIMM, the maximum DQS delay is expressed by the following equation:</p> $\max(DQS_1PathDelay \text{ rank 1}, DQS_2Path Delay \text{ rank 1}, DQS_1Path Delay \text{ rank 2}, DQS_2Path Delay \text{ rank 2})$
Average delay difference between DQS and CK	<p>The average delay difference between the DQS signals and the CK signal, calculated by averaging the longest and smallest DQS delay minus the CK delay. Positive values represent DQS signals that are longer than CK signals and negative values represent DQS signals that are shorter than CK signals. The Quartus Prime software uses this skew to optimize the delay of the DQS signals for appropriate setup and hold margins.</p> $\frac{\max_r[\max_{n,m}\{(DQS_{m,r}Delay - CK_{n,r}Delay)\}] - \min_r[\min_{n,m}\{(DQS_{m,r}Delay - CK_{n,r}Delay\}]}{2}$

*continued...*



Parameter	Description/Equation
	<p>Where <math>n</math> is the number of memory clock, <math>m</math> is the number of DQS, and <math>r</math> is the number of rank of DIMM/device.</p> <p>When using discrete components, the calculation differs slightly. Find the minimum and maximum values for (DQS-CK) over all groups and then divide by 2. Calculate the (DQS-CK) for each DQS group, by using the appropriate CLK for that group.</p> <p>For example, in a configuration with 5 x16 components, with each component having two DQS groups: To find the minimum and maximum, calculate the minimum and maximum of (DQS0 - CK0, DQS1 - CK0, DQS2 - CK1, DQS3 - CK1, and so forth) and then divide the result by 2.</p>
Maximum Board skew within DQS group	<p>The largest skew between all DQ and DM pins in a DQS group. Enter your board skew only. Package skew is calculated automatically, based on the memory interface configuration, and added to this value. This value affects the read capture and write margins.</p> $\left[ \max_g^{groups} [maxDQ_g - minDQ_g] \right]$
Maximum skew between DQS groups	<p>The largest skew between DQS signals in different DQS groups.</p> $\left[ \max_g^{groups} [DQS_g] \right] - \left[ \min_g^{groups} [DQS_g] \right]$
Maximum system skew within address/command bus	$(MaxAC - Min AC)$ <p>The largest skew between the address and command signals. Enter combined board and package skew. In the case of a component, find the maximum address/command and minimum address/command values across all component address signals.</p>
Average delay difference between address/command and CK	<p>A value equal to the average of the longest and smallest address/command signal delays, minus the delay of the CK signal. The value can be positive or negative.</p> <p>The average delay difference between the address/command and CK is expressed by the following equation:</p> $\sum_{n=1}^{n=n} \left[ \left( \frac{\text{Longest AC Path Delay} + \text{Shortest AC Path Delay}}{2} \right) - CK_n \text{PathDelay} \right]$ <p>where <math>n</math> is the number of memory clocks.</p>
Maximum delay difference between DIMMs/devices	<p>The largest propagation delay on DQ signals between ranks. For example, in a two-rank configuration where you place DIMMs in different slots there is also a propagation delay for DQ signals going to and coming back from the furthest</p>



Parameter	Description/Equation
	<p>DIMM compared to the nearest DIMM. This parameter is applicable only when there is more than one rank.</p> $\text{Maxr} \{ \max n, m [ (DQn\_r \text{ path delay} - DQn\_r+1 \text{ path delay}), (DQS_m\_r \text{ path delay} - DQS_m\_r+1 \text{ path delay}) ] \}$ <p>Where <math>n</math> is the number of DQ, <math>m</math> is the number of DQS and <math>r</math> is number of rank of DIMM/device .</p>

#### 7.4.9.2. Equations for QDR-IV Board Skew Parameters

**Table 288. Parameter Equations**

Parameter	Description/Equation
Maximum system skew within address/command bus	<p>The largest skew between the address and command signals. Enter combined board and package skew.</p> $(MaxAC - Min AC)$
Average delay difference between address/command and CK	<p>The average delay difference between the address and command signals and the CK signal, calculated by averaging the longest and smallest Address/Command signal delay minus the CK delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals. The Quartus Prime software uses this skew to optimize the delay of the address and command signals to have appropriate setup and hold margins.</p> $\sum_{n=1}^N \left[ \left( \frac{\text{Longest AC Path Delay} + \text{Shortest AC Path Delay}}{2} \right) - CK_n \text{PathDelay} \right]$ <p>where <math>n</math> is the number of memory clocks.</p>
Maximum System skew within QK group	<p>The largest skew between all DQ and DM pins in a QK group. Enter combined board and package skew. This value affects the read capture and write margins.</p> $\max_n (\max DQ_n - \min DQ_n)$ <p>Where <math>n</math> includes both DQa and DQb</p>
Maximum CK delay to device	<p>The delay of the longest CK trace from the FPGA to any device.</p> $[\max_n CK_n \text{PathDelay}]$ <p>where <math>n</math> is the number of memory clocks.</p>
Maximum DK delay to device	<p>The delay of the longest DK trace from the FPGA to any device.</p>

*continued...*



Parameter	Description/Equation
	$\max_{n} (DK_n \text{PathDelay})$ where $n$ is the number of DK.
Average delay difference between DK and CK	The average delay difference between the DK signals and the CK signal, calculated by averaging the longest and smallest DK delay minus the CK delay. Positive values represent DK signals that are longer than CK signals and negative values represent DK signals that are shorter than CK signals. The Quartus Prime software uses this skew to optimize the delay of the DK signals to have appropriate setup and hold margins. $\frac{\min_{n,m} (CK_n \text{PathDelay} - DK_m \text{PathDelay}) + \max_{n,m} (CK_n \text{PathDelay} - DK_m \text{PathDelay})}{2}$ CDO:/content/authoring/rto1474984235656.xml where $n$ is the number of memory clocks and $m$ is the number of DK.
Maximum skew between DK groups	The largest skew between DK signals in different DK groups. $\max_n (\max_{n_i} DK_n - \min_{n_i} DK_n)$ where $n$ is the number of DK. Where $n$ includes both DQa and DQb.

#### 7.4.9.3. Equations for QDRII, QDRII+, and QDRII+ Xtreme Board Skew Parameters

**Table 289. Parameter Equations**

Parameter	Description/Equation
Maximum system skew within address/command bus	$(MaxAC - Min AC)$ The largest skew between the address and command signals. Enter combined board and package skew.
Average delay difference between address/command and K	The average delay difference between the address and command signals and the K signal, calculated by averaging the longest and smallest Address/Command signal delay minus the K delay. Positive values represent address and command signals that are longer than K signals and negative values represent address and command signals that are shorter than K signals. The Quartus Prime software uses this skew to optimize the delay of the address and command signals to have appropriate setup and hold margins. $\sum_{n=1}^{n=n} \left[ \left( \frac{\text{Longest AC Path Delay} + \text{Shortest AC Path Delay}}{2} \right) - K_n \text{PathDelay} \right]$ where $n$ is the number of K clocks.

*continued...*



Parameter	Description/Equation
Maximum board skew within Q group	The largest skew between all Q pins in a Q group. Enter your board skew only. Package skew is calculated automatically, based on the memory interface configuration, and added to this value. This value affects the read capture and write margins. $\max_g [maxQ_g - minQ_g]$ where $g$ is the number of Q group.
Maximum board skew within D group	The largest skew between all D and BWS# pins in a D group. Enter your board skew only. Package skew is calculated automatically, based on the memory interface configuration, and added to this value. This value affects the read capture and write margins. $\max_g [maxD_g - minD_g]$ where $g$ is the number of D group.
Maximum K delay to device	$\max_n K_n PathDelay$ where $n$ is the number of K clocks.

#### 7.4.9.4. Equations for RLDRAM 3 Board Skew Parameters

**Table 290. Parameter Equations**

Parameter	Description/Equation
Maximum CK delay to device	The delay of the longest CK trace from the FPGA to any device. $\max_n CK_n PathDelay$ where $n$ is the number of memory clocks. For example, the maximum CK delay for two pairs of memory clocks is expressed by the following equation: $\max_2 (CK_1 PathDelay, CK_2 PathDelay)$
Maximum DK delay to device	The delay of the longest DK trace from the FPGA to any device. $\max_n DK_n PathDelay$ where $n$ is the number of DK. For example, the maximum DK delay for two DK is expressed by the following equation: $\max_2 (DK_1 PathDelay, DK_2 PathDelay)$
Average delay difference between DK and CK	The average delay difference between the DK signals and the CK signal, calculated by averaging the longest and smallest DK delay minus the CK delay. Positive values

*continued...*



Parameter	Description/Equation
	<p>represent DK signals that are longer than CK signals and negative values represent DK signals that are shorter than CK signals. The Quartus Prime software uses this skew to optimize the delay of the DK signals to have appropriate setup and hold margins.</p> $\frac{\max_{n,m}(CK_n PathDelay - DK_m PathDelay) + \min_{n,m}(CK_n PathDelay - DK_m PathDelay)}{2}$ <p>where n is the number of memory clocks and m is the number of DK.</p>
Maximum system skew within address/command bus	$(MaxAC - Min AC)$ <p>The largest skew between the address and command signals. Enter combined board and package skew.</p>
Average delay difference between address/command and CK	<p>The average delay difference between the address and command signals and the CK signal, calculated by averaging the longest and smallest Address/Command signal delay minus the CK delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals. The Quartus Prime software uses this skew to optimize the delay of the address and command signals to have appropriate setup and hold margins.</p> $\sum_{n=1}^{n=n} \left[ \left( \frac{\text{Longest AC Path Delay} + \text{Shortest AC Path Delay}}{2} \right) - CK_n PathDelay \right]$ <p>where n is the number of memory clocks.</p>
Maximum board skew within QK group	<p>The largest skew between all DQ and DM pins in a QK group. Enter your board skew only. Package skew will be calculated automatically, based on the memory interface configuration, and added to this value. This value affects the read capture and write margins.</p> $\max_n(\max DQ_n - \min DQ_n)$ <p>where n is the number of DQ.</p>
Maximum skew between DK groups	<p>The largest skew between DK signals in different DK groups.</p> $\max_n(\max DK_n - \min DK_n)$ <p>where n is the number of DQ.</p>



## 7.5. Intel Stratix 10 External Memory Interface IP

This section contains information about parameterizing Intel Stratix 10 External Memory Interface IP.

### 7.5.1. Qsys Interfaces

The interfaces in the Stratix 10 External Memory Interface IP each have signals that can be connected in Qsys. The following tables list the signals available for each interface and provide a description and guidance on how to connect those interfaces.

#### Stratix 10 External Memory Interface IP Interfaces

**Table 291. Interface: afi\_clk\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
afi_clk	Output	<ul style="list-style-type: none"> <li>DDR3, DDR4, LPDDR3, RLDRAM 3, QDR IV</li> <li>Hard PHY only</li> </ul>	The Altera PHY Interface (AFI) clock output signal. The clock frequency in relation to the memory clock frequency depends on the <b>Clock rate of user logic</b> value set in the parameter editor. Connect this interface to the (clock input) conduit of the custom AFI-based memory controller connected to the afi_conduit_end or any user logic block that requires the generated clock frequency.

**Table 292. Interface: afi\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
afi_cal_success	Output		
afi_cal_fail	Output		
afi_cal_req	Input		
afi_rlat	Output		
afi_wlat	Output		
afi_addr	Input		
afi_RST_n	Input		
afi_wdata_valid	Input		
afi_wdata	Input		
afi_rdata_en_full	Input		
afi_rdata	Output		
afi_rdata_valid	Output		
afi_rrank	Input		
afi_wrank	Input		

*continued...*



Signals in Interface	Direction	Availability	Description
afi_ba	Input	<ul style="list-style-type: none"><li>DDR3, DDR4, RLDRAM 3</li><li>Hard PHY only</li></ul>	
afi_cs_n	Input	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3, RLDRAM 3</li><li>Hard PHY only</li></ul>	
afi_cke	Input	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3</li><li>Hard PHY only</li></ul>	
afi_odt	Input		
afi_dqs_burst	Input		
afi_ap	Input	<ul style="list-style-type: none"><li>QDR IV</li><li>Hard PHY only</li></ul>	
afi_pe_n	Output		
afi_ainv	Input		
afi_id_n	Input		
afi_rw_n	Input		
afi_cfg_n	Input		
afi_lbk0_n	Input		
afi_lbk1_n	Input		
afi_rdata_dinv	Output	<ul style="list-style-type: none"><li>QDR IV</li><li>Hard PHY only</li></ul>	The Altera PHY Interface (AFI) signals between the external memory interface IP and the custom AFI-based memory controller.
afi_wdata_dinv	Input		Connect this interface to the AFI conduit of the custom AFI-based memory controller.
afi_we_n	Input	<ul style="list-style-type: none"><li>DDR3, RLDRAM 3</li><li>Hard PHY only</li></ul>	The Altera PHY Interface (AFI) signals between the external memory interface IP and the custom AFI-based memory controller.
afi_dm	Input	<ul style="list-style-type: none"><li>DDR3, LPDDR3, RLDRAM 3</li><li>Hard PHY only</li><li><b>Enable DM pins=True</b></li></ul>	Connect this interface to the AFI conduit of the custom AFI-based memory controller.
afi_ras_n	Input	<ul style="list-style-type: none"><li>DDR3</li><li>Hard PHY only</li></ul>	For more information, refer to the <a href="#">AFI 4.0 Specification</a> .
afi_cas_n	Input		
afi_rm	Input	<ul style="list-style-type: none"><li>DDR3</li><li>Hard PHY only</li><li>LRDIMM with <b>Number of rank multiplication pins &gt; 0</b></li></ul>	
afi_par	Input	<ul style="list-style-type: none"><li>DDR3</li><li>Hard PHY only</li><li>RDIMM/LRDIMM</li><li>DDR4</li><li>Hard PHY only</li><li><b>Enable alert_n/par pins = True</b></li></ul>	
afi_bg	Input	<ul style="list-style-type: none"><li>DDR4</li><li>Hard PHY only</li></ul>	

continued...



Signals in Interface	Direction	Availability	Description
afi_act_n	Input	<ul style="list-style-type: none"> <li>• DDR4</li> <li>• Hard PHY only</li> <li>• <b>Enable DM pins=True</b></li> </ul>	
afi_dm_n	Input		
afi_ref_n	Input		

**Table 293. Interface: afi\_half\_clk\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
afi_half_clk	Output	<ul style="list-style-type: none"> <li>• DDR3, DDR4, LPDDR3, RLDRAM 3, QDR IV</li> <li>• Hard PHY only</li> </ul>	<p>The Altera PHY Interface (AFI) half clock output signal. The clock runs at half the frequency of the AFI clock (afi_clk clock). Connect this interface to the clock input conduit of the user logic block that needs to be clocked at the generated clock frequency.</p>

**Table 294. Interface: afi\_reset\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
afi_reset_n	Output	<ul style="list-style-type: none"> <li>• DDR3, DDR4, LPDDR3, RLDRAM 3, QDR IV</li> <li>• Hard PHY only</li> </ul>	<p>The Altera PHY Interface (AFI) reset output signal. Asserted when the PLL becomes unlocked or when the PHY is reset. Asynchronous assertion and synchronous deassertion. Connect this interface to the reset input conduit of the custom AFI-based memory controller connected to the afi_conduit_end and all the user logic blocks that are under the AFI clock domain afi_clk or afi_half_clk clock).</p>

**Table 295. Interface: cal\_debug\_avalon\_slave**

Interface type: Avalon Memory-Mapped Slave

Signals in Interface	Direction	Availability	Description
cal_debug_waitrequest	Output	<ul style="list-style-type: none"> <li>• EMIF Debug Toolkit</li> <li>• <b>On-Chip Debug Port=Export</b></li> </ul>	<p>The Avalon-MM signals between the external memory interface IP and the external memory interface Debug Component. Connect this interface to the (to_ioaux) Avalon-MM master of the <b>Stratix 10 EMIF Debug Component</b> IP or to (cal_debug_out_avalon_master) Avalon-MM master of the other external memory interface IP that has exported the interface. If you are not using the Altera</p>
cal_debug_read	Input		
cal_debug_write	Input		
cal_debug_addr	Input		
cal_debug_read_data	Output		
cal_debug_write_data	Input		
cal_debug_bytelenable	Input		
cal_debug_read_data_valid	Output		

*continued...*



Signals in Interface	Direction	Availability	Description
			EMIF Debug Toolkit, connect this interface to the Avalon-MM master of the custom debug logic. When in daisy-chaining mode, ensure one of the connected Avalon masters is either the <b>Stratix 10 EMIF Debug Component</b> IP or the external memory interface IP with <b>EMIF Debug Toolkit/On-Chip Debug Port</b> set to <b>Add EMIF Debug Interface</b> .

**Table 296. Interface: cal\_debug\_clk\_clock\_sink**

Interface type: Clock Input

Signals in Interface	Direction	Availability	Description
cal_debug_clk	Input	<ul style="list-style-type: none"><li><b>EMIF Debug Toolkit / On-Chip Debug Port</b>=Export</li></ul>	The calibration debug clock input signal. Connect this interface to the (avl_clk_out) clock output of the <b>Stratix 10 EMIF Debug Component</b> IP or to (cal_debug_out_clk_clock_source) clock input of the other external memory interface IP, depending on which IP the cal_debug_avalon_slave interface is connecting to. If you are not using the Altera EMIF Debug Toolkit, connect this interface to the clock output of the custom debug logic.

**Table 297. Interface: cal\_debug\_out\_avalon\_master**

Interface type: Avalon Memory-Mapped Master

Signals in Interface	Direction	Availability	Description
cal_debug_out_waitrequest	Input	<ul style="list-style-type: none"><li><b>EMIF Debug Toolkit / On-Chip Debug Port</b>=Export</li><li>Add EMIF Debug Interface with <b>Enable Daisy-Chaining for EMIF Debug Toolkit/ On-Chip Debug Port</b>=True</li></ul>	The Avalon-MM signals between the external memory interface IP and the other external memory interface IP. Connect this interface to the (cal_debug_avalon_slave) Avalon-MM Master of the external memory interface IP that has exported the interface .
cal_debug_out_read	Output		
cal_debug_out_write	Output		
cal_debug_out_addr	Output		
cal_debug_out_read_data	Input		
cal_debug_out_write_data	Output		
cal_debug_out_bytelenable	Output		
cal_debug_out_read_data_valid	Input		



**Table 298. Interface: cal\_debug\_out\_clk\_clock\_source**

Interface type: Clock Output

Signals in Interface	Direction	Availability	Description
cal_debug_out_clk	Output	<ul style="list-style-type: none"> <li>• <b>EMIF Debug Toolkit / On-Chip Debug Port=Export</b></li> <li>• Add EMIF Debug Interface with <b>Enable Daisy-Chaining for EMIF Debug Toolkit/ On-Chip Debug Port=True</b></li> </ul>	<p>The calibration debug clock output signal.</p> <p>For <b>EMIF Debug Toolkit/On-Chip Debug Port=Export</b> with <b>Enable Daisy-Chaining for EMIF Debug Toolkit/ On-Chip Debug Port=True</b>, the clock frequency follows the <code>cal_debug_clk</code> frequency. Otherwise, the clock frequency in relation to the memory clock frequency depends on the <b>Clock rate of user logic</b> value set in the parameter editor. Connect this interface to the <code>(cal_debug_out_reset_reset_source)</code> clock input of the other external memory interface IP where the <code>cal_debug_avalon_master</code> interface is being connected to or to any user logic block that needs to be clocked at the generated clock frequency.</p>

**Table 299. Interface: cal\_debug\_out\_reset\_reset\_source**

Interface type: Reset Output

Signals in Interface	Direction	Availability	Description
cal_debug_out_reset_n	Output	<ul style="list-style-type: none"> <li>• <b>EMIF Debug Toolkit / On-Chip Debug Port=Export</b></li> <li>• Add EMIF Debug Interface with <b>Enable Daisy-Chaining for EMIF Debug Toolkit/ On-Chip Debug Port=True</b></li> </ul>	<p>The calibration debug reset output signal. Asynchronous assertion and synchronous deassertion. Connect this interface to the <code>(cal_debug_reset_reset_sink)</code> reset input of the other external memory interface IP where the <code>cal_debug_avalon_master</code> interface being connected to and all the user logic blocks that are under the calibration debug clock domain (<code>cal_debug_out_clk</code> clock reset). If you are not using the Altera EMIF Debug Toolkit, connect this interface to the reset output of the custom debug logic.</p>

**Table 300. Interface: cal\_debug\_reset\_reset\_sink**

Interface type: Reset Input

Signals in Interface	Direction	Availability	Description
cal_debug_reset_n	Input	<ul style="list-style-type: none"> <li>• <b>EMIF Debug Toolkit / On-Chip Debug Port=Export</b></li> </ul>	<p>The calibration debug reset input signal. Require asynchronous assertion and synchronous deassertion. Connect this interface to the <code>(avl_rst_out)</code> reset output of the <b>Stratix 10 EMIF Debug Component</b> IP or to <code>(cal_debug_out_reset_reset</code></p>



Signals in Interface	Direction	Availability	Description
			_source) clock input of the other external memory interface IP, depending on which IP the cal_debug_avalon_slave interface is being connected to.

**Table 301. Interface: clks\_sharing\_master\_out\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
clks_sharing_master_out	Input	• <b>Core clocks sharing</b> =Master	The core clock output signals. Connect this interface to the (clks_sharing_slave_in_conduit_end) conduit of the other external memory interface IP with the Core clock sharing set to slave or other PLL Slave.

**Table 302. Interface: clks\_sharing\_slave\_in\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
clks_sharing_slave_in	Input	• <b>Core clocks sharing</b> =Slave	The core clock input signals. Connect this interface to the (clks_sharing_master_out_conduit_end) conduit of the other external memory interface IP with the Core clock sharing set to Master or other PLL Master.

**Table 303. Interface: ctrl\_amm\_avalon\_slave**

Interface type: Avalon Memory-Mapped Slave

Signals in Interface	Direction	Availability	Description
amm_ready	Output	• DDR3, DDR4 with Hard PHY & Hard Controller • QDR II/II+/II+ Xtreme, QDR IV	The Avalon-MM signals between the external memory interface IP and the user logic. Connect this interface to the Avalon-MM Master of the user logic that needs to access the external memory device. For QDR II/II+/II+ Xtreme, connect the ctrl_amm_avalon_slave_0 to the user logic for read request and connect the ctrl_amm_avalon_slave_1 to the user logic for write request. In Ping Pong PHY mode, each interface controls only one memory device. Connect ctrl_amm_avalon_slave_0 to the user logic that will access the first memory device, and connect ctrl_amm_avalon_slave_1 to the user logic that will access the secondary memory device.
amm_read	Input		
amm_write	Input		
amm_address	Input		
amm_readdata	Output		
amm_writedata	Input		
amm_burstcount	Input		
amm_readdatavalid	Output		
amm_bytetable	Input	• DDR3, DDR4 with Hard PHY & Hard Controller and Enable DM pins=True • QDR II/II+/II+ Xtreme with Enable BWS# pins=True	



**Table 304. Interface: ctrl\_auto\_purge\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
ctrl_auto_purge_req	Input	<ul style="list-style-type: none"> <li>DDR3, DDR4 with Hard PHY &amp; Hard Controller and <b>Enable Auto-Purge Control</b>=True</li> </ul>	The auto-purge control input signal. Asserting the <code>ctrl_auto_purge_req</code> signal while issuing a read or write burst instructs the external memory interface IP to issue read or write with auto-purge to the external memory device. This purges the row immediately after the command currently accessing it finishes, potentially speeding up a future access to a different row of the same bank. Connect this interface to the conduit of the user logic block that controls when the external memory interface IP needs to issue read or write with auto-purge to the external memory device.

**Table 305. Interface: ctrl\_ecc\_user\_interrupt\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
ctrl_ecc_user_interrupt	Output	<ul style="list-style-type: none"> <li>DDR3, DDR4 with Hard PHY &amp; Hard Controller and <b>Enable Error Detection and Correction Logic</b> = True</li> </ul>	Controller ECC user interrupt interface for connection to a custom control block that must be notified when ECC errors occur.

**Table 306. Interface: ctrl\_mmr\_avalon\_slave**

Interface type: Avalon Memory-Mapped Slave

Signals in Interface	Direction	Availability	Description
mmr_waitrequest	Output	<ul style="list-style-type: none"> <li>DDR3, DDR4, LPDDR3 with Hard PHY &amp; Hard Controller and <b>Enable Memory-Mapped Configuration and Status Register (MMR)</b>=True</li> </ul>	The Avalon-MM signals between the external memory interface IP and the user logic. Connect this interface to the Avalon-MM master of the user logic that needs to access the Memory-Mapped Configuration and Status Register (MMR) in the external memory interface IP.
mmr_read	Input		
mmr_write	Input		
mmr_address	Input		
mmr_readdata	Output		
mmr_writedata	Input		
mmr_burstcount	Input		
mmr_bytelenable	Input		
mmr_beginbursttransfer	Input		
mmr_readdatavalid	Output		

**Table 307. Interface: ctrl\_power\_down\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
ctrl_power_down_ack	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3 with Hard PHY &amp; Hard Controller and <b>Enable Auto Power Down=True</b></li></ul>	The auto power-down acknowledgment signals. When the <code>ctrl_power_down_ack</code> signal is asserted, it indicates that the external memory interface IP is placing the external memory device into power-down mode. Connect this interface to the conduit of the user logic block that requires the auto power-down status, or leave it unconnected.

**Table 308. Interface: ctrl\_user\_priority\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
ctrl_user_priority_hi	Input	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3 with Hard PHY &amp; Hard Controller</li><li>Avalon Memory-Mapped and <b>Enable Command Priority Control=true</b></li></ul>	The command priority control input signal. Asserting the <code>ctrl_user_priority_hi</code> signal while issuing a read or write request instructs the external memory interface to treat it as a high-priority command. The external memory interface attempts to execute high-priority commands sooner, to reduce latency. Connect this interface to the conduit of the user logic block that determines when the external memory interface IP treats the read or write request as a high-priority command.

**Table 309. Interface: emif\_usr\_clk\_clock\_source**

Interface type: Clock Output

Signals in Interface	Direction	Availability	Description
emif_usr_clk	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3, with Hard PHY &amp; Hard Controller</li><li>QDR II/II+/II+ Xtreme</li><li>QDR IV</li></ul>	The user clock output signal. The clock frequency in relation to the memory clock frequency depends on the <b>Clock rate of user logic</b> value set in the parameter editor. Connect this interface to the clock input of the respective user logic connected to the <code>ctrl_amm_avalon_slave_0</code> interface, or to any user logic block that must be clocked at the generated clock frequency.



**Table 310. Interface: emif\_usr\_reset\_reset\_source**

Interface type: Reset Output

Signals in Interface	Direction	Availability	Description
emif_usr_reset_n	Output	<ul style="list-style-type: none"> <li>DDR3, DDR4, LPDDR3 with Hard PHY &amp; Hard Controller</li> <li>QDR II/II+/II+ Xtreme</li> <li>QDR IV</li> </ul>	<p>The user reset output signal. Asserted when the PLL becomes unlocked or the PHY is reset. Asynchronous assertion and synchronous deassertion.</p> <p>Connect this interface to the clock input of the respective user logic connected to the <code>ctrl_amm_avalon_slave_0</code> interface, or to any user logic block that must be clocked at the generated clock frequency.</p>

**Table 311. Interface: emif\_usr\_clk\_sec\_clock\_source**

Interface type: Clock Output

Signals in Interface	Direction	Availability	Description
emif_usr_clk_sec	Output	<ul style="list-style-type: none"> <li>DDR3, DDR4, with Ping Pong PHY</li> </ul>	<p>The user clock output signal. The clock frequency in relation to the memory clock frequency depends on the <b>Clock rate of user logic</b> value set in the parameter editor.</p> <p>Connect this interface to the clock input of the respective user logic connected to the <code>ctrl_amm_avalon_slave_1</code> interface, or to any user logic block that must be clocked at the generated clock frequency.</p>

**Table 312. Interface: emif\_usr\_reset\_sec\_reset\_source**

Interface type: Reset Output

Signals in Interface	Direction	Availability	Description
emif_usr_reset_n_sec	Output	<ul style="list-style-type: none"> <li>DDR3, DDR4, with Ping Pong PHY</li> </ul>	<p>The user reset output signal. Asserted when the PLL becomes unlocked or the PHY is reset. Asynchronous assertion and synchronous deassertion.</p> <p>Connect this interface to the clock input of the respective user logic connected to the <code>ctrl_amm_avalon_slave_1</code> interface, or to any user logic block that must be clocked at the generated clock frequency.</p>

**Table 313. Interface: global\_reset\_reset\_sink**

Interface type: Reset Input

Signals in Interface	Direction	Availability	Description
global_reset_n	Input	<ul style="list-style-type: none"> <li><b>Core Clock Sharing</b>=No Sharing / Master</li> </ul>	<p>The global reset input signal. Asserting the <code>global_reset_n</code> signal causes the external memory interface IP to be reset and recalibrated.</p>



Signals in Interface	Direction	Availability	Description
			Connect this interface to the reset output of the asynchronous or synchronous reset source that controls when the external memory interface IP needs to be reset and recalibrated.

**Table 314. Interface: mem\_conduit\_end**

Interface type: Conduit

The memory interface signals between the external memory interface IP and the external memory device.

Export this interface to the top level for I/O assignments. Typically `mem_rm[0]` and `mem_rm[1]` connect to CS2# and CS3# of the memory buffer of all LRDIMM slots.

Signals in Interface	Direction	Availability
mem_ck	Output	Always available
mem_ck_n	Output	
mem_reset_n	Output	
mem_a	Output	
mem_k_n	Output	<ul style="list-style-type: none"><li>QDR II</li></ul>
mem_ras_n	Output	<ul style="list-style-type: none"><li>DDR3</li></ul>
mem_cas_n	Output	
mem_odt	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3</li></ul>
mem_dqs	Bidirectional	
mem_dqs_n	Bidirectional	
mem_ba	Output	
mem_cs_n	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3, RLDRAM 3</li></ul>
mem_dq	Bidirectional	
mem_we_n	Output	<ul style="list-style-type: none"><li>DDR3, RLDRAM 3</li></ul>
mem_dm	Output	<ul style="list-style-type: none"><li>DDR3, LPDDR3, RLDRAM 3 with <b>Enable DM pins=True</b></li></ul>
mem_rm	Output	<ul style="list-style-type: none"><li>DDR3, RLDRAM 3 with <b>Memory format=LRDIMM</b> and <b>Number of rank multiplication pins &gt; 0</b></li></ul>
mem_par	Output	<ul style="list-style-type: none"><li>DDR3 with <b>Memory format=RDIMM / LRDIMM</b></li><li>DDR4 with <b>Enable alert_n/par pins=True</b></li></ul>
mem_alert_n	Input	
mem_cke	Output	<ul style="list-style-type: none"><li>DDR3, DDR4, LPDDR3</li></ul>
mem_bg	Output	<ul style="list-style-type: none"><li>DDR4</li></ul>
mem_act_n	Output	
mem_db1_n	Bidirectional	<ul style="list-style-type: none"><li>DDR4 with <b>Enable DM pins=True</b> or <b>Write DBI=True</b> or <b>Read DBI=True</b></li></ul>
mem_k	Output	<ul style="list-style-type: none"><li>QDR II/II+/II+ Xtreme</li></ul>
mem_wps_n	Output	
mem_rps_n	Output	

*continued...*



Signals in Interface	Direction	Availability
mem_doff_n	Output	
mem_d	Output	
mem_q	Input	
mem_cq	Input	
mem_cq_n	Input	
mem_bws_n	Output	
mem_dk	Output	
mem_dk_n	Output	
mem_ref_n	Output	
mem_qk	Input	<ul style="list-style-type: none"> <li>QDR II/II+/II+ Xtreme with <b>Enable BWS# pins=True</b></li> </ul>
mem_qk_n	Input	<ul style="list-style-type: none"> <li>RDRAM 3</li> </ul>
mem_ap	Output	<ul style="list-style-type: none"> <li>QDR IV with <b>Use Address Parity Bit=True</b></li> </ul>
mem_pe_n	Input	<ul style="list-style-type: none"> <li>QDR IV with <b>Use Address Parity Bit=True</b></li> </ul>
mem_ainv	Output	<ul style="list-style-type: none"> <li>QDR IV with <b>Address Bus Inversion=True</b></li> </ul>
mem_lda_n	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_lda_b	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_rwa_n	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_rwb_n	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_cfg_n	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_lbk0_n	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_lbk1_n	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_dka	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_dka_n	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_dkb	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_dkb_n	Output	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_qka	Input	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_qka_n	Input	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_qkb	Input	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_qkb_n	Input	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_dqa	Bidirectional	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_dqb	Bidirectional	<ul style="list-style-type: none"> <li>QDR IV</li> </ul>
mem_dinva	Bidirectional	<ul style="list-style-type: none"> <li>QDR IV with <b>Data Bus Inversion=True</b></li> </ul>
mem_dinvb	Bidirectional	<ul style="list-style-type: none"> <li>QDR IV with <b>Data Bus Inversion=True</b></li> </ul>

**Table 315. Interface: oct\_conduit\_end**

Interface type: Conduit

Signals in Interface	Direction	Availability	Description
oct_rzqin	Input	Always available	The On-Chip Termination (OCT) RZQ reference resistor input signal. Export this interface to the top level for I/O assignments.

**Table 316. Interface: pll\_ref\_clk\_clock\_sink**

Signals in Interface	Interface Type	Direction	Availability	Description
pll_ref_clk	Clock Input	Input	• <b>Core clock sharing</b> =No Sharing / Master	The PLL reference clock input signal. Connect this interface to the clock output of the clock source that matches the <b>PLL reference clock frequency</b> value set in the parameter editor.

**Table 317. Interface: status\_conduit\_end**

Signals in Interface	Interface Type	Direction	Availability	Description
local_cal_success	Conduit	Output	Always available	The PHY calibration status output signals. When the local_cal_success signal is asserted, it indicates that the PHY calibration was successful. Otherwise, if local_cal_fail signal is asserted, it indicates that PHY calibration has failed. Connect this interface to the conduit of the user logic block that requires the calibration status information, or leave it unconnected.
local_cal_fail				

### 7.5.2. Generated Files for Stratix 10 External Memory Interface IP

When you complete the IP generation flow, there are generated files created in your project directory. The directory structure created varies somewhat, depending on the tool used to parameterize and generate the IP.

**Note:** The PLL parameters are statically defined in the `<variation_name>_parameters.tcl` at generation time. To ensure timing constraints and timing reports are correct, when you edit the PLL parameters, apply those changes to the PLL parameters in this file.

The following table lists the generated directory structure and key files created when generating the IP.



**Table 318. Generated Directory Structure and Key Files for Synthesis**

Directory	File Name	Description
<code>working_dir/</code>	<code>working_dir/&lt;Top-level Name&gt;/</code>	The Qsys files for your IP component or system based on your configuration.
<code>working_dir/&lt;Top-level Name&gt;/</code>	<code>*.ppf</code>	Pin Planner File for use with the Pin Planner.
<code>working_dir/&lt;Top-level Name&gt;/synth/</code>	<code>&lt;Top-level Name&gt;.v or &lt;Top-level Name&gt;.vhd</code>	Qsys generated top-level wrapper for synthesis.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_S10&lt;acds version&gt;/synth/</code>	<code>*.v or (*.v and *.vhd)</code>	Stratix 10 EMIF (non-HPS) top-level dynamic wrapper files for synthesis. This wrapper instantiates the EMIF ECC and EMIF Debug Interface IP core.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_s10_hps_&lt;acds version&gt;/synth/</code>	<code>*.v or (*.v and *.vhd)</code>	Stratix 10 EMIF for HPS top-level dynamic wrapper files for synthesis.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_arch_nd_&lt;acds version&gt;/synth/</code>	<code>*.sv, *.sdc, *.tcl and *.hex and *_readme.txt</code>	Stratix 10 EMIF Core RTL, constraints files, ROM content files and information files for synthesis. Whether the file type is set to Verilog or VHDL, all the Stratix 10 EMIF Core RTL files will be generated as a SystemVerilog file. The <code>readme.txt</code> file contains information and guidelines specific to your configuration.
<code>working_dir/&lt;Top-level Name&gt;/&lt;other components&gt;_&lt;acds version&gt;/synth/</code>	<code>*</code>	Other EMIF ECC, EMIF Debug Interface IP or Merlin Interconnect component files for synthesis.

**Table 319. Generated Directory Structure and Key Files for Simulation**

Directory	File Name	Description
<code>working_dir/&lt;Top-level Name&gt;/sim/</code>	<code>&lt;Top-level Name&gt;.v or &lt;Top-level Name&gt;.vhd</code>	Qsys generated top-level wrapper for simulation.
<code>working_dir/&lt;Top-level Name&gt;/sim/&lt;simulator vendor&gt;/</code>	<code>*.tcl, *cds.lib, *.lib, *.var, *.sh, *.setup</code>	Simulator-specific simulation scripts.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_s10&lt;acds version&gt;/sim/</code>	<code>*.v or *.vhd</code>	Stratix 10 EMIF (non-HPS) top-level dynamic wrapper files for simulation. This wrapper instantiates the EMIF ECC and EMIF Debug Interface IP cores.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_s10_hps_&lt;acds version&gt;/sim/</code>	<code>*.v or *.vhd</code>	Stratix 10 EMIF for HPS top-level dynamic wrapper files for simulation.
<code>working_dir/&lt;Top-level Name&gt;/altera_emif_arch_nd_&lt;acds version&gt;/sim/</code>	<code>*.sv or (*.sv and *.vhd), *.hex and *_readme.txt</code>	Stratix 10 EMIF RTL, ROM content files, and information files for simulation. For SystemVerilog / Mix language simulator, you may directly use the files from this folder. For VHDL-only simulator, other than the ROM content files, you have to use files in <code>&lt;current folder&gt;/mentor</code> directory instead. The <code>readme.txt</code> file contains information and guidelines specific to your configuration.

*continued...*



Directory	File Name	Description
<code>working_dir/&lt;Top-level Name&gt;/&lt;other components&gt;_&lt;acds version&gt;/sim/</code>		Other EMIF ECC, EMIF Debug Interface IP, or Merlin Interconnect component files for simulation

**Table 320. Generated Directory Structure and Key Files for Qsys-Generated Testbench System**

Directory	File Name	Description
<code>working_dir/&lt;Top-level Name&gt;_tb/</code>	<code>*.qsys</code>	The Qsys files for the QSYS generated testbench system.
<code>working_dir/&lt;Top-level Name&gt;_tb/sim/</code>	<code>&lt;Top-level Name&gt;.v</code> or <code>&lt;Top-level Name&gt;.vhdl</code>	Qsys generated testbench file for simulation. This wrapper instantiates BFM components. For Stratix 10 EMIF IP, this module should instantiate the memory model for the memory conduit being exported from your created system.
<code>working_dir/&lt;Top-level Name&gt;_tb/&lt;Top-level Name&gt;_&lt;id&gt;/sim/</code>	<code>&lt;Top-level Name&gt;.v</code> or <code>&lt;Top-level Name&gt;.vhdl</code>	Qsys generated top-level wrapper for simulation.
<code>working_dir/&lt;Top-level Name&gt;_tb/sim/&lt;simulator vendor&gt;/</code>	<code>*.tcl, *cds.lib, *.lib, *.var, *.sh, *.setup</code>	Simulator-specific simulation scripts.
<code>working_dir/&lt;Top-level Name&gt;_tb/sim/&lt;simulator vendor&gt;/</code>	<code>*.v</code> or <code>*.vhdl</code>	Stratix 10 EMIF (non-HPS) top-level dynamic wrapper files for simulation. This wrapper instantiates the EMIF ECC and EMIF Debug Interface IP cores.
<code>working_dir/&lt;Top-level Name&gt;_tb/altera_emif_a10_hps_&lt;acds version&gt;/sim/</code>	<code>*.v</code> or <code>*.vhdl</code>	Stratix 10 EMIF for HPS top-level dynamic wrapper files for simulation.
<code>working_dir/&lt;Top-level Name&gt;_tb/altera_emif_arch_nf_&lt;acds version&gt;/sim/</code>	<code>*sv or (*.sv and *.vhdl), *.hex and *_readme.txt</code>	Stratix 10 EMIF Core RTL, ROM content files and information files for simulation. For SystemVerilog / Mix language simulator, you may use the files from this folder. For VHDL-only simulator, other than the ROM content files, you must use files in the <code>&lt;current folder&gt;/mentor</code> directory instead. The <code>readme.txt</code> file contains information and guidelines specific to your configuration.
<code>working_dir/&lt;Top-level Name&gt;_tb/sim/altera_emif_arch_nf_&lt;acds version&gt;/sim/mentor/</code>	<code>*.sv</code> and <code>*.vhdl</code>	Stratix 10 EMIF Core RTL for simulation.

*continued...*



Directory	File Name	Description
		Only available when you create a VHDL simulation model. All .sv files are Mentor-tagged encrypted IP (IEEE Encrypted Verilog) for VHDL-only simulator support.
<code>working_dir/&lt;Top-level Name&gt;_tb/&lt;other components&gt;_&lt;acds version&gt;/sim/</code>	*	Other EMIF ECC, EMIF Debug Interface IP or Merlin Interconnect component files for simulation.
<code>working_dir/&lt;Top-level Name&gt;_tb/&lt;other components&gt;_&lt;acds version&gt;/sim/mentor/</code>	*	Other EMIF ECC, EMIF Debug Interface IP or Merlin Interconnect component files for simulation. Only available depending on individual component simulation model support and when creating a VHDL simulation model. All files in this folder are Mentor-tagged encrypted IP (IEEE Encrypted Verilog) for VHDL-only simulator support.

**Table 321. Generated Directory Structure and Key Files for Example Simulation Designs**

Directory	File Name	Description
<code>working_dir/*_example_design*/</code>	<code>*.qsys, *.tcl</code> and <code>readme.txt</code>	Qsys files, generation scripts, and information for generating the Stratix 10 EMIF IP example design. These files are available only when you generate an example design. You may open the .qsys file in Qsys to add more components to the example design.
<code>working_dir/*_example_design*/sim/ed_sim/sim/</code>	<code>*.v</code> or <code>*.vhd</code>	Qsys-generated top-level wrapper for simulation.
<code>working_dir/*_example_design*/sim/ed_sim/&lt;simulator vendor&gt;/</code>	<code>*.tcl, *cds.lib, *.lib, *.var, *.sh, *.setup</code>	Simulator-specific simulation scripts.
<code>working_dir/*_example_design*/sim/ip/ed_sim/ed_sim_emif_s10_0/altera_emif_s10_&lt;acds_version&gt;/sim/</code>	<code>*.v</code> or <code>*.vhd</code>	Stratix 10 EMIF (non-HPS) top-level dynamic wrapper files for simulation. This wrapper instantiates the EMIF ECC and EMIF Debug Interface IP cores.
<code>working_dir/*_example_design*/sim/ip/ed_sim/ed_sim_emif_s10_0/altera_emif_arch_nd_&lt;acds_version&gt;/sim/</code>	<code>*sv</code> or <code>(*.sv and *.vhd)</code> , <code>*.hex</code> and <code>*_readme.txt</code>	Stratix 10 EMIF RTL, ROM content files, and information files for simulation. For SystemVerilog / Mix language simulator, you may directly use the files from this folder. For VHDL-only simulator, other than the ROM content files, you have to use files in <code>&lt;current folder&gt;/mentor</code> directory instead. The <code>readme.txt</code> file contains information and guidelines specific to your configuration.
<code>working_dir/*_example_design*/sim/ed_sim/&lt;other components&gt;_&lt;acds_version&gt;/sim/</code>	*	Other EMIF ECC, EMIF Debug Interface IP, or Merlin Interconnect component files for simulation

*continued...*



Directory	File Name	Description
and <i>working_dir/</i> * <i>_example_design*</i> /sim/ip/ ed_sim/ <other_components>/sim/ and <i>working_dir/</i> * <i>_example_design*</i> /sim/ip/ ed_sim/<other_components>/ <other_components>_ <i>&lt;acds_version&gt;</i> /sim/		

**Table 322. Generated Directory Structure and Key Files for Example Synthesis Designs**

Directory	File Name	Description
<i>working_dir/</i> * <i>_example_design*</i> /	*.qsys, *.tcl and readme.txt	Qsys files, generation scripts, and information for generating the Stratix 10 EMIF IP example design. These files are available only when you generate an example design. You may open the .qsys file in Qsys to add more components to the example design.
<i>working_dir/</i> * <i>_example_design*</i> /qii/ ed_synth/synth	*.v or (*.v and *.vhd)	Qsys-generated top-level wrapper for synthesis.
<i>working_dir/</i> * <i>_example_design*</i> /qii/ip/ ed_synth/ end_synth_emif_s10_0/ altera_emif_s10_<i> <i>&lt;acds_version&gt;</i> </i>/synth	*.v or (*.v and *.vhd)	Stratix 10 EMIF (non HPS) top-level dynamic wrapper files for synthesis. This wrapper instantiates the EMIF ECC and EMIF debug interface core IP.
<i>working_dir/</i> * <i>_example_design*</i> /qii/ip/ ed_synth/ed_synth_emif_s10_0/ altera_emif_arch_nd_<i> <i>&lt;acds_version&gt;</i> </i>/synth/	*.sv, *.sdc, *.tcl and *.hex and *_readme.txt	Stratix 10 EMIF Core RTL, constraints files, ROM content files and information files for synthesis. Whether the file type is set to Verilog or VHDL, all the Stratix 10 EMIF Core RTL files will be generated as a System verilog file. The readme.txt file contains information and guidelines specific to your configuration.
<i>working_dir/</i> * <i>_example_design*</i> /qii/ ed_synth/<other components>_ <i>&lt;acds_version&gt;</i> / synth and <i>working_dir/</i> * <i>_example_design*</i> /qii/ip/ ed_synth/<other components>/ synth and <i>working_dir/</i> * <i>_example_design*</i> /qii/ip/ ed_synth/<other components>/ <other components>_ <i>&lt;acds_version&gt;</i> / synth	*	Other EMIF ECC, EMIF debug interface IP, or Merlin interconnect component files for synthesis.



### 7.5.3. Stratix 10 EMIF IP DDR4 Parameters

The Stratix 10 EMIF IP parameter editor allows you to parameterize settings for the Stratix 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Stratix 10 EMIF IP.

#### 7.5.3.1. Stratix 10 EMIF IP DDR4 Parameters: General

**Table 323. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 324. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 325. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out

*continued...*

<b>Display Name</b>	<b>Identifier</b>	<b>Description</b>
		signal from the master interface to the <code>clks_sharing_slave_in</code> signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, <code>emif_usr_clk</code> , <code>afi_clk</code> ), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Use recommended PLL reference clock frequency</b>	<code>PHY_DDR4_DEFAULT_REF_CLK_FREQ</code>	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Memory clock frequency</b>	<code>PHY_MEM_CLK_FREQ_MHZ</code>	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Clock rate of user logic</b>	<code>PHY_RATE_ENUM</code>	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	<code>PHY_REF_CLK_FREQ_MHZ</code>	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	<code>PHY_REF_CLK_JITTER_PS</code>	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	<code>PLL_ADD_EXTRA_CLKS</code>	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as <code>emif_usr_clk</code> or <code>afi_clk</code> ). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 326. Group: General / Additional Core Clocks**

<b>Display Name</b>	<b>Identifier</b>	<b>Description</b>
<b>Number of additional core clocks</b>	<code>PLL_USER_NUM_OF_EXTRA_CLKS</code>	Specifies the number of additional output clocks to create from the PLL.

### 7.5.3.2. Stratix 10 EMIF IP DDR4 Parameters: Memory



**Table 327. Group: Memory / Topology**

Display Name	Identifier	Description
<b>DQS group of ALERT#</b>	MEM_DDR4_ALERT_N_DQS_GROUP	Select the DQS group with which the ALERT# pin is placed.
<b>ALERT# pin placement</b>	MEM_DDR4_ALERT_N_PLACEMENT_ENUM	Specifies placement for the mem_alert_n signal. If you select "I/O Lane with Address/Command Pins", you can pick the I/O lane and pin index in the add/cmd bank with the subsequent drop down menus. If you select "I/O Lane with DQS Group", you can specify the DQS group with which to place the mem_alert_n pin. If you select "Automatically select a location", the IP automatically selects a pin for the mem_alert_n signal. If you select this option, no additional location constraints can be applied to the mem_alert_n pin, or a fitter error will result during compilation. For optimum signal integrity, you should choose "I/O Lane with Address/Command Pins". For interfaces containing multiple memory devices, it is recommended to connect the ALERT# pin on the FPGA together to the ALERT# pin on the FPGA.
<b>Enable ALERT#/PAR pins</b>	MEM_DDR4_ALERT_PAR_EN	Allows address/command calibration, which may provide better margins on the address/command bus. The alert_n signal is not accessible in the AFI or Avalon domains. This means there is no way to know whether a parity error has occurred during user mode. The parity pin is a dedicated pin in the address/command bank, but the alert_n pin can be placed in any bank that spans the memory interface. You should explicitly choose the location of the alert_n pin and place it in the address/command bank.
<b>Bank address width</b>	MEM_DDR4_BANK_ADDRESS_WIDTH	Specifies the number of bank address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of bank address pins needed for access to all available banks.
<b>Bank group width</b>	MEM_DDR4_BANK_GROUP_WIDTH	Specifies the number of bank group pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of bank group pins needed for access to all available bank groups.
<b>Chip ID width</b>	MEM_DDR4_CHIP_ID_WIDTH	Specifies the number of chip ID pins. Only applicable to registered and load-reduced DIMMs that use 3DS/TSV memory devices.
<b>Number of clocks</b>	MEM_DDR4_CK_WIDTH	Specifies the number of CK/CK# clock pairs exposed by the memory interface. Usually more than 1 pair is required for RDIMM/LRDIMM formats. The value of this parameter depends on the memory device selected; refer to the data sheet for your memory device.
<b>Column address width</b>	MEM_DDR4_COL_ADDRESS_WIDTH	Specifies the number of column address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of address pins needed for access to all available columns.
<b>Number of chip selects per DIMM</b>	MEM_DDR4_CS_PER_DIMM	Specifies the number of chip selects per DIMM.
<b>Number of chip selects</b>	MEM_DDR4_DISCRETE_CS_WIDTH	Specifies the total number of chip selects in the interface, up to a maximum of 4. This parameter applies to discrete components only.
<b>Data mask</b>	MEM_DDR4_DM_EN	Indicates whether the interface uses data mask (DM) pins. This feature allows specified portions of the data bus to be written to memory (not available in x4 mode). One DM pin exists per DQS group.

*continued...*



Display Name	Identifier	Description
<b>Number of DQS groups</b>	MEM_DDR4_DQS_WIDTH	Specifies the total number of DQS groups in the interface. This value is automatically calculated as the DQ width divided by the number of DQ pins per DQS group.
<b>DQ pins per DQS group</b>	MEM_DDR4_DQ_PER_DQS	Specifies the total number of DQ pins per DQS group.
<b>DQ width</b>	MEM_DDR4_DQ_WIDTH	Specifies the total number of data pins in the interface. The maximum supported width is 144, or 72 in Ping Pong PHY mode.
<b>Memory format</b>	MEM_DDR4_FORMAT_ENUM	Specifies the format of the external memory device. The following formats are supported: Component - a Discrete memory device; UDIMM - Unregistered/Unbuffered DIMM where address/control, clock, and data are unbuffered; RDIMM - Registered DIMM where address/control and clock are buffered; LRDIMM - Load Reduction DIMM where address/control, clock, and data are buffered. LRDIMM reduces the load to increase memory speed and supports higher densities than RDIMM; SODIMM - Small Outline DIMM is similar to UDIMM but smaller in size and is typically used for systems with limited space. Some memory protocols may not be available in all formats.
<b>Number of DIMMs</b>	MEM_DDR4_NUM_OF_DIMMS	Total number of DIMMs.
<b>Number of physical ranks per DIMM</b>	MEM_DDR4_RANKS_PER_DIMM	Number of ranks per DIMM. For LRDIMM, this represents the number of physical ranks on the DIMM behind the memory buffer
<b>Read DBI</b>	MEM_DDR4_READ_DB_I	Specifies whether the interface uses read data bus inversion (DBI). Enable this feature for better signal integrity and read margin. This feature is not available in x4 configurations.
<b>Row address width</b>	MEM_DDR4_ROW_ADDRESS_WIDTH	Specifies the number of row address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of address pins needed for access to all available rows.
<b>Write DBI</b>	MEM_DDR4_WRITE_DB_I	Indicates whether the interface uses write data bus inversion (DBI). This feature provides better signal integrity and write margin. This feature is unavailable if Data Mask is enabled or in x4 mode.

**Table 328. Group: Memory / Latency and Burst**

Display Name	Identifier	Description
<b>Addr/CMD parity latency</b>	MEM_DDR4_AC_PARITY_LATENCY	Additional latency incurred by enabling address/command parity check. Select a value to enable address/command parity with the latency associated with the selected value. Select Disable to disable address/command parity.
<b>Memory additive CAS latency setting</b>	MEM_DDR4_ATCL_ENUM	Determines the posted CAS additive latency of the memory device. Enable this feature to improve command and bus efficiency, and increase system bandwidth.
<b>Burst Length</b>	MEM_DDR4_BL_ENUM	Specifies the DRAM burst length which determines how many consecutive addresses should be accessed for a given read/write command.

*continued...*



Display Name	Identifier	Description
<b>Read Burst Type</b>	MEM_DDR4_BT_ENUM	Indicates whether accesses within a given burst are in sequential or interleaved order. Select sequential if you are using the Intel-provided memory controller.
<b>Memory CAS latency setting</b>	MEM_DDR4_TCL	Specifies the number of clock cycles between the read command and the availability of the first bit of output data at the memory device. Overall read latency equals the additive latency (AL) + the CAS latency (CL). Overall read latency depends on the memory device selected; refer to the datasheet for your device.
<b>Memory write CAS latency setting</b>	MEM_DDR4_WTCL	Specifies the number of clock cycles from the release of internal write to the latching of the first data in at the memory device. This value depends on the memory device selected; refer to the datasheet for your device.

**Table 329. Group: Memory / Mode Register Settings**

Display Name	Identifier	Description
<b>Auto self-refresh method</b>	MEM_DDR4_ASR_ENUM	Indicates whether to enable or disable auto self-refresh. Auto self-refresh allows the controller to issue self-refresh requests, rather than manually issuing self-refresh in order for memory to retain data.
<b>Fine granularity refresh</b>	MEM_DDR4_FINE_GRANULARITY_REFRESH	Increased frequency of refresh in exchange for shorter refresh. Shorter tRFC and increased cycle time can produce higher bandwidth.
<b>Internal VrefDQ monitor</b>	MEM_DDR4_INTERNAL_VREFDQ_MONITOR	Indicates whether to enable the internal VrefDQ monitor.
<b>ODT input buffer during powerdown mode</b>	MEM_DDR4_ODT_IN_POWERDOWN	Indicates whether to enable on-die termination (ODT) input buffer during powerdown mode.
<b>Read preamble</b>	MEM_DDR4_READ_PREAMBLE	Number of read preamble cycles. This mode register setting determines the number of cycles DQS (read) will go low before starting to toggle.
<b>Self refresh abort</b>	MEM_DDR4_SELF_RFSH_ABORT	Self refresh abort for latency reduction.
<b>Temperature controlled refresh enable</b>	MEM_DDR4_TEMP_CONTROLLED_RFSH_ENA	Indicates whether to enable temperature controlled refresh, which allows the device to adjust the internal refresh period to be longer than tREFI of the normal temperature range by skipping external refresh commands.
<b>Temperature controlled refresh range</b>	MEM_DDR4_TEMP_CONTROLLED_RFSH_RANGE	Indicates temperature controlled refresh range where normal temperature mode covers 0C to 85C and extended mode covers 0C to 95C.
<b>Write preamble</b>	MEM_DDR4_WRITE_PREAMBLE	Write preamble cycles.

### 7.5.3.3. Stratix 10 EMIF IP DDR4 Parameters: Mem I/O

**Table 330. Group: Mem I/O / Memory I/O Settings**

Display Name	Identifier	Description
<b>DB Host Interface DQ Driver</b>	MEM_DDR4_DB_DQ_DRV_ENUM	Specifies the driver impedance setting for the host interface of the data buffer. This parameter determines the value of the control word BC03 of the data buffer. Perform board simulation to obtain the optimal value for this setting.
<b>DB Host Interface DQ RTT_NOM</b>	MEM_DDR4_DB_RTT_NOM_ENUM	Specifies the RTT_NOM setting for the host interface of the data buffer. Only "RTT_NOM disabled" is supported. This parameter determines the value of the control word BC00 of the data buffer.
<b>DB Host Interface DQ RTT_PARK</b>	MEM_DDR4_DB_RTT_PARK_ENUM	Specifies the RTT_PARK setting for the host interface of the data buffer. This parameter determines the value of control word BC02 of the data buffer. Perform board simulation to obtain the optimal value for this setting.
<b>DB Host Interface DQ RTT_WR</b>	MEM_DDR4_DB_RTT_WR_ENUM	Specifies the RTT_WR setting of the host interface of the data buffer. This parameter determines the value of the control word BC01 of the data buffer. Perform board simulation to obtain the optimal value for this setting.
<b>Use recommended initial VrefDQ value</b>	MEM_DDR4_DEFAULT_VREFOUT	Specifies to use the recommended initial VrefDQ value. This value is used as a starting point and may change after calibration.
<b>Output drive strength setting</b>	MEM_DDR4_DRV_STR_ENUM	Specifies the output driver impedance setting at the memory device. To obtain optimum signal integrity performance, select option based on board simulation results.
<b>RCD CA Input Bus Termination</b>	MEM_DDR4_RCD_CA_IBT_ENUM	Specifies the input termination setting for the following pins of the registering clock driver: DA0..DA17, DBA0..DBA1, DBG0..DBG1, DACT_n, DC2, DPAR. This parameter determines the value of bits DA[1:0] of control word RC7x of the registering clock driver. Perform board simulation to obtain the optimal value for this setting.
<b>RCD DCKE Input Bus Termination</b>	MEM_DDR4_RCD_CKE_IBT_ENUM	Specifies the input termination setting for the following pins of the registering clock driver: DCKE0, DCKE1. This parameter determines the value of bits DA[5:4] of control word RC7x of the registering clock driver. Perform board simulation to obtain the optimal value for this setting.
<b>RCD DCS[3:0]_n Input Bus Termination</b>	MEM_DDR4_RCD_CS_IBT_ENUM	Specifies the input termination setting for the following pins of the registering clock driver: DCS[3:0]_n. This parameter determines the value of bits DA[3:2] of control word RC7x of the registering clock driver. Perform board simulation to obtain the optimal value for this setting.
<b>RCD DODT Input Bus Termination</b>	MEM_DDR4_RCD_ODT_IBT_ENUM	Specifies the input termination setting for the following pins of the registering clock driver: DODT0, DODT1. This parameter determines the value of bits DA[7:6] of control word RC7x of the registering clock driver. Perform board simulation to obtain the optimal value for this setting.
<b>ODT Rtt nominal value</b>	MEM_DDR4_RTT_NOM_ENUM	Determines the nominal on-die termination value applied to the DRAM. The termination is applied any time that ODT is asserted. If you specify a different value for RTT_WR, that value takes precedence over the values mentioned here. For optimum signal integrity performance, select your option based on board simulation results.
<b>RTT PARK</b>	MEM_DDR4_RTT_PAR_K	If set, the value is applied when the DRAM is not being written AND ODT is not asserted HIGH.

*continued...*



Display Name	Identifier	Description
<b>Dynamic ODT (Rtt_WR) value</b>	MEM_DDR4_RTT_WR_ENUM	Specifies the mode of the dynamic on-die termination (ODT) during writes to the memory device (used for multi-rank configurations). For optimum signal integrity performance, select this option based on board simulation results.
<b>RCD and DB Manufacturer (LSB)</b>	MEM_DDR4_SPD_133_RCD_DB_VENDOR_LSB	Specifies the LSB of the ID code of the registering clock driver and data buffer manufacturer. The value must come from Byte 133 of the SPD from the DIMM vendor.
<b>RCD and DB Manufacturer (MSB)</b>	MEM_DDR4_SPD_134_RCD_DB_VENDOR_MSB	Specifies the MSB of the ID code of the registering clock driver and data buffer manufacturer. The value must come from Byte 134 of the SPD from the DIMM vendor.
<b>RCD Revision Number</b>	MEM_DDR4_SPD_135_RCD_REV	Specifies the die revision of the registering clock driver. The value must come from Byte 135 of the SPD from the DIMM vendor.
<b>SPD Byte 137 - RCD Drive Strength for Command/Address</b>	MEM_DDR4_SPD_137_RCD_CA_DRV	Specifies the drive strength of the registering clock driver's control and command/address outputs to the DRAM. The value must come from Byte 137 of the SPD from the DIMM vendor.
<b>SPD Byte 138 - RCD Drive Strength for CK</b>	MEM_DDR4_SPD_138_RCD_CK_DRV	Specifies the drive strength of the registering clock driver's clock outputs to the DRAM. The value must come from Byte 138 of the SPD from the DIMM vendor.
<b>DB Revision Number</b>	MEM_DDR4_SPD_139_DB_REV	Specifies the die revision of the data buffer. The value must come from Byte 139 of the SPD from the DIMM vendor.
<b>SPD Byte 140 - DRAM VrefDQ for Package Rank 0</b>	MEM_DDR4_SPD_140_DRAM_VREFDQ_R0	Specifies the VrefDQ setting for package rank 0 of an LRDIMM. The value must come from Byte 140 of the SPD from the DIMM vendor.
<b>SPD Byte 141 - DRAM VrefDQ for Package Rank 1</b>	MEM_DDR4_SPD_141_DRAM_VREFDQ_R1	Specifies the VrefDQ setting for package rank 1 of an LRDIMM. The value must come from Byte 141 of the SPD from the DIMM vendor.
<b>SPD Byte 142 - DRAM VrefDQ for Package Rank 2</b>	MEM_DDR4_SPD_142_DRAM_VREFDQ_R2	Specifies the VrefDQ setting for package rank 2 (if it exists) of an LRDIMM. The value must come from Byte 142 of the SPD from the DIMM vendor.
<b>SPD Byte 143 - DRAM VrefDQ for Package Rank 3</b>	MEM_DDR4_SPD_143_DRAM_VREFDQ_R3	Specifies the VrefDQ setting for package rank 3 (if it exists) of an LRDIMM. The value must come from Byte 143 of the SPD from the DIMM vendor.
<b>SPD Byte 144 - DB VrefDQ for DRAM Interface</b>	MEM_DDR4_SPD_144_DB_VREFDQ	Specifies the VrefDQ setting of the data buffer's DRAM interface. The value must come from Byte 144 of the SPD from the DIMM vendor.
<b>SPD Byte 145-147 - DB MDQ Drive Strength and RTT</b>	MEM_DDR4_SPD_145_DB_MDQ_DRV	Specifies the drive strength of the MDQ pins of the data buffer's DRAM interface. The value must come from either Byte 145 (data rate = 1866), 146 (1866 data rate = 2400), or 147 (2400 data rate = 3200) of the SPD from the DIMM vendor.
<b>SPD Byte 148 - DRAM Drive Strength</b>	MEM_DDR4_SPD_148_DRAM_DRV	Specifies the drive strength of the DRAM. The value must come from Byte 148 of the SPD from the DIMM vendor.
<b>SPD Byte 149-151 - DRAM ODT (RTT_WR and RTT_NOM)</b>	MEM_DDR4_SPD_149_DRAM_RTT_WR_NOM	Specifies the RTT_WR and RTT_NOM setting of the DRAM. The value must come from either Byte 149 (data rate = 1866), 150 (1866 data rate = 2400), or 151 (2400 data rate = 3200) of the SPD from the DIMM vendor.

*continued...*



Display Name	Identifier	Description
<b>SPD Byte 152-154 - DRAM ODT (RTT_PARK)</b>	MEM_DDR4_SPD_152_DRAM_RTT_PARK	Specifies the RTT_PARK setting of the DRAM. The value must come from either Byte 152 (data rate = 1866), 153 (1866 data rate = 2400), or 154 (2400 data rate = 3200) of the SPD from the DIMM vendor.
<b>VrefDQ training range</b>	MEM_DDR4_VREFDQ_TRAINING_RANGE	VrefDQ training range.
<b>VrefDQ training value</b>	MEM_DDR4_VREFDQ_TRAINING_VALUE	VrefDQ training value.

**Table 331. Group: Mem I/O / ODT Activation**

Display Name	Identifier	Description
<b>Use Default ODT Assertion Tables</b>	MEM_DDR4_USE_DEFAULT_ODT	Enables the default ODT assertion pattern as determined from vendor guidelines. These settings are provided as a default only; you should simulate your memory interface to determine the optimal ODT settings and assertion patterns.

#### 7.5.3.4. Stratix 10 EMIF IP DDR4 Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 332. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_DDR4_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_DDR4_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.



**Table 333. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_DDR4_USER_AC_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR4_USER_AC_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_DDR4_USER_AC_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 334. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_DDR4_USER_CK_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR4_USER_CK_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_DDR4_USER_CK_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 335. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_DDR4_USER_AUTO_STARTING_VREFIN_N_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_DDR4_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_DDR4_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR4_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_DDR4_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by

*continued...*



Display Name	Identifier	Description
		calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 336. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_DDR4_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_DDR4_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.5.3.5. Stratix 10 EMIF IP DDR4 Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 337. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_DDR4_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>TdiVW_total</b>	MEM_DDR4_TDIVW_TOTAL_UI	TdiVW_total describes the minimum horizontal width of the DQ eye opening required by the receiver (memory device/ DIMM). It is measured in UI (1UI = half the memory clock period).
<b>tDQSCK</b>	MEM_DDR4_TDQSCK_PS	tDQSCK describes the skew between the memory clock (CK) and the input data strobes (DQS) used for reads. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDQSQ</b>	MEM_DDR4_TDQSQ_U_I	tDQSQ describes the latest valid transition of the associated DQ pins for a READ. tDQSQ specifically refers to the DQS, DQS# to DQ skew. It is the length of time between the DQS, DQS# crossing to the last valid transition of the slowest DQ pin in the DQ group associated with that DQS strobe.
<b>tDQSS</b>	MEM_DDR4_TDQSS_CYC	tDQSS describes the skew between the memory clock (CK) and the output data strobes used for writes. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDSH</b>	MEM_DDR4_TDSH_CYC	tDSH specifies the write DQS hold time. This is the time difference between the rising CK edge and the falling edge of DQS, measured as a percentage of tCK.
<b>tDSS</b>	MEM_DDR4_TDSS_CYC	tDSS describes the time between the falling edge of DQS to the rising edge of the next CK transition.

*continued...*



Display Name	Identifier	Description
<b>tIH (base) DC level</b>	MEM_DDR4_TIH_DC_MV	tIH (base) DC level refers to the voltage level which the address/command signal must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tIH (base)</b>	MEM_DDR4_TIH_PS	tIH (base) refers to the hold time for the Address/Command (A) bus after the rising edge of CK. Depending on what AC level the user has chosen for a design, the hold margin can vary (this variance will be automatically determined when the user chooses the "tIH (base) AC level").
<b>tINIT</b>	MEM_DDR4_TINIT_US	tINIT describes the time duration of the memory initialization after a device power-up. After RESET_n is de-asserted, wait for another 500us until CKE becomes active. During this time, the DRAM will start internal initialization; this will be done independently of external clocks.
<b>tIS (base) AC level</b>	MEM_DDR4_TIS_AC_MV	tIS (base) AC level refers to the voltage level which the address/command signal must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tIS (base)</b>	MEM_DDR4_TIS_PS	tIS (base) refers to the setup time for the Address/Command/Control (A) bus to the rising edge of CK.
<b>tMRD</b>	MEM_DDR4_TMRD_CK_CYC	The mode register set command cycle time, tMRD is the minimum time period required between two MRS commands.
<b>tQH</b>	MEM_DDR4_TQH_UI	tQH specifies the output hold time for the DQ in relation to DQS, DQS#. It is the length of time between the DQS, DQS# crossing to the earliest invalid transition of the fastest DQ pin in the DQ group associated with that DQS strobe.
<b>tQSH</b>	MEM_DDR4_TQSH_CYCLE	tQSH refers to the differential High Pulse Width, which is measured as a percentage of tCK. It is the time during which the DQS is high for a read.
<b>tRAS</b>	MEM_DDR4_TRAS_NS	tRAS describes the activate to precharge duration. A row cannot be deactivated until the tRAS time has been met. Therefore tRAS determines how long the memory has to wait after a activate command before a precharge command can be issued to close the row.
<b>tRCD</b>	MEM_DDR4_TRCD_NS	tRCD, row command delay, describes the amount of delay between the activation of a row through the RAS command and the access to the data through the CAS command.
<b>tRP</b>	MEM_DDR4_TRP_NS	tRP refers to the Precharge (PRE) command period. It describes how long it takes for the memory to disable access to a row by precharging and before it is ready to activate a different row.
<b>tWLH</b>	MEM_DDR4_TWHL_PS	tWLH describes the write leveling hold time from the rising edge of DQS to the rising edge of CK.

*continued...*



Display Name	Identifier	Description
<b>tWLS</b>	MEM_DDR4_TWLS_PS	tWLS describes the write leveling setup time. It is measured from the rising edge of CK to the rising edge of DQS.
<b>tWR</b>	MEM_DDR4_TWR_NS	tWR refers to the Write Recovery time. It specifies the amount of clock cycles needed to complete a write before a precharge command can be issued.
<b>VdiVW_total</b>	MEM_DDR4_VDIVW_TOTAL	VdiVW_total describes the Rx Mask voltage, or the minimum vertical width of the DQ eye opening required by the receiver (memory device/DIMM). It is measured mV.

**Table 338. Group: Mem Timing / Parameters dependent on Speed Bin, Operating Frequency, and Page Size**

Display Name	Identifier	Description
<b>tCCD_L</b>	MEM_DDR4_TCCD_L_CYC	tCCD_L refers to the CAS_n-to-CAS_n delay (long). It is the minimum time interval between two read/write (CAS) commands to the same bank group.
<b>tCCD_S</b>	MEM_DDR4_TCCD_S_CYC	tCCD_S refers to the CAS_n-to-CAS_n delay (short). It is the minimum time interval between two read/write (CAS) commands to different bank groups.
<b>tFAW_dlr</b>	MEM_DDR4_TFAW_DL_R_CYC	tFAW_dlr refers to the four activate window to different logical ranks. It describes the period of time during which only four banks can be active across all logical ranks within a 3DS DDR4 device.
<b>tFAW</b>	MEM_DDR4_TFAW_NS	tFAW refers to the four activate window time. It describes the period of time during which only four banks can be active.
<b>tRRD_dlr</b>	MEM_DDR4_TRRD_DL_R_CYC	tRRD_dlr refers to the Activate to Activate Command Period to Different Logical Ranks. It is the minimum time interval (measured in memory clock cycles) between two activate commands to different logical ranks within a 3DS DDR4 device.
<b>tRRD_L</b>	MEM_DDR4_TRRD_L_CYC	tRRD_L refers to the Activate to Activate Command Period (long). It is the minimum time interval (measured in memory clock cycles) between two activate commands to the same bank group.
<b>tRRD_S</b>	MEM_DDR4_TRRD_S_CYC	tRRD_S refers to the Activate to Activate Command Period (short). It is the minimum time interval between two activate commands to the different bank groups.
<b>tRTP</b>	MEM_DDR4_TRTP_CYC	tRTP refers to the internal READ Command to PRECHARGE Command delay. It is the number of memory clock cycles that is needed between a read command and a precharge command to the same rank.
<b>tWTR_L</b>	MEM_DDR4_TWTR_L_CYC	tWTR_L or Write Timing Parameter describes the delay from start of internal write transaction to internal read command, for accesses to the same bank group. The delay is measured from the first rising memory clock edge after the last write data is received to the rising memory clock edge when a read command is received.
<b>tWTR_S</b>	MEM_DDR4_TWTR_S_CYC	tWTR_S or Write Timing Parameter describes the delay from start of internal write transaction to internal read command, for accesses to the different bank group. The delay is measured from the first rising memory clock edge after the last write data is received to the rising memory clock edge when a read command is received.



**Table 339. Group: Mem Timing / Parameters dependent on Density and Temperature**

Display Name	Identifier	Description
tREFI	MEM_DDR4_TREFI_US	tREFI refers to the average periodic refresh interval. It is the maximum amount of time the memory can tolerate in between each refresh command
tRFC_dlr	MEM_DDR4_TRFC_DL_R_NS	tRFC_dlr refers to the Refresh Cycle Time to different logical rank. It is the amount of delay after a refresh command to one logical rank before an activate command can be accepted by another logical rank within a 3DS DDR4 device. This parameter is dependent on the memory density and is necessary for proper hardware functionality.
tRFC	MEM_DDR4_TRFC_NS	tRFC refers to the Refresh Cycle Time. It is the amount of delay after a refresh command before an activate command can be accepted by the memory. This parameter is dependent on the memory density and is necessary for proper hardware functionality.

#### 7.5.3.6. Stratix 10 EMIF IP DDR4 Parameters: Board

**Table 340. Group: Board / Intersymbol Interference/Crosstalk**

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_DDR4_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total loss of margin on both the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQS/DQS# ISI/crosstalk</b>	BOARD_DDR4_USER_RCLK_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQS/DQS# signal when driven by the memory device during a read. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQ ISI/crosstalk</b>	BOARD_DDR4_USER_RDATA_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQ signal when driven by the memory device during a read. The number to be entered is the total loss of margin on the setup and hold side (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQS/DQS# ISI/crosstalk</b>	BOARD_DDR4_USER_WCLK_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQS/DQS# signal when driven by the FPGA during a write. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_DDR4_USER_WDATA_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side).

*continued...*



Display Name	Identifier	Description
		setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_DDR4_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.

**Table 341. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_DDR4_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Maximum board skew within address/command bus</b>	BOARD_DDR4_BRD_SKEW_WITHIN_AC_NS	The largest skew between the address and command signals.
<b>Maximum board skew within DQS group</b>	BOARD_DDR4_BRD_SKEW_WITHIN_DQS_NS	The largest skew between all DQ and DM pins in a DQS group. This value affects the read capture and write margins.
<b>Average delay difference between DQS and CK</b>	BOARD_DDR4_DQS_TO_CK_SKEW_NS	The average delay difference between the DQS signals and the CK signal, calculated by averaging the longest and smallest DQS trace delay minus the CK trace delay. Positive values represent DQS signals that are longer than CK signals and negative values represent DQS signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_DDR4_IS_SKEW_WITHIN_AC_DESKEWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (DQS group)</b>	BOARD_DDR4_IS_SKEW_WITHIN_DQS_DESKEWED	Enable this parameter if you are compensating for package skew on the DQ, DQS, and DM buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to DIMM/device</b>	BOARD_DDR4_MAX_CK_DELAY_NS	The delay of the longest CK trace from the FPGA to any DIMM/device.
<b>Maximum DQS delay to DIMM/device</b>	BOARD_DDR4_MAX_DQS_DELAY_NS	The delay of the longest DQS trace from the FPGA to any DIMM/device
<b>Maximum delay difference between DIMMs/devices</b>	BOARD_DDR4_SKEW_BETWEEN_DIMMS_NS	The largest propagation delay on DQ signals between ranks (applicable only when there is more than one rank). For example: when you configure two ranks using one DIMM there is a short distance between the ranks for the same DQ pin; when you implement two ranks using two DIMMs the distance is larger.
<b>Maximum skew between DQS groups</b>	BOARD_DDR4_SKEW_BETWEEN_DQS_NS	The largest skew between DQS signals.

#### 7.5.3.7. Stratix 10 EMIF IP DDR4 Parameters: Controller



**Table 342. Group: Controller / Low Power Mode**

Display Name	Identifier	Description
<b>Auto Power-Down Cycles</b>	CTRL_DDR4_AUTO_POWER_DOWN_CYCS	Specifies the number of idle controller cycles after which the memory device is placed into power-down mode. You can configure the idle waiting time. The supported range for number of cycles is from 1 to 65534.
<b>Enable Auto Power-Down</b>	CTRL_DDR4_AUTO_POWER_DOWN_EN	Enable this parameter to have the controller automatically place the memory device into power-down mode after a specified number of idle controller clock cycles. The idle wait time is configurable. All ranks must be idle to enter auto power-down.

**Table 343. Group: Controller / Efficiency**

Display Name	Identifier	Description
<b>Address Ordering</b>	CTRL_DDR4_ADDR_ORDER_ENUM	Controls the mapping between Avalon addresses and memory device addresses. By changing the value of this parameter, you can change the mappings between the Avalon-MM address and the DRAM address. (CS = chip select, CID = chip ID in 3DS/TSV devices, BG = bank group address, Bank = bank address, Row = row address, Col = column address)
<b>Enable Auto-Precache Control</b>	CTRL_DDR4_AUTO_PRECHARGE_EN	Select this parameter to enable the auto-precharge control on the controller top level. If you assert the auto-precharge control signal while requesting a read or write burst, you can specify whether the controller should close (auto-precharge) the currently open page at the end of the read or write burst, potentially making a future access to a different page of the same bank faster.
<b>Enable Reordering</b>	CTRL_DDR4_REORDER_EN	Enable this parameter to allow the controller to perform command and data reordering. Reordering can improve efficiency by reducing bus turnaround time and row/bank switching time. Data reordering allows the single-port memory controller to change the order of read and write commands to achieve highest efficiency. Command reordering allows the controller to issue bank management commands early based on incoming patterns, so that the desired row in memory is already open when the command reaches the memory interface. For more information, refer to the Data Reordering topic in the EMIF Handbook.
<b>Starvation limit for each command</b>	CTRL_DDR4_STARVE_LIMIT	Specifies the number of commands that can be served before a waiting command is served. The controller employs a counter to ensure that all requests are served after a pre-defined interval -- this ensures that low priority requests are not ignored, when doing data reordering for efficiency. The valid range for this parameter is from 1 to 63. For more information, refer to the Starvation Control topic in the EMIF Handbook.
<b>Enable Command Priority Control</b>	CTRL_DDR4_USER_PRIORITY_EN	Select this parameter to enable user-requested command priority control on the controller top level. This parameter instructs the controller to treat a read or write request as high-priority. The controller attempts to fill high-priority requests sooner, to reduce latency. Connect this interface to the conduit of your logic block that determines when the external memory interface IP treats the read or write request as a high-priority command.

**Table 344. Group: Controller / Configuration, Status, and Error Handling**

Display Name	Identifier	Description
<b>Enable Auto Error Correction</b>	CTRL_DDR4_ECC_AUTO_CORRECTION_EN	Specifies that the controller perform auto correction when a single-bit error is detected by the ECC logic.
<b>Enable Error Detection and Correction Logic with ECC</b>	CTRL_DDR4_ECC_EN	Enables error-correction code (ECC) for single-bit error correction and double-bit error detection. Your memory interface must have a width of 16, 24, 40, or 72 bits to use ECC. ECC is implemented as soft logic.
<b>Enable Memory-Mapped Configuration and Status Register (MMR) Interface</b>	CTRL_DDR4_MMR_EN	Enable this parameter to change or read memory timing parameters, memory address size, mode register settings, controller status, and request sideband operations.

**Table 345. Group: Controller / Data Bus Turnaround Time**

Display Name	Identifier	Description
<b>Additional read-to-read turnaround time (different ranks)</b>	CTRL_DDR4_RD_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a read of another logical rank. This can resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (different ranks)</b>	CTRL_DDR4_RD_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (same rank)</b>	CTRL_DDR4_RD_TO_WR SAME CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read to a write within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (different ranks)</b>	CTRL_DDR4_WR_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a read of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (same rank)</b>	CTRL_DDR4_WR_TO_RD SAME CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write to a read within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-write turnaround time (different ranks)</b>	CTRL_DDR4_WR_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.



### 7.5.3.8. Stratix 10 EMIF IP DDR4 Parameters: Diagnostics

**Table 346. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_DDR4_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 347. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Skip address/command deskew calibration</b>	DIAG_DDR4_SKIP_CA_DESKEW	Specifies to skip the address/command deskew calibration stage. Address/command deskew performs per-bit deskew for the address and command pins.
<b>Skip address/command leveling calibration</b>	DIAG_DDR4_SKIP_CA_LEVEL	Specifies to skip the address/command leveling stage during calibration. Address/command leveling attempts to center the memory clock edge against CS# by adjusting delay elements inside the PHY, and then applying the same delay offset to the rest of the address and command pins.
<b>Skip VREF calibration</b>	DIAG_DDR4_SKIP_VREF_CAL	Specifies to skip the VREF stage of calibration. Enable this parameter for debug purposes only; generally, you should include the VREF calibration stage during normal operation.
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface", an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLITE cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip

*continued...*



Display Name	Identifier	Description
		Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.

**Table 348. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX_DESIGN_ISP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX_DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 349. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the <code>altera_emif_avl_tg_2_tb.sv</code> file.

*continued...*



Display Name	Identifier	Description
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 350. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG_EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 351. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

### 7.5.3.9. Stratix 10 EMIF IP DDR4 Parameters: Example Designs

**Table 352. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_DDR4_SEL_DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 353. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_DDR4_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl

*continued...*



Display Name	Identifier	Description
		from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_DDR_4_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 354. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_DDR_4_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.

**Table 355. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_DDR_4_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

### 7.5.3.10. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPOLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.



When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

#### 7.5.4. Stratix 10 EMIF IP DDR3 Parameters

The Stratix 10 EMIF IP parameter editor allows you to parameterize settings for the Stratix 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Stratix 10 EMIF IP.

##### 7.5.4.1. Stratix 10 EMIF IP DDR3 Parameters: General

**Table 356. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 357. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 358. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the <code>clks_sharing_master_out</code> signal from the master interface to the <code>clks_sharing_slave_in</code> signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, <code>emif_usr_clk</code> , <code>afi_clk</code> ), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Use recommended PLL reference clock frequency</b>	PHY_DDR3_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as <code>emif_usr_clk</code> or <code>afi_clk</code> ). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.



**Table 359. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

#### 7.5.4.2. Stratix 10 EMIF IP DDR3 Parameters: Memory

**Table 360. Group: Memory / Topology**

Display Name	Identifier	Description
<b>DQS group of ALERT#</b>	MEM_DDR3_ALERT_N_DQS_GROUP	Select the DQS group with which the ALERT# pin is placed.
<b>ALERT# pin placement</b>	MEM_DDR3_ALERT_N_PLACEMENT_ENUM	Specifies placement for the mem_alert_n signal. If you select "I/O Lane with Address/Command Pins", you can pick the I/O lane and pin index in the add/cmd bank with the subsequent drop down menus. If you select "I/O Lane with DQS Group", you can specify the DQS group with which to place the mem_alert_n pin. If you select "Automatically select a location", the IP automatically selects a pin for the mem_alert_n signal. If you select this option, no additional location constraints can be applied to the mem_alert_n pin, or a fitter error will result during compilation. For optimum signal integrity, you should choose "I/O Lane with Address/Command Pins". For interfaces containing multiple memory devices, it is recommended to connect the ALERT# pins together to the ALERT#pin on the FPGA.
<b>Bank address width</b>	MEM_DDR3_BANK_ADDRESS_WIDTH	Specifies the number of bank address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of bank address pins needed for access to all available banks.
<b>Number of clocks</b>	MEM_DDR3_CK_WIDTH	Specifies the number of CK/CK# clock pairs exposed by the memory interface. Usually more than 1 pair is required for RDIMM/LRDIMM formats. The value of this parameter depends on the memory device selected; refer to the data sheet for your memory device.
<b>Column address width</b>	MEM_DDR3_COL_ADDRESS_WIDTH	Specifies the number of column address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of address pins needed for access to all available columns.
<b>Number of chip selects per DIMM</b>	MEM_DDR3_CS_PER_DIMM	Specifies the number of chip selects per DIMM.
<b>Number of chip selects</b>	MEM_DDR3_DISCRETE_CS_WIDTH	Specifies the total number of chip selects in the interface, up to a maximum of 4. This parameter applies to discreet components only.
<b>Enable DM pins</b>	MEM_DDR3_DM_EN	Indicates whether the interface uses data mask (DM) pins. This feature allows specified portions of the data bus to be written to memory (not available in x4 mode). One DM pin exists per DQS group
<b>Number of DQS groups</b>	MEM_DDR3_DQS_WIDTH	Specifies the total number of DQS groups in the interface. This value is automatically calculated as the DQ width divided by the number of DQ pins per DQS group.
<b>DQ pins per DQS group</b>	MEM_DDR3_DQ_PER_DQS	Specifies the total number of DQ pins per DQS group.

*continued...*



Display Name	Identifier	Description
<b>DQ width</b>	MEM_DDR3_DQ_WIDTH	Specifies the total number of data pins in the interface. The maximum supported width is 144, or 72 in Ping Pong PHY mode.
<b>Memory format</b>	MEM_DDR3_FORMAT_ENUM	Specifies the format of the external memory device. The following formats are supported: Component - a Discrete memory device; UDIMM - Unregistered/Unbuffered DIMM where address/control, clock, and data are unbuffered; RDIMM - Registered DIMM where address/control and clock are buffered; LRDIMM - Load Reduction DIMM where address/control, clock, and data are buffered. LRDIMM reduces the load to increase memory speed and supports higher densities than RDIMM; SODIMM - Small Outline DIMM is similar to UDIMM but smaller in size and is typically used for systems with limited space. Some memory protocols may not be available in all formats.
<b>Number of DIMMs</b>	MEM_DDR3_NUM_OF_DIMMS	Total number of DIMMs.
<b>Number of physical ranks per DIMM</b>	MEM_DDR3_RANKS_PER_DIMM	Number of ranks per DIMM. For LRDIMM, this represents the number of physical ranks on the DIMM behind the memory buffer
<b>Number of rank multiplication pins</b>	MEM_DDR3_RM_WIDTH	Number of rank multiplication pins used to access all physical ranks on an LRDIMM. Rank multiplication is a ratio between the number of physical ranks for an LRDIMM and the number of logical ranks for the controller. These pins should be connected to CS#[2] and/or CS#[3] of all LRDIMMs in the system
<b>Row address width</b>	MEM_DDR3_ROW_ADDRESS_WIDTH	Specifies the number of row address pins. Refer to the data sheet for your memory device. The density of the selected memory device determines the number of address pins needed for access to all available rows.

**Table 361. Group: Memory / Latency and Burst**

Display Name	Identifier	Description
<b>Memory additive CAS latency setting</b>	MEM_DDR3_ATCL_ENUM	Determines the posted CAS additive latency of the memory device. Enable this feature to improve command and bus efficiency, and increase system bandwidth.
<b>Burst Length</b>	MEM_DDR3_BL_ENUM	Specifies the DRAM burst length which determines how many consecutive addresses should be accessed for a given read/write command.
<b>Read Burst Type</b>	MEM_DDR3_BT_ENUM	Indicates whether accesses within a given burst are in sequential or interleaved order. Select sequential if you are using the Intel-provided memory controller.
<b>Memory CAS latency setting</b>	MEM_DDR3_TCL	Specifies the number of clock cycles between the read command and the availability of the first bit of output data at the memory device. Overall read latency equals the additive latency (AL) + the CAS latency (CL). Overall read latency depends on the memory device selected; refer to the datasheet for your device.
<b>Memory write CAS latency setting</b>	MEM_DDR3_WTCL	Specifies the number of clock cycles from the release of internal write to the latching of the first data in at the memory device. This value depends on the memory device selected; refer to the datasheet for your device.



**Table 362. Group: Memory / Mode Register Settings**

Display Name	Identifier	Description
<b>Auto self-refresh method</b>	MEM_DDR3_ASR_ENUM	Indicates whether to enable or disable auto self-refresh. Auto self-refresh allows the controller to issue self-refresh requests, rather than manually issuing self-refresh in order for memory to retain data.
<b>DDR3 RDIMM/LRDIMM additional control words</b>	MEM_DDR3_LRDIMM_EXTENDED_CONFIG	Each 4-bit setting can be obtained from the manufacturer's data sheet and should be entered in hexadecimal, starting with BC0F on the left and ending with BC00 on the right
<b>DLL precharge power down</b>	MEM_DDR3_PD_ENUM	Specifies whether the DLL in the memory device is off or on during precharge power-down
<b>DDR3 RDIMM/LRDIMM control words</b>	MEM_DDR3_RDIMM_CONFIG	Each 4-bit/8-bit setting can be obtained from the manufacturer's data sheet and should be entered in hexadecimal, starting with the 8-bit setting RCBx on the left and continuing to RC1x followed by the 4-bit setting RCOF and ending with RC00 on the right
<b>Self-refresh temperature</b>	MEM_DDR3_SRT_ENUM	Specifies the self-refresh temperature as "Normal" or "Extended" mode. More information on Normal and Extended temperature modes can be found in the memory device datasheet.

#### 7.5.4.3. Stratix 10 EMIF IP DDR3 Parameters: Mem I/O

**Table 363. Group: Mem I/O / Memory I/O Settings**

Display Name	Identifier	Description
<b>Output drive strength setting</b>	MEM_DDR3_DRV_STR_ENUM	Specifies the output driver impedance setting at the memory device. To obtain optimum signal integrity performance, select option based on board simulation results.
<b>ODT Rtt nominal value</b>	MEM_DDR3_RTT_NOM_ENUM	Determines the nominal on-die termination value applied to the DRAM. The termination is applied any time that ODT is asserted. If you specify a different value for RTT_WR, that value takes precedence over the values mentioned here. For optimum signal integrity performance, select your option based on board simulation results.
<b>Dynamic ODT (Rtt_WR) value</b>	MEM_DDR3_RTT_WR_ENUM	Specifies the mode of the dynamic on-die termination (ODT) during writes to the memory device (used for multi-rank configurations). For optimum signal integrity performance, select this option based on board simulation results.

**Table 364. Group: Mem I/O / ODT Activation**

Display Name	Identifier	Description
<b>Use Default ODT Assertion Tables</b>	MEM_DDR3_USE_DEFAULT_ODT	Enables the default ODT assertion pattern as determined from vendor guidelines. These settings are provided as a default only; you should simulate your memory interface to determine the optimal ODT settings and assertion patterns.

#### 7.5.4.4. Stratix 10 EMIF IP DDR3 Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 365. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_DDR3_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_DDR3_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 366. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_DDR3_USER_AC_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR3_USER_AC_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_DDR3_USER_AC_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 367. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_DDR3_USER_CK_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR3_USER_CK_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_DDR3_USER_CK_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.



**Table 368. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_DDR3_USER_AU_TO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_DDR3_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_DDR3_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_DDR3_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_DDR3_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 369. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_DDR3_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_DDR3_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.5.4.5. Stratix 10 EMIF IP DDR3 Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 370. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_DDR3_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tDH (base) DC level</b>	MEM_DDR3_TD�_DC_MV	tDH (base) DC level refers to the voltage level which the data bus must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tDH (base)</b>	MEM_DDR3_TD�_PS	tDH (base) refers to the hold time for the Data (DQ) bus after the rising edge of CK.

*continued...*



Display Name	Identifier	Description
<b>tDQSK</b>	MEM_DDR3_TDQSK_PS	tDQSK describes the skew between the memory clock (CK) and the input data strobes (DQS) used for reads. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDQSQ</b>	MEM_DDR3_TDQSQ_PS	tDQSQ describes the latest valid transition of the associated DQ pins for a READ. tDQSQ specifically refers to the DQS, DQS# to DQ skew. It is the length of time between the DQS, DQS# crossing to the last valid transition of the slowest DQ pin in the DQ group associated with that DQS strobe.
<b>tDQSS</b>	MEM_DDR3_TDQSS_CYC	tDQSS describes the skew between the memory clock (CK) and the output data strobes used for writes. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDSH</b>	MEM_DDR3_TDSH_CYC	tDSH specifies the write DQS hold time. This is the time difference between the rising CK edge and the falling edge of DQS, measured as a percentage of tCK.
<b>tDSS</b>	MEM_DDR3_TDSS_CYC	tDSS describes the time between the falling edge of DQS to the rising edge of the next CK transition.
<b>tDS (base) AC level</b>	MEM_DDR3_TDS_AC_MV	tDS (base) AC level refers to the voltage level which the data bus must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tDS (base)</b>	MEM_DDR3_TDS_PS	tDS(base) refers to the setup time for the Data (DQ) bus before the rising edge of the DQS strobe.
<b>tIH (base) DC level</b>	MEM_DDR3_TIH_DC_MV	tIH (base) DC level refers to the voltage level which the address/command signal must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tIH (base)</b>	MEM_DDR3_TIH_PS	tIH (base) refers to the hold time for the Address/Command (A) bus after the rising edge of CK. Depending on what AC level the user has chosen for a design, the hold margin can vary (this variance will be automatically determined when the user chooses the "tIH (base) AC level").
<b>tINIT</b>	MEM_DDR3_TINIT_US	tINIT describes the time duration of the memory initialization after a device power-up. After RESET_n is de-asserted, wait for another 500us until CKE becomes active. During this time, the DRAM starts internal initialization; this happens independently of external clocks.
<b>tIS (base) AC level</b>	MEM_DDR3_TIS_AC_MV	tIS (base) AC level refers to the voltage level which the address/command signal must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tIS (base)</b>	MEM_DDR3_TIS_PS	tIS (base) refers to the setup time for the Address/Command/Control (A) bus to the rising edge of CK.
<b>tMRD</b>	MEM_DDR3_TMRD_CK_CYC	The mode register set command cycle time, tMRD is the minimum time period required between two MRS commands.

*continued...*



Display Name	Identifier	Description
<b>tQH</b>	MEM_DDR3_TQH_CYC	tQH specifies the output hold time for the DQ in relation to DQS, DQS#. It is the length of time between the DQS, DQS# crossing to the earliest invalid transition of the fastest DQ pin in the DQ group associated with that DQS strobe.
<b>tQSH</b>	MEM_DDR3_TQSH_CYC	tQSH refers to the differential High Pulse Width, which is measured as a percentage of tCK. It is the time during which the DQS is high for a read.
<b>tRAS</b>	MEM_DDR3_TRAS_NS	tRAS describes the activate to precharge duration. A row cannot be deactivated until the tRAS time has been met. Therefore tRAS determines how long the memory has to wait after a activate command before a precharge command can be issued to close the row.
<b>tRCD</b>	MEM_DDR3_TRCD_NS	tRCD, row command delay, describes the amount of delay between the activation of a row through the RAS command and the access to the data through the CAS command.
<b>tRP</b>	MEM_DDR3_TRP_NS	tRP refers to the Precharge (PRE) command period. It describes how long it takes for the memory to disable access to a row by precharging and before it is ready to activate a different row.
<b>tWLH</b>	MEM_DDR3_TWHL_PS	tWLH describes the write leveling hold time from the rising edge of DQS to the rising edge of CK.
<b>tWLS</b>	MEM_DDR3_TWLS_PS	tWLS describes the write leveling setup time. It is measured from the rising edge of CK to the rising edge of DQS.
<b>tWR</b>	MEM_DDR3_TWR_NS	tWR refers to the Write Recovery time. It specifies the amount of clock cycles needed to complete a write before a precharge command can be issued.

**Table 371. Group: Mem Timing / Parameters dependent on Speed Bin, Operating Frequency, and Page Size**

Display Name	Identifier	Description
<b>tFAW</b>	MEM_DDR3_TFAW_NS	tFAW refers to the four activate window time. It describes the period of time during which only four banks can be active.
<b>tRRD</b>	MEM_DDR3_TRRD_CYC	tRRD refers to the Row Active to Row Active Delay. It is the minimum time interval (measured in memory clock cycles) between two activate commands to rows in different banks in the same rank
<b>tRTP</b>	MEM_DDR3_TRTP_CYC	tRTP refers to the internal READ Command to PRECHARGE Command delay. It is the number of memory clock cycles that is needed between a read command and a precharge command to the same rank.
<b>tWTR</b>	MEM_DDR3_TWTR_CYC	tWTR or Write Timing Parameter describes the delay from start of internal write transaction to internal read command, for accesses to the same bank. The delay is measured from the first rising memory clock edge after the last write data is received to the rising memory clock edge when a read command is received.

**Table 372. Group: Mem Timing / Parameters dependent on Density and Temperature**

Display Name	Identifier	Description
tREFI	MEM_DDR3_TREFI_US	tREFI refers to the average periodic refresh interval. It is the maximum amount of time the memory can tolerate in between each refresh command
tRFC	MEM_DDR3_TRFC_NS	tRFC refers to the Refresh Cycle Time. It is the amount of delay after a refresh command before an activate command can be accepted by the memory. This parameter is dependent on the memory density and is necessary for proper hardware functionality.

#### 7.5.4.6. Stratix 10 EMIF IP DDR3 Parameters: Board

**Table 373. Group: Board / Intersymbol Interference/Crosstalk**

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_DDR3_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total loss of margin on both the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQS/DQS# ISI/crosstalk</b>	BOARD_DDR3_USER_RCLK_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQS/DQS# signal when driven by the memory device during a read. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQ ISI/crosstalk</b>	BOARD_DDR3_USER_RDATA_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQ signal when driven by the memory device during a read. The number to be entered is the total loss of margin on the setup and hold side (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQS/DQS# ISI/crosstalk</b>	BOARD_DDR3_USER_WCLK_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQS/DQS# signal when driven by the FPGA during a write. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_DDR3_USER_WDATA_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total loss of margin on the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_DDR3_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.



**Table 374. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_DDR3_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Maximum board skew within DQS group</b>	BOARD_DDR3_BRD_SKew_WITHIN_DQS_NS	The largest skew between all DQ and DM pins in a DQS group. This value affects the read capture and write margins.
<b>Average delay difference between DQS and CK</b>	BOARD_DDR3_DQS_TO_CK_SKEW_NS	The average delay difference between the DQS signals and the CK signal, calculated by averaging the longest and smallest DQS trace delay minus the CK trace delay. Positive values represent DQS signals that are longer than CK signals and negative values represent DQS signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_DDR3_IS_SKew_WITHIN_AC_DESKewed	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (DQS group)</b>	BOARD_DDR3_IS_SKew_WITHIN_DQS_DESKewed	Enable this parameter if you are compensating for package skew on the DQ, DQS, and DM buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to DIMM/device</b>	BOARD_DDR3_MAX_CK_DELAY_NS	The delay of the longest CK trace from the FPGA to any DIMM/device.
<b>Maximum DQS delay to DIMM/device</b>	BOARD_DDR3_MAX_DQS_DELAY_NS	The delay of the longest DQS trace from the FPGA to any DIMM/device
<b>Maximum system skew within address/command bus</b>	BOARD_DDR3_PKG_BRD_SKew_WITHIN_AC_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.
<b>Maximum delay difference between DIMMs/devices</b>	BOARD_DDR3_SKew_BETWEEN_DIMMS_NS	The largest propagation delay on DQ signals between ranks (applicable only when there is more than one rank). For example: when you configure two ranks using one DIMM there is a short distance between the ranks for the same DQ pin; when you implement two ranks using two DIMMs the distance is larger.
<b>Maximum skew between DQS groups</b>	BOARD_DDR3_SKew_BETWEEN_DQS_NS	The largest skew between DQS signals.

#### 7.5.4.7. Stratix 10 EMIF IP DDR3 Parameters: Controller

**Table 375. Group: Controller / Low Power Mode**

Display Name	Identifier	Description
<b>Auto Power-Down Cycles</b>	CTRL_DDR3_AUTO_POWER_DOWN_CYCS	Specifies the number of idle controller cycles after which the memory device is placed into power-down mode. You can configure the idle waiting time. The supported range for number of cycles is from 1 to 65534.
<b>Enable Auto Power-Down</b>	CTRL_DDR3_AUTO_POWER_DOWN_EN	Enable this parameter to have the controller automatically place the memory device into power-down mode after a specified number of idle controller clock cycles. The idle wait time is configurable. All ranks must be idle to enter auto power-down.

**Table 376. Group: Controller / Efficiency**

Display Name	Identifier	Description
<b>Address Ordering</b>	CTRL_DDR3_ADDR_ORDER_ENUM	Controls the mapping between Avalon addresses and memory device addresses. By changing the value of this parameter, you can change the mappings between the Avalon-MM address and the DRAM address.
<b>Enable Auto-Precharge Control</b>	CTRL_DDR3_AUTO_PRECHARGE_EN	Select this parameter to enable the auto-precharge control on the controller top level. If you assert the auto-precharge control signal while requesting a read or write burst, you can specify whether the controller should close (auto-precharge) the currently open page at the end of the read or write burst, potentially making a future access to a different page of the same bank faster.
<b>Enable Reordering</b>	CTRL_DDR3_REORDER_EN	Enable this parameter to allow the controller to perform command and data reordering. Reordering can improve efficiency by reducing bus turnaround time and row/bank switching time. Data reordering allows the single-port memory controller to change the order of read and write commands to achieve highest efficiency. Command reordering allows the controller to issue bank management commands early based on incoming patterns, so that the desired row in memory is already open when the command reaches the memory interface. For more information, refer to the Data Reordering topic in the EMIF Handbook.
<b>Starvation limit for each command</b>	CTRL_DDR3_STARVE_LIMIT	Specifies the number of commands that can be served before a waiting command is served. The controller employs a counter to ensure that all requests are served after a pre-defined interval -- this ensures that low priority requests are not ignored, when doing data reordering for efficiency. The valid range for this parameter is from 1 to 63. For more information, refer to the Starvation Control topic in the EMIF Handbook.
<b>Enable Command Priority Control</b>	CTRL_DDR3_USER_PRIORITY_EN	Select this parameter to enable user-requested command priority control on the controller top level. This parameter instructs the controller to treat a read or write request as high-priority. The controller attempts to fill high-priority requests sooner, to reduce latency. Connect this interface to the conduit of your logic block that determines when the external memory interface IP treats the read or write request as a high-priority command.



**Table 377. Group: Controller / Configuration, Status, and Error Handling**

Display Name	Identifier	Description
<b>Enable Auto Error Correction</b>	CTRL_DDR3_ECC_AUTO_CORRECTION_EN	Specifies that the controller perform auto correction when a single-bit error is detected by the ECC logic.
<b>Enable Error Detection and Correction Logic with ECC</b>	CTRL_DDR3_ECC_EN	Enables error-correction code (ECC) for single-bit error correction and double-bit error detection. Your memory interface must have a width of 16, 24, 40, or 72 bits to use ECC. ECC is implemented as soft logic.
<b>Enable Memory-Mapped Configuration and Status Register (MMR) Interface</b>	CTRL_DDR3_MMR_EN	Enable this parameter to change or read memory timing parameters, memory address size, mode register settings, controller status, and request sideband operations.

**Table 378. Group: Controller / Data Bus Turnaround Time**

Display Name	Identifier	Description
<b>Additional read-to-read turnaround time (different ranks)</b>	CTRL_DDR3_RD_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a read of another logical rank. This can resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (different ranks)</b>	CTRL_DDR3_RD_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (same rank)</b>	CTRL_DDR3_RD_TO_WR_SAME_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read to a write within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (different ranks)</b>	CTRL_DDR3_WR_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a read of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (same rank)</b>	CTRL_DDR3_WR_TO_RD_SAME_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write to a read within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-write turnaround time (different ranks)</b>	CTRL_DDR3_WR_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.



#### 7.5.4.8. Stratix 10 EMIF IP DDR3 Parameters: Diagnostics

**Table 379. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_DDR3_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 380. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface," an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLite cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.



**Table 381. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX_DESIGN_IS_SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX_DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 382. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 383. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 384. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

#### 7.5.4.9. Stratix 10 EMIF IP DDR3 Parameters: Example Designs

**Table 385. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_DDR3_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 386. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_DDR3_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_DDR3_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 387. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_DDR3_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.



**Table 388. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
Select board	EX_DESIGN_GUI_DDR3_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

#### 7.5.4.10. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

#### 7.5.5. Stratix 10 EMIF IP LPDDR3 Parameters

The Stratix 10 EMIF IP parameter editor allows you to parameterize settings for the Stratix 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.



The following tables describe the parameterization settings available in the parameter editor for the Stratix 10 EMIF IP.

### 7.5.5.1. Stratix 10 EMIF IP LPDDR3 Parameters: General

**Table 389. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 390. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 391. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Use recommended PLL reference clock frequency</b>	PHY_LPDDR3_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.

*continued...*



Display Name	Identifier	Description
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 392. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

### 7.5.5.2. Stratix 10 EMIF IP LPDDR3 Parameters: Memory

**Table 393. Group: Memory / Topology**

Display Name	Identifier	Description
<b>Bank address width</b>	MEM_LPDDR3_BANK_ADDR_WIDTH	The number of bank address bits.
<b>Number of clocks</b>	MEM_LPDDR3_CK_WIDT	Number of CK/CK# clock pairs exposed by the memory interface.
<b>Column address width</b>	MEM_LPDDR3_COL_A_DDR_WIDTH	The number of column address bits.
<b>Number of chip selects</b>	MEM_LPDDR3_DISCRETE_CS_WIDTH	Total number of chip selects in the interface.

*continued...*



Display Name	Identifier	Description
<b>Enable DM pins</b>	MEM_LPDDR3_DM_EN	Indicates whether interface uses data mask (DM) pins. This feature allows specified portions of the data bus to be written to memory (not available in x4 mode). One DM pin exists per DQS group
<b>Number of DQS groups</b>	MEM_LPDDR3_DQS_W_IDTH	Specifies the total number of DQS groups in the interface. This value is automatically calculated as the DQ width divided by the number of DQ pins per DQS group.
<b>DQ width</b>	MEM_LPDDR3_DQ_WI_DTH	Total number of DQ pins in the interface.
<b>Row address width</b>	MEM_LPDDR3_ROW_A_DDR_WIDTH	The number of row address bits.

**Table 394. Group: Memory / Latency and Burst**

Display Name	Identifier	Description
<b>Burst length</b>	MEM_LPDDR3_BL	Burst length of the memory device.
<b>Data latency</b>	MEM_LPDDR3_DATA_L_ATENCY	Determines the mode register setting that controls the data latency. Sets both READ and WRITE latency (RL and WL).
<b>DQ ODT</b>	MEM_LPDDR3_DQODT	The ODT setting for the DQ pins during writes.
<b>Power down ODT</b>	MEM_LPDDR3_PDODT	Turn on turn off ODT during power down.
<b>WL set</b>	MEM_LPDDR3_WLSELECT	The set of the currently selected write latency. Only certain memory devices support WL Set B. Refer to the WRITE Latency table in the memory vendor data sheet.

#### 7.5.5.3. Stratix 10 EMIF IP LPDDR3 Parameters: Mem I/O

**Table 395. Group: Mem I/O / ODT Activation**

Display Name	Identifier	Description
<b>Use Default ODT Assertion Tables</b>	MEM_LPDDR3_USE_DEFAULT_ODT	Enables the default ODT assertion pattern as determined from vendor guidelines. These settings are provided as a default only; you should simulate your memory interface to determine the optimal ODT settings and assertion patterns.

#### 7.5.5.4. Stratix 10 EMIF IP LPDDR3 Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 396. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_LPDDR3_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal

*continued...*



Display Name	Identifier	Description
		integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_LPDDR3_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 397. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_LPDDR3_USER_A_C_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_LPDDR3_USER_A_C_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_LPDDR3_USER_A_C_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 398. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_LPDDR3_USER_C_K_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_LPDDR3_USER_C_K_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_LPDDR3_USER_C_K_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 399. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_LPDDR3_USER_AUTO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_LPDDR3_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_LPDDR3_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_LPDDR3_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_LPDDR3_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 400. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_LPDDR3_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_LPDDR3_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

### 7.5.5.5. Stratix 10 EMIF IP LPDDR3 Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 401. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_LPDDR3_SPEED_BIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tDH (base) DC level</b>	MEM_LPDDR3_TDHC_DC_MV	tDH (base) DC level refers to the voltage level which the data bus must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tDH (base)</b>	MEM_LPDDR3_TDHS_P	tDH (base) refers to the hold time for the Data (DQ) bus after the rising edge of CK.

*continued...*



Display Name	Identifier	Description
<b>tDQSKC</b>	MEM_LPDDR3_TDQSC_K_PS	tDQSKC describes the skew between the memory clock (CK) and the input data strobes (DQS) used for reads. It is the time between the rising data strobe edge (DQS, DQS#) relative to the rising CK edge.
<b>tDQSQ</b>	MEM_LPDDR3_TDQSQ_PS	tDQSQ describes the latest valid transition of the associated DQ pins for a READ. tDQSQ specifically refers to the DQS, DQS# to DQ skew. It is the length of time between the DQS, DQS# crossing to the last valid transition of the slowest DQ pin in the DQ group associated with that DQS strobe.
<b>tDSH</b>	MEM_LPDDR3_TDSH_CYC	tDSH specifies the write DQS hold time. This is the time difference between the rising CK edge and the falling edge of DQS, measured as a percentage of tCK.
<b>tDSS</b>	MEM_LPDDR3_TDSS_CYC	tDSS describes the time between the falling edge of DQS to the rising edge of the next CK transition.
<b>tDS (base) AC level</b>	MEM_LPDDR3_TDS_A_C_MV	tDS (base) AC level refers to the voltage level which the data bus must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tDS (base)</b>	MEM_LPDDR3_TDS_P_S	tDS(base) refers to the setup time for the Data (DQ) bus before the rising edge of the DQS strobe.
<b>tIHCA (base) DC level</b>	MEM_LPDDR3_TIH_D_C_MV	DC level of tIH (base) for derating purpose
<b>tIHCA (base)</b>	MEM_LPDDR3_TIH_PS	Address and control hold after CK clock rise
<b>tINIT</b>	MEM_LPDDR3_TINIT_US	tINIT describes the time duration of the memory initialization after a device power-up. After RESET_n is de-asserted, wait for another 500us until CKE becomes active. During this time, the DRAM will start internal initialization; this will be done independently of external clocks.
<b>tISCA (base) AC level</b>	MEM_LPDDR3_TIS_AC_MV	AC level of tIS (base) for derating purpose
<b>tISCA (base)</b>	MEM_LPDDR3_TIS_PS	Address and control setup to CK clock rise
<b>tMRR</b>	MEM_LPDDR3_TMRR_CK_CYC	tMRR describes the minimum MODE REGISTER READ command period.
<b>tMRW</b>	MEM_LPDDR3_TMRW_CK_CYC	tMRW describes the minimum MODE REGISTER WRITE command period.
<b>tQH</b>	MEM_LPDDR3_TQH_C_YC	tQH specifies the output hold time for the DQ in relation to DQS, DQS#. It is the length of time between the DQS, DQS# crossing to the earliest invalid transition of the fastest DQ pin in the DQ group associated with that DQS strobe.
<b>tQSH</b>	MEM_LPDDR3_TQSH_CYC	tQSH refers to the differential High Pulse Width, which is measured as a percentage of tCK. It is the time during which the DQS is high for a read.
<b>tRAS</b>	MEM_LPDDR3_TRAS_NS	tRAS describes the activate to precharge duration. A row cannot be deactivated until the tRAS time has been met. Therefore tRAS determines how long the memory has to wait after a activate command before a precharge command can be issued to close the row.
<b>tRCD</b>	MEM_LPDDR3_TRCD_NS	tRCD, row command delay, describes the amount of delay between the activation of a row through the RAS command and the access to the data through the CAS command.

*continued...*



Display Name	Identifier	Description
<b>tWLH</b>	MEM_LPDDR3_TWHL_PS	tWLH describes the write leveling hold time from the rising edge of DQS to the rising edge of CK.
<b>tWLS</b>	MEM_LPDDR3_TWLS_PS	tWLS describes the write leveling setup time. It is measured from the rising edge of CK to the rising edge of DQS.
<b>tWR</b>	MEM_LPDDR3_TWR_NS	tWR refers to the Write Recovery time. It specifies the amount of clock cycles needed to complete a write before a precharge command can be issued.

**Table 402. Group: Mem Timing / Parameters dependent on Speed Bin, Operating Frequency, and Page Size**

Display Name	Identifier	Description
<b>tFAW</b>	MEM_LPDDR3_TFAW_NS	tFAW refers to the four activate window time. It describes the period of time during which only four banks can be active.
<b>tRRD</b>	MEM_LPDDR3_TRRD_CYC	tRRD refers to the Row Active to Row Active Delay. It is the minimum time interval (measured in memory clock cycles) between two activate commands to rows in different banks in the same rank
<b>tRTP</b>	MEM_LPDDR3_TRTP_CYC	tRTP refers to the internal READ Command to PRECHARGE Command delay. It is the number of memory clock cycles that is needed between a read command and a precharge command to the same rank.
<b>tWTR</b>	MEM_LPDDR3_TWTR_CYC	tWTR or Write Timing Parameter describes the delay from start of internal write transaction to internal read command, for accesses to the same bank. The delay is measured from the first rising memory clock edge after the last write data is received to the rising memory clock edge when a read command is received.

**Table 403. Group: Mem Timing / Parameters dependent on Density and Temperature**

Display Name	Identifier	Description
<b>tREFI</b>	MEM_LPDDR3_TREFI_US	tREFI refers to the average periodic refresh interval. It is the maximum amount of time the memory can tolerate in between each refresh command
<b>tRFCab</b>	MEM_LPDDR3_TRFC_NS	Auto-refresh command interval (all banks)

### 7.5.5.6. Stratix 10 EMIF IP LPDDR3 Parameters: Board

**Table 404. Group: Board / Intersymbol Interference/Crosstalk**

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_AC_ISI_NS	The address and command window reduction due to intersymbol interference and crosstalk effects. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQS/DQS# ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_RCLK_ISI_NS	The reduction of the read data window due to intersymbol interference and crosstalk effects on the DQS/DQS# signal when driven by the memory device during a read. The

*continued...*



Display Name	Identifier	Description
		number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQ ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_RDATA_ISI_NS	The reduction of the read data window due to intersymbol interference and crosstalk effects on the DQ signal when driven by the memory device during a read. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQS/DQS# ISI/crosstalk</b>	BOARD_LPDDR3_USE_R_WCLK_ISI_NS	The reduction of the write data window due to intersymbol interference and crosstalk effects on the DQS/DQS# signal when driven by the FPGA during a write. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk values</b>	BOARD_LPDDR3_USE_DEFAULT_ISI_VALUES	The reduction of the write data window due to intersymbol interference and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_LPDDR3_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx)*, and manually enter values based on your simulation results, instead of using the default values.

**Table 405. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_LPDDR3_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Maximum board skew within DQS group</b>	BOARD_LPDDR3_BRD_SKEW_WITHIN_DQS_NS	The largest skew between all DQ and DM pins in a DQS group. This value affects the read capture and write margins.
<b>Average delay difference between DQS and CK</b>	BOARD_LPDDR3_DQS_TO_CK_SKEW_NS	The average delay difference between the DQS signals and the CK signal, calculated by averaging the longest and smallest DQS trace delay minus the CK trace delay. Positive values represent DQS signals that are longer than CK signals and negative values represent DQS signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_LPDDR3_IS_SKEWED_WITHIN_AC_DE_SKEWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.

*continued...*



Display Name	Identifier	Description
<b>Package deskewed with board layout (DQS group)</b>	BOARD_LPDDR3_IS_SKew_WITHIN_DQS_ESKEWED	Enable this parameter if you are compensating for package skew on the DQ, DQS, and DM buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to device</b>	BOARD_LPDDR3_MAX_CK_DELAY_NS	The maximum CK delay to device refers to the delay of the longest CK trace from the FPGA to any device.
<b>Maximum DQS delay to device</b>	BOARD_LPDDR3_MAX_DQS_DELAY_NS	The maximum DQS delay to device refers to the delay of the longest DQS trace from the FPGA to any device
<b>Maximum system skew within address/command bus</b>	BOARD_LPDDR3_PKG_BRD_SKEW_WITHIN_AC_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.
<b>Maximum delay difference between devices</b>	BOARD_LPDDR3_SKEW_BETWEEN_DIMMS_NS	This parameter describes the largest propagation delay on the DQ signals between ranks. For example, in a two-rank configuration where devices are placed in series, there is an extra propagation delay for DQ signals going to and coming back from the furthest device compared to the nearest device. This parameter is only applicable when there is more than one rank.
<b>Maximum skew between DQS groups</b>	BOARD_LPDDR3_SKEW_BETWEEN_DQS_NS	The largest skew between DQS signals.

### 7.5.5.7. Stratix 10 EMIF IP LPDDR3 Parameters: Controller

**Table 406. Group: Controller / Low Power Mode**

Display Name	Identifier	Description
<b>Auto Power-Down Cycles</b>	CTRL_LPDDR3_AUTO_POWER_DOWN_CYCS	Specifies the number of idle controller cycles after which the memory device is placed into power-down mode. You can configure the idle waiting time. The supported range for number of cycles is from 1 to 65534.
<b>Enable Auto Power-Down</b>	CTRL_LPDDR3_AUTO_POWER_DOWN_EN	Enable this parameter to have the controller automatically place the memory device into power-down mode after a specified number of idle controller clock cycles. The idle wait time is configurable. All ranks must be idle to enter auto power-down.

**Table 407. Group: Controller / Efficiency**

Display Name	Identifier	Description
<b>Address Ordering</b>	CTRL_LPDDR3_ADDR_ORDER_ENUM	Controls the mapping between Avalon addresses and memory device addresses. By changing the value of this parameter, you can change the mappings between the Avalon-MM address and the DRAM address.
<b>Enable Auto-Precharge Control</b>	CTRL_LPDDR3_AUTO_PRECHARGE_EN	Select this parameter to enable the auto-precharge control on the controller top level. If you assert the auto-precharge control signal while requesting a read or write burst, you can specify whether the controller should close (auto-precharge) the currently open page at the end of the read or write burst, potentially making a future access to a different page of the same bank faster.
<b>Enable Reordering</b>	CTRL_LPDDR3_REORDER_EN	Enable this parameter to allow the controller to perform command and data reordering. Reordering can improve efficiency by reducing bus turnaround time and row/bank

*continued...*



Display Name	Identifier	Description
		switching time. Data reordering allows the single-port memory controller to change the order of read and write commands to achieve highest efficiency. Command reordering allows the controller to issue bank management commands early based on incoming patterns, so that the desired row in memory is already open when the command reaches the memory interface. For more information, refer to the Data Reordering topic in the EMIF Handbook.
<b>Starvation limit for each command</b>	CTRL_LPDDR3_STARVE_LIMIT	Specifies the number of commands that can be served before a waiting command is served. The controller employs a counter to ensure that all requests are served after a pre-defined interval -- this ensures that low priority requests are not ignored, when doing data reordering for efficiency. The valid range for this parameter is from 1 to 63. For more information, refer to the Starvation Control topic in the EMIF Handbook.
<b>Enable Command Priority Control</b>	CTRL_LPDDR3_USER_PRIORITY_EN	Select this parameter to enable user-requested command priority control on the controller top level. This parameter instructs the controller to treat a read or write request as high-priority. The controller attempts to fill high-priority requests sooner, to reduce latency. Connect this interface to the conduit of your logic block that determines when the external memory interface IP treats the read or write request as a high-priority command.

**Table 408. Group: Controller / Configuration, Status, and Error Handling**

Display Name	Identifier	Description
<b>Enable Memory-Mapped Configuration and Status Register (MMR) Interface</b>	CTRL_LPDDR3_MMR_EN	Enable this parameter to change or read memory timing parameters, memory address size, mode register settings, controller status, and request sideband operations.

**Table 409. Group: Controller / Data Bus Turnaround Time**

Display Name	Identifier	Description
<b>Additional read-to-read turnaround time (different ranks)</b>	CTRL_LPDDR3_RD_TO_RD_DIFF_CHIP_DELT_A_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a read of another logical rank. This can resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (different ranks)</b>	CTRL_LPDDR3_RD_TO_WR_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional read-to-write turnaround time (same rank)</b>	CTRL_LPDDR3_RD_TO_WR_SAME_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a read to a write within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (different ranks)</b>	CTRL_LPDDR3_WR_TO_RD_DIFF_CHIP_DELTA_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a read of another logical rank. This can help resolve bus contention problems specific to your board

*continued...*



Display Name	Identifier	Description
		topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-read turnaround time (same rank)</b>	CTRL_LPDDR3_WR_TO_RD_SAME_CHIP_DELAY_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write to a read within the same logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.
<b>Additional write-to-write turnaround time (different ranks)</b>	CTRL_LPDDR3_WR_TO_WR_DIFF_CHIP_DELAY_CYCS	Specifies additional number of idle controller (not DRAM) cycles when switching the data bus from a write of one logical rank to a write of another logical rank. This can help resolve bus contention problems specific to your board topology. The value is added to the default which is calculated automatically. Use the default setting unless you suspect a problem exists.

#### 7.5.5.8. Stratix 10 EMIF IP LPDDR3 Parameters: Diagnostics

**Table 410. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_LPDDR3_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 411. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug

*continued...*



Display Name	Identifier	Description
		interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface", an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLITE cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Skip address/command deskew calibration</b>	DIAG_LPDDR3_SKIP_CA_DESKEW	Specifies to skip the address/command deskew calibration stage. Address/command deskew performs per-bit deskew for the address and command pins.
<b>Skip address/command leveling calibration</b>	DIAG_LPDDR3_SKIP_CA_LEVEL	Specifies to skip the address/command leveling stage during calibration. Address/command leveling attempts to center the memory clock edge against CS# by adjusting delay elements inside the PHY, and then applying the same delay offset to the rest of the address and command pins.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.

**Table 412. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX DESIGN_IS SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 413. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic

*continued...*



Display Name	Identifier	Description
		generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 414. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 415. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

### 7.5.5.9. Stratix 10 EMIF IP LPDDR3 Parameters: Example Designs

**Table 416. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX DESIGN GUI LPDDR3_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.



**Table 417. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_LPD DR3_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_LPD DR3_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 418. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_LPD DR3_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.

**Table 419. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_LPD DR3_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

### 7.5.5.10. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPOLOGY CAPACITY (INTERNAL-ORGANIZATION)



For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

### 7.5.6. Stratix 10 EMIF IP QDR-IV Parameters

The Stratix 10 EMIF IP parameter editor allows you to parameterize settings for the Stratix 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Stratix 10 EMIF IP.

#### 7.5.6.1. Stratix 10 EMIF IP QDR-IV Parameters: General

**Table 420. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 421. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft

*continued...*



Display Name	Identifier	Description
		Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 422. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Use recommended PLL reference clock frequency</b>	PHY_QDR4_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the

*continued...*



Display Name	Identifier	Description
		clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 423. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

### 7.5.6.2. Stratix 10 EMIF IP QDR-IV Parameters: Memory

**Table 424. Group: Memory / Topology**

Display Name	Identifier	Description
<b>Address width</b>	MEM_QDR4_ADDR_WIDTH	Number of address pins.
<b>DINVA / DINVB width</b>	MEM_QDR4_DINV_PER_PORT_WIDTH	Number of DINV pins for port A or B of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled. Two memory input pins without expansion and four pins with width expansion.
<b>DKA / DKB width</b>	MEM_QDR4_DK_PER_PORT_WIDTH	Number of DK clock pairs for port A or B of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled. Two memory input pins without expansion and four pins with width expansion.
<b>DQ width per device</b>	MEM_QDR4_DQ_PER_PORT_PER_DEVICE	Specifies number of DQ pins per RLDRAM3 device and number of DQ pins per port per QDR IV device. Available widths for DQ are x18 and x36.

*continued...*



Display Name	Identifier	Description
<b>DQA / DQB width</b>	MEM_QDR4_DQ_PER_PORT_WIDTH	Number of DQ pins for port A or B of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled. The interface supports a width expansion configuration up to 72-bits.
<b>QKA / QKB width</b>	MEM_QDR4_QK_PER_PORT_WIDTH	Number of QK clock pairs for port A or B of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled. Two memory input pins without expansion and four pins with width expansion.
<b>Enable width expansion</b>	MEM_QDR4_WIDTH_EXPANDED	Indicates whether to combine two memory devices to double the data bus width. With two devices, the interface supports a width expansion configuration up to 72-bits. For width expansion configuration, the address and control signals are routed to 2 devices.

**Table 425. Group: Memory / Configuration Register Settings**

Display Name	Identifier	Description
<b>ODT (Address/Command)</b>	MEM_QDR4_AC_ODT_MODE_ENUM	Determines the configuration register setting that controls the address/command ODT setting.
<b>Address bus inversion</b>	MEM_QDR4_ADDR_IN_V_ENA	Enable address bus inversion. AINV are all active high at memory device.
<b>ODT (Clock)</b>	MEM_QDR4_CK_ODT_MODE_ENUM	Determines the configuration register setting that controls the clock ODT setting.
<b>Data bus inversion</b>	MEM_QDR4_DATA_IN_V_ENA	Enable data bus inversion for DQ pins. DINVA[1:0] and DINVB[1:0] are all active high. When set to 1, the corresponding bus is inverted at memory device. If the data inversion feature is programmed to be OFF, then the DINVA/DINVB output bits will always be driven to 0.
<b>ODT (Data)</b>	MEM_QDR4_DATA_ODT_MODE_ENUM	Determines the configuration register setting that controls the data ODT setting.
<b>Output drive (pull-down)</b>	MEM_QDR4_PD_OUTPUT_DRIVE_MODE_ENUM	Determines the configuration register setting that controls the pull-down output drive setting.
<b>Output drive (pull-up)</b>	MEM_QDR4_PU_OUTPUT_DRIVE_MODE_ENUM	Determines the configuration register setting that controls the pull-up output drive setting.

### 7.5.6.3. Stratix 10 EMIF IP QDR-IV Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 426. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_QDR4_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal

*continued...*



Display Name	Identifier	Description
		integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_QDR4_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 427. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_QDR4_USER_AC_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR4_USER_AC_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_QDR4_USER_AC_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 428. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_QDR4_USER_CK_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR4_USER_CK_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_QDR4_USER_CK_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.



**Table 429. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_QDR4_USER_AU_TO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_QDR4_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_QDR4_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR4_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_QDR4_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 430. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_QDR4_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_QDR4_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.5.6.4. Stratix 10 EMIF IP QDR-IV Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 431. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_QDR4_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tASH</b>	MEM_QDR4_TASH_PS	tASH provides the setup/hold window requirement for the address bus in relation to the CK clock. Because the individual signals in the address bus may not be perfectly aligned with each other, this parameter describes the intersection window for all the individual address signals setup/hold margins.

*continued...*

<b>Display Name</b>	<b>Identifier</b>	<b>Description</b>
<b>tCKDK_max</b>	MEM_QDR4_TCKDK_M AX_PS	tCKDK_max refers to the maximum skew from the memory clock (CK) to the write strobe (DK).
<b>tCKDK_min</b>	MEM_QDR4_TCKDK_M IN_PS	tCKDK_min refers to the minimum skew from the memory clock (CK) to the write strobe (DK).
<b>tCKQK_max</b>	MEM_QDR4_TCKQK_M AX_PS	tCKQK_max refers to the maximum skew from the memory clock (CK) to the read strobe (QK).
<b>tCSH</b>	MEM_QDR4_TCSH_PS	tCSH provides the setup/hold window requirement for the control bus (LD#, RW#) in relation to the CK clock. Because the individual signals in the control bus may not be perfectly aligned with each other, this parameter describes the intersection window for all the individual control signals setup/hold margins.
<b>tISH</b>	MEM_QDR4_TISH_PS	tISH provides the setup/hold window requirement for the entire data bus (DK or DINV) in all the data groups with respect to the DK clock. After deskew calibration, this parameter describes the intersection window for all the individual data bus signals setup/hold margins.
<b>tQH</b>	MEM_QDR4_TQH_CYC	tQH specifies the output hold time for the DQ/DINV in relation to QK.
<b>tQKQ_max</b>	MEM_QDR4_TQKQ_M AX_PS	tQKQ_max describes the maximum skew between the read strobe (QK) clock edge to the data bus (DQ/DINV) edge.

### 7.5.6.5. Stratix 10 EMIF IP QDR-IV Parameters: Board

**Table 432. Group: Board / Intersymbol Interference/Crosstalk**

<b>Display Name</b>	<b>Identifier</b>	<b>Description</b>
<b>Address and command ISI/crosstalk</b>	BOARD_QDR4_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>QK/QK# ISI/crosstalk</b>	BOARD_QDR4_USER_RCLK_ISI_NS	QK/QK# ISI/crosstalk describes the reduction of the read data window due to intersymbol interference and crosstalk effects on the QK/QK# signal when driven by the memory device during a read. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read DQ ISI/crosstalk</b>	BOARD_QDR4_USER_RDATA_ISI_NS	The reduction of the read data window due to ISI and crosstalk effects on the DQ signal when driven by the memory device during a read. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>DK/DK# ISI/crosstalk</b>	BOARD_QDR4_USER_WCLK_ISI_NS	DK/DK# ISI/crosstalk describes the reduction of the write data window due to intersymbol interference and crosstalk effects on the DK/DK# signal when driven by the FPGA during a write. The number to be entered in the Quartus Prime software is the total of the measured loss of margin

*continued...*



Display Name	Identifier	Description
		on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_QDR4_USER_WDATA_ISI_NS	The reduction of the write data window due to intersymbol interference and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_QDR4_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.

**Table 433. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_QDR4_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Average delay difference between DK and CK</b>	BOARD_QDR4_DK_TO_CK_SKEW_NS	This parameter describes the average delay difference between the DK signals and the CK signal, calculated by averaging the longest and smallest DK trace delay minus the CK trace delay. Positive values represent DK signals that are longer than CK signals and negative values represent DK signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_QDR4_IS_SKew_WITHIN_AC_DESKEWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (QK group)</b>	BOARD_QDR4_IS_SKew_WITHIN_QK_DESKEWED	If you are compensating for package skew on the QK bus in the board layout (hence checking the box here), please include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to device</b>	BOARD_QDR4_MAX_CK_DELAY_NS	The maximum CK delay to device refers to the delay of the longest CK trace from the FPGA to any device.
<b>Maximum DK delay to device</b>	BOARD_QDR4_MAX_DK_DELAY_NS	The maximum DK delay to device refers to the delay of the longest DK trace from the FPGA to any device.
<b>Maximum system skew within address/command bus</b>	BOARD_QDR4_PKG_BRD_SKEW_WITHIN_AC_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.
<b>Maximum system skew within QK group</b>	BOARD_QDR4_PKG_BRD_SKEW_WITHIN_QK_NS	Maximum system skew within QK group refers to the largest skew between all DQ and DM pins in a QK group. This value can affect the read capture and write margins.
<b>Maximum delay difference between devices</b>	BOARD_QDR4_SKew_BETWEEN_DIMMS_NS	This parameter describes the largest propagation delay on the DQ signals between ranks. For example, in a two-rank configuration where devices are placed in series, there is an extra propagation delay for DQ signals going to and coming

*continued...*



Display Name	Identifier	Description
		back from the furthest device compared to the nearest device. This parameter is only applicable when there is more than one rank.
<b>Maximum skew between DK groups</b>	BOARD_QDR4_SKEW_BETWEEN_DK_NS	This parameter describes the largest skew between DK signals in different DK groups.

#### 7.5.6.6. Stratix 10 EMIF IP QDR-IV Parameters: Controller

**Table 434. Group: Controller**

Display Name	Identifier	Description
<b>Generate power-of-2 data bus widths for Qsys</b>	CTRL_QDR4_AVL_ENA_BLE_POWER_OF_TWO_BUS	If enabled, the Avalon data bus width is rounded down to the nearest power-of-2. The width of the symbols within the data bus is also rounded down to the nearest power-of-2. You should only enable this option if you know you will be connecting the memory interface to Qsys interconnect components that require the data bus and symbol width to be a power-of-2. If this option is enabled, you cannot utilize the full density of the memory device. For example, in x36 data width upon selecting this parameter, will define the Avalon data bus to 256-bit. This will ignore the upper 4-bit of data width.
<b>Maximum Avalon-MM burst length</b>	CTRL_QDR4_AVL_MAX_BURST_COUNT	Specifies the maximum burst length on the Avalon-MM bus. This will be used to configure the FIFOs to be able to manage the maximum data burst. More core logic will require for increase in FIFO length.

#### 7.5.6.7. Stratix 10 EMIF IP QDR-IV Parameters: Diagnostics

**Table 435. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_QDR4_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.



**Table 436. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_A_VALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_A_VALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface", an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLITE cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Skip VREF_in calibration</b>	DIAG_QDR4_SKIP_VREF_CAL	Users can check this option to skip the VREF stage of calibration. Users should enable this option for debug purpose only; it is recommended to keep enabled during normal operation.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.

**Table 437. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX_DESIGN_ISP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX_DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 438. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 439. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 440. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

#### 7.5.6.8. Stratix 10 EMIF IP QDR-IV Parameters: Example Designs



**Table 441. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_QDR_4_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 442. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_QDR_4_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_QDR_4_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 443. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_QDR_4_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.

**Table 444. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_QDR_4_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.



### 7.5.6.9. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPOLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

### 7.5.7. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters

The Stratix 10 EMIF IP parameter editor allows you to parameterize settings for the Stratix 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.

The following tables describe the parameterization settings available in the parameter editor for the Stratix 10 EMIF IP.

#### 7.5.7.1. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: General

**Table 445. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.



**Table 446. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 447. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.
<b>Use recommended PLL reference clock frequency</b>	PHY_QDR2_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the

*continued...*

Display Name	Identifier	Description
		clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 448. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

### 7.5.7.2. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Memory

**Table 449. Group: Memory / Topology**

Display Name	Identifier	Description
<b>Address width</b>	MEM_QDR2_ADDR_WIDTH	Number of address pins.
<b>Burst length</b>	MEM_QDR2_BL	Burst length of the memory device.
<b>Enable BWS# pins</b>	MEM_QDR2_BWS_EN	Indicates whether the interface uses the BWS#( Byte Write Select) pins. If enabled, 1 BWS# pin for every 9 D pins will be added.
<b>BWS# width</b>	MEM_QDR2_BWS_N_WIDTH	Number of BWS# (Byte Write Select) pins of the memory interface. Automatically calculated based on the data width per device and whether width expansion is enabled. BWS# pins are used to select which byte is written into the device during the current portion of the write operations. Bytes not written remain unaltered.
<b>CQ width</b>	MEM_QDR2_CQ_WIDTH	Width of the CQ (read strobe) clock on the memory device.
<b>Data width per device</b>	MEM_QDR2_DATA_PER_DEVICE	Number of D and Q pins per QDR II device.

*continued...*



Display Name	Identifier	Description
<b>Data width</b>	MEM_QDR2_DATA_WIDTH	Number of D and Q pins of the memory interface. Automatically calculated based on the D and Q width per device and whether width expansion is enabled.
<b>K width</b>	MEM_QDR2_K_WIDTH	Width of the K (address, command and write strobe) clock on the memory device.
<b>Enable width expansion</b>	MEM_QDR2_WIDTH_EXPANDED	Indicates whether to combine two memory devices to double the data bus width. With two devices, the interface supports a width expansion configuration up to 72-bits. For width expansion configuration, the address and control signals are routed to 2 devices.

### 7.5.7.3. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 450. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_QDR2_DEFAULT_IO	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_QDR2_IO_VOLTAGE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC_OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 451. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_QDR2_USER_ADDRESS_COMMAND_IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR2_USER_ADDRESS_COMMAND_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_QDR2_USER_ADDRESS_COMMAND_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 452. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_QDR2_USER_CK_IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR2_USER_CK_MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_QDR2_USER_CK_SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 453. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_QDR2_USER_AU_TO_STARTING_VREFIN_N_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_QDR2_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_QDR2_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_QDR2_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_QDR2_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 454. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_QDR2_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_QDR2_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.



#### 7.5.7.4. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 455. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Internal Jitter</b>	MEM_QDR2_INTERNAL_JITTER_NS	QDRII internal jitter.
<b>Speed bin</b>	MEM_QDR2_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tCCQO</b>	MEM_QDR2_TCCQO_NS	tCCQO describes the skew between the rising edge of the C clock to the rising edge of the echo clock (CQ) in QDRII memory devices.
<b>tCQDOH</b>	MEM_QDR2_TCQDOH_NS	tCQDOH refers to the minimum time expected between the echo clock (CQ or CQ#) edge and the last of the valid Read data (Q).
<b>tCQD</b>	MEM_QDR2_TCQD_NS	tCQD refers to the maximum time expected between an echo clock edge and valid data on the Read Data bus (Q).
<b>tCQH</b>	MEM_QDR2_TCQH_NS	tCQH describes the time period during which the echo clock (CQ, #CQ) is considered logically high.
<b>tHA</b>	MEM_QDR2_THA_NS	tHA refers to the hold time after the rising edge of the clock (K) to the address and command control bus (A). The address and command control bus must remain stable for at least tHA after the rising edge of K.
<b>tHD</b>	MEM_QDR2_THD_NS	tHD refers to the hold time after the rising edge of the clock (K) to the data bus (D). The data bus must remain stable for at least tHD after the rising edge of K.
<b>tRL</b>	MEM_QDR2TRL_CYC	tRL refers to the QDR memory specific read latency. This parameter describes the length of time after a Read command has been registered on the rising edge of the Write Clock (K) at the QDR memory before the first piece of read data (Q) can be expected at the output of the memory. It is measured in Write Clock (K) cycles. The Read Latency is specific to a QDR memory device and cannot be modified to a different value. The Read Latency (tRL) can have the following values: 1.5, 2, 2.5 clk cycles.
<b>tSA</b>	MEM_QDR2_TSA_NS	tSA refers to the setup time for the address and command bus (A) before the rising edge of the clock (K). The address and command bus must be stable for at least tSA before the rising edge of K.
<b>tSD</b>	MEM_QDR2_TSD_NS	tSD refers to the setup time for the data bus (D) before the rising edge of the clock (K). The data bus must be stable for at least tSD before the rising edge of K.
<b>tWL</b>	MEM_QDR2_TWLCYC	tWL refers to the write latency requirement at the QDR memory. This parameter describes the length of time after a Write command has been registered at the memory on the rising edge of the Write clock (K) before the memory expects the Write Data (D). It is measured in (K) clock cycles and is usually 1.

#### 7.5.7.5. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Board

**Table 456. Group: Board / Intersymbol Interference/Crosstalk**

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_QDR2_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>CQ/CQ# ISI/crosstalk</b>	BOARD_QDR2_USER_RCLK_ISI_NS	CQ/CQ# ISI/crosstalk describes the reduction of the read data window due to intersymbol interference and crosstalk effects on the CQ/CQ# signal when driven by the memory device during a read. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Read Q ISI/crosstalk</b>	BOARD_QDR2_USER_RDATA_ISI_NS	Read Q ISI/crosstalk describes the reduction of the read data window due to intersymbol interference and crosstalk effects on the CQ/CQ# signal when driven by the memory device during a read. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>K/K# ISI/crosstalk</b>	BOARD_QDR2_USER_WCLK_ISI_NS	K/K# ISI/crosstalk describes the reduction of the write data window due to intersymbol interference and crosstalk effects on the K/K# signal when driven by the FPGA during a write. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write D ISI/crosstalk</b>	BOARD_QDR2_USER_WDATA_ISI_NS	Write D ISI/crosstalk describes the reduction of the write data window due to intersymbol interference and crosstalk effects on the signal when driven by driven by the FPGA during a write. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_QDR2_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.

**Table 457. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and K</b>	BOARD_QDR2_AC_TO_K_SKEW_NS	This parameter refers to the average delay difference between the Address/Command signals and the K signal, calculated by averaging the longest and smallest Address/Command trace delay minus the maximum K trace delay. Positive values represent address and command signals that

*continued...*



Display Name	Identifier	Description
		are longer than K signals and negative values represent address and command signals that are shorter than K signals.
<b>Maximum board skew within D group</b>	BOARD_QDR2_BRD_SKEW_WITHIN_D_NS	This parameter refers to the largest skew between all D and BWS# signals in a D group. D pins are used for driving data signals to the memory device during a write operation. BWS# pins are used as Byte Write Select signals to control which byte(s) are written to the memory during a write operation. Users should enter their board skew only. Package skew will be calculated automatically, based on the memory interface configuration, and added to this value. This value affects the read capture and write margins.a
<b>Maximum board skew within Q group</b>	BOARD_QDR2_BRD_SKEW_WITHIN_Q_NS	This parameter describes the largest skew between all Q signals in a Q group. Q pins drive the data signals from the memory to the FPGA when the read operation is active. Users should enter their board skew only. Package skew will be calculated automatically, based on the memory interface configuration, and added to this value. This value affects the read capture and write margins.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_QDR2_IS_SKEW_WITHIN_AC_DESK_EWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (D group)</b>	BOARD_QDR2_IS_SKEW_WITHIN_D_DESK_EWED	If you are compensating for package skew on the D and BWS# signals in the board layout (hence checking the box here), please include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (Q group)</b>	BOARD_QDR2_IS_SKEW_WITHIN_Q_DESK_EWED	If you are compensating for package skew on the Q bus in the board layout (hence checking the box here), please include package skew in calculating the following board skew parameters.
<b>Maximum K delay to device</b>	BOARD_QDR2_MAX_K_DELAY_NS	The maximum K delay to device refers to the delay of the longest K trace from the FPGA to any device
<b>Maximum system skew within address/command bus</b>	BOARD_QDR2_PKG_BUS_SKEW_WITHIN_ADDRESS_COMMAND_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.

#### 7.5.7.6. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Controller

**Table 458. Group: Controller**

Display Name	Identifier	Description
<b>Generate power-of-2 data bus widths for Qsys</b>	CTRL_QDR2_AVL_ENA_BLE_POWER_OF_TWO_BUS	If enabled, the Avalon data bus width is rounded down to the nearest power-of-2. The width of the symbols within the data bus is also rounded down to the nearest power-of-2. You should only enable this option if you know you will be connecting the memory interface to Qsys interconnect components that require the data bus and symbol width to be a power-of-2. If this option is enabled, you cannot utilize the full density of the memory device. For example, in x36 data width upon selecting this parameter, will define the Avalon data bus to 256-bit. This will ignore the upper 4-bit of data width.
<b>Maximum Avalon-MM burst length</b>	CTRL_QDR2_AVL_MAX_BURST_COUNT	Specifies the maximum burst length on the Avalon-MM bus. This will be used to configure the FIFOs to be able to manage the maximum data burst. More core logic will require for increase in FIFO length.



### 7.5.7.7. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Diagnostics

**Table 459. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_QDR2_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 460. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface," an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLite cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.



**Table 461. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX_DESIGN_IS_SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX_DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 462. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 463. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 464. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

### 7.5.7.8. Stratix 10 EMIF IP QDR II/II+/II+ Xtreme Parameters: Example Designs

**Table 465. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_QDR2_SEL DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 466. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_QDR2_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_QDR2_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 467. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_QDR2_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.



**Table 468. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_QDR2_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

### 7.5.7.9. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.

When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

### 7.5.8. Stratix 10 EMIF IP RLDRAM 3 Parameters

The Stratix 10 EMIF IP parameter editor allows you to parameterize settings for the Stratix 10 EMIF IP.

The text window at the bottom of the parameter editor displays information about the memory interface, as well as warning and error messages. You should correct any errors indicated in this window before clicking the **Finish** button.

**Note:** Default settings are the minimum required to achieve timing, and may vary depending on memory protocol.



The following tables describe the parameterization settings available in the parameter editor for the Stratix 10 EMIF IP.

### 7.5.8.1. Stratix 10 EMIF IP RLDRAM 3 Parameters: General

**Table 469. Group: General / FPGA**

Display Name	Identifier	Description
<b>Speed grade</b>	PHY_FPGA_SPEEDGRADE_GUI	Indicates the device speed grade, and whether it is an engineering sample (ES) or production device. This value is based on the device that you select in the parameter editor. If you do not specify a device, the system assumes a default value. Ensure that you always specify the correct device during IP generation, otherwise your IP may not work in hardware.

**Table 470. Group: General / Interface**

Display Name	Identifier	Description
<b>Configuration</b>	PHY_CONFIG_ENUM	Specifies the configuration of the memory interface. The available options depend on the protocol in use. Options include Hard PHY and Hard Controller, Hard PHY and Soft Controller, or Hard PHY only. If you select Hard PHY only, the AFI interface is exported to allow connection of a custom memory controller or third-party IP.
<b>Instantiate two controllers sharing a Ping Pong PHY</b>	PHY_PING_PONG_EN	Specifies the instantiation of two identical memory controllers that share an address/command bus through the use of Ping Pong PHY. This parameter is available only if you specify the Hard PHY and Hard Controller option. When this parameter is enabled, the IP exposes two independent Avalon interfaces to the user logic, and a single external memory interface with double width for the data bus and the CS#, CKE, ODT, and CK/CK# signals.

**Table 471. Group: General / Clocks**

Display Name	Identifier	Description
<b>Core clocks sharing</b>	PHY_CORE_CLKS_SHARING_ENUM	When a design contains multiple interfaces of the same protocol, rate, frequency, and PLL reference clock source, they can share a common set of core clock domains. By sharing core clock domains, they reduce clock network usage and avoid clock synchronization logic between the interfaces. To share core clocks, denote one of the interfaces as "Master", and the remaining interfaces as "Slave". In the RTL, connect the clks_sharing_master_out signal from the master interface to the clks_sharing_slave_in signal of all the slave interfaces. Both master and slave interfaces still expose their own output clock ports in the RTL (for example, emif_usr_clk, afi_clk), but the physical signals are equivalent, hence it does not matter whether a clock port from a master or a slave is used. As the combined width of all interfaces sharing the same core clock increases, you may encounter timing closure difficulty for transfers between the FPGA core and the periphery.
<b>Memory clock frequency</b>	PHY_MEM_CLK_FREQ_MHZ	Specifies the operating frequency of the memory interface in MHz. If you change the memory frequency, you should update the memory latency parameters on the "Memory" tab and the memory timing parameters on the "Mem Timing" tab.

*continued...*



Display Name	Identifier	Description
<b>Clock rate of user logic</b>	PHY_RATE_ENUM	Specifies the relationship between the user logic clock frequency and the memory clock frequency. For example, if the memory clock sent from the FPGA to the memory device is toggling at 800MHz, a quarter-rate interface means that the user logic in the FPGA runs at 200MHz.
<b>PLL reference clock frequency</b>	PHY_REF_CLK_FREQ_MHZ	Specifies the PLL reference clock frequency. You must configure this parameter only if you do not check the "Use recommended PLL reference clock frequency" parameter. To configure this parameter, select a valid PLL reference clock frequency from the list. The values in the list can change if you change the memory interface frequency and/or the clock rate of the user logic. For best jitter performance, you should use the fastest possible PLL reference clock frequency.
<b>PLL reference clock jitter</b>	PHY_REF_CLK_JITTER_PS	Specifies the peak-to-peak jitter on the PLL reference clock source. The clock source of the PLL reference clock must meet or exceed the following jitter requirements: 10ps peak to peak, or 1.42ps RMS at 1e-12 BER, 1.22ps at 1e-16 BER.
<b>Use recommended PLL reference clock frequency</b>	PHY_RLD3_DEFAULT_REF_CLK_FREQ	Specifies that the PLL reference clock frequency is automatically calculated for best performance. If you want to specify a different PLL reference clock frequency, uncheck the check box for this parameter.
<b>Specify additional core clocks based on existing PLL</b>	PLL_ADD_EXTRA_CLKS	Displays additional parameters allowing you to create additional output clocks based on the existing PLL. This parameter provides an alternative clock-generation mechanism for when your design exhausts available PLL resources. The additional output clocks that you create can be fed into the core. Clock signals created with this parameter are synchronous to each other, but asynchronous to the memory interface core clock domains (such as emif_usr_clk or afi_clk). You must follow proper clock-domain-crossing techniques when transferring data between clock domains.

**Table 472. Group: General / Additional Core Clocks**

Display Name	Identifier	Description
<b>Number of additional core clocks</b>	PLL_USER_NUM_OF_EXTRA_CLKS	Specifies the number of additional output clocks to create from the PLL.

### 7.5.8.2. Stratix 10 EMIF IP RLDRAM 3 Parameters: Memory

**Table 473. Group: Memory / Topology**

Display Name	Identifier	Description
<b>Address width</b>	MEM_RLD3_ADDR_WIDTH	Number of address pins.
<b>CS# width</b>	MEM_RLD3_CS_WIDTH	Number of chip selects of the memory interface.
<b>Enable depth expansion using twin die package</b>	MEM_RLD3_DEPTH_EXPANDED	Indicates whether to combine two RLDRAM3 devices to double the address space, resulting in more density.
<b>DK width</b>	MEM_RLD3_DK_WIDTH	Number of DK clock pairs of the memory interface. This is equal to the number of write data groups, and is automatically calculated based on the DQ width per device and whether width expansion is enabled.
<i>continued...</i>		



Display Name	Identifier	Description
<b>DQ width per device</b>	MEM_RLD3_DQ_PER_DEVICE	Specifies number of DQ pins per RLDRAM3 device and number of DQ pins per port per QDR IV device. Available widths for DQ are x18 and x36.
<b>DQ width</b>	MEM_RLD3_DQ_WIDTH	Number of data pins of the memory interface. Automatically calculated based on the DQ width per device and whether width expansion is enabled.
<b>QK width</b>	MEM_RLD3_QK_WIDTH	Number of QK output clock pairs of the memory interface. This is equal to the number of read data groups, and is automatically calculated based on the DQ width per device and whether width expansion is enabled.
<b>Enable width expansion</b>	MEM_RLD3_WIDTH_EXPANDED	Indicates whether to combine two memory devices to double the data bus width. With two devices, the interface supports a width expansion configuration up to 72-bits. For width expansion configuration, the address and control signals are routed to 2 devices.

**Table 474. Group: Memory / Mode Register Settings**

Display Name	Identifier	Description
<b>AREF protocol</b>	MEM_RLD3_AREF_PROTOCOL_ENUM	Determines the mode register setting that controls the AREF protocol setting. The AUTO REFRESH (AREF) protocol is selected by setting mode register 1. There are two ways in which AREF commands can be issued to the RLDRAM, the memory controller can either issue bank address-controlled or multibank AREF commands. Multibank refresh protocol allows for the simultaneous refreshing of a row in up to four banks
<b>Data Latency</b>	MEM_RLD3_DATA_LATENCY_MODE_ENUM	Determines the mode register setting that controls the data latency. Sets both READ and WRITE latency (RL and WL).
<b>ODT</b>	MEM_RLD3_ODT_MODE_ENUM	Determines the mode register setting that controls the ODT setting.
<b>Output drive</b>	MEM_RLD3_OUTPUT_DRIVE_MODE_ENUM	Determines the mode register setting that controls the output drive setting.
<b>tRC</b>	MEM_RLD3_T_RC_MODE_ENUM	Determines the mode register setting that controls the tRC(activate to activate timing parameter). Refer to the tRC table in the memory vendor data sheet. Set the tRC according to the memory speed grade and data latency. Full name of tRC
<b>Write protocol</b>	MEM_RLD3_WRITE_PROTOCOL_ENUM	Determines the mode register setting that controls the write protocol setting. When multiple bank (dual bank or quad bank) is selected, identical data is written to multiple banks.

### 7.5.8.3. Stratix 10 EMIF IP RLDRAM 3 Parameters: FPGA I/O

You should use Hyperlynx\* or similar simulators to determine the best settings for your board. Refer to the EMIF Simulation Guidance wiki page for additional information.

**Table 475. Group: FPGA IO / FPGA IO Settings**

Display Name	Identifier	Description
<b>Use default I/O settings</b>	PHY_RLD3_DEFAULT_I_O	Specifies that a legal set of I/O settings are automatically selected. The default I/O settings are not necessarily optimized for a specific board. To achieve optimal signal <b>continued...</b>



Display Name	Identifier	Description
		integrity, perform I/O simulations with IBIS models and enter the I/O settings manually, based on simulation results.
<b>Voltage</b>	PHY_RLD3_IO_VOLTA GE	The voltage level for the I/O pins driving the signals between the memory device and the FPGA memory interface.
<b>Periodic OCT re-calibration</b>	PHY_USER_PERIODIC _OCT_RECAL_ENUM	Specifies that the system periodically recalibrate on-chip termination (OCT) to minimize variations in termination value caused by changing operating conditions (such as changes in temperature). By recalibrating OCT, I/O timing margins are improved. When enabled, this parameter causes the PHY to halt user traffic about every 0.5 seconds for about 1900 memory clock cycles, to perform OCT recalibration. Efficiency is reduced by about 1% when this option is enabled.

**Table 476. Group: FPGA IO / Address/Command**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_RLD3_USER_AC_ IO_STD_ENUM	Specifies the I/O electrical standard for the address/command pins of the memory interface. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_RLD3_USER_AC_ MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_RLD3_USER_AC_ SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 477. Group: FPGA IO / Memory Clock**

Display Name	Identifier	Description
<b>I/O standard</b>	PHY_RLD3_USER_CK_ IO_STD_ENUM	Specifies the I/O electrical standard for the memory clock pins. The selected I/O standard configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_RLD3_USER_CK_ MODE_ENUM	This parameter allows you to change the current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Slew rate</b>	PHY_RLD3_USER_CK_ SLEW_RATE_ENUM	Specifies the slew rate of the address/command output pins. The slew rate (or edge rate) describes how quickly the signal can transition, measured in voltage per unit time. Perform board simulations to determine the slew rate that provides the best eye opening for the address and command signals.

**Table 478. Group: FPGA IO / Data Bus**

Display Name	Identifier	Description
<b>Use recommended initial Vrefin</b>	PHY_RLD3_USER_AUTO_STARTING_VREFIN_EN	Specifies that the initial Vrefin setting is calculated automatically, to a reasonable value based on termination settings.
<b>Input mode</b>	PHY_RLD3_USER_DATA_IN_MODE_ENUM	This parameter allows you to change the input termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>I/O standard</b>	PHY_RLD3_USER_DATA_IO_STD_ENUM	Specifies the I/O electrical standard for the data and data clock/strobe pins of the memory interface. The selected I/O standard option configures the circuit within the I/O buffer to match the industry standard.
<b>Output mode</b>	PHY_RLD3_USER_DATA_OUT_MODE_ENUM	This parameter allows you to change the output current drive strength or termination settings for the selected I/O standard. Perform board simulation with IBIS models to determine the best settings for your design.
<b>Initial Vrefin</b>	PHY_RLD3_USER_STARTING_VREFIN	Specifies the initial value for the reference voltage on the data pins (Vrefin). This value is entered as a percentage of the supply voltage level on the I/O pins. The specified value serves as a starting point and may be overridden by calibration to provide better timing margins. If you choose to skip Vref calibration (Diagnostics tab), this is the value that is used as the Vref for the interface.

**Table 479. Group: FPGA IO / PHY Inputs**

Display Name	Identifier	Description
<b>PLL reference clock I/O standard</b>	PHY_RLD3_USER_PLL_REF_CLK_IO_STD_ENUM	Specifies the I/O standard for the PLL reference clock of the memory interface.
<b>RZQ I/O standard</b>	PHY_RLD3_USER_RZQ_IO_STD_ENUM	Specifies the I/O standard for the RZQ pin used in the memory interface.
<b>RZQ resistor</b>	PHY_RZQ	Specifies the reference resistor used to calibrate the on-chip termination value. You should connect the RZQ pin to GND through an external resistor of the specified value.

#### 7.5.8.4. Stratix 10 EMIF IP RLDRAM 3 Parameters: Mem Timing

These parameters should be read from the table in the datasheet associated with the speed bin of the memory device (not necessarily the frequency at which the interface is running).

**Table 480. Group: Mem Timing / Parameters dependent on Speed Bin**

Display Name	Identifier	Description
<b>Speed bin</b>	MEM_RLD3_SPEEDBIN_ENUM	The speed grade of the memory device used. This parameter refers to the maximum rate at which the memory device is specified to run.
<b>tCKDK_max</b>	MEM_RLD3_TCKDK_MAX_CYC	tCKDK_max refers to the maximum skew from the memory clock (CK) to the write strobe (DK).
<b>tCKDK_min</b>	MEM_RLD3_TCKDK_MIN_CYC	tCKDK_min refers to the minimum skew from the memory clock (CK) to the write strobe (DK).
<b>tCKQK_max</b>	MEM_RLD3_TCKQK_MAX_PS	tCKQK_max refers to the maximum skew from the memory clock (CK) to the read strobe (QK).

*continued...*



Display Name	Identifier	Description
<b>tDH (base) DC level</b>	MEM_RLD3_TD�_DC_MV	tDH (base) DC level refers to the voltage level which the data bus must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tDH (base)</b>	MEM_RLD3_TD�_PS	tDH (base) refers to the hold time for the Data (DQ) bus after the rising edge of CK.
<b>tDS (base) AC level</b>	MEM_RLD3_TDS_AC_MV	tDS (base) AC level refers to the voltage level which the data bus must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tDS (base)</b>	MEM_RLD3_TDS_PS	tDS(base) refers to the setup time for the Data (DQ) bus before the rising edge of the DQS strobe.
<b>tIH (base) DC level</b>	MEM_RLD3_TIH_DC_MV	tIH (base) DC level refers to the voltage level which the address/command signal must not cross during the hold window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire hold period.
<b>tIH (base)</b>	MEM_RLD3_TIH_PS	tIH (base) refers to the hold time for the Address/Command (A) bus after the rising edge of CK. Depending on what AC level the user has chosen for a design, the hold margin can vary (this variance will be automatically determined when the user chooses the "tIH (base) AC level").
<b>tIS (base) AC level</b>	MEM_RLD3_TIS_AC_MV	tIS (base) AC level refers to the voltage level which the address/command signal must cross and remain above during the setup margin window. The signal is considered stable only if it remains above this voltage level (for a logic 1) or below this voltage level (for a logic 0) for the entire setup period.
<b>tIS (base)</b>	MEM_RLD3_TIS_PS	tIS (base) refers to the setup time for the Address/Command/Control (A) bus to the rising edge of CK.
<b>tQH</b>	MEM_RLD3_TQH_CYC	tQH specifies the output hold time for the DQ/DINV in relation to QK.
<b>tQKQ_max</b>	MEM_RLD3_TQKQ_MAX_PS	tQKQ_max describes the maximum skew between the read strobe (QK) clock edge to the data bus (DQ/DINV) edge.

### 7.5.8.5. Stratix 10 EMIF IP RLDRAM 3 Parameters: Board

Table 481. Group: Board / Intersymbol Interference/Crosstalk

Display Name	Identifier	Description
<b>Address and command ISI/crosstalk</b>	BOARD_RLD3_USER_AC_ISI_NS	The address and command window reduction due to ISI and crosstalk effects. The number to be entered is the total loss of margin on both the setup and hold sides (measured loss on the setup side + measured loss on the hold side). Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>QK/QK# ISI/crosstalk</b>	BOARD_RLD3_USER_RCLK_ISI_NS	QK/QK# ISI/crosstalk describes the reduction of the read data window due to intersymbol interference and crosstalk effects on the QK/QK# signal when driven by the memory device during a read. The number to be entered in the Quartus Prime software is the total of the measured loss of

*continued...*



Display Name	Identifier	Description
		margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>DK/DK# ISI/crosstalk</b>	BOARD_RLD3_USER_WCLK_ISI_NS	DK/DK# ISI/crosstalk describes the reduction of the write data window due to intersymbol interference and crosstalk effects on the DK/DK# signal when driven by the FPGA during a write. The number to be entered in the Quartus Prime software is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Write DQ ISI/crosstalk</b>	BOARD_RLD3_USER_WDATA_ISI_NS	The reduction of the write data window due to ISI and crosstalk effects on the DQ signal when driven by the FPGA during a write. The number to be entered is the total of the measured loss of margin on the setup side plus the measured loss of margin on the hold side. Refer to the EMIF Simulation Guidance wiki page for additional information.
<b>Use default ISI/crosstalk values</b>	BOARD_RLD3_USE_DEFAULT_ISI_VALUES	You can enable this option to use default intersymbol interference and crosstalk values for your topology. Note that the default values are not optimized for your board. For optimal signal integrity, it is recommended that you do not enable this parameter, but instead perform I/O simulation using IBIS models and Hyperlynx*, and manually enter values based on your simulation results, instead of using the default values.

**Table 482. Group: Board / Board and Package Skews**

Display Name	Identifier	Description
<b>Average delay difference between address/command and CK</b>	BOARD_RLD3_AC_TO_CK_SKEW_NS	The average delay difference between the address/command signals and the CK signal, calculated by averaging the longest and smallest address/command signal trace delay minus the maximum CK trace delay. Positive values represent address and command signals that are longer than CK signals and negative values represent address and command signals that are shorter than CK signals.
<b>Maximum board skew within QK group</b>	BOARD_RLD3_BRD_SKEW_WITHIN_QK_NS	Maximum board skew within QK group refers to the largest skew between all DQ and DM pins in a QK group. This value can affect the read capture and write margins.
<b>Average delay difference between DK and CK</b>	BOARD_RLD3_DK_TO_CK_SKEW_NS	This parameter describes the average delay difference between the DK signals and the CK signal, calculated by averaging the longest and smallest DK trace delay minus the CK trace delay. Positive values represent DK signals that are longer than CK signals and negative values represent DK signals that are shorter than CK signals.
<b>Package deskewed with board layout (address/command bus)</b>	BOARD_RLD3_IS_SKEW_WITHIN_AC_DESK_EWED	Enable this parameter if you are compensating for package skew on the address, command, control, and memory clock buses in the board layout. Include package skew in calculating the following board skew parameters.
<b>Package deskewed with board layout (QK group)</b>	BOARD_RLD3_IS_SKEW_WITHIN_QK_DESK_EWED	If you are compensating for package skew on the QK bus in the board layout (hence checking the box here), please include package skew in calculating the following board skew parameters.
<b>Maximum CK delay to device</b>	BOARD_RLD3_MAX_CK_DELAY_NS	The maximum CK delay to device refers to the delay of the longest CK trace from the FPGA to any device.
<b>Maximum DK delay to device</b>	BOARD_RLD3_MAX_DK_DELAY_NS	The maximum DK delay to device refers to the delay of the longest DK trace from the FPGA to any device.

*continued...*



Display Name	Identifier	Description
<b>Maximum system skew within address/command bus</b>	BOARD_RLD3_PKG_BRD_SKew_WITHIN_A_C_NS	Maximum system skew within address/command bus refers to the largest skew between the address and command signals.
<b>Maximum delay difference between devices</b>	BOARD_RLD3_SKew_BETWEEN_DIMMS_NS	This parameter describes the largest propagation delay on the DQ signals between ranks. For example, in a two-rank configuration where devices are placed in series, there is an extra propagation delay for DQ signals going to and coming back from the furthest device compared to the nearest device. This parameter is only applicable when there is more than one rank.
<b>Maximum skew between DK groups</b>	BOARD_RLD3_SKew_BETWEEN_DK_NS	This parameter describes the largest skew between DK signals in different DK groups.

#### 7.5.8.6. Stratix 10 EMIF IP RLDRAM 3 Parameters: Diagnostics

**Table 483. Group: Diagnostics / Simulation Options**

Display Name	Identifier	Description
<b>Abstract phy for fast simulation</b>	DIAG_RLD3_ABSTRACT_PHY	Specifies that the system use Abstract PHY for simulation. Abstract PHY replaces the PHY with a model for fast simulation and can reduce simulation time by 2-3 times. Abstract PHY is available for certain protocols and device families, and only when you select Skip Calibration.
<b>Calibration mode</b>	DIAG_SIM_CAL_MODE_ENUM	Specifies whether to skip memory interface calibration during simulation, or to simulate the full calibration process. Simulating the full calibration process can take hours (or even days), depending on the width and depth of the memory interface. You can achieve much faster simulation times by skipping the calibration process, but that is only expected to work when the memory model is ideal and the interconnect delays are zero. If you enable this parameter, the interface still performs some memory initialization before starting normal operations. Abstract PHY is supported with skip calibration.

**Table 484. Group: Diagnostics / Calibration Debug Options**

Display Name	Identifier	Description
<b>Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_MASTER	Specifies that the IP export an Avalon-MM master interface (cal_debug_out) which can connect to the cal_debug interface of other EMIF cores residing in the same I/O column. This parameter applies only if the EMIF Debug Toolkit or On-Chip Debug Port is enabled. Refer to the Debugging Multiple EMIFs wiki page for more information about debugging multiple EMIFs.
<b>Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port</b>	DIAG_EXPORT_SEQ_AVALON_SLAVE	Specifies the connectivity of an Avalon slave interface for use by the Quartus Prime EMIF Debug Toolkit or user core logic. If you set this parameter to "Disabled," no debug features are enabled. If you set this parameter to "Export," an Avalon slave interface named "cal_debug" is exported from the IP. To use this interface with the EMIF Debug Toolkit, you must instantiate and connect an EMIF debug interface IP core to it, or connect it to the cal_debug_out interface of another EMIF core. If you select "Add EMIF Debug Interface", an EMIF debug interface component containing a JTAG Avalon Master is connected to the debug port, allowing the core to be accessed by the EMIF Debug

*continued...*



Display Name	Identifier	Description
		Toolkit. Only one EMIF debug interface should be instantiated per I/O column. You can chain additional EMIF or PHYLite cores to the first by enabling the "Enable Daisy-Chaining for Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option for all cores in the chain, and selecting "Export" for the "Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port" option on all cores after the first.
<b>Interface ID</b>	DIAG_INTERFACE_ID	Identifies interfaces within the I/O column, for use by the EMIF Debug Toolkit and the On-Chip Debug Port. Interface IDs should be unique among EMIF cores within the same I/O column. If the Quartus Prime EMIF Debug Toolkit/On-Chip Debug Port parameter is set to Disabled, the interface ID is unused.
<b>Use Soft NIOS Processor for On-Chip Debug</b>	DIAG_SOFT_NIOS_MODE	Enables a soft Nios processor as a peripheral component to access the On-Chip Debug Port. Only one interface in a column can activate this option.

**Table 485. Group: Diagnostics / Example Design**

Display Name	Identifier	Description
<b>Enable In-System-Sources-and-Probes</b>	DIAG_EX DESIGN_IS SP_EN	Enables In-System-Sources-and-Probes in the example design for common debug signals, such as calibration status or example traffic generator per-bit status. This parameter must be enabled if you want to do driver margining.
<b>Number of core clocks sharing slaves to instantiate in the example design</b>	DIAG_EX DESIGN_NUM_OF_SLAVES	Specifies the number of core clock sharing slaves to instantiate in the example design. This parameter applies only if you set the "Core clocks sharing" parameter in the "General" tab to Master or Slave.

**Table 486. Group: Diagnostics / Traffic Generator**

Display Name	Identifier	Description
<b>Bypass the default traffic pattern</b>	DIAG_BYPASS_DEFAULT_PATTERN	Specifies that the controller/interface bypass the traffic generator 2.0 default pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the traffic generator repeated-writes/repeated-reads test pattern</b>	DIAG_BYPASS_REPEAT_STAGE	Specifies that the controller/interface bypass the traffic generator's repeat test stage. If you do not enable this parameter, every write and read is repeated several times.
<b>Bypass the traffic generator stress pattern</b>	DIAG_BYPASS_STRESS_STAGE	Specifies that the controller/interface bypass the traffic generator's stress pattern stage. (Stress patterns are meant to create worst-case signal integrity patterns on the data pins.) If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface.
<b>Bypass the user-configured traffic stage</b>	DIAG_BYPASS_USER_STAGE	Specifies that the controller/interface bypass the user-configured traffic generator's pattern after reset. If you do not enable this parameter, the traffic generator does not assert a pass or fail status until the generator is configured and signaled to start by its Avalon configuration interface. Configuration can be done by connecting to the traffic generator via the EMIF Debug Toolkit, or by using custom logic connected to the Avalon-MM configuration slave port on the traffic generator. Configuration can also be simulated using the example testbench provided in the altera_emif_avl_tg_2_tb.sv file.

*continued...*



Display Name	Identifier	Description
<b>Run diagnostic on infinite test duration</b>	DIAG_INFI_TG2_ERR_TEST	Specifies that the traffic generator run indefinitely until the first error is detected.
<b>Export Traffic Generator 2.0 configuration interface</b>	DIAG_TG_AVL_2_EXPORT_CFG_INTERFACE	Specifies that the IP export an Avalon-MM slave port for configuring the Traffic Generator. This is required only if you are configuring the traffic generator through user logic and not through the EMIF Debug Toolkit.
<b>Use configurable Avalon traffic generator 2.0</b>	DIAG_USE_TG_AVL_2	This option allows users to add the new configurable Avalon traffic generator to the example design.

**Table 487. Group: Diagnostics / Performance**

Display Name	Identifier	Description
<b>Enable Efficiency Monitor</b>	DIAG_EFFICIENCY_MONITOR	Adds an Efficiency Monitor component to the Avalon-MM interface of the memory controller, allowing you to view efficiency statistics of the interface. You can access the efficiency statistics using the EMIF Debug Toolkit.

**Table 488. Group: Diagnostics / Miscellaneous**

Display Name	Identifier	Description
<b>Use short Qsys interface names</b>	SHORT_QSYS_INTERFACE_NAMES	Specifies the use of short interface names, for improved usability and consistency with other Qsys components. If this parameter is disabled, the names of Qsys interfaces exposed by the IP will include the type and direction of the interface. Long interface names are supported for backward-compatibility and will be removed in a future release.

### 7.5.8.7. Stratix 10 EMIF IP RLDRAM 3 Parameters: Example Designs

**Table 489. Group: Example Designs / Available Example Designs**

Display Name	Identifier	Description
<b>Select design</b>	EX_DESIGN_GUI_RLD3_SEL_DESIGN	Specifies the creation of a full Quartus Prime project, instantiating an external memory interface and an example traffic generator, according to your parameterization. After the design is created, you can specify the target device and pin location assignments, run a full compilation, verify timing closure, and test the interface on your board using the programming file created by the Quartus Prime assembler. The 'Generate Example Design' button lets you generate simulation or synthesis file sets.

**Table 490. Group: Example Designs / Example Design Files**

Display Name	Identifier	Description
<b>Simulation</b>	EX_DESIGN_GUI_RLD3_GEN_SIM	Specifies that the 'Generate Example Design' button creates all necessary file sets for simulation. Expect a short additional delay as the file set is created. If you do not enable this parameter, simulation file sets are not created. Instead, the output directory will contain the ed_sim.qsys file which holds Qsys details of the simulation example design, and a make_sim_design.tcl file with other corresponding tcl files. You can run make_sim_design.tcl

*continued...*



Display Name	Identifier	Description
		from a command line to generate the simulation example design. The generated example designs for various simulators are stored in the /sim sub-directory.
<b>Synthesis</b>	EX_DESIGN_GUI_RLD_3_GEN_SYNTH	Specifies that the 'Generate Example Design' button creates all necessary file sets for synthesis. Expect a short additional delay as the file set is created. If you do not enable this parameter, synthesis file sets are not created. Instead, the output directory will contain the ed_synth.qsys file which holds Qsys details of the synthesis example design, and a make_qii_design.tcl script with other corresponding tcl files. You can run make_qii_design.tcl from a command line to generate the synthesis example design. The generated example design is stored in the /qii sub-directory.

**Table 491. Group: Example Designs / Generated HDL Format**

Display Name	Identifier	Description
<b>Simulation HDL format</b>	EX_DESIGN_GUI_RLD_3_HDL_FORMAT	This option lets you choose the format of HDL in which generated simulation files are created.

**Table 492. Group: Example Designs / Target Development Kit**

Display Name	Identifier	Description
<b>Select board</b>	EX_DESIGN_GUI_RLD_3_TARGET_DEV_KIT	Specifies that when you select a development kit with a memory module, the generated example design contains all settings and fixed pin assignments to run on the selected board. You must select a development kit preset to generate a working example design for the specified development kit. Any IP settings not applied directly from a development kit preset will not have guaranteed results when testing the development kit. To exclude hardware support of the example design, select 'none' from the 'Select board' pull down menu. When you apply a development kit preset, all IP parameters are automatically set appropriately to match the selected preset. If you want to save your current settings, you should do so before you apply the preset. You can save your settings under a different name using File->Save as.

### 7.5.8.8. About Memory Presets

Presets help simplify the process of copying memory parameter values from memory device data sheets to the EMIF parameter editor.

For DDRx protocols, the memory presets are named using the following convention:

PROTOCOL-SPEEDBIN LATENCY FORMAT-AND-TOPOLOGY CAPACITY (INTERNAL-ORGANIZATION)

For example, the preset named DDR4-2666U CL18 Component 1CS 2Gb (512Mb x 4) refers to a DDR4 x4 component rated at the DDR4-2666U JEDEC speed bin, with nominal CAS latency of 18 cycles, one chip-select, and a total memory space of 2Gb. The JEDEC memory specification defines multiple speed bins for a given frequency (that is, DDR4-2666U and DDR4-2666V). You may be able to determine the exact speed bin implemented by your memory device using its nominal latency. When in doubt, contact your memory vendor.

For RLDRAMx and QDRx protocols, the memory presets are named based on the vendor's device part number.



When the preset list does not contain the exact configuration required, you can still minimize data entry by selecting the preset closest to your configuration and then modify parameters as required.

Prior to production you should always review the parameter values to ensure that they match your memory device data sheet, regardless of whether a preset is used or not. Incorrect memory parameters can cause functional failures.

## 7.6. Document Revision History

Date	Version	Changes
May 2017	2017.05.08	<ul style="list-style-type: none"><li>Added <i>Stratix 10 External Memory Interface IP</i> section.</li><li>Rebranded as Intel.</li></ul>
October 2016	2016.10.31	<ul style="list-style-type: none"><li>Updated <i>Generated Directory Structure and Key Files for Example Simulation Designs</i> and <i>Generated Directory Structure and Key Files for Example Synthesis Designs</i> tables.</li><li>Removed <i>Parameterizing Arria 10 External Memory Interface IP</i> section.</li><li>Added protocol-specific parameter sections for Arria 10 EMIF IP.</li><li>Added <i>Equations for Arria 10 EMIF IP Board Skew Parameters</i> section.</li></ul>
May 2016	2016.05.02	<ul style="list-style-type: none"><li>Modified window duration in descriptions of <i>tDQSCK Delta Medium</i> and <i>tDQSCK Delta Long</i> in the <i>Parameter Descriptions</i> table in <i>Memory Timing Parameters for DDR2, DDR3, and LPDDR2 SDRAM for UniPHY IP</i>.</li><li>Modified description of <i>Enable the Efficiency Monitor and Protocol Checker on the Controller Avalon Interface</i> in the <i>Simulation Options</i> table in <i>Diagnostics for UniPHY IP</i>.</li><li>Removed <i>cal_debug_burstcount</i> from <i>Interface: cal_debug_avalon_slave</i> table in <i>Qsys Interfaces</i>.</li><li>Modified <i>Direction</i> information for entries in <i>Interface: cal_debug_out_avalon_master</i> table in <i>Qsys Interfaces</i>.</li><li>Removed two rows from the <i>Generated Directory Structure and Key Files for Simulation</i> table in <i>Generated Files for Arria 10 External Memory Interface IP</i>.</li><li>Replaced <i>Generated Directory Structure and Key Files for Example Designs</i> table with <i>Generated Directory Structure and Key Files for Example Simulation Designs</i> and <i>Generated Directory Structure and Key Files for Example Synthesis Designs</i> tables in <i>Generated Files for Arria 10 External Memory Interface IP</i>.</li><li>Added entry for <i>Slew rate to Group: I/O / Address/Command</i> and <i>Group: I/O / Memory Clock</i> tables in <i>I/O Parameters for Arria 10 EMIF IP</i>.</li><li>Modified description of <i>DDR3 LRDIMM additional control words</i> entry in <i>Group: Memory Topology / Mode Register Settings</i> table in <i>Memory Topology Parameters for Arria 10 EMIF IP</i>.</li><li>Modified supported protocol information for <i>Enable Error Detection and Correction Logic</i> and <i>Enable Auto Error Correction</i> entries in <i>Group: Controller / Configuration, Status and Error Handling</i> table in <i>Controller Parameters for Arria 10 EMIF IP</i>.</li><li>Minor change to description of <i>Calibration mode</i> in <i>Group: Diagnostic / Simulation Options</i> table in <i>Diagnostic Parameters for Arria 10 EMIF IP</i>.</li><li>Added QDR-IV to supported protocols for <i>Skip VREF calibration</i> in <i>Group: Diagnostic / Calibration Debug Options</i> table in <i>Diagnostic Parameters for Arria 10 EMIF IP</i>.</li><li>Added <i>Calibration address 0</i>, <i>Calibration address 1</i>, and <i>Enable automatic calibration after reset</i> to <i>Group: Diagnostic / Calibration Debug Options</i> table in <i>Diagnostic Parameters for Arria 10 EMIF IP</i>.</li><li>Added <i>Group: Diagnostic / Traffic Generator</i> table to <i>Diagnostic Parameters for Arria 10 EMIF IP</i>. Moved some entries from <i>Group: Diagnostic / Example Design</i> table to new <i>Group: Diagnostic / Traffic Generator</i> table.</li></ul>

*continued...*



Date	Version	Changes
November 2015	2015.11.02	<ul style="list-style-type: none"><li>Added note to descriptions of <i>Minimum delay difference between CK and DQS</i> and <i>Maximum delay difference between CK and DQS</i> in <i>Board Skew parameters</i> for LPDDR2/DDR2/DDR3 SDRAM.</li><li>Added text to description of Maximum skew between DQS groups in <i>Board Skew parameters</i> for LPDDR2/DDR2/DDR3 SDRAM.</li><li>Changed description of <i>emif_usr_clk</i> in the <i>Interface: emif_usr_clk_clock_source</i> table in <i>Qsys Interfaces</i>.</li><li>Changed description of <i>emif_usr_reset_n</i> in the <i>Interface: emif_usr_reset_reset_source</i> table in <i>Qsys Interfaces</i>.</li><li>Added <i>Interface: emif_usr_clk_sec_clock_source</i> and <i>Interface: emif_usr_reset_sec_reset_source</i> tables in <i>Qsys Interfaces</i>.</li><li>Added several options to the <i>Simulation Options</i>, <i>Calibration Debug Options</i>, and <i>Example Design</i> tables in <i>Diagnostic Parameters for Arria 10 EMIF IP</i>.</li><li>Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i>.</li><li>Added LPDDR3</li><li>Removed <b>Use address parity bit</b> parameter for QDR IV</li><li>Removed following DDR4 parameters:<ul style="list-style-type: none"><li>— <b>Write CRC enable</b></li><li>— <b>DDR4 gardown mode</b></li><li>— <b>Per-DRAM addressability</b></li><li>— <b>Temperature sensor readout</b></li><li>— <b>Write CMD latency for CRC/DM enable</b></li><li>— <b>MPR read format</b></li><li>— <b>CS to Addr/CMD Latency</b></li><li>— <b>Enable DM pins</b></li><li>— <b>Addr/CMD persistent error</b></li><li>— <b>Write DBI</b></li><li>— <b>Read DBI</b></li></ul></li><li>Removed group board timing/slew rates table</li><li>Removed <b>Maximum system skew within QK group</b> parameters for RLDRAM 3</li><li>Removed <b>Maximum system skew within Q group</b> and <b>Maximum skew within D group</b> parameters</li></ul>
May 2015	2015.05.04	Added information to the <i>Description</i> column for the <i>cal_debug_avalon_slave</i> , <i>cal_debug_clk_clock_sink</i> , and <i>cal_debug_out_reset_reset_source</i> tables in the <i>Qsys Interfaces</i> topic.
December 2014	2014.12.15	<ul style="list-style-type: none"><li>Added MAX 10 device support to the <i>PHY Parameters</i> table in <i>PHY Settings for UniPHY IP</i></li><li>Changed <i>Memory Parameters</i> for LPDDR2, DDR2, and DDR3 SDRAM table to accommodate MAX 10 devices.</li><li>Added <i>Enable Deep Power-Down Controls</i> parameter to <i>Controller Settings</i> table in <i>Controller Settings for UniPHY IP</i> section.</li><li><i>Arria 10 External Memory Interface IP</i> section:<ul style="list-style-type: none"><li>Removed references to Arria 10 devices from <i>Board Settings</i> topic in <i>UniPHY-Based External Memory Interface IP</i> section. Added new <i>Board Timing</i> topic.</li><li>Added QDR IV support to tables in <i>Qsys Interfaces</i> section.</li><li>Removed <i>afi_i_c</i> from <i>afi_conduit_end</i> table and <i>mem_c</i> from <i>mem_conduit_end</i> table.</li><li>Changed description of <i>global_reset_n</i> signal in the <i>Interface: global_reset_reset_sink</i> table.<ul style="list-style-type: none"><li>— Added <i>Board Timing</i> section and subtopics.</li></ul></li></ul></li><li>Changed <i>Memory Initialization Options</i> for DDR3 table to accommodate MAX 10 devices.</li></ul>

**continued...**



Date	Version	Changes
		<ul style="list-style-type: none"> <li>Replaced <i>Parameter Descriptions</i> table in <i>Memory Timing Parameters for DDR2, DDR3, and LPDDR2 SDRAM for UniPHY IP</i> section with new table including LPDDR2.</li> <li>Removed <i>General Settings</i> for Arria 10 EMIF IP section.</li> <li><i>Parameterizing Arria 10 External Memory Interface IP</i> section: <ul style="list-style-type: none"> <li>Added DDR4 support to tables.</li> <li>Changed descriptions of RDIMM/LRDIMM control words and LRDIMM additional control words in <i>Group: Memory Topology / Mode Register Settings</i> table.</li> <li>Removed <b>Chip ID Width</b> from <i>Group: Memory Topology / Topology</i> table.</li> <li>Added entry for <b>Instantiate two controllers sharing a Ping Pong PHY</b>, and expanded description of <b>Configuration</b>, in the <i>Group: General / Interface</i> table.</li> <li>Changed <b>Total interface width</b> entry to <b>DQ width</b> and changed <b>Place ALERT# pin to ALERT# pin placement</b> in <i>Group: Memory Topology / Topology</i> table.</li> </ul> </li> </ul>
August 2014	2014.08.15	<ul style="list-style-type: none"> <li>Added notes about Arria 10 EMIF IP to beginning of chapter.</li> <li>Added IP Catalog to <i>Design Flows</i> figure.</li> <li>Replaced <i>MegaWizard Plug-In Manager Flow</i> with <i>IP Catalog Design Flow</i>.</li> <li>Revised <i>Specify Parameters for the Qsys Flow</i> and <i>Completing the Qsys System</i> sections.</li> <li>Added information to description of <i>mem_doff_n</i> in <i>QDR II and QDR II+ SRAM Controller with UniPHY Interfaces</i> table.</li> <li>Reorganized into separate sections for <i>UniPHY-Based External Memory Interfaces</i> and <i>Arria 10 External Memory Interface IP</i>.</li> <li>Replaced the following sections with the new <i>Arria 10 EMIF IP Interfaces</i> section: <ul style="list-style-type: none"> <li><i>DDR3 Controller with Arria 10 EMIF Interfaces</i></li> <li><i>LPDDR2 SDRAM Controller with Arria 10 EMIF Interfaces</i></li> <li><i>QDR II/II+ Controller with Arria 10 EMIF Interfaces</i></li> <li><i>RLDRAM II Controller with Arria 10 EMIF Interfaces</i></li> <li><i>RLDRAM 3 Controller with Arria 10 EMIF Interfaces</i></li> </ul> </li> <li>Changed name of <i>Generated Files for Memory Controllers with Arria 10 EMIF IP</i> section to <i>Generated Files for Arria 10 External Memory Interface IP</i>, and revised content.</li> <li>Revised content of <i>General Settings</i> for Arria 10 EMIF IP section.</li> <li>Added <i>Parameterizing Arria 10 External Memory Interface IP</i> section.</li> <li>Revised content of <i>Memory Topology for LPDDR2, DDR3 and DDR4 SDRAM for Arria 10 EMIF IP</i> section.</li> <li>Added <i>Memory Parameters for QDR IV for Arria 10 EMIF IP</i> section.</li> <li>Revised <i>Memory Parameters for RLDRAM 3 for Arria 10 EMIF IP</i> section.</li> <li>Added slew rate information for QDR II, QDR II+, and QDR II+ Xtreme to table in <i>Slew Rates for Arria 10 EMIF IP</i> section.</li> <li>Revised <i>ISI Parameters</i> table in <i>Intersymbol Interference Channel Signal Integrity for UniPHY IP</i> section.</li> <li>Revised description of fly-by topology for UDIMMs in <i>Board Skews for UniPHY IP</i> section.</li> <li>Added MAX 10 to <i>Simulation Options</i> table in <i>Diagnostics for UniPHY IP</i> section.</li> <li>Added note to descriptions of Minimum delay difference between CK and DQS and Maximum delay difference between CK and DQS in <i>Board Skew Parameters for LPDDR2/DDR2/DDR3 SDRAM</i> section.</li> <li>Revised content of <i>Parameter Descriptions</i> table in <i>Board and Package Skews for LPDDR2/DDR3/DDR4 SDRAM for Arria 10 EMIF IP</i>.</li> <li>Revised content of <i>Parameter Descriptions</i> table in <i>Board and Package Skews for QDR II, QDR II+, and QDR II+ Xtreme for Arria 10 EMIF IP</i>.</li> </ul>

continued...



Date	Version	Changes
		<ul style="list-style-type: none"><li>Revised content of <i>Parameter Descriptions</i> table in <i>Board and Package Skews for RLDRAM II and RLDRAM 3 for Arria 10 EMIF IP</i>.</li><li>Revised content of <i>Controller Settings</i> table in <i>Controller Settings for Arria 10 EMIF IP</i>.</li><li>Added <i>Diagnostics for Arria 10 EMIF IP</i>.</li></ul>
December 2013	2013.12.16	<ul style="list-style-type: none"><li>Removed references to ALTMEMPHY.</li><li>Removed references to HardCopy.</li><li>Removed references to Stratix II devices.</li><li>Removed references to SOPC Builder.</li><li>Added Arria 10 information to <i>Qsys Interface</i>, <i>Generated Files</i>, <i>Parameter Settings</i>, <i>Board Settings</i>, and <i>Controller Settings</i> sections.</li><li>Added descriptions of several registered DIMM parameters to <i>Memory Parameters for LPDDR2, DDR2, and DDR3 SDRAM</i> table.</li><li>Added steps for compiling example project.</li><li>Added clock information to <i>Adding Pins and DQ Group Assignments</i>.</li><li>Updated <i>Intersymbol Interference for UniPHY IP</i> to <i>Intersymbol Interference Channel Signal Integrity for UniPHY IP</i>.</li><li>Added <i>Intersymbol Interference Channel Signal Integrity for Arria 10 EMIF IP</i>.</li></ul>
November 2012	6.0	<ul style="list-style-type: none"><li>Added RLDRAM 3 information.</li><li>Added LPDDR2 information.</li><li>Changed chapter number from 8 to 9.</li></ul>
June 2012	5.0	<ul style="list-style-type: none"><li>Added number of sharing interfaces parameters to <i>Clock Parameters</i> table.</li><li>Added <b>DQ/DQS Package Deskew</b> and <b>Address/Command Package Deskew</b> descriptions to <i>Board Skew Parameters</i> table.</li><li>Added equations for multiple boards to several parameter descriptions in <i>Board Skew Parameters</i> table.</li><li>Added Feedback icon.</li></ul>
November 2011	4.0	<ul style="list-style-type: none"><li>Updated <i>Installation and Licensing</i> section.</li><li>Combined <i>Qsys</i> and <i>SOPC Builder Interfaces</i> sections.</li><li>Combined parameter settings for DDR, DDR2, DDR3 SDRAM, QDRII SRAM, and RLDRAM II for both ALTMEMPHY and UniPHY IP.</li><li>Added parameter usage details to <i>Parameterizing Memory Controllers with UniPHY IP</i> section.</li><li>Moved <i>Functional Description</i> section for DDR, DDR2, DDR3 SDRAM, QDRII SRAM, and RLDRAM II to volume 3 of the <i>External Memory Interface Handbook</i>.</li></ul>
June 2011	3.0	<ul style="list-style-type: none"><li>Removed references to High-Performance Controller.</li><li>Updated High-Performance Controller II information.</li><li>Removed HardCopy III, HardCopy IV E, HardCopy IV GX, Stratix III, and Stratix IV support.</li><li>Updated <i>Generated Files</i> lists.</li><li>Added <i>Qsys</i> and <i>SOPC Builder Interfaces</i> section.</li></ul>
December 2010	2.1	<ul style="list-style-type: none"><li>Updated Design Flows and Generated Files information.</li><li>Updated <i>Parameterizing Memory Controllers with UniPHY IP</i> chapter</li></ul>
July 2010	2.0	<ul style="list-style-type: none"><li>Added information for new GUI parameters: <b>Controller latency</b>, <b>Enable reduced bank tracking for area optimization</b>, and <b>Number of banks to track</b>.</li><li>Removed information about IP Advisor. This feature is removed from the DDR/DDR2 SDRAM IP support for version 10.0.</li></ul>
February 2010	1.3	Corrected typos.

*continued...*



Date	Version	Changes
February 2010	1.2	<ul style="list-style-type: none"><li>• Full support for Stratix IV devices.</li><li>• Added timing diagrams for initialization and calibration stages for HPC.</li></ul>
November 2009	1.1	Minor corrections.
November 2009	1.0	Initial release.



## 8. Simulating Memory IP

---

To simulate your design you require the following components:

- A simulator—The simulator must be any Intel-supported VHDL or Verilog HDL simulator
- A design using one of Intel's external memory IP
- An example driver (to initiate read and write transactions)
- A testbench and a suitable memory simulation model

The Intel External Memory Interface IP is not compatible with the Qsys Testbench System generation feature. Instead, use the simulation example design of your generated IP as a reference for how to create a simulatable design, completed with a memory interface, a memory model, and a traffic generator.

### Memory Simulation Models

There are two types of memory simulation models that you can use:

- Intel-provided generic memory model.

The Quartus® Prime software generates this model together with the example design and this model adheres to all the memory protocol specifications. You can parameterize the generic memory model.

- Vendor-specific memory model.

Memory vendors such as Micron and Samsung provide simulation models for specific memory components that you can download from their websites.

*Note:* Intel does not provide support for vendor-specific memory models.

### 8.1. Simulation Options

The following simulation options are available with the example testbench to improve simulation speed:

- Full calibration—Calibrates the same way as in hardware, and includes all phase, delay sweeps, and centering on every data bit.

*Note:* Arria 10 EMIF full calibration simulation will be available in a future release of the Quartus Prime software.

- Quick calibration—Calibrates the read and write latency only, skipping per bit deskew. (Not available for Arria 10 EMIF IP.)
- Skip calibration—Provides the fastest simulation. It loads the settings calculated from the memory configuration and enters user mode.



By default, the UniPHY IP generates abstract PHY, which uses skip calibration regardless of the simulation options that you chose in the parameter editor.

**Note:** For proper simulation of DQS Tracking, you must enable either full calibration or quick calibration.

The following table lists typical simulation times implemented using UniPHY IP. The simulation times in the table are estimates based on average run times of a few example designs. The simulation times for your design may vary depending on the memory interface specifications, simulator, or the system you are using.

**Table 493. Typical Simulation Times Using UniPHY IP**

Calibration Mode/Run Time <sup>(1)</sup>	Estimated Simulation Time	
	Small Interface ( $\times 8$ Single Rank)	Large Interface ( $\times 72$ Quad Rank)
Full <ul style="list-style-type: none"> <li>• Full calibration</li> <li>• Includes all phase/delay sweeps and centering</li> </ul>	10 minutes	~ 1 day
Quick <ul style="list-style-type: none"> <li>• Scaled down calibration</li> <li>• Calibrate one pin</li> </ul>	3 minutes	4 hours
Skip <ul style="list-style-type: none"> <li>• Skip all calibration, jump to user mode</li> <li>• Preload calculated settings</li> </ul>	3 minutes	20 minutes
<p><b>Note to Table:</b></p> <ol style="list-style-type: none"> <li>1. Uses one loop of driver test. One loop of driver is approximately 600 read or write requests, with burst length up to 64.</li> <li>2. Simulation times shown in this table are approximate measurements made using Synopsys VCS. Simulation times can vary considerably, depending on the IP configuration, the simulator used, and the computer or server used.</li> </ol>		

**Table 494. Typical Simulation Times Using Arria 10 EMIF IP**

Calibration Mode/Run Time <sup>(1)</sup>	Estimated Simulation Time	
	Small Interface ( $\times 8$ Single Rank)	Large Interface ( $\times 72$ Quad Rank)
Full <ul style="list-style-type: none"><li>• Full calibration</li><li>• Includes all phase/delay sweeps and centering</li></ul>	20 minutes	$\sim$ 1 day
Skip <ul style="list-style-type: none"><li>• Skip all calibration, jump to user mode</li><li>• Preload calculated settings</li></ul>	10 minutes	25 minutes
Abstract PHY <ul style="list-style-type: none"><li>• Replace PHY and external memory model with a single abstract PHY model.</li><li>• <b>IMPORTANT:</b> <i>External memory model is NOT used in this mode. No I/O switching occurs to the external memory model.</i></li></ul>	1 minute	5 minutes
Note to Table: 1. Uses one loop of driver test. One loop of driver is approximately 600 read or write requests, with burst length up to 64. 2. Simulation times shown in this table are approximate measurements made using Synopsys VCS. Simulation times can vary considerably, depending on the IP configuration, the simulator used, and the computer or server used.		

For more information about steps to follow before simulating, modifying the vendor memory model, and simulation flow for UniPHY IPs, refer to the “Simulation Walkthrough with UniPHY IP”.

#### Related Information

[Simulation Walkthrough with UniPHY IP](#) on page 470

## 8.2. Simulation Walkthrough with UniPHY IP

Simulation of the whole memory interface is a good way to determine the latency of your system. However, the latency found in simulation may be different than the latency found on the board because functional simulation does not take into account board trace delays and different process, voltage, and temperature scenarios.

For a given design on a given board, the latency found may differ by one clock cycle (for full-rate designs) or two clock cycles (for half-rate designs) upon resetting the board. Different boards can also show different latencies even with the same design.

The UniPHY IP supports only functional simulation. Functional simulation is supported at the RTL level and after generating a post-fit functional simulation netlist. The post-fit netlist for designs that contain UniPHY IP is a hybrid of the gate level (for FPGA core) and RTL level (for the external memory interface IP). Intel recommends that you validate the functional operation of your design using RTL simulation, and the timing of your design using TimeQuest Timing Analysis.

For UniPHY-based external memory interfaces, you can perform functional simulation using an example design that is generated with your IP core. The example design files are created in the \<variation\_name>\_example\_design directory.



You can use the IP functional simulation model with any supported VHDL or Verilog HDL simulator.

After you have generated the memory IP, view the README.txt file located in the `\<variation_name>_example_design\simulation` directory for instructions on how to generate the simulation example design for Verilog HDL or VHDL. The **README.txt** file also explains how to run simulation using the ModelSim - Intel FPGA Edition software. Simulation scripts for the Mentor Graphics, Cadence, Aldec, and Synopsys simulators are provided; however, detailed instructions on how to perform simulation using these third-party simulators are not provided.

### 8.2.1. Simulation Scripts

The Quartus Prime software generates three simulation scripts during project generation for four different third party simulation tools—Cadence, Synopsys, Aldec, and Mentor Graphics.

The simulation scripts reduce the number of files that you need to compile separately before simulating a design. These scripts are located in three separate folders under the `<project_directory>\<varitation_name>_sim` directory, each named after the names of the simulation tools. The example designs also provide equivalent scripts after you run the **.tcl** script from the project located in the `\<variation_name>_example_design\simulation` directory.

The order of the files in the simulation scripts is important. Ensure that you maintain the files in order, to avoid error messages and warning messages during simulation. If you choose not to use the Intel-generated simulation scripts in your simulation environment, you must maintain the specified file order when compiling the memory controller with the user-generated simulation script.

### 8.2.2. Preparing the Vendor Memory Model

You can replace the Intel-supplied memory model with a vendor-specific memory model. In general, you may find vendor-specific models to be standardized, thorough, and well supported, but sometimes more complex to setup and use.

**Note:** Intel does not provide support for vendor-specific memory models. If you do want to replace the Intel-supplied memory model with a vendor-supplied memory model, you should observe the following guidelines:

- Ensure that you have the correct vendor-supplied memory model for your memory device.
- Disconnect all signals from the default memory model and reconnect them to the vendor-supplied memory model.
- If you intend to run simulation from the Quartus Prime software, ensure that the **.qip** file points to the vendor-supplied memory model.



When you are using a vendor-supplied memory model instead of the generated functional simulation model, you must modify the vendor memory model and the testbench files by following these steps:

1. Obtain and copy the vendor memory model to the `\<variation_name>_example_design\simulation\<variation_name>_sim\submodules` directory. For example, obtain the **ddr2.v** and **ddr2\_parameters.vh** simulation model files from the Micron website and save them in the directory.
  - The auto-generated generic SDRAM model may be used as a placeholder for a specific vendor memory model.
  - Some vendor DIMM memory models do not use data mask (DM) pin operation, which can cause calibration failures. In these cases, use the vendor's component simulation models directly.
2. Open the vendor memory model file in a text editor and specify the speed grade and device width at the top of the file. For example, you can add the following statements for a DDR2 SDRAM model file:

```
'define sg25  
'define x8
```

The first statement specifies the memory device speed grade as -25 (for 400 MHz operation). The second statement specifies the memory device width per DQS.

3. Check that the following statement is included in the vendor memory model file. If not, include it at the top of the file. This example is for a DDR2 SDRAM model file:  
``include "ddr2_parameters.vh"`
4. Save the vendor memory model file.
5. Open the simulation example project file `<variation_name>_example_sim.qpf`, located in the `<variation_name>_example_design\simulation` directory.
6. On the Tools menu, select **TCL scripts** to run the **generate\_sim\_verilog\_example\_design.tcl** file, in which generates the simulation example design.
7. To enable vendor memory model simulation, you have to include and compile the vendor memory model by adding it into the simulation script. Open the **.tcl** script, **msim\_setup.tcl**, located in the `<variation_name>_example_design\simulation\verilog\mentor` directory in the text editor. Add in the following line in the '# Compile the design files in correct order' section:

```
vlog      +incdir+$QSYS_SIMDIR/submodules/  
          "$QSYS_SIMDIR/submodules/<vendor_memory>.v"  
          -work <variation_name>_example_sim_work
```

8. Open the simulation example design, `<variation_name>_example_sim.v`, located in the `<variation_name>_example_design\simulation\verilog` directory in a text editor and delete the following module:

```
alt_mem_if_<memory_type>_mem_model_top_<memory_type>_mem_if_dm_pins_en_mem_  
if_dqsn_en
```

*Note:* The actual name of the pin may differ slightly depending on the memory controller you are using.



9. Instantiate the downloaded memory model and connect its signals to the rest of the design.
10. Ensure that the ports names and capitalization in the memory model match the port names and capitalization in the testbench.

*Note:* The vendor memory model may use different pin names and capitalization than the generated functional simulation model.

11. Save the testbench file.

The original instantiation may be similar to the following code:

```
alt_mem_if_ddr2_mem_model_top_mem_if_dm_pins_en_mem_if_dqsn_en #(
    .MEM_IF_ADDR_WIDTH          (13),
    .MEM_IF_ROW_ADDR_WIDTH      (12),
    .MEM_IF_COL_ADDR_WIDTH      (8),
    .MEM_IF_CS_PER_RANK         (1),
    .MEM_IF_CONTROL_WIDTH       (1),
    .MEM_IF_DQS_WIDTH          (1),
    .MEM_IF_CS_WIDTH            (1),
    .MEM_IF_BANKADDR_WIDTH     (3),
    .MEM_IF_DQ_WIDTH           (8),
    .MEM_IF_CK_WIDTH           (1),
    .MEM_IF_CLK_EN_WIDTH       (1),
    .DEVICE_WIDTH               (1),
    .MEM_TRCD                  (6),
    .MEM_TRTP                  (3),
    .MEM_DQS_TO_CLK_CAPTURE_DELAY (100),
    .MEM_IF_ODT_WIDTH          (1),
    .MEM_MIRROR_ADDRESSING_DEC (0),
    .MEM_REGDIMM_ENABLED        (0),
    .DEVICE_DEPTH               (1),
    .MEM_INIT_EN                (0),
    .MEM_INIT_FILE              (""),
    .DAT_DATA_WIDTH             (32)
) m0 (
    .mem_a   (e0_memory_mem_a), // memory.mem_a
    .mem_ba  (e0_memory_mem_ba), // .mem_ba
    .mem_ck  (e0_memory_mem_ck), // .mem_ck
    .mem_ck_n (e0_memory_mem_ck_n), // .mem_ck_n
    .mem_cke (e0_memory_mem_cke), // .mem_cke
    .mem_cs_n (e0_memory_mem_cs_n), // .mem_cs_n
    .mem_dm  (e0_memory_mem_dm), // .mem_dm
    .mem_ras_n (e0_memory_mem_ras_n), // .mem_ras_n
    .mem_cas_n (e0_memory_mem_cas_n), // .mem_cas_n
    .mem_we_n (e0_memory_mem_we_n), // .mem_we_n
    .mem_dq   (e0_memory_mem_dq), // .mem_dq
    .mem_dqs  (e0_memory_mem_dqs), // .mem_dqs
    .mem_dqs_n (e0_memory_mem_dqs_n), // .mem_dqs_n
    .mem_odt (e0_memory_mem_odt) // .mem_odt
);
```

Replace the original code with the following code:

```
ddr2 memory_0 (
    .addr (e0_memory_mem_a), // memory.mem_a
    .ba (e0_memory_mem_ba), // .mem_ba
    .clk (e0_memory_mem_ck), // .mem_ck
    .clk_n (e0_memory_mem_ck_n), // .mem_ck_n
    .cke (e0_memory_mem_cke), // .mem_cke
    .cs_n (e0_memory_mem_cs_n), // .mem_cs_n
    .dm_rdqs (e0_memory_mem_dm), // .mem_dm
    .ras_n (e0_memory_mem_ras_n), // .mem_ras_n
    .cas_n (e0_memory_mem_cas_n), // .mem_cas_n
    .we_n (e0_memory_mem_we_n), // .mem_we_n
    .dq (e0_memory_mem_dq), // .mem_dq
    .dqs (e0_memory_mem_dqs), // .mem_dqs
```



```
.rdqs_n (), // .mem_dqs_n  
.dqs_n (e0_memory_mem_dqs_n), // .mem_dqs_n  
.odt (e0_memory_mem_odt) // .mem_odt);
```

If you are interfacing with a DIMM or multiple memory components, you need to instantiate all the memory components in the simulation file.

### 8.2.3. Functional Simulation with Verilog HDL

Simulation scripts for the Synopsys, Cadence, Aldec, and Mentor Graphics simulators are provided for you to run the example design.

The simulation scripts are located in the following main folder locations:

Simulation scripts in the simulation folders are located as follows:

- <variation\_name>\_example\_design\simulation\verilog\mentor\msim\_setup.tcl
- <variation\_name>\_example\_design\simulation\verilog\synopsys\vcs\vcs\_setup.sh
- <variation\_name>\_example\_design\simulation\verilog\synopsys\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_example\_design\simulation\verilog\aldec\rivierapro\_setup.tcl
- <variation\_name>\_example\_design\simulation\verilog\cadence\ncsim\_setup.sh

Simulation scripts in the <>\_sim\_folder are located as follows:

- <variation\_name>\_sim\mentor\msim\_setup.tcl
- <variation\_name>\_sim\cadence\ncsim\_setup.sh
- <variation\_name>\_sim\synopsys\vcs\vcs\_setup.sh
- <variation\_name>\_sim\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_sim\aldec\rivierapro\_setup.tcl

For more information about simulating Verilog HDL or VHDL designs using command lines, refer to the *Mentor Graphics ModelSim and QuestaSim Support* chapter in volume 3 of the Quartus Prime Handbook.

#### Related Information

[Mentor Graphics ModelSim and QuestaSim Support](#)

### 8.2.4. Functional Simulation with VHDL

The UniPHY VHDL file set is provided for customers who want to generate the top-level RTL instance of their UniPHY cores in VHDL.

Prior to Quartus Prime version 15.1, the VHDL fileset was composed entirely of VHDL files. Beginning with Quartus Prime version 15.1, only the top-level IP instance file is guaranteed to be written in VHDL; submodules can still be written in Verilog/SystemVerilog (encrypted or plaintext), or in VHDL. Note that the ModelSim - Intel



FPGA Edition is no longer restricted to a single HDL language, as of version 15.1; however, some files may still be encrypted in order to be excluded from the maximum unencrypted module limit of this tool.

Because the VHDL fileset consists of both VHDL and Verilog files, you must follow certain mixed-language simulation guidelines. The general guideline for mixed-language simulation is that you must always link the Verilog files (whether encrypted or not) against the Verilog version of the libraries, and the VHDL files (whether simgen-generated or pure VHDL) against the VHDL libraries.

Simulation scripts for the Synopsys, Cadence, Aldec, and Mentor Graphics simulators are provided for you to run the example design. These simulation scripts are located in the following main folder locations:

Simulation scripts in the simulation folders are located as follows:

- <variation\_name>\_example\_design\simulation\vhdl\mentor\msim\_setup.tcl
- <variation\_name>\_example\_design\simulation\vhdl\synopsys\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_example\_design\simulation\vhdl\cadence\ncsim\_setup.sh
- <variation\_name>\_example\_design\simulation\vhdl\aldec\rivierapro\_setup.tcl

Simulation scripts in the <>\_sim\_folder are located as follows:

- <variation\_name>\_sim\mentor\msim\_setup.tcl
- <variation\_name>\_sim\cadence\ncsim\_setup.sh
- <variation\_name>\_sim\synopsys\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_sim\aldec\rivierapro\_setup.tcl

For more information about simulating Verilog HDL or VHDL designs using command lines, refer to the *Mentor Graphics ModelSim and QuestaSim Support* chapter in volume 3 of the Quartus Prime Handbook.

#### Related Information

[Mentor Graphics ModelSim and QuestaSim Support](#)

### 8.2.5. Simulating the Example Design

This topic describes how to simulate the example design in Cadence, Synopsys, Mentor Graphics, and Aldec simulators.

To simulate the example design in the Quartus Prime software using the Cadence simulator, follow these steps:

1. At the Linux\* shell command prompt, change directory to  
<name>\_example\_design\simulation\<verilog/vhdl>\cadence
2. Run the simulation by typing the following command at the command prompt:

```
sh ncsim_setup.sh
```



To simulate the example design in the Quartus Prime software using the Synopsys simulator, follow these steps:

1. At the Linux shell command prompt, change directory to <name>\_example\_design\simulation\<verilog/vhdl>\synopsys\vcsmx
2. Run the simulation by typing the following command at the command prompt:

```
sh vcsmx_setup.sh
```

To simulate the example design in the Quartus Prime software using the Mentor simulator, follow these steps:

1. At the Linux or Windows shell command prompt, change directory to <name>\_example\_design\simulation\<verilog/vhdl>\mentor
2. Execute the **msim\_setup.tcl** script that automatically compiles and runs the simulation by typing the following command at the Linux or Windows command prompt:

```
vsim -do run.do
```

or

Type the following command at the ModelSim command prompt:

```
do run.do
```

To simulate the example design in the Quartus Prime software using the Aldec simulator, follow these steps:

1. At the Linux or Windows shell command prompt, change directory to <name>\_example\_design\simulation\<verilog/vhdl>\aldec
2. Execute the **rivierapro\_setup.tcl** script that automatically compiles and runs the simulation by typing the following command at the Linux or Windows command prompt: vsim -do rivierapro.tcl
3. To compile and elaborate the design after the script loads, type `ld_debug`.
4. Type `run -all` to run the simulation.

For more information about simulation, refer to the *Simulating Designs* chapter in volume 3 of the Quartus Prime Handbook.

If your Quartus Prime project appears to be configured correctly but the example testbench still fails, check the known issues on the Intel FPGA Knowledge Base before filing a service request.

#### Related Information

- [Simulating Intel FPGA Designs](#)
- [Knowledge Database](#)

### 8.2.6. UniPHY Abstract PHY Simulation

UniPHY IP generates both synthesizable and abstract models for simulation, with the abstract model as default. The UniPHY abstract model replaces the PLL with simple fixed-delay model, and the detailed models of the hard blocks with simple cycle-accurate functional models.



Full calibration mode cannot be used with abstract models, which is the default model type for all devices except Arria V and Cyclone V. In addition to enabling full calibration during generation, you must also disable the use of abstract models by modifying the generated simulation scripts as described below. For VHDL, the UniPHY abstract model is the only option because you cannot switch to regular simulation model. The PLL frequencies in simulation may differ from the real time simulation due to pico-second timing rounding.

However, you can switch to regular simulation models for Verilog HDL language. The full and quick calibration modes are available for regular simulation models.

Add an additional command line to the compilation script for the two relevant files to enable regular PHY simulation:

```
+define+ALTERA_ALT_MEM_IF_PHY_FAST_SIM_MODEL=0
```

The two relevant files are:

- In *<variation\_name>\_example\_design/simulation/verilog* submodules:

```
<variation_name>_example_sim_e0_if0_p0.sv  
and  
<variation_name>_example_sim_e0_if0_pll0.sv
```

or

- In *<variation\_name>\_sim/submodules*:

```
<variation_name>_p0.sv  
and  
<variation_name>_pll0.sv
```

To switch to regular simulation models for the Verilog HDL language on the example simulation design, follow the appropriate steps for your simulator:

- For the Mentor simulator, edit the **msim\_setup.tcl** file as follows:

```
vlog  
- sv "$QSYS_SIMDIR/submodules/  
<variation_name>_example_sim_e0_if0_p0.sv"  
+define+ALTERA_ALT_MEM_IF_PHY_FAST_SIM_MODEL=0  
-work <variation_name>_example_sim_work
```

```
vlog - sv  
"$QSYS_SIMDIR/submodules/<variation_name>/  
_example_sim_e0_if0_pll0.sv"  
+define+ALTERA_ALT_MEM_IF_PHY_FAST_SIM_MODEL=0  
-work <variation_name>_example_sim_work
```

- For the Cadence simulator, edit the **ncsim\_setup.sh** file as follows:

```
ncvlog  
- sv "$QSYS_SIMDIR/submodules/  
<variation_name>_example_sim_e0_if0_p0.sv"
```



```
+define+ALTERA_ALT_MEM_IF_PHY_FAST_SIM_MODEL=0
-work <variation_name>_example_sim_work
- cdslib ./cds_libs/skip_example_sim_work.cds.lib

ncvlog - sv
"${QSYS_SIMDIR}/submodules/
<variation_name>_example_sim_e0_if0_pll0.sv"
+define+ALTERA_ALT_MEM_IF_PHY_FAST_SIM_MODEL=0
-work <variation_name>_example_sim_work
- cdslib ./cds_libs/<variation_name>_example_sim_work.cds.lib
```

- For the Synopsys simulator, edit the **vscmx\_setup.sh** file as follows:

```
vlogan +v2k - sverilog
"${QSYS_SIMDIR}/submodules/<variation_name>_example_sim_e0_if0_p0.sv"
+define+ALTERA_ALT_MEM_IF_PHY_FAST_SIM_MODEL=0
- work <variation_name>_example_sim_work

vlogan +v2k - sverilog
"${QSYS_SIMDIR}/submodules/
<variation_name>_example_sim_e0_if0_pll0.sv"
+define+ALTERA_ALT_MEM_IF_PHY_FAST_SIM_MODEL=0
- work <variation_name>_example_sim_work
```

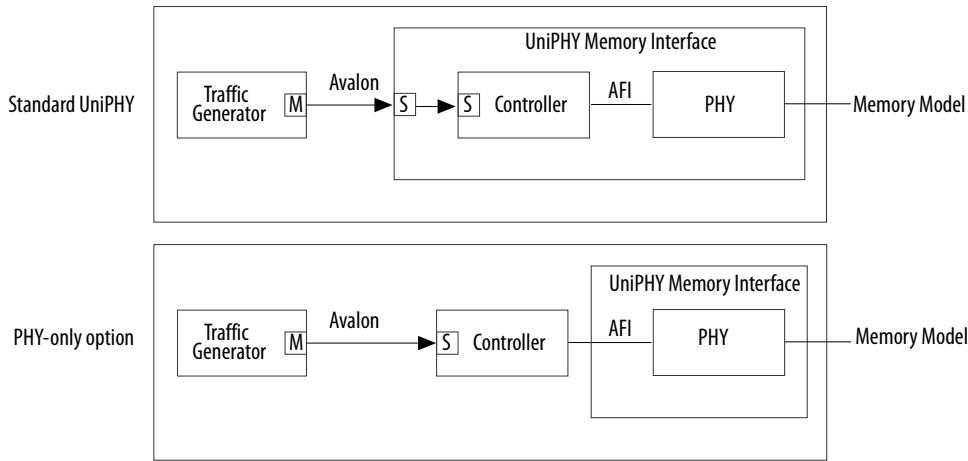
If you use the UniPHY abstract model, the simulation is two times faster in magnitude if compared to the real simulation model. Instantiating a standalone UniPHY IP in your design further improves the simulation time if you use a half-rate controller with UniPHY or a larger memory DQ width.

### 8.2.7. PHY-Only Simulation

To enable PHY-only simulation in the parameter editor, under **PHY Settings** tab, in the FPGA section, turn on **Generate PHY only**. This setting also applies to designs using Qsys. This option allows you to replace the Intel high-performance memory controllers with your own custom controller.

When you are using a standard UniPHY memory interface, by default, the parameter editor generates an external memory interface with a controller and a PHY. The controller and PHY are connected internally with the Altera PHY interface (AFI). The memory interface has an Avalon slave port that connects to the controller to allow communication from the user logic. When you turn on the PHY-only option, the parameter editor generates the PHY without the controller. In this case, the PHY is accessed via the AFI port, which can be externally connected to a custom controller. In the example design, a controller is instantiated externally to the memory interface. This provides a fully functional example design and demonstrates how to connect the controller to manage the transactions from the traffic generator.

The following figure shows the difference in the UniPHY memory interface when the PHY-only option is enabled.

**Figure 65. PHY-only Option**

### 8.2.8. Post-fit Functional Simulation

The post-fit functional simulation does not work for the UniPHY IP core because of the following inherent problems:

- The UniPHY sample 'X's during calibration, which causes an issue during timing simulation
- Some internal transfers that are 0-cycle require delays to properly function in a post-fit netlist

To enable functional simulation for a design that uses UniPHY IP core, a quasi-post-fit scheme is implemented. This scheme allows gate-level simulation of the full design (excluding the UniPHY IP), while you use RTL simulation for the UniPHY IP. The quasi-post-fit scheme involves partitioning blocks in the EMIF and swapping them with simulation RTL. With this workaround the memory interface is partially post-fit RTL and partially premap RTL, therefore the simulation flow is not impeded.

Gate simulation for the hard memory controller is not supported.

#### 8.2.8.1. Running Post-fit Simulation

Assuming that the UniPHY IP has been generated and inserted in some larger design, follow these steps to run post-fit simulation:

1. In the Quartus Prime software, set up a project that contains a UniPHY IP core.
2. On the Assignments menu, click **Assignment Editor**.
3. In the assignment editor, add the global assignment VERILOG\_MACRO and set the value to SYNTH\_FOR\_SIM=1.
4. On the Assignments menu, click **Settings**.
5. In the **Category** list, under **EDA Tools Settings**, select **Simulation**.
6. On the Simulation page, select a tool name (for example, ModelSim - Intel FPGA Edition).
7. In the **Format for output netlist** list, select a HDL language.



8. In the **Output directory** box, type or browse to the location where you want output files saved.
9. Click **More EDA Netlist Writer Settings** to choose from a list of other options.
10. Set the value for **Maintain hierarchy** to **PARTITION\_ONLY**, and click **OK**.
11. Elaborate the project. On the Processing menu, select **Start** and click **Start Hierarchy Elaboration**.
12. In the Project Navigator window, click the **Hierarchy** tab. In the **Entity** box, locate the instances for the following devices:
  - a. For instances in Stratix III, Stratix IV, Arria II GX, Arria II GZ , click the + icon to expand the following top-level design entities, right-click on the lower-level entities, select **Design Partition**, and click **Set as Design Partition**:
    - <hierarchy path to UniPHY top-level>  
  \<name>\_if0:if0\<name>\_if0\_p0:p0
    - <hierarchy path to UniPHY top-level>  
  \<name>\_if0:if0\<name>\_if0\_s0:s0
  - b. For instances in Arria V or Stratix V, click the + icon to expand the following top-level design entity, right-click on the lower-level entities, select **Design Partition**, and click **Set as Design Partition**:
    - <hierarchy path to UniPHY top-level>  
  \<name>\_if0:if0\<name>\_if0\_s0:s0
- For instances of hard memory interfaces in Arria V, no design partition is necessary.
13. In the **Design Partitions Window**, ensure that the netlist type value of the design partitions listed in Step12 a and 12b are set to **Post-synthesis**.
14. On the Processing menu, select **Start** and click **Start Analysis and Synthesis**.
15. Run the Pin assignments script. To run the pin assignment script, follow these steps:
  - a. On the **Tools** menu,click **TCL Scripts**.
  - b. In the **Libraries** list, locate the **<name>\_pin\_assignment.tcl**.
  - c. Click **Run**.
16. On the Processing menu, select **Start** and click **Partition Merge**.
17. On the Processing menu, select **Start** and click **Start Fitter**.
18. On the Processing menu, select **Start** and click **Start EDA netlist writer** .
19. The output post-fit netlist is located in the directory you chose in Step 8.
20. Assume that the netlist filename is **dut.vo** (or **dut.vho** for VHDL). Replace the instance of the partitioned modules (specified in step 12) in **dut.vo** and instantiate the original instance of the RTL. As a result, the RTL of those modules will simulate correctly instead of the the post-fit netlist. For example, you can delete the definition of the **<name>\_if0\_s0** (and **<name>\_if0\_p0**, if appropriate) modules in the post-fit netlist, and ensure that your simulator compiles the post-fit netlist and all the UniPHY RTL in order to properly link these modules for simulation.  
(This step does not apply to hard memory interfaces on Arria V devices.)



21. To match the post-fit netlist instantiation of `s0` (and `p0`, if appropriate) with the original RTL module definition (specified in step 12), you must also account for three device input ports that are added to the post-fit netlist. The easiest way to do this is to delete the following three connections from the `s0` (and `p0`, if appropriate) instances in the post-fit netlist:

- `.devpor(devpor)`
- `.devclrn(devclrн)(`
- `.devoe(devpoe)`

(This step does not apply to hard memory interfaces on Arria V devices.)

22. For Stratix V the `<name>_if0_s0` instance in the post-fit netlist will also have a connection `.QIC_GND_PORT(<wire name>)` that you must delete because it does not match with the original RTL module.

(This step does not apply to hard memory interfaces on Arria V devices.)

23. Set up and run your simulator.

### 8.2.9. Simulation Issues

When you simulate an example design in ModelSim, you might see the following warnings, which are expected and not harmful:

```
# ** Warning: (vsim-3015)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_controller_phy.sv(402
): [PCDPC] - Port size (1 or 1) does not match connection size (7) for port
'local_size'.

#           Region:
/uniphy_s4_example_top_tb/dut/mem_if/controller_phy_inst/
alt_ddrx_controller_inst

# ** Warning: (vsim-3015)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_controller_phy.sv(402
): [PCDPC] - Port size (9 or 9) does not match connection size (1) for port
'ctl_cal_byte_lane_sel_n'.

#           Region:
/uniphy_s4_example_top_tb/dut/mem_if/controller_phy_inst/
alt_ddrx_controller_inst

# ** Warning: (vsim-3015)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_controller_phy.sv(402
): [PCDPC] - Port size (18 or 18) does not match connection size (1) for port
'afi_doing_read'.

#           Region:
/uniphy_s4_example_top_tb/dut/mem_if/controller_phy_inst/
alt_ddrx_controller_inst

# ** Warning: (vsim-3015)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_controller_phy.sv(402
): [PCDPC] - Port size (2 or 2) does not match connection size (1) for port
'afi_rdata_valid'.

#           Region:
/uniphy_s4_example_top_tb/dut/mem_if/controller_phy_inst/
alt_ddrx_controller_inst
```



```
# ** Warning: (vsim-3015)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_controller_phy.sv(402
): [PCDPC] - Port size (112 or 112) does not match connection size (1) for
port
'bank_information'.

#           Region:
/uniphy_s4_example_top_tb/dut/mem_if/controller_phy_inst/
alt_ddrx_controller_inst

# ** Warning: (vsim-3015)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_controller_phy.sv(402
): [PCDPC] - Port size (8 or 8) does not match connection size (1) for port
'bank_open'.

#           Region:
/uniphy_s4_example_top_tb/dut/mem_if/controller_phy_inst/
alt_ddrx_controller_inst

# ** Warning: (vsim-3015)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_alt_ddrx_bank_timer_
wrapper.v(1191): [TFMPC] - Too few port connections. Expected 127, found 126.

#           Region:
/uniphy_s4_example_top_tb/dut/mem_if/controller_phy_inst/
alt_ddrx_controller_inst
/bank_timer_wrapper_inst/bank_timer_inst

# ** Warning: (vsim-3722)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_alt_ddrx_bank_timer_
wrapper.v(1191): [TFMPC] - Missing connection for port 'wr_to_rd_to_pch_all'.

# ** Warning: (vsim-3015)
D:/design_folder/iptest10/simulation/uniphy_s4/rtl/
uniphy_s4_alt_ddrx_bank_timer_
wrapper.v(1344): [PCDPC] - Port size (5 or 5) does not match connection size
(1)
for port 'wr_to_rd_to_pch_all'.

#           Region:
/uniphy_s4_example_top_tb/dut/mem_if/controller_phy_inst/alt_ddrx_controller_
inst/bank_timer_wrapper_inst/rank_monitor_inst

# ** Warning: (vsim-8598) Non-positive replication multiplier inside concat.
Replication will be ignored

Warning-[OSPA-N] Overriding same parameter again

/p/eda/acd/altera/quartusII/10.1/quartus/eda/sim_lib/synopsys/stratixv_atoms_
ncrypt.v, 8499

Warning-[ZONMCM] Zero or negative multiconcat multiplier
../quartus_stratix5/ddr3_ctlr_sim/ddr3_ctlr_sequencer.sv, 916

Zero or negative multiconcat multiplier is found in design. It will be
replaced by 1'b0.

Source info: {INIT_COUNT_WIDTH {1'b0} }

Warning-[PCWM-W] Port connection width mismatch
../quartus_stratix5/ddr3_ctlr_sim/ddr3_ctlr_sequencer_cpu.v, 2830

"the_sequencer_cpu_nios2_oci_itrace"

The following 38-bit expression is connected to 16-bit port "jdo" of module
"ddr3_ctlr_sequencer_cpu_nios2_oci_itrace", instance
"the_sequencer_cpu_nios2_oci_itrace".
```



```
Expression: jdo
use +lint=PCWM for more details
```

## 8.3. Simulation Walkthrough with Arria 10 EMIF IP

Simulation of the whole memory interface is a good way to determine the latency of your system. However, the latency found in simulation may be different than the latency found on the board because functional simulation does not take into account board trace delays and different process, voltage, and temperature scenarios.

For a given design on a given board, the latency found may differ by one clock cycle (for full-rate designs) or two clock cycles (for half-rate designs) upon resetting the board. Different boards can also show different latencies even with the same design.

The Arria 10 EMIF IP supports only functional simulation. Functional simulation is supported at the RTL level and after generating a post-fit functional simulation netlist. The post-fit netlist for designs that contain Arria 10 EMIF IP is a hybrid of the gate level (for FPGA core) and RTL level (for the external memory interface IP). You should validate the functional operation of your design using RTL simulation, and the timing of your design using TimeQuest Timing Analysis.

For Arria 10 EMIF IP, you can perform functional simulation of an example design that is generated with your IP core. The example design files are created in the `\<variation_name>_example_design` directory.

You can use the IP functional simulation model with any supported VHDL or Verilog HDL simulator.

After you have generated the memory IP, view the `README.txt` file located in the `\<variation_name>_example_design` directory for instructions on how to generate the simulation example design for Verilog HDL or VHDL. Simulation filesets for both Verilog HDL and VHDL are located in `\<variation_name>_example_design\sim`. The **README.txt** file also explains how to run simulation using the ModelSim - Intel FPGA Edition. Simulation scripts for the Mentor Graphics, Cadence, Aldec, and Synopsys simulators are provided; however, detailed instructions on how to perform simulation using these third-party simulators are not provided.

### 8.3.1. Skip Calibration Versus Full Calibration

Calibration must occur shortly after the memory device is initialized, to compensate for uncertainties of the hardware system, including silicon PVT variation, circuit board trace delays, and skewed arrival times. Such variations are usually not present in an RTL simulation environment, therefore there are two options for how the calibration algorithm behaves during simulation: Skip Calibration mode (which is the default), and Full Calibration mode.

#### Skip Calibration Mode

In Skip Calibration mode, the calibration processor assumes an ideal hardware environment, where PVT variations, board delays, and trace skews are all zero. Instead of running the actual calibration routine, the calibration processor calculates the expected arrival time of read data based on the memory latency values that you provide, thus reducing much simulation processing. Skip calibration mode is



recommended for use during system development, because it allows you to focus on interacting with the controller and optimizing your memory access patterns, thus facilitating rapid RTL development.

### Full Calibration Mode

Full Calibration mode simulates every stage of the calibration algorithm immediately after memory device initialization. Because the calibration algorithm processes each data group sequentially and each pin in each group individually, simulation time increases with the number of groups and data pins in your interface. You can observe how the calibration algorithm compensates for various delays in the system by incorporating your own board delay model based on trace delays from your PCB design tools. Due to the large simulation overhead, Full Calibration simulation mode is not recommended for rapid development of IP cores.

### VHDL Support

VHDL support for mixed-language simulators is implemented by generating the top-level wrapper for the core in VHDL, while all submodules are provided as clear text SystemVerilog files.

A set of precompiled device libraries is provided for use with the ModelSim - Intel FPGA Edition single-language simulator which is supplied with the Quartus Prime software. Submodules normally provided as cleartext SystemVerilog files are encrypted using IEEE Verilog HDL encryption for ModelSim - Intel FPGA Edition.

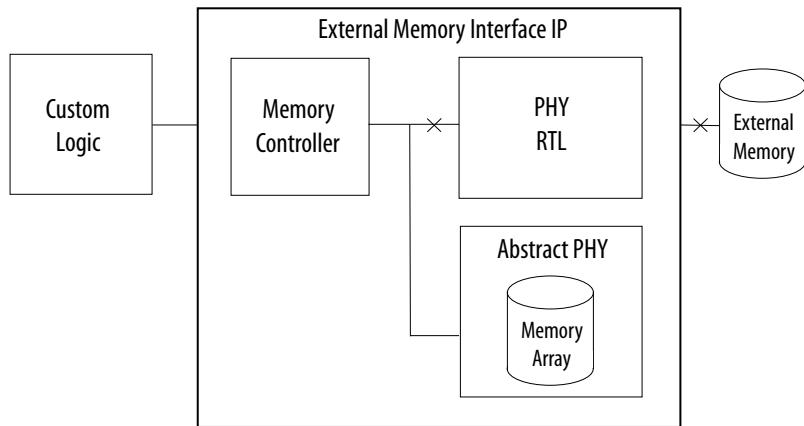
### 8.3.2. Arria 10 Abstract PHY Simulation

The abstract PHY is a simulation model of the EMIF PHY that can decrease simulation time by 3-10 times. The abstract PHY replaces the lane and the external memory model with a single model containing an internal memory array. No switching of the I/Os to the external memory model occurs when simulating with the abstract PHY.

Abstract PHY reduces simulation time by two mechanisms:

- The Nios processor has been disabled and replaced by HDL forces that are applied at the beginning of simulation. The HDL forces are a minimum set of configuration registers that allow the EMIF to be configured properly for simulation. The write and read latency values applied by the HDL forces are not representative of the post-calibration values applied to the EMIF running on hardware. However, as long as the customer logic is avalon- and afi-compliant, these values allow for successful RTL simulation.
- The abstract PHY eliminates the need for full-speed clocks and therefore simulation of the abstract PHY does not require full-speed clock simulation events.

To use the abstract PHY, turn on **Simulation Options > Abstract PHY for fast simulation** on the **Diagnostic** tab. When you turn on the abstract PHY, the EMIF IP is configured as shown below. The PHY RTL and external memory model are disconnected from the data path and in their place is the abstract PHY containing an internal memory array. The abstract PHY is designed with no high-speed clocks, resulting in the removal of all high-speed clock simulator events.

**Figure 66. Abstract PHY**

**Note:** You cannot observe the external memory device signals when you are using the abstract PHY.

If the memory controller is normally part of the EMIF IP, it will continue to be instantiated and used when simulating with the abstract PHY. Because the memory controller regulates the throughput characteristics of data to the external memory interface, these throughput characteristics are maintained when simulating with the abstract PHY. It is important to understand that the abstract PHY is not a cycle-accurate mode of the EMIF IP, and therefore you should not expect to see cycle-accurate behavior.

The HDL forces are created by the Quartus Prime software at IP generation, and therefore you can run abstract PHY simulations immediately upon generation of the EMIF IP.

### 8.3.3. Simulation Scripts

The Quartus Prime software generates three simulation scripts during project generation for four different third party simulation tools—Cadence, Synopsys, Aldec, and Mentor.

The simulation scripts reduce the number of files that you need to compile separately before simulating a design. These scripts are located in four separate folders under the `<project_directory>\<variation_name>_sim` directory, each named after the names of the simulation tools. The example designs also provide equivalent scripts after you run the `.tcl` script from the project located in the `\<variation_name>_example_design\sim` directory.

### 8.3.4. Functional Simulation with Verilog HDL

Simulation scripts for the Synopsys, Cadence, Aldec, and Mentor Graphics simulators are provided for you to run the example design.



The simulation scripts are located in the following main folder locations:

Simulation scripts in the simulation folders are located as follows:

- <variation\_name>\_example\_design\sim\mentor\msim\_setup.tcl
- <variation\_name>\_example\_design\sim\synopsys\vcs\vcs\_setup.sh
- <variation\_name>\_example\_design\sim\synopsys\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_example\_design\sim\aldec\rivierapro\_setup.tcl
- <variation\_name>\_example\_design\sim\cadence\ncsim\_setup.sh

Simulation scripts in the <>\_sim\_folder are located as follows:

- <variation\_name>\_sim\mentor\msim\_setup.tcl
- <variation\_name>\_sim\cadence\ncsim\_setup.sh
- <variation\_name>\_sim\synopsys\vcs\vcs\_setup.sh
- <variation\_name>\_sim\synopsys\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_sim\aldec\rivierapro\_setup.tcl

For more information about simulating Verilog HDL or VHDL designs using command lines, refer to the *Mentor Graphics ModelSim and QuestaSim Support* chapter in volume 3 of the Quartus Prime Handbook.

#### Related Information

[Mentor Graphics ModelSim and QuestaSim Support](#)

### 8.3.5. Functional Simulation with VHDL

The Arria 10 EMIF VHDL fileset is provided for customers that wish to generate the top-level RTL instance of their Arria 10 EMIF cores in VHDL.

Prior to Quartus Prime version 15.1, the VHDL fileset was comprised entirely of VHDL files. Beginning with Quartus Prime version 15.1, only the top-level IP instance file is guaranteed to be written in VHDL; submodules can still be deployed as Verilog/ SystemVerilog (encrypted or plaintext) files, or VHDL files. Note that the ModelSim - Intel FPGA Edition is no longer restricted to a single HDL language as of Quartus 15.1; however, some files may still be encrypted in order to be excluded from the maximum unencrypted module limit of this tool.

Because the VHDL fileset consists of both VHDL and Verilog files, you must follow certain mixed-language simulation guidelines. The general guideline for mixed-language simulation is that you must always link the Verilog files (whether encrypted or not) against the Verilog version of the libraries, and the VHDL files (whether SimGen-generated or pure VHDL) against the VHDL libraries.



Simulation scripts for the Synopsys, Cadence, Aldec, and Mentor Graphics simulators are provided for you to run the example design. These simulation scripts are located in the following main folder locations:

Simulation scripts in the simulation folders are located as follows:

- <variation\_name>\_example\_design\sim\mentor\msim\_setup.tcl
- <variation\_name>\_example\_design\sim\synopsys\vcs\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_example\_design\sim\synopsys\vcs\vcs\_setup.sh
- <variation\_name>\_example\_design\sim\cadence\ncsim\_setup.sh
- <variation\_name>\_example\_design\sim\aldec\rivierapro\_setup.tcl

Simulation scripts in the <>\_sim\_folder are located as follows:

- <variation\_name>\_sim\mentor\msim\_setup.tcl
- <variation\_name>\_sim\cadence\ncsim\_setup.sh
- <variation\_name>\_sim\synopsys\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_sim\aldec\rivierapro\_setup.tcl

For more information about simulating Verilog HDL or VHDL designs using command lines, refer to the *Mentor Graphics ModelSim and QuestaSim Support* chapter in volume 3 of the Quartus Prime Handbook.

#### Related Information

[Mentor Graphics ModelSim and QuestaSim Support](#)

### 8.3.6. Simulating the Example Design

This topic describes how to simulate the example design in Cadence, Synopsys, Mentor Graphics, and Aldec simulators.

To simulate the example design in the Quartus Prime software using the Cadence simulator, follow these steps:

1. At the Linux shell command prompt, change directory to  
    <name>\_example\_design\sim\cadence
2. Run the simulation by typing the following command at the command prompt:

```
sh ncsim_setup.sh
```

To simulate the example design in the Quartus Prime software using the Synopsys simulator, follow these steps:

1. At the Linux shell command prompt, change directory to  
    <name>\_example\_design\sim\synopsys\vcsmx
2. Run the simulation by typing the following command at the command prompt:

```
sh vcsmx_setup.sh
```



To simulate the example design in the Quartus Prime software using the Mentor simulator, follow these steps:

1. At the Linux or Windows shell command prompt, change directory to `<name>_example_design\sim\mentor`
2. Execute the **msim\_setup.tcl** script that automatically compiles and runs the simulation by typing the following command at the Linux or Windows command prompt:

```
vsim -do msim_setup.tcl
```

or

Type the following command at the ModelSim command prompt:

```
do msim_setup.tcl
```

**Note:**

Intel does not provide the `run.do` file for the example design with the Arria 10 EMIF interface.

To simulate the example design in the Quartus Prime software using the Aldec simulator, follow these steps:

1. At the Linux or Windows shell command prompt, change directory to `<name>_example_design\sim\aldec`
2. Execute the **rivierapro\_setup.tcl** script that automatically compiles and runs the simulation by typing the following command at the Linux or Windows command prompt:`vsim -do rivierapro.tcl`
3. To compile and elaborate the design after the script loads, type `ld_debug`.
4. Type `run -all` to run the simulation.

For more information about simulation, refer to the *Simulating Designs* chapter in volume 3 of the Quartus Prime Handbook.

If your Quartus Prime project appears to be configured correctly but the example testbench still fails, check the known issues on the Intel FPGA Knowledge Base before filing a service request.

#### Related Information

- [Simulating Intel FPGA Designs](#)
- [Knowledge Database](#)

## 8.4. Simulation Walkthrough with Stratix 10 EMIF IP

Simulation of the whole memory interface is a good way to determine the latency of your system. However, the latency found in simulation may be different than the latency found on the board because functional simulation does not take into account board trace delays and different process, voltage, and temperature scenarios.

For a given design on a given board, the latency found may differ by one clock cycle (for full-rate designs) or two clock cycles (for half-rate designs) upon resetting the board. Different boards can also show different latencies even with the same design.



The Stratix 10 EMIF IP supports only functional simulation. Functional simulation is supported at the RTL level and after generating a post-fit functional simulation netlist. The post-fit netlist for designs that contain Stratix 10 EMIF IP is a hybrid of the gate level (for FPGA core) and RTL level (for the external memory interface IP). You should validate the functional operation of your design using RTL simulation, and the timing of your design using TimeQuest Timing Analysis.

For Stratix 10 EMIF IP, you can perform functional simulation of an example design that is generated with your IP core. The example design files are created in the `\<variation_name>_example_design` directory.

You can use the IP functional simulation model with any supported VHDL or Verilog HDL simulator.

After you have generated the memory IP, view the `README.txt` file located in the `\<variation_name>_example_design` directory for instructions on how to generate the simulation example design for Verilog HDL or VHDL. Simulation filesets for both Verilog HDL and VHDL are located in `\<variation_name>_example_design\sim`. The `README.txt` file also explains how to run simulation using the ModelSim - Intel FPGA Edition. Simulation scripts for the Mentor Graphics, Cadence, Aldec, and Synopsys simulators are provided; however, detailed instructions on how to perform simulation using these third-party simulators are not provided.

#### **8.4.1. Skip Calibration Versus Full Calibration**

Calibration must occur shortly after the memory device is initialized, to compensate for uncertainties of the hardware system, including silicon PVT variation, circuit board trace delays, and skewed arrival times. Such variations are usually not present in an RTL simulation environment, therefore there are two options for how the calibration algorithm behaves during simulation: Skip Calibration mode (which is the default), and Full Calibration mode.

##### **Skip Calibration Mode**

In Skip Calibration mode, the calibration processor assumes an ideal hardware environment, where PVT variations, board delays, and trace skews are all zero. Instead of running the actual calibration routine, the calibration processor calculates the expected arrival time of read data based on the memory latency values that you provide, thus reducing much simulation processing. Skip calibration mode is recommended for use during system development, because it allows you to focus on interacting with the controller and optimizing your memory access patterns, thus facilitating rapid RTL development.

##### **Full Calibration Mode**

Full Calibration mode simulates every stage of the calibration algorithm immediately after memory device initialization. Because the calibration algorithm processes each data group sequentially and each pin in each group individually, simulation time increases with the number of groups and data pins in your interface. You can observe how the calibration algorithm compensates for various delays in the system by incorporating your own board delay model based on trace delays from your PCB design tools. Due to the large simulation overhead, Full Calibration simulation mode is not recommended for rapid development of IP cores.



## VHDL Support

VHDL support for mixed-language simulators is implemented by generating the top-level wrapper for the core in VHDL, while all submodules are provided as clear text SystemVerilog files.

A set of precompiled device libraries is provided for use with the ModelSim - Intel FPGA Edition single-language simulator which is supplied with the Quartus Prime software. Submodules normally provided as cleartext SystemVerilog files are encrypted using IEEE Verilog HDL encryption for ModelSim - Intel FPGA Edition.

### 8.4.2. Simulation Scripts

The Quartus Prime software generates three simulation scripts during project generation for four different third party simulation tools—Cadence, Synopsys, Aldec, and Mentor.

The simulation scripts reduce the number of files that you need to compile separately before simulating a design. These scripts are located in four separate folders under the `<project_directory>\<variation_name>_sim` directory, each named after the names of the simulation tools. The example designs also provide equivalent scripts after you run the `.tcl` script from the project located in the `\<variation_name>_example_design\sim` directory.

### 8.4.3. Functional Simulation with Verilog HDL

Simulation scripts for the Synopsys, Cadence, Aldec, and Mentor Graphics simulators are provided for you to run the example design.

The simulation scripts are located in the following main folder locations:

Simulation scripts in the simulation folders are located as follows:

- `<variation_name>_example_design\sim\mentor\msim_setup.tcl`
- `<variation_name>_example_design\sim\synopsys\vcs\vcs_setup.sh`
- `<variation_name>_example_design\sim\synopsys\vcsmx\vcsmx_setup.sh`
- `<variation_name>_example_design\sim\aldec\rivierapro_setup.tcl`
- `<variation_name>_example_design\sim\cadence\ncsim_setup.sh`

Simulation scripts in the `<>_sim_folder` are located as follows:

- `<variation_name>_sim\mentor\msim_setup.tcl`
- `<variation_name>_sim\cadence\ncsim_setup.sh`
- `<variation_name>_sim\synopsys\vcs\vcs_setup.sh`
- `<variation_name>_sim\synopsys\vcsmx\vcsmx_setup.sh`
- `<variation_name>_sim\aldec\rivierapro_setup.tcl`

For more information about simulating Verilog HDL or VHDL designs using command lines, refer to the *Mentor Graphics ModelSim and QuestaSim Support* chapter in volume 3 of the Quartus Prime Handbook.



### Related Information

Mentor Graphics ModelSim and QuestaSim Support

#### 8.4.4. Functional Simulation with VHDL

The Stratix 10 EMIF VHDL fileset is provided for customers who want to generate the top-level RTL instance of their Stratix 10 EMIF cores in VHDL.

The top-level IP instance file is guaranteed to be written in VHDL; submodules can be deployed as Verilog/SystemVerilog (encrypted or plaintext) files, or VHDL files. ModelSim - Intel FPGA Edition is not restricted to a single HDL language; however, some files may be encrypted to be excluded from the maximum unencrypted module limit of this tool.

Because the VHDL fileset consists of both VHDL and Verilog files, you must follow certain mixed-language simulation guidelines. The general guideline for mixed-language simulation is that you must always link the Verilog files (whether encrypted or not) against the Verilog version of the libraries, and the VHDL files (whether SimGen-generated or pure VHDL) against the VHDL libraries.

Simulation scripts for the Synopsys, Cadence, Aldec, and Mentor Graphics simulators are provided for you to run the example design. These simulation scripts are located in the following main folder locations:

Simulation scripts in the simulation folders are located as follows:

- <variation\_name>\_example\_design\sim\mentor\msim\_setup.tcl
- <variation\_name>\_example\_design\sim\synopsys\vcs\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_example\_design\sim\synopsys\vcs\vcs\_setup.sh
- <variation\_name>\_example\_design\sim\cadence\ncsim\_setup.sh
- <variation\_name>\_example\_design\sim\aldec\rivierapro\_setup.tcl

Simulation scripts in the <>\_sim\_folder are located as follows:

- <variation\_name>\_sim\mentor\msim\_setup.tcl
- <variation\_name>\_sim\cadence\ncsim\_setup.sh
- <variation\_name>\_sim\synopsys\vcsmx\vcsmx\_setup.sh
- <variation\_name>\_sim\aldec\rivierapro\_setup.tcl

For more information about simulating Verilog HDL or VHDL designs using command lines, refer to the *Mentor Graphics ModelSim and QuestaSim Support* chapter in volume 3 of the Quartus Prime Handbook.

#### 8.4.5. Simulating the Example Design

This topic describes how to simulate the example design in Cadence, Synopsys, Mentor Graphics, and Aldec simulators.



To simulate the example design in the Quartus Prime software using the Cadence simulator, follow these steps:

1. At the Linux shell command prompt, change directory to  
`<name>_example_design\sim\cadence`
2. Run the simulation by typing the following command at the command prompt:

```
sh ncsim_setup.sh
```

To simulate the example design in the Quartus Prime software using the Synopsys simulator, follow these steps:

1. At the Linux shell command prompt, change directory to  
`<name>_example_design\sim\synopsys\vcsmx`
2. Run the simulation by typing the following command at the command prompt:

```
sh vcsmx_setup.sh
```

To simulate the example design in the Quartus Prime software using the Mentor simulator, follow these steps:

1. At the Linux or Windows shell command prompt, change directory to  
`<name>_example_design\sim\mentor`
2. Execute the **msim\_setup.tcl** script that automatically compiles and runs the simulation by typing the following command at the Linux or Windows command prompt:

```
vsim -do msim_setup.tcl
```

or

Type the following command at the ModelSim command prompt:

```
do msim_setup.tcl
```

*Note:*

Intel does not provide the `run.do` file for the example design with the Stratix 10 EMIF interface.

To simulate the example design in the Quartus Prime software using the Aldec simulator, follow these steps:

1. At the Linux or Windows shell command prompt, change directory to  
`<name>_example_design\sim\aldec`
2. Execute the **rivierapro\_setup.tcl** script that automatically compiles and runs the simulation by typing the following command at the Linux or Windows command prompt:  
`vsim -do rivierapro.tcl`
3. To compile and elaborate the design after the script loads, type `ld_debug`.
4. Type `run -all` to run the simulation.

For more information about simulation, refer to the *Simulating Designs* chapter in volume 3 of the Quartus Prime Handbook.

If your Quartus Prime project appears to be configured correctly but the example testbench still fails, check the known issues on the Intel FPGA Knowledge Base before filing a service request.



## 8.5. Document Revision History

Date	Version	Changes
May 2017	2017.05.08	<ul style="list-style-type: none"> <li>Added <i>Simulation Walkthrough with Stratix 10 EMIF IP</i> section.</li> <li>Rebranded as Intel.</li> </ul>
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	<ul style="list-style-type: none"> <li>Added paragraph about incompatibility with the Qsys Testbench System generation feature to <i>Simulating Memory IP</i>.</li> <li>Updated <i>Simulation Options</i> and <i>Abstract PHY Simulation</i> topics for changes to Abstract PHY.</li> <li>Added note to tables 10-1 and 10-2.</li> <li>Updated <i>Functional Simulation with VHDL</i> topic in both <i>Simulation Walkthrough with UniPHY IP</i> and <i>Simulation Walkthrough with Arria 10 EMIF IP</i> sections.</li> </ul>
November 2015	2015.11.02	<ul style="list-style-type: none"> <li>Added <i>Abstract PHY Simulation</i> topic.</li> <li>Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i>.</li> </ul>
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	Removed references to MegaWizard Plug-In Manager.
December 2013	2013.12.16	<ul style="list-style-type: none"> <li>Removed references to ALTMEMPHY.</li> <li>Removed references to HardCopy.</li> <li>Removed references to SOPC Builder.</li> <li>Added <i>Simulation Walkthrough with Arria 10 EMIF IP</i>.</li> <li>Clarified explanation of full calibration mode with abstract models in <i>Abstract PHY</i> section.</li> </ul>
November 2012	6.0	Changed chapter number from 9 to 10.
June 2012	5.0	<ul style="list-style-type: none"> <li>Added path to simulation scripts for Riviera-PRO to <i>Functional Simulations</i> section.</li> <li>Added simulation procedure for Riviera-PRO to <i>Simulating the Example Design</i> section.</li> <li>Updated the <i>Abstract PHY</i> section.</li> <li>Updated the <i>Post-fit Functional Simulation</i> procedure.</li> <li>Added Feedback icon.</li> </ul>
November 2011	4.0	<ul style="list-style-type: none"> <li>Added the <i>PHY-Only Simulation</i> section.</li> <li>Added the <i>Post-fit Functional Simulation</i> section.</li> <li>Updated the <i>Simulation Walkthrough with UniPHY IP</i> section.</li> </ul>
June 2011	3.0	<ul style="list-style-type: none"> <li>Added an overview about memory simulation.</li> <li>Added the <i>Simulation Walkthrough with UniPHY IP</i> section.</li> </ul>
December 2010	2.1	Updated fr 10.1 release.
July 2010	2.0	Updated for 10.0 release.
January 2010	1.1	Corrected typos.
November 2009	1.0	Initial release.

## 9. Analyzing Timing of Memory IP

---

The external memory physical layer (PHY) interface offers a combination of source-synchronous and self-calibrating circuits to maximize system timing margins. The physical layer interface is a plug-and-play solution that the Quartus Prime TimeQuest Timing Analyzer timing constrains and analyzes.

The Intel FPGA IP and the numerous device features offered by Arria® II, Arria V, Intel Arria 10, Cyclone® V, Stratix® III, Stratix IV, and Stratix V FPGAs, greatly simplify the implementation of an external memory interface.

This chapter details the various timing paths that determine overall external memory interface performance, and describes the timing constraints and assumptions that the PHY IP uses to analyze these paths.

This chapter focuses on timing constraints for external memory interfaces based on the UniPHY IP. For information about timing constraints and analysis of external memory interfaces and other source-synchronous interfaces based on the ALTDQ\_DQS and ALTDQ\_DQS2 IP cores, refer to AN 433: *Constraining and Analyzing Source-Synchronous Interfaces* and the *Quartus Prime TimeQuest Timing Analyzer* chapter in volume 3 of the *Quartus Prime Handbook*.

External memory interface timing analysis is supported only by the TimeQuest Timing Analyzer, for the following reasons:

- The wizard-generated timing constraint scripts support only the TimeQuest analyzer.
- The Classic Timing Analyzer does not offer analysis of source-synchronous outputs. For example, write data, address, and command outputs.
- The Classic Timing Analyzer does not support detailed rise and fall delay analysis.

The performance of an FPGA interface to an external memory device is dependent on the following items:

- Read datapath timing
- Write datapath timing
- Address and command path timing
- Clock to strobe timing ( $t_{DQSS}$  in DDR and DDR2 SDRAM, and  $t_{KHK#H}$  in QDR II and QDRII+ SRAM)
- Read resynchronization path timing (applicable for DDR, DDR2, and DDR3 SDRAM in Arria II, Arria 10, Stratix III, Stratix IV, and Stratix V devices)
- Write leveling path timing (applicable for DDR2 and DDR3 SDRAM with UniPHY, and DDR3 and DDR4 SDRAM with Arria 10 EMIF IP.)
- PHY timing paths between I/O element and core registers
- PHY and controller internal timing paths (core  $f_{MAX}$  and reset recovery/removal)



- I/O toggle rate
- Output clock specifications
- Bus turnaround timing (applicable for RLDRAM II and DDR2 and DDR3 SDRAM with UniPHY)

**Note:** External memory interface performance depends on various timing components, and overall system level performance is limited by performance of the slowest link (that is, the path with the smallest timing margins).

#### Related Information

- [AN 433: Constraining and Analyzing Source-Synchronous Interfaces](#)
- [Quartus Prime TimeQuest Timing Analyzer](#)

## 9.1. Memory Interface Timing Components

There are several categories of memory interface timing components, including source-synchronous timing paths, calibrated timing paths, internal FPGA timing paths, and other FPGA timing parameters.

Understanding the nature of timing paths enables you to use an appropriate timing analysis methodology and constraints. The following section examines these aspects of memory interface timing paths.

### 9.1.1. Source-Synchronous Paths

Source-synchronous timing paths are those where clock and data signals pass from the transmitting device to the receiving device.

An example of a source-synchronous timing path is the FPGA-to-memory write datapath. The FPGA device transmits DQ output data signals to the memory along with a center-aligned DQS output strobe signal. The memory device uses the DQS signal to clock the data on the DQ pins into its internal registers.

**Note:** For brevity, the following topics refer to data signals and clock strobe signals as DQ signals and DQS signals, respectively. While the terminology is formally correct only for DDR-type interfaces and does not match QDR II, QDR II+ and RLDRAM II pin names, the behavior is similar enough that most timing properties and concepts apply to both. The clock that captures address and command signals is always referred to as CK/CK# too.

### 9.1.2. Calibrated Paths

Calibrated timing paths are those where the clock used to capture data is dynamically positioned within the data valid window (DVW) to maximize timing margin.

For UniPHY-based controllers and Arria 10 EMIF controllers, the sequencer block analyzes all path delays between the read capture registers and the read FIFO buffer to set up the FIFO write clock phase for optimal timing margin. The read postamble calibration process is implemented in a similar manner to the read resynchronization calibration. In addition, the sequencer block calibrates a read data valid signal to the delay between a controller issuing a read command and read data returning to controller.



In DDR2, DDR3, and RLDRAM II with UniPHY, and in Arria 10 EMIF, the IP calibrates the write-leveling chains and programmable output delay chain to align the DQS edge with the CK edge at memory to meet the  $t_{DQSS}$ ,  $t_{DSS}$ , and  $t_{DSH}$  specifications.

Both UniPHY IP and Arria 10 EMIF IP enable the dynamic deskew calibration with the Nios II sequencer for read and write paths. Dynamic deskew process uses the programmable delay chains that exist within the read and write data paths to adjust the delay of each DQ and DQS pin to remove the skew between different DQ signals and to centre-align the DQS strobe in the DVW of the DQ signals. This process occurs at power up for the read and the write paths.

### 9.1.3. Internal FPGA Timing Paths

Other timing paths that have an impact on memory interface timing include FPGA internal  $f_{MAX}$  paths for PHY and controller logic.

This timing analysis is common to all FPGA designs. With appropriate timing constraints on the design (such as clock settings), the TimeQuest Timing Analyzer reports the corresponding timing margins.

For more information about the TimeQuest Timing Analyzer, refer to the *Quartus Prime TimeQuest Timing Analyzer* chapter in volume 3 of the *Quartus Prime Handbook*.

#### Related Information

[Quartus Prime TimeQuest Timing Analyzer](#)

### 9.1.4. Other FPGA Timing Parameters

Some FPGA data sheet parameters, such as I/O toggle rate and output clock specifications, can limit memory interface performance.

I/O toggle rates vary based on speed grade, loading, and I/O bank location— top/bottom versus left/right. This toggle rate is also a function of the termination used (OCT or external termination) and other settings such as drive strength and slew rate.

**Note:** Ensure you check the I/O performance in the overall system performance calculation. Intel recommends that you perform signal integrity analysis for the specified drive strength and output pin load combination.

For information about signal integrity, refer to the board design guidelines chapters and [AN 476: Impact of I/O Settings on Signal Integrity in Stratix III Devices](#).

Output clock specifications include clock period jitter, half-period jitter, cycle-to-cycle jitter, and skew between FPGA clock outputs. You can obtain these specifications from the FPGA data sheet and must meet memory device requirements. You can use these specifications to determine the overall data valid window for signals transmitted between the memory and FPGA device.

#### Related Information

[AN 476: Impact of I/O Settings on Signal Integrity in Stratix III Devices](#)



## 9.2. FPGA Timing Paths

The following topics describe the FPGA timing paths, the timing constraints examples, and the timing assumptions that the constraint scripts use.

In Arria II, Arria V, Arria V GZ, Arria 10, Cyclone V, Stratix III, Stratix IV, and Stratix V devices, the interface margin is reported based on a combination of the TimeQuest Timing Analyzer and further steps to account for calibration that occurs at runtime. First the TimeQuest analyzer returns the base setup and hold slacks, and then further processing adjusts the slacks to account for effects which cannot be modeled in TimeQuest.

### 9.2.1. Arria II Device PHY Timing Paths

The following table lists all Arria II devices external memory interface timing paths.

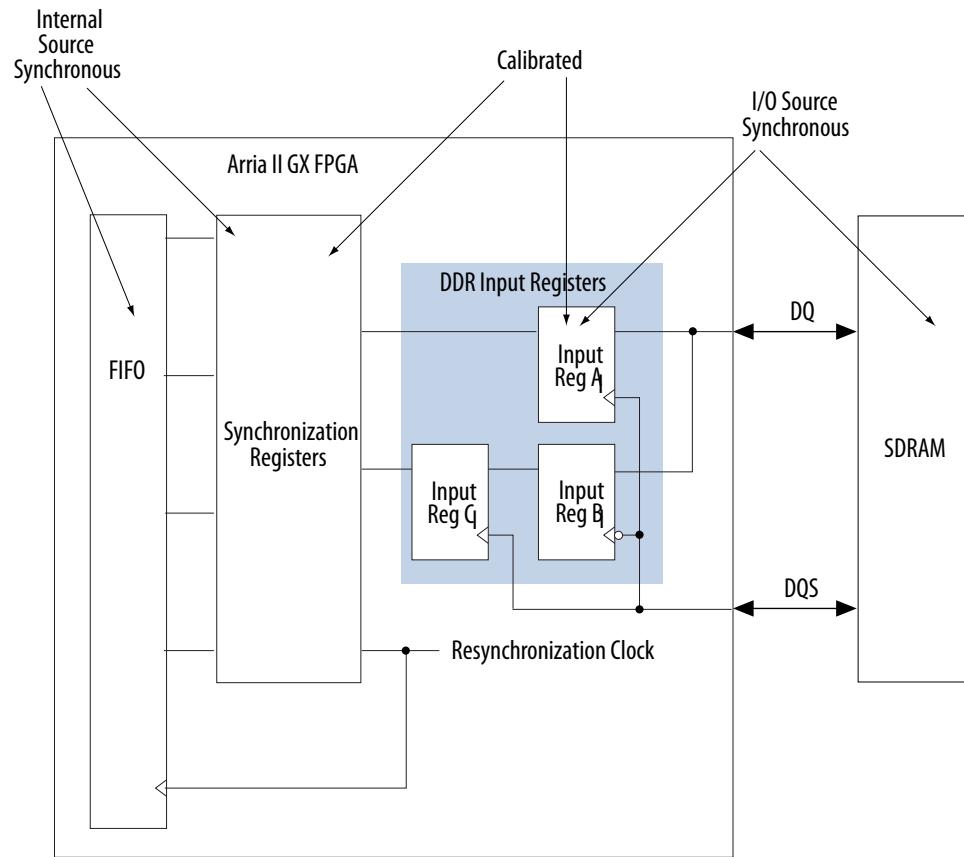
**Table 495. Arria II Devices External Memory Interface Timing Paths (1)**

Timing Path	Circuit Category	Source	Destination
Read Data (2) (6)	Source-Synchronous	Memory DQ, DQS Pins	DQ Capture Registers in IOE
Write Data (2) (6)	Source-Synchronous	FPGA DQ, DQS Pins	Memory DQ, DM, and DQS Pins
Address and command (2)	Source-Synchronous	FPGA CK/CK# and Addr/Cmd Pins	Memory Input Pins
Clock-to-Strobe (2)	Source-Synchronous	FPGA CK/CK# and DQS Output Pins	Memory Input Pins
Read Resynchronization (2)	Calibrated	IOE Capture Registers	IOE Resynchronization Registers
Read Resynchronization (2) (5)	Calibrated	IOE Capture Registers	Read FIFO in FPGA Core
PHY IOE-Core Paths (2)	Source-Synchronous	IOE Resynchronization Registers	FIFO in FPGA Core
PHY and Controller Internal Paths (2)	Internal Clock $f_{MAX}$	Core Registers	Core Registers
I/O Toggle Rate (3)	I/O	FPGA Output Pin	Memory Input Pins
Output Clock Specifications (Jitter, DCD) (4)	I/O	FPGA Output Pin	Memory Input Pins
Notes to Table:			
1. Timing paths applicable for an interface between Arria II devices and SDRAM component. 2. Timing margins for this path are reported by the TimeQuest Timing Analyzer Report DDR function. 3. Intel recommends that you perform signal integrity simulations to verify I/O toggle rate. 4. For output clock specifications, refer to the <i>Arria II Device Data Sheet</i> chapter of the <i>Arria II Handbook</i> . 5. Only for UniPHY IP. 6. Arria II GX devices use source-synchronous and calibrated path.			

The following figure shows the Arria II GX devices input datapath registers and circuit types.

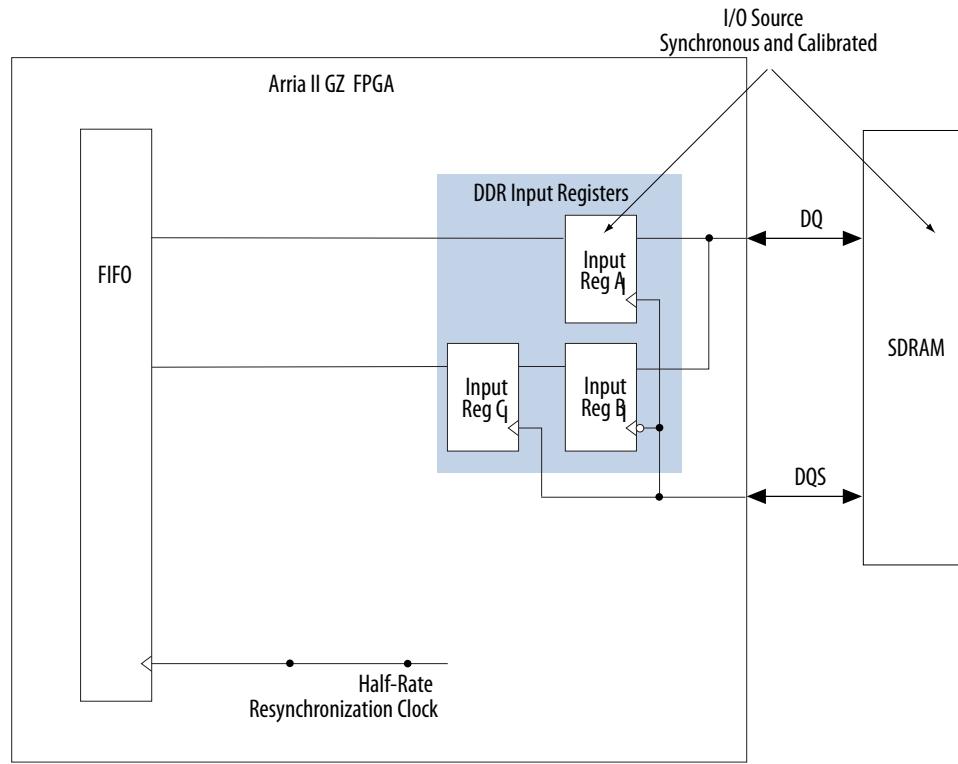
*Note:* UniPHY IP interfaces bypass the synchronization registers.

**Figure 67. Arria II GX Devices Input Data Path Registers and Circuit Types in SDRAM Interface**



The following figure shows the Arria II GZ devices input datapath registers and circuit types.

**Figure 68. Arria II GZ Devices Input Data Path Registers and Circuit Types in SDRAM Interface**



#### Related Information

[Device Datasheet for Arria II Devices](#)

### 9.2.2. Stratix III and Stratix IV PHY Timing Paths

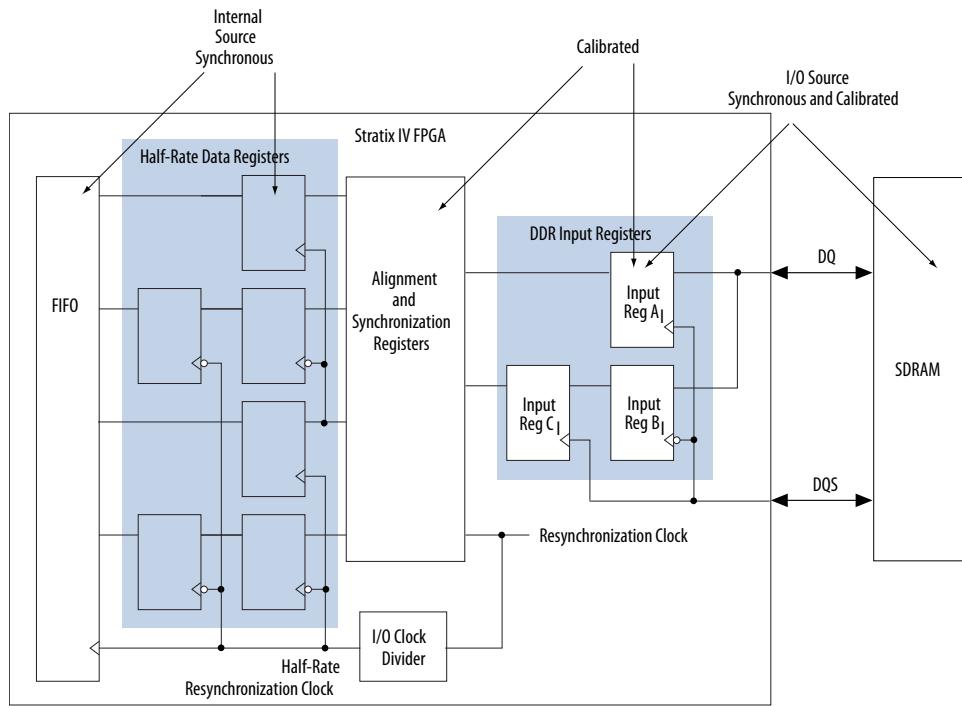
A close look at all the register transfers occurring in the Stratix III and Stratix IV input datapath reveals many source-synchronous and calibrated circuits.

**Note:** The information in the following figure and table is based on Stratix IV devices, but is also applicable to Stratix III devices.

The following figure shows a block diagram of this input path with some of these paths identified for Stratix IV devices. The output datapath contains a similar set of circuits.

**Note:** UniPHY IP interfaces bypass the alignment and synchronization registers.

**Figure 69. Stratix IV Input Path Registers and Circuit Types in SDRAM Interface**



The following table lists the timing paths applicable for an interface between Stratix IV devices and half-rate SDRAM components.

**Note:**

The timing paths are also applicable to Stratix III devices, but Stratix III devices use only source-synchronous path for read and write data paths.

**Table 496. Stratix IV External Memory Interface Timing Paths (Part 1 of 2)**

Timing Path	Circuit Category	Source	Destination
Read Data <sup>(1)</sup>	Source-Synchronous and Calibrated	Memory DQ, DQS Pins	DQ Capture Registers in IOE
Write Data <sup>(1)</sup>	Source-Synchronous and Calibrated	FPGA DQ, DQS Pins	Memory DQ, DM, and DQS Pins
Address and command <sup>(1)</sup>	Source-Synchronous	FPGA CK/CK# and Addr/Cmd Pins	Memory Input Pins
Clock-to-Strobe <sup>(1)</sup>	Source-Synchronous	FPGA CK/CK# and DQS Output Pins	Memory Input Pins
Read Resynchronization <sup>(1)</sup>	Calibrated	IOE Capture Registers	IOE Alignment and Resynchronization Registers
Read Resynchronization <sup>(1)</sup> <sup>(4)</sup>	Calibrated	IOE Capture Registers	Read FIFO in FPGA Core
PHY IOE-Core Paths <sup>(1)</sup>	Source-Synchronous	IOE Half Data Rate Registers and Half-Rate Resynchronization Clock	FIFO in FPGA Core
PHY & Controller Internal Paths <sup>(1)</sup>	Internal Clock $f_{MAX}$	Core registers	Core registers

*continued...*



Timing Path	Circuit Category	Source	Destination
I/O Toggle Rate <sup>(2)</sup>	I/O – Data sheet	FPGA Output Pin	Memory Input Pins
Output Clock Specifications (Jitter, DCD) <sup>(3)</sup>	I/O – Data sheet	FPGA Output Pin	Memory Input Pins

Notes to Table:

1. Timing margins for this path are reported by the TimeQuest Timing Analyzer Report DDR function.
2. Intel recommends that you perform signal integrity simulations to verify I/O toggle rate.
3. For output clock specifications, refer to the DC and Switching Characteristics chapter of the Stratix IV Device Handbook.
4. Only for UniPHY IP.

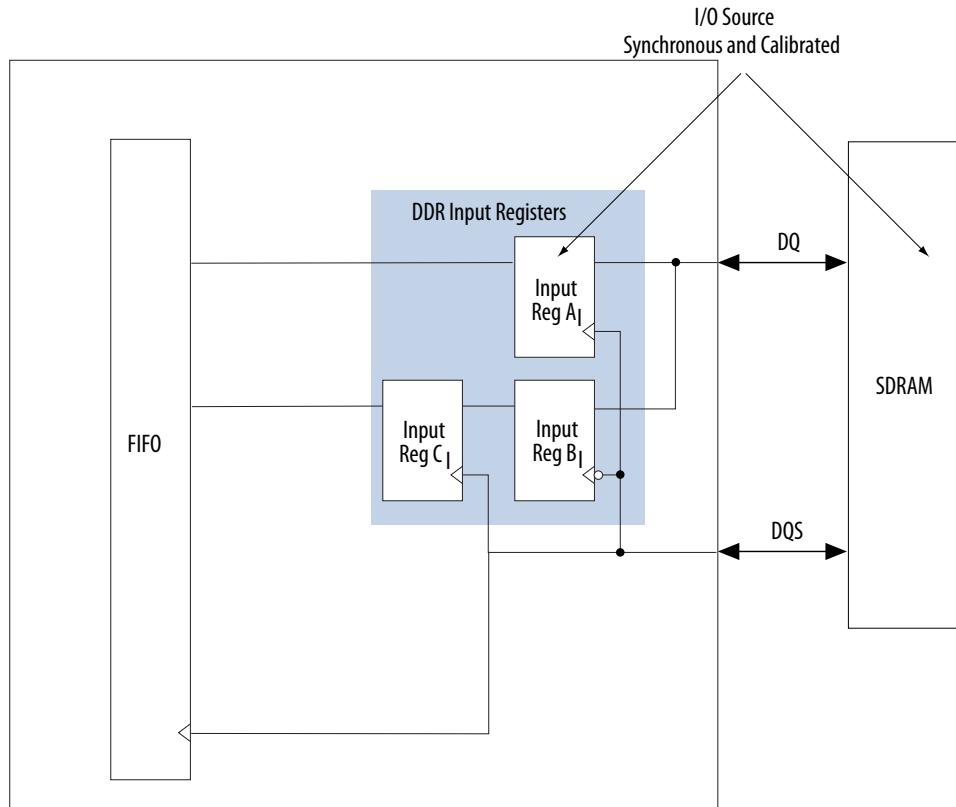
### Related Information

- DC and Switching Characteristics for Stratix III Devices
- DC and Switching Characteristics for Stratix IV Devices

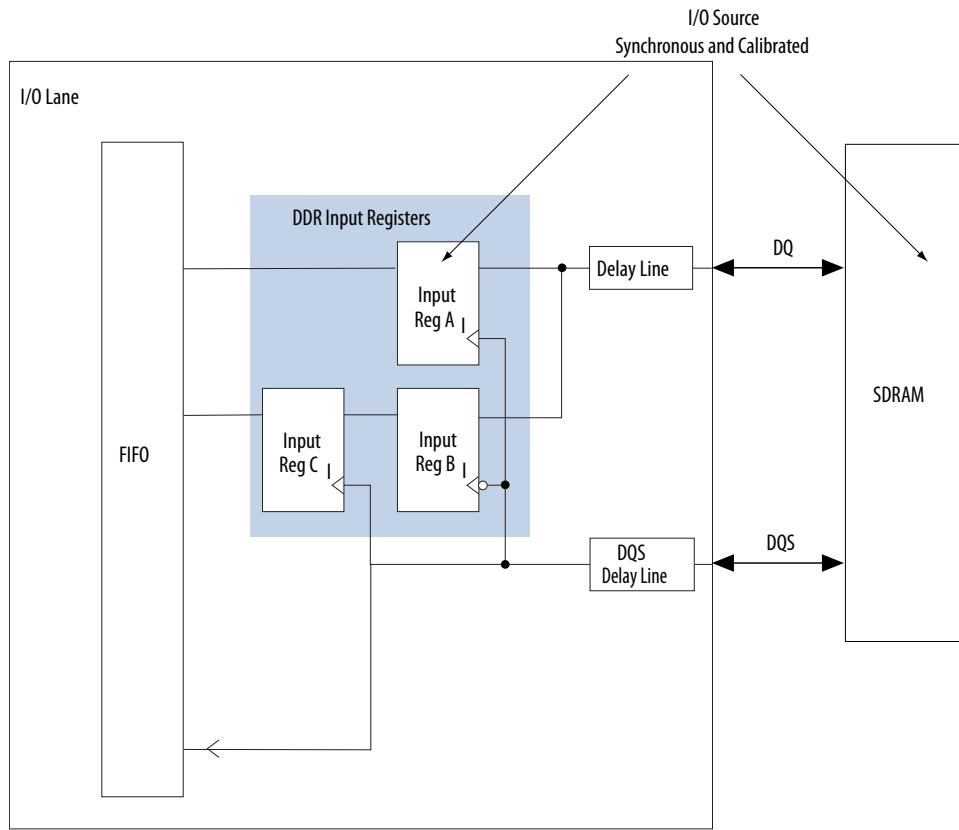
### 9.2.3. Arria V, Arria V GZ, Arria 10, Cyclone V, and Stratix V Timing paths

The following figures show block diagrams of the input data paths for Arria V, Arria V GZ, Cyclone V, and Stratix V devices, and for Arria 10 devices.

**Figure 70. Arria V, Arria V GZ, Cyclone V, and Stratix V Input Data Path**



The following figure shows a block diagram of the Arria 10 input data path.

**Figure 71. Arria 10 Input Data Path**


The following table lists all Arria V, Arria V GZ, Arria 10, Cyclone V, and Stratix V devices external memory interface timing paths.

**Table 497. Arria V, Arria V GZ, Arria 10, Cyclone V, and Stratix V External Memory Interface Timing Paths (1)**

Timing Path	Circuit Category	Source	Destination
Read Data (2)	Source-Synchronous and Calibrated	Memory DQ, DQS Pins	DQ Capture Registers in IOE
Write Data (2)	Source-Synchronous and Calibrated	FPGA DQ, DM, DQS Pins	Memory DQ, DM, and DQS Pins
Address and command (2)	Source-Synchronous	FPGA CK/CK# and Addr/Cmd Pins	Memory Input Pins
Clock-to-Strobe (2)	Source-Synchronous	FPGA CK/CK# and DQS Output Pins	Memory Input Pins
Read Resynchronization (2)	Source-Synchronous	IOE Capture Registers	Read FIFO in IOE
PHY & Controller Internal Paths (2)	Internal Clock fMAX	Core Registers	Core Registers

*continued...*



Timing Path	Circuit Category	Source	Destination
i/O Toggle Rate <sup>(3)</sup>	I/O – Data sheet	FPGA Output Pin	Memory Input Pins
Output Clock Specifications (Jitter, DCD) <sup>(4)</sup>	I/O – Data sheet	FPGA Output Pin	Memory Input Pins

Notes to Table:

1. This table lists the timing paths applicable for an interface between Arria V, Arria V GZ, Cyclone V, and Stratix V devices and half-rate SDRAM components.
2. Timing margins for this path are reported by the TimeQuest Timing Analyzer Report DDR function.
3. Intel recommends that you perform signal integrity simulations to verify I/O toggle rate.
4. For output clock specifications, refer to the *DC and Switching Characteristics* chapter of the respective *Device Handbook*.

The following table lists the Arria 10 external memory interface timing paths.

Timing Path	Circuit Category	Source	Destination
Read Data <sup>(1)</sup>	Source-Synchronous and Calibrated	Memory DQ, DQS Pins	DQ Capture Registers in IOE
Write Data <sup>(1)</sup>	Source-Synchronous and Calibrated	FPGA DQ, DM, DQS Pins	Memory DQ, DM, and DQS Pins
Address and Command <sup>(1)</sup>	Source-Synchronous	FPGA CK/CK# and Address/Command Pins	Memory Input Pins
Clock-to-Strobe <sup>(1)</sup>	Source-Synchronous	FPGA CK/CK# and DQS Output Pins	Memory Input Pins
PHY & Controller Internal Paths	Internal Clock fmax	Core Registers	Core Registers
I/O Toggle Rate <sup>(2)</sup>	I/O Data sheet	FPGA Output Pin	Memory Input Pins
Output Clock Specifications (Jitter, DCD) <sup>(3)</sup>	I/O Data sheet	FPGA Output Pin	Memory Input Pins

Notes to Table:

1. The Report DDR function in the TimeQuest Timing Analyzer reports the timing margins for this path.
2. You should perform signal integrity simulations for verify I/O toggle rate.
3. For output clock verifications, refer to the *DC and Switching Characteristics* chapter of the *Intel Arria 10 Device Handbook*.

### 9.3. Timing Constraint and Report Files for UniPHY IP

To ensure a successful external memory interface operation, the UniPHY IP generates two sets of files for timing constraints but in different folders and with slightly different filenames.

One set of files are used for the synthesis project, which is available under the `<variation_name>` folder located in the main project folder while the other set of files are the example designs, located in the `<variation_name>example design \example_project` folder.

The project folders contain the following files for timing constraints and reporting scripts:

- `<variation_name>.sdc`
- `<variation_name>_timing.tcl`
- `<variation_name>_report_timing.tcl`



- <variation\_name>**\_report\_timing\_core.tcl**
- <variation\_name>**\_pin\_map.tcl**
- <variation\_name>**\_parameters.tcl**

#### **<variation\_name>.sdc**

The <variation\_name>**.sdc** is listed in the wizard-generated Quartus Prime IP File (**.qip**). Including this file in the project allows the Quartus Prime Synthesis and Fitter to use the timing driven compilation to optimize the timing margins.

To analyze the timing margins for all UniPHY timing paths, execute the Report DDR function in the TimeQuest Timing Analyzer.

The UniPHY IP uses the **.sdc** to constrain internal FPGA timing paths, address and command paths, and clock-to-strobe timing paths, and more specifically:

- Creating clocks on PLL inputs
- Creating generated clocks
- Calling derive\_clock\_uncertainty
- Cutting timing paths for specific reset paths
- Setting input and output delays on DQ inputs and outputs
- Setting output delays on address and command outputs (versus CK/CK# outputs)

#### **<variation\_name>\_timing.tcl**

This script includes the memory, FPGA, and board timing parameters for your variation. It is included within <variation\_name>**\_report\_timing.tcl** and <variation\_name>**.sdc**.

#### **<variation\_name>\_report\_timing.tcl**

This script reports the timing slack for your variation. It runs automatically during compilation (during static timing analysis). You can also run this script with the Report DDR task in the TimeQuest Timing Analyzer. This script is run for every instance of the same variation.

#### **<variation\_name>\_report\_timing\_core.tcl**

This script contains high-level procedures that the <variation\_name>**\_report\_timing.tcl** script uses to compute the timing slack for your variation. This script runs automatically during compilation.

#### **<variation\_name>\_pin\_map.tcl**

This script is a library of functions and procedures that the <variation\_name>**\_report\_timing.tcl** and <variation\_name>**.sdc** scripts use. The <variation\_name>**\_pin\_assignments.tcl** script, which is not relevant to timing constraints, also uses this library.



#### **<variation\_name>\_parameters.tcl**

This script defines some of the parameters that describe the geometry of the core and the PLL configuration. Do not change this file, except when you modify the PLL through the parameter editor. In this case, the changes to the PLL parameters do not automatically propagate to this file and you must manually apply those changes in this file.

## 9.4. Timing Constraint and Report Files for Arria 10 EMIF IP

To ensure a successful external memory interface operation, the Arria 10 EMIF IP generates two sets of files for timing constraints but in different folders and with slightly different filenames.

One set of files are used for synthesis project, which is available under the *<variation\_name>* folder located in the main project folder while the other set of files are the example designs, located in the *<variation\_name>\_example* design\qii folder.

The project folders contain the following files for timing constraints and reporting scripts:

- *<variation\_name>.sdc*
- *<variation\_name>\_ip\_parameters.tcl*
- *<variation\_name>\_parameters.tcl*
- *<variation\_name>\_pin\_map.tcl*
- *<variation\_name>\_report\_timing.tcl*

#### **<variation\_name>.sdc**

The *<variation\_name>.sdc* file is listed in the Quartus Prime IP File (.qip), which you generate by running **make\_qii\_design.tcl**. The *<variation\_name>.sdc* file allows the Quartus Prime fitter to optimize timing margins with timing-driven compilation.

To analyze timing margins for all Arria 10 external memory interface IP timing paths, run the **Report DDR** function in the TimeQuest Timing Analyzer.

The Arria 10 EMIF IP uses the **.sdc** file for the following operations:

- Creating clocks on PLL inputs
- Creating generated clocks
- Calling **derive\_clock\_uncertainty**
- Creating a false path from a user clock to a hard memory controller clock, and vice versa
- Setting output delays on address and command outputs (versus CK/CK# outputs)

#### **<variation\_name>\_ip\_parameters.tcl**

The *<variation\_name>\_ip\_parameters.tcl* file is a script that lists the Arria 10 EMIF IP memory parameters and board parameters defined in the MegaWizard, which are used in the **.sdc** file and timing report scripts.



#### **<variation\_name>\_parameters.tcl**

The **<variation\_name>\_parameters.tcl** file is a script that lists the Arria 10 EMIF IP device and speed grade dependent values, which are used in the **.sdc** file and report timing scripts:

- Jitter
- Simultaneous switching noise
- Duty cycle distortion
- Calibration uncertainties

#### **<variation\_name>\_pin\_map.tcl**

The **<variation\_name>\_pin\_map.tcl** file is a library of functions and procedures that the **<variation\_name>report\_timing.tcl** and **<variation\_name>.sdc** scripts use.

#### **<variation\_name>\_report\_timing.tcl**

The **<variation\_name>\_report\_timing.tcl** file is a script that contains timing analysis flow and reports the timing slack for your variation. This script runs automatically during calibration (during static timing analysis) by sourcing the following files:

- **<variation\_name>\_ip\_parameters.tcl**
- **<variation\_name>\_parameters.tcl**
- **<variation\_name>\_pin\_map.tcl**
- **<variation\_name>\_report\_timing\_core.tcl**

You can also run **<variation\_name>\_report\_timing.tcl** with the **Report DDR** function in the TimeQuest Timing Analyzer. This script runs for every instance of the same variation.

#### **<variation\_name>\_report\_timing\_core.tcl**

The **<variation\_name>\_report\_timing\_core.tcl** file is a script that **<variation\_name>\_report\_timing.tcl** uses to calculate the timing slack for your variation. **<variation\_name>\_report\_timing\_core.tcl** runs automatically during compilation.

#### **<variation\_name>\_report\_io\_timing.tcl**

The **<variation\_name>\_report\_io\_timing.tcl** file is a script that contains an early I/O estimation for your external memory interface design, excluding FPGA core timing analysis. This script allows you to determine early I/O margins without having to compile your design.

## **9.5. Timing Constraint and Report Files for Stratix 10 EMIF IP**

To ensure a successful external memory interface operation, the Stratix 10 EMIF IP generates two sets of files for timing constraints but in different folders and with slightly different filenames.

One set of files are used for synthesis project, which is available under the **<variation\_name>** folder located in the main project folder while the other set of files are the example designs, located in the **<variation\_name>\_example\_design\qii\ip** folder.



The project folders contain the following files for timing constraints and reporting scripts:

- **<variation\_name>.sdc**
- **<variation\_name>\_ip\_parameters.tcl**
- **<variation\_name>\_parameters.tcl**
- **<variation\_name>\_pin\_map.tcl**
- **<variation\_name>\_report\_timing.tcl**

#### **<variation\_name>.sdc**

The **<variation\_name>.sdc** file allows the Quartus Prime fitter to optimize timing margins with timing-driven compilation.

To analyze timing margins for all Stratix 10 external memory interface IP timing paths, run the **Report DDR** function in the TimeQuest Timing Analyzer.

The Stratix 10 EMIF IP uses the **.sdc** file for the following operations:

- Creating clocks on PLL inputs
- Creating generated clocks
- Calling **derive\_clock\_uncertainty**
- Creating a false path from a user clock to a hard memory controller clock, and vice versa
- Setting output delays on address and command outputs (versus CK/CK# outputs)

#### **<variation\_name>\_ip\_parameters.tcl**

The **<variation\_name>\_ip\_parameters.tcl** file is a script that lists the Stratix 10 EMIF IP memory parameters and board parameters defined in the MegaWizard, which are used in the **.sdc** file and timing report scripts.

#### **<variation\_name>\_parameters.tcl**

The **<variation\_name>\_parameters.tcl** file is a script that lists the Stratix 10 EMIF IP device and speed grade dependent values, which are used in the **.sdc** file and report timing scripts:

- Jitter
- Simultaneous switching noise
- Duty cycle distortion
- Calibration uncertainties

#### **<variation\_name>\_pin\_map.tcl**

The **<variation\_name>\_pin\_map.tcl** file is a library of functions and procedures that the **<variation\_name>report\_timing.tcl** and **<variation\_name>.sdc** scripts use.



### **<variation\_name>\_report\_timing.tcl**

The **<variation\_name>\_report\_timing.tcl** file is a script that contains timing analysis flow and reports the timing slack for your variation. This script runs automatically during calibration (during static timing analysis) by sourcing the following files:

- **<variation\_name>\_ip\_parameters.tcl**
- **<variation\_name>\_parameters.tcl**
- **<variation\_name>\_pin\_map.tcl**
- **<variation\_name>\_report\_timing\_core.tcl**

You can also run **<variation\_name>\_report\_timing.tcl** with the **Report DDR** function in the TimeQuest Timing Analyzer. This script runs for every instance of the same variation.

### **<variation\_name>\_report\_timing\_core.tcl**

The **<variation\_name>\_report\_timing\_core.tcl** file is a script that **<variation\_name>\_report\_timing.tcl** uses to calculate the timing slack for your variation. **<variation\_name>\_report\_timing\_core.tcl** runs automatically during compilation.

### **<variation\_name>\_report\_io\_timing.tcl**

The **<variation\_name>\_report\_io\_timing.tcl** file is a script that contains an early I/O estimation for your external memory interface design, excluding FPGA core timing analysis. This script allows you to determine early I/O margins without having to compile your design.

## **9.6. Timing Analysis Description**

The following sections describe the timing analysis using the respective FPGA data sheet specifications and the user-specified memory data sheet parameters.



- Core to core (C2C) transfers have timing constraint created and are timing analyzed by TimeQuest Analyzer. Core timing does not include user logic timing within core or to and from EMIF block. Both UniPHY-based IP and Arria 10 EMIF IP provide the constrained clock to the customer logic.
- Core to periphery (C2P) transfers have timing constraint created and are timing analyzed by TimeQuest Analyzer. This is common for UniPHY and Arria 10 External Memory Interfaces IP. Because of the increased number of C2P/P2C signals in 20nm families compared to previous families, more work is expected to ensure that these special timing arcs are properly modeled, both during TimeQuest and compilation.
- Periphery to core (P2C) transfers have timing constraint created and are timing analyzed by TimeQuest Analyzer. This is common for UniPHY and Arria 10 External Memory Interfaces IP. Because of the increased number of C2P/P2C signals in 20nm families compared to previous families, more work is expected to ensure that these special timing arcs are properly modeled, both during TimeQuest and compilation.
- Periphery to periphery (P2P) transfers are modeled entirely by a minimum pulse width violation on the hard block, and have no internal timing arc. In UniPHY-based IP, P2P transfers are reported as part of Core Timing analysis. For Arria 10 EMIF IP, P2P transfers are modeled only by a minimum pulse width violation on hardened block.

To account for the effects of calibration, the UniPHY IP and Arria 10 EMIF IP include additional scripts that are part of the

`<phy_variation_name>_report_timing.tcl` and `<phy_variation_name>_report_timing_core.tcl` files that determine the timing margin after calibration. These scripts use the setup and hold slacks of individual pins to emulate what is occurring during calibration to obtain timing margins that are representative of calibrated PHYs. The effects considered as part of the calibrated timing analysis include improvements in margin because of calibration, and quantization error and calibration uncertainty because of voltage and temperature changes after calibration. The calibration effects do not apply to Stratix III devices.

### Related Information

[Timing Constraint and Report Files for UniPHY IP](#) on page 503

## 9.6.1. UniPHY IP Timing Analysis

The following topics describe timing analysis for UniPHY-based external memory interface IP.

### 9.6.1.1. Address and Command

Address and command signals are single data rate signals latched by the memory device using the FPGA output clock.

Some of the address and command signals are half-rate data signals, while others, such as the chip select, are full-rate signals. The TimeQuest Timing Analyzer analyzes the address and command timing paths using the `set_output_delay (max and min)` constraints.



### 9.6.1.2. PHY or Core

Timing analysis of the PHY or core path includes the path of soft registers in the device and the register in the I/O element.

However, the analysis does not include the paths through the pin or the calibrated path. The PHY or core analyzes this path by calling the `report_timing` command in `<variation_name>_report_timing.tcl` and `<variation_name>_report_timing_core.tcl`.

### 9.6.1.3. PHY or Core Reset

The PHY or core reset is the internal timing of the asynchronous reset signals to the UniPHY IP.

The PHY or core analyzes this path by calling the `report_timing` command in `<variation_name>_report_timing.tcl` and `<variation_name>_report_timing_core.tcl`.

### 9.6.1.4. Read Capture and Write

Stratix III memory interface designs perform read capture and write timing analysis using the TCCS and SW timing specification.

Read capture and write timing analysis for Arria II, Cyclone IV, Stratix IV, and Stratix V memory interface designs are based on the timing slacks obtained from the TimeQuest Timing Analyzer and all the effects included with the Quartus Prime timing model such as die-to-die and within-die variations, aging, systematic skew, and operating condition variations. Because the PHY IP adjusts the timing slacks to account for the calibration effects, there are two sets of read capture and write timing analysis numbers—**Before Calibration** and **After Calibration**.

#### 9.6.1.4.1. Stratix III

This topic details the timing margins, such as the read data and write data timing paths, which the TimeQuest Timing Analyzer callates for Stratix III designs. Timing paths internal to the FPGA are either guaranteed by design and tested on silicon, or analyzed by the TimeQuest Timing Analyzer using corresponding timing constraints.

For design guidelines about implementing and analyzing your external memory interface using the PHY in Stratix III and Stratix IV devices, refer to the design tutorials on the *List of designs using Intel FPGA External Memory IP* page of [www.alterawiki.com](http://www.alterawiki.com).

Timing margins for chip-to-chip data transfers can be defined as:

Margin = bit period – transmitter uncertainties – receiver requirements

where:

- Sum of all transmitter uncertainties = transmitter channel-to-channel skew (TCCS).

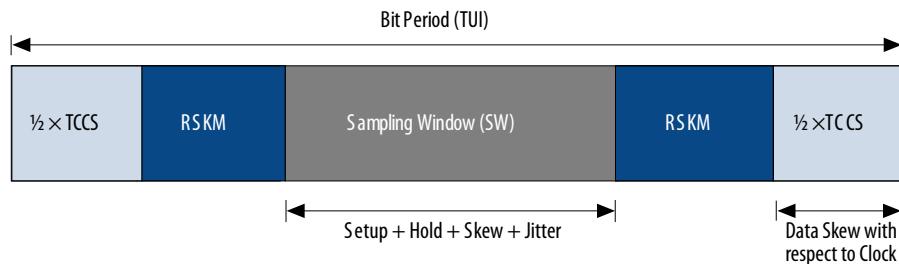
The timing difference between the fastest and slowest output edges on data signals, including  $t_{CO}$  variation, clock skew, and jitter. The clock is included in the TCCS measurement and serves as the time reference.



- Sum of all receiver requirements = receiver sampling window (SW) requirement.  
The period of time during which the data must be valid to capture it correctly. The setup and hold times determine the ideal strobe position within the sampling window.
- Receiver skew margin (RSKM) = margin or slack at the receiver capture register.  
For TCCS and SW specifications, refer to the *DC and Switching Characteristics* chapter of the *Stratix III Device Handbook*.

The following figure relates this terminology to a timing budget diagram.

**Figure 72. Sample Timing Budget Diagram**



The timing budget regions marked " $\frac{1}{2} \times \text{TCCS}$ " represent the latest data valid time and earliest data invalid times for the data transmitter. The region marked sampling window is the time required by the receiver during which data must stay stable. This sampling window comprises the following:

- Internal register setup and hold requirements
- Skew on the data and clock nets within the receiver device
- Jitter and uncertainty on the internal capture clock

**Note:** The sampling window is not the capture margin or slack, but instead the requirement from the receiver. The margin available is denoted as RSKM.

The simple example illustrated in the preceding figure does not consider any board level uncertainties, assumes a center-aligned capture clock at the middle of the receiver sampling window region, and assumes an evenly distributed TCCS with respect to the transmitter clock pin. In this example, the left end of the bit period corresponds to time  $t = 0$ , and the right end of the bit period corresponds to time  $t = \text{TUI}$  (where TUI stands for time unit interval). Therefore, the center-aligned capture clock at the receiver is best placed at time  $t = \text{TUI}/2$ .

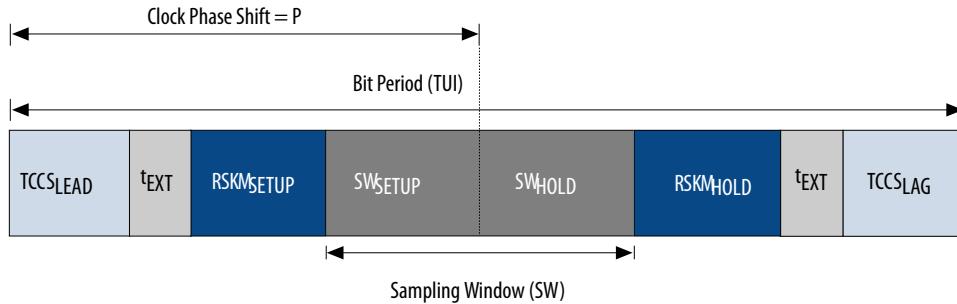
Therefore:

$$\text{the total margin} = 2 \times \text{RSKM} = \text{TUI} - \text{TCCS} - \text{SW}.$$

Consider the case where the clock is not center-aligned within the bit period (clock phase shift =  $P$ ), and the transmitter uncertainties are unbalanced ( $\text{TCCS}_{\text{LEAD}}$  and  $\text{TCCS}_{\text{LAG}}$ ).  $\text{TCCS}_{\text{LEAD}}$  is defined as the skew between the clock signal and latest data valid signal.  $\text{TCCS}_{\text{LAG}}$  is defined as the skew between the clock signal and earliest data invalid signal. Also, the board level skew across data and clock traces are specified as  $t_{\text{EXT}}$ . For this condition, you should compute independent setup and hold margins at the receiver ( $\text{RSKM}_{\text{SETUP}}$  and  $\text{RSKM}_{\text{HOLD}}$ ). In this example, the sampling window requirement is split into a setup side requirement ( $\text{SW}_{\text{SETUP}}$ ) and hold side ( $\text{SW}_{\text{HOLD}}$ )

requirement. The following figure illustrates the timing budget for this condition. A timing budget similar to that shown is used for Stratix III FPGA read and write data timing paths.

**Figure 73. Sample Timing Budget with Unbalanced (TCCS and SW) Timing Parameters**



Therefore:

$$\text{Setup margin} = \text{RSKM}_{\text{SETUP}} = P - \text{TCCS}_{\text{LEAD}} - \text{SW}_{\text{SETUP}} - t_{\text{EXT}}$$

$$\text{Hold margin} = \text{RSKM}_{\text{HOLD}} = (\text{TUI} - P) - \text{TCCS}_{\text{LAG}} - \text{SW}_{\text{HOLD}} - t_{\text{EXT}}$$

The timing budget illustrated in the first figure with balanced timing parameters applies for calibrated paths where the clock is dynamically center-aligned within the data valid window. The timing budget illustrated in the second figure with unbalanced timing parameters applies for circuits that employ a static phase shift using a DLL or PLL to place the clock within the data valid window.

#### Related Information

- [Read Capture](#) on page 512
- [List of designs using Intel FPGA External Memory IP](#)
- [Stratix III Device Handbook](#)

#### Read Capture

Memory devices provide edge-aligned DQ and DQS outputs to the FPGA during read operations. Stratix III FPGAs center-aligns the DQS strobe using static DLL-based delays. Stratix III devices use a source synchronous circuit for data capture.

When applying this methodology to read data timing, the memory device is the transmitter and the FPGA device is the receiver.

The transmitter channel-to-channel skew on outputs from the memory device is available from the corresponding device data sheet. Let us examine the TCCS parameters for a DDR2 SDRAM component.

For DQS-based capture:

- The time between DQS strobe and latest data valid is defined as t<sub>DQSQ</sub>
- The time between earliest data invalid and next strobe is defined as t<sub>QHS</sub>
- Based on earlier definitions, TCCS<sub>LEAD</sub> = t<sub>DQSQ</sub> and TCCS<sub>LAG</sub> = t<sub>QHS</sub>



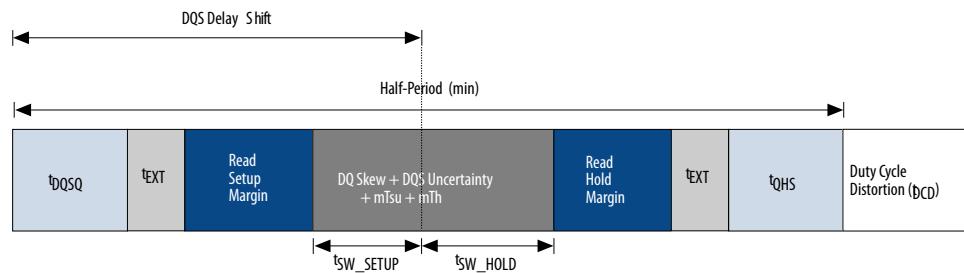
The sampling window at the receiver, the FPGA, includes several timing parameters:

- Capture register micro setup and micro hold time requirements
- DQS clock uncertainties because of DLL phase shift error and phase jitter
- Clock skew across the DQS bus feeding DQ capture registers
- Data skew on DQ paths from pin to input register including package skew

For TCCS and SW specifications, refer to the *DC and Switching Characteristics* chapter of the *Stratix III Device Handbook*.

The following figure shows the timing budget for a read data timing path.

**Figure 74. Timing Budget for Read Data Timing Path**



The following table lists a read data timing analysis for a Stratix III -2 speed-grade device interfacing with a 400-MHz DDR2 SDRAM component.

**Table 498. Read Data Timing Analysis for Stratix III Device with a 400-MHz DDR2 SDRAM (1)**

Parameter	Specifications	Value (ps)	Description
Memory Specifications (1)	t <sub>HP</sub>	1250	Average half period as specified by the memory data sheet, $t_{HP} = 1/2 * t_{CK}$
	t <sub>DCD</sub>	50	Duty cycle distortion = $2\% \times t_{CK} = 0.02 \times 2500 \text{ ps}$
	t <sub>DQSQ</sub>	200	Skew between DQS and DQ from memory
	t <sub>QHS</sub>	300	Data hold skew factor as specified by memory
FPGA Specifications	t <sub>SW_SETUP</sub>	181	FPGA sampling window specifications for a given configuration (DLL mode, width, location, and so on.)
	t <sub>SW_HOLD</sub>	306	
Board Specifications	t <sub>EXT</sub>	20	Maximum board trace variation allowed between any two signal traces (user specified parameter)
Timing Calculations	t <sub>DVW</sub>	710	$t_{HP} - t_{DCD} - t_{DQSQ} - t_{QHS} - 2 \times t_{EXT}$
	t <sub>DQS_PHASE_DELAY</sub>	500	Ideal phase shift delay on DQS capture strobe $= (\text{DLL phase resolution} \times \text{number of delay stages} \times t_{CK}) / 360^\circ = (36^\circ \times 2 \text{ stages} \times 2500 \text{ ps}) / 360^\circ = 500 \text{ ps}$

*continued...*

Parameter	Specifications	Value (ps)	Description
Results	Setup margin	99	$RSKM_{SETUP} = t_{DQSQ\_PHASE\_DELAY} - t_{DQSQ} - t_{SW\_SETUP} - t_{EXT}$
	Hold margin	74	$RSKM_{HOLD} = t_{HP} - t_{DCD} - t_{DQS\_PHASE\_DELAY} - t_{QHS} - t_{SW\_HOLD} - t_{EXT}$
Notes to Table:			
1. This sample calculation uses memory timing parameters from a 72-bit wide 256-MB micron MT9HTF3272AY-80E 400-MHz DDR2 SDRAM DIMM.			

### Related Information

[Stratix III Device Handbook](#)

### Write Capture

During write operations, the FPGA generates a DQS strobe and a center-aligned DQ data bus using multiple PLL-driven clock outputs. The memory device receives these signals and captures them internally. The Stratix III family contains dedicated DDIO (double data rate I/O) blocks inside the IOEs.

For write operations, the FPGA device is the transmitter and the memory device is the receiver. The memory device's data sheet specifies data setup and data hold time requirements based on the input slew rate on the DQ/DQS pins. These requirements make up the memory sampling window, and include all timing uncertainties internal to the memory.

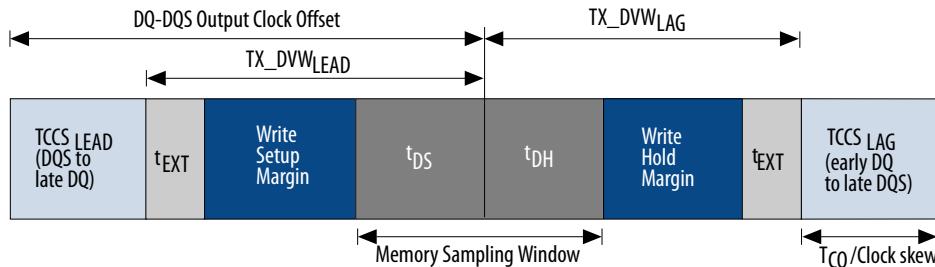
Output skew across the DQ and DQS output pins on the FPGA make up the TCCS specification. TCCS includes contributions from numerous internal FPGA circuits, including:

- Location of the DQ and DQS output pins
- Width of the DQ group
- PLL clock uncertainties, including phase jitter between different output taps used to center-align DQS with respect to DQ
- Clock skew across the DQ output pins, and between DQ and DQS output pins
- Package skew on DQ and DQS output pins

Refer to the *DC and Switching Characteristics* chapter of the *Stratix III Device Handbook* for TCCS and SW specifications.

The following figure illustrates the timing budget for a write data timing path.

**Figure 75. Timing Budget for Write Data Timing Path**





The following table lists a write data timing analysis for a Stratix III -2 speed-grade device interfacing with a DDR2 SDRAM component at 400 MHz. This timing analysis assumes the use of a differential DQS strobe with 2.0-V/ns edge rates on DQS, and 1.0-V/ns edge rate on DQ output pins. Consult your memory device's data sheet for derated setup and hold requirements based on the DQ/DQS output edge rates from your FPGA.

**Table 499. Write Data Timing Analysis for 400-MHz DDR2 SDRAM Stratix III Device <sup>(1)</sup>**

Parameter	Specifications	Value (ps)	Description
Memory Specifications <sup>(1)</sup>	t <sub>HP</sub>	1250	Average half period as specified by the memory data sheet
	t <sub>DSA</sub>	250	Memory setup requirement (derated for DQ/DQS edge rates and V <sub>REF</sub> reference voltage)
	t <sub>DHA</sub>	250	Memory hold requirement (derated for DQ/DQS edge rates and V <sub>REF</sub> reference voltage)
FPGA Specifications	TCCS <sub>LEAD</sub>	229	FPGA transmitter channel-to-channel skew for a given configuration (PLL setting, location, and width).
	TCCS <sub>LAG</sub>	246	
Board Specifications	t <sub>EXT</sub>	20	Maximum board trace variation allowed between any two signal traces (user specified parameter)
Timing Calculations	t <sub>OUTPUT_CLOCK_OFFSET</sub>	625	Output clock phase offset between DQ & DQS output clocks = 90°. $t_{OUTPUT\_CLOCK\_OFFSET} = (\text{output clock phase DQ and DQS offset} \times t_{CK})/360^\circ = (90^\circ \times 2500)/360^\circ = 625$
	TX_DVW <sub>LEAD</sub>	396	Transmitter data valid window = t <sub>OUTPUT_CLOCK_OFFSET</sub> - TCCS <sub>LEAD</sub>
	TX_DVW <sub>LAG</sub>	379	Transmitter data valid window = t <sub>HP</sub> - t <sub>OUTPUT_CLOCK_OFFSET</sub> - TCCS <sub>LAG</sub>
Results	Setup margin	126	TX_DVW <sub>LEAD</sub> - t <sub>EXT</sub> - t <sub>DSA</sub>
	Hold margin	109	TX_DVW <sub>LAG</sub> - t <sub>EXT</sub> - t <sub>DHA</sub>

Notes to Table:

1. This sample calculation uses memory timing parameters from a 72-bit wide 256-MB micron MT9HTF3272AY-80E 400-MHz DDR2 SDRAM DIMM

### Related Information

- [Read Capture](#) on page 512
- [Stratix III Device Handbook](#)

#### 9.6.1.4.2. Arria II, Arria V, Cyclone V, Stratix IV and Stratix V

Read capture timing analysis indicates the amount of slack on the DDR DQ signals that are latched by the FPGA using the DQS strobe output of the memory device.

### Read Capture

The read capture timing paths are analyzed by a combination of the TimeQuest Timing Analyzer using the set\_input\_delay (max and min), set\_max\_delay, and set\_min\_delay constraints, and further steps to account for calibration that occurs at runtime. The UniPHY IP include timing constraints in the



<phy\_variation\_name>.sdc file, and further slack analysis in <phy\_variation\_name>\_report\_timing.tcl and <phy\_variation\_name>\_report\_timing\_core.tcl files.

In Arria II and Stratix IV devices, the margin is reported based on a combination of the TimeQuest Timing Analyzer calculation results and further processing steps that account for the calibration that occurs at runtime. First, the TimeQuest analyzer returns the base setup and hold slacks, and further processing steps adjust the slacks to account for effects which the TimeQuest analyzer cannot model.

### Write

Write timing analysis indicates the amount of slack on the DDR DQ signals that are latched by the memory device using the DQS strobe output from the FPGA device. The write timing paths are analyzed by a combination of the TimeQuest Timing Analyzer using the set\_output\_delay (max and min) and further steps to account for calibration that occurs at runtime. The UniPHY IP includes timing constraints in the <phy\_variation\_name>.sdc (UniPHY) file, and further slack analysis in the <phy\_variation\_name>\_report\_timing.tcl and <phy\_variation\_name>\_report\_timing\_core.tcl files.

#### 9.6.1.5. Read Resynchronization

In a UniPHY interface, a FIFO buffer synchronizes the data transfer from the data capture to the core.

The calibration process sets the depth of the FIFO buffer and no dedicated synchronization clock is required. Refer to <phy\_variation\_name>\_report\_timing\_core.tcl for more information about the resynchronization timing margin equation.

#### 9.6.1.6. DQS versus CK—Arria II GX and Cyclone IV Devices

The DQS versus CK timing path indicates the skew requirement for the arrival time of the DQS strobe at the memory with respect to the arrival time of CK/CK# at the memory. Arria II GX and Cyclone IV devices require the DQS strobes and CK clocks to arrive edge aligned.

There are two timing constraints for DQS versus CK timing path to account for duty cycle distortion. The DQS/DQS# rising edge to CK/CK# rising edge ( $t_{DQSS}$ ) requires the rising edge of DQS to align with the rising edge of CK to within 25% of a clock cycle, while the DQS/DQS# falling edge setup/hold time from CK/CK# rising edge ( $t_{DSS}/t_{DSH}$ ) requires the falling edge of DQS to be more than 20% of a clock cycle away from the rising edge of CK.

The TimeQuest Timing Analyzer analyzes the DQS vs CK timing paths using the set\_output\_delay (max and min) constraints. For more information, refer to <phy\_variation\_name>\_phy\_ddr\_timing.sdc.

#### 9.6.1.7. Write Leveling tDQSS

In DDR2 SDRAM and DDR3 SDRAM interfaces, write leveling  $t_{DQSS}$  timing is a calibrated path that details skew margin for the arrival time of the DQS strobe with respect to the arrival time of CK/CK# at the memory side.



For proper write leveling configuration, DLL delay chain must be equal to 8. The PHY IP reports the margin through an equation. For more information, refer to `<phy_variation_name> _report_timing_core.tcl`.

#### 9.6.1.8. Write Leveling tDSH/tDSS

In DDR2 SDRAM and DDR3 SDRAM interfaces, write leveling  $t_{DSH}/t_{DSS}$  timing details the setup and hold margin for the DQS falling edge with respect to the CK clock at the memory.

The PHY IP reports the margin through an equation. For more information, refer to `<phy_variation_name> _report_timing_core.tcl`.

#### 9.6.1.9. DK versus CK (RLDRAM II with UniPHY)

In RDRAM II with UniPHY designs using the Nios-based sequencer, DK versus CK timing is a calibrated path that details skew margin for the arrival time of the DK clock versus the arrival time of CK/CK# on the memory side.

The PHY IP reports the margin through an equation. For more information, refer to `<phy_variation_name> _report_timing_core.tcl`.

#### 9.6.1.10. Bus Turnaround Time

In DDR2 and DDR3 SDRAM, and RDRAM II (CIO) with UniPHY designs that use bidirectional data bus, you may have potential encounter with data bus contention failure when a write command follows a read command. The bus-turnaround time analysis determines how much margin there is on the switchover time and prevents bus contention.

If the timing is violated, you can either increase the controller's bus turnaround time, which may reduce efficiency or board traces delay. Refer to `<variation> _report_timing_core.tcl` for the equation. You can find this analysis in the timing report. This analysis is only available for DDR2/3 SDRAM and RDRAM II UniPHY IPs in Arria II GZ, Arria V, Cyclone V, Stratix IV, and Stratix V devices.

To determine whether the bus turnaround time issue is the cause of your design failure and to overcome this timing violation, follow these steps:

1. When the design fails, change the default values of `MEM_IF_WR_TO_RD_TURNAROUND_OCT` and `MEM_IF_RD_TO_WR_TURNAROUND_OCT` parameters in the controller wrapper file to a maximum value of 5. If the design passes after the change, it is a bus turnaround issue.
2. To solve the bus turnaround time issue, reduce the values of the `MEM_IF_WR_TO_RD_TURNAROUND_OCT` and `MEM_IF_RD_TO_WR_TURNAROUND_OCT` parameters gradually until you reach the minimum value needed for the design to pass on board.

#### 9.6.2. Timing Analysis Description for Arria 10 EMIF IP

Timing analysis of Arria 10 external memory interface IP is somewhat simpler than that of UniPHY-based IP, because Arria 10 devices have more hardened blocks and there are fewer soft logic registers to be analyzed, because most are user logic registers.



Your Arria 10 EMIF IP includes a Synopsys Design Constraints File (.sdc) which contains timing constraints specific to your IP. The .sdc file also contains Tool Command Language (.tcl) scripts which perform various timing analyses specific to memory interfaces.

Two timing analysis flows are available for Arria 10 EMIF IP:

- Early I/O Timing Analysis, which is a precompilation flow.
- Full Timing Analysis, which is a post-compilation flow.

### 9.6.2.1. PHY or Core

Timing analysis of the PHY or core path includes the path from the last set of registers in the core to the first set of registers in the periphery (C2P), path from the last set of registers in the periphery to the first of registers in the core (P2C) and ECC related path if it is enabled.

As with 28 nm devices, core timing analysis excludes user logic timing to or from EMIF blocks. The EMIF IP provides a constrained clock (for example: ddr3\_usr\_clk) with which to clock customer logic. In 28 nm devices, pll\_afi\_clk serves this purpose

The PHY or core analyzes this path by calling the report\_timing command in <variation\_name>\_report\_timing.tcl and <variation\_name>\_report\_timing\_core.tcl.

**Note:** In version 14.1 and later, the *Spatial Pessimism Removal* slack values in the **Core to Periphery** and **Periphery to Core** tables are always equal to zero. This occurs because pessimism removal is integrated into the base timing analysis.

### 9.6.2.2. I/O Timing

I/O timing analysis includes analysis of read capture, write, address and command, DQS gating, and write leveling.

The TimeQuest Timing Analyzer provides a breakdown of the timing budgets which details margin loss due to transmitter, receiver, and channel. TimeQuest displays the total margin in the last row of the timing report.

The I/O timing analysis described in the following topics is based on a 2 speed-grade device, interfacing with a DDR3 SDRAM UDIMM at 1066 MHz. A 1066 MHz DDR3 SDRAM UDIMM is used for the analysis.

#### 9.6.2.2.1. Read Capture

Read capture timing analysis indicates the amount of slack on the DQ signals that are latched by the FPGA using the DQS strobe output of the memory device.

The TimeQuest Timing Analyzer analyzes read capture timing paths through conventional static timing analysis and further processing steps that account for memory calibration (which may include pessimism removal) and calibration uncertainties as shown in the following figure.



**Figure 76. Read Capture Timing Analysis**

ed_synth_ddr3w Read Capture		
	Operation	Margin
1	Ideal Timing Window	0.469
2	ISI	0.063
3	SSI	0.023
4	Slew Rate Derating	0.050
5	tDQSQ effect	0.075
6	tQH effect	0.112
7	Memory Calibration	-0.075
8	Jitter Effects	0.072
9	Duty Cycle Distortion	0.031
10	Setup/Hold Time	0.016
11	EOL	0.025
12	Calibration Uncertainty	0.036
13	Skew Effect	0.000
14	Final Read Margin	0.040

#### 9.6.2.2.2. Write

Write timing analysis indicates the amount of slack on the DQ signals that are latched by the memory device using the DQS strobe output from the FPGA device.

As with read capture, the TimeQuest Timing Analyzer analyzes write timing paths through conventional static timing analysis and further processing steps that account for memory calibration (which may include pessimism removal) and calibration uncertainties as shown in the following figure.

**Figure 77. Write Timing Analysis**

ed_synth_ddr3w Write		
	Operation	Margin
1	Ideal Timing Window	0.469
2	ISI	0.063
3	SSO	0.047
4	Slew Rate Derating	0.118
5	tDS effect	0.053
6	tDH effect	0.055
7	Memory Calibration	-0.043
8	Jitter Effects	0.039
9	Duty Cycle Distortion	0.016
10	EOL	0.013
11	Calibration Uncertainty	0.038
12	Skew Effect	0.000
13	Final Write Margin	0.071

#### 9.6.2.2.3. Address and Command

Address and command signals are single data rate signals latched by the memory device using the FPGA output clock; some are half-rate data signals, while others, such as the chip select, are full-rate signals.

The TimeQuest Timing Analyzer analyzes the address and command timing paths through conventional static timing analysis and further processing steps that account for memory pessimism removal (as shown in the following figure). Depending on the

memory protocol in use, if address command calibration is performed, calibration uncertainty is subtracted from the timing window while PVT variation and skew effects are not subtracted, and vice versa

**Figure 78. Address and Command Timing Analysis**

ed_synth_ddr3w Address/Command		
	Operation	Margin
1	Ideal Timing Window	0.937
2	ISI	0.094
3	SSO	0.023
4	Slew Rate Derating	0.118
5	tIS effect	0.060
6	tIH effect	0.095
7	Memory Calibration	-0.046
8	Jitter Effects	0.083
9	Duty Cycle Distortion	0.016
10	EOL	0.013
11	Calibration Uncertainty	0.041
12	PVT variation	0.027
13	Skew Effect	0.000
14	Final CA Margin	0.442

#### 9.6.2.2.4. DQS Gating / Postamble

Postamble timing is a setup period during which the DQS signal goes low after all the DQ data has been received from the memory device during a read operation. After postamble time, the DQS signal returns from a low-impedance to a high-impedance state to disable DQS and disallow any glitches from writing false data over valid data.

The TimeQuest Timing Analyzer analyzes the postamble timing path in DDRx memory protocols only through an equation which considers memory calibration, calibration uncertainty, and tracking uncertainties as shown in the following figure.

**Figure 79. DQS Gating Timing Analysis**

ed_synth_ddr3w DQS Gating		
	Operation	Margin
1	Ideal Timing Window	0.750
2	ISI	0.094
3	SSI	0.012
4	Slew Rate Derating	0.050
5	tDQSCK	0.360
6	Memory Calibration	-0.144
7	Jitter Effects	0.167
8	Duty Cycle Distortion	0.002
9	EOL	0.002
10	Calibration Uncertainty	0.020
11	Tracking Uncertainty	0.078
12	Setup/Hold Time	0.016
13	Skew Effect	0.000
14	Final DQS Gating Margin	0.093

#### 9.6.2.2.5. Write Leveling

In DDR3 SDRAM and DDR4 SDRAM interfaces, write leveling details the margin for the DQS strobe with respect to CK/CK# at the memory side.



The TimeQuest Timing Analyzer analyzes the write leveling timing path through an equation which considers memory calibration, calibration uncertainty and PVT variation as shown in the following figure.

**Figure 80. Write Leveling Timing Analysis**

ed_synth_ddr3w Write Levelling		
	Operation	Margin
1	Ideal Timing Window	0.937
2	ISI	0.031
3	SSO	0.035
4	tDQSS/tDSS/tDSH Effect	0.431
5	Memory Calibration	-0.172
6	tWLS/tWLH effect	0.000
7	Jitter Effects	0.165
8	Duty Cycle Distortion	0.016
9	EOL	0.000
10	Calibration Uncertainty	0.030
11	PVT variation	0.862
12	Skew Effect	0.000
13	Final Write Levelling Margin	0.401

### 9.6.3. Timing Analysis Description for Stratix 10 EMIF IP

Your Stratix 10 EMIF IP includes a Synopsys Design Constraints File (.sdc) which contains timing constraints specific to your IP.

The .sdc file also contains Tool Command Language (.tcl) scripts which perform various timing analyses specific to memory interfaces.

Two timing analysis flows are available for Stratix 10 EMIF IP:

- Early I/O Timing Analysis, which is a precompilation flow.
- Full Timing Analysis, which is a post-compilation flow.

#### 9.6.3.1. PHY or Core

Timing analysis of the PHY or core path includes the path from the last set of registers in the core to the first set of registers in the periphery (C2P), path from the last set of registers in the periphery to the first of registers in the core (P2C) and ECC related path if it is enabled.

As with 28 nm devices, core timing analysis excludes user logic timing to or from EMIF blocks. The EMIF IP provides a constrained clock (for example: ddr3\_usr\_clk) with which to clock customer logic. In 28 nm devices, pll\_afi\_clk serves this purpose

The PHY or core analyzes this path by calling the report\_timing command in <variation\_name>\_report\_timing.tcl and <variation\_name>\_report\_timing\_core.tcl.

**Note:** In version 14.1 and later, the *Spatial Pessimism Removal* slack values in the **Core to Periphery** and **Periphery to Core** tables are always equal to zero. This occurs because pessimism removal is integrated into the base timing analysis.

### 9.6.3.2. I/O Timing

I/O timing analysis includes analysis of read capture, write, address and command, DQS gating, and write leveling.

The TimeQuest Timing Analyzer provides a breakdown of the timing budgets which details margin loss due to transmitter, receiver, and channel. TimeQuest displays the total margin in the last row of the timing report.

The I/O timing analysis described in the following topics is based on a 2 speed-grade device, interfacing with a DDR3 SDRAM UDIMM at 1066 MHz. A 1066 MHz DDR3 SDRAM UDIMM is used for the analysis.

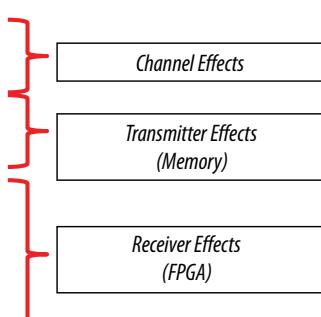
#### 9.6.3.2.1. Read Capture

Read capture timing analysis indicates the amount of slack on the DQ signals that are latched by the FPGA using the DQS strobe output of the memory device.

The TimeQuest Timing Analyzer analyzes read capture timing paths through conventional static timing analysis and further processing steps that account for memory calibration (which may include pessimism removal) and calibration uncertainties as shown in the following figure.

**Figure 81. Read Capture Timing Analysis**

ed_synth_ddr3w Read Capture		
	Operation	Margin
1	Ideal Timing Window	0.469
2	ISI	0.063
3	SSI	0.023
4	Slew Rate Derating	0.050
5	tDQSQ effect	0.075
6	tQH effect	0.112
7	Memory Calibration	-0.075
8	Jitter Effects	0.072
9	Duty Cycle Distortion	0.031
10	Setup/Hold Time	0.016
11	EOL	0.025
12	Calibration Uncertainty	0.036
13	Skew Effect	0.000
14	Final Read Margin	0.040



The diagram illustrates the breakdown of the read capture timing margins. A vertical red bracket on the right side of the table groups the rows into three categories: 'Channel Effects' (rows 1-4), 'Transmitter Effects (Memory)' (rows 5-11), and 'Receiver Effects (FPGA)' (rows 12-14). The 'Final Read Margin' is shown as the sum of these effects.

#### 9.6.3.2.2. Write

Write timing analysis indicates the amount of slack on the DQ signals that are latched by the memory device using the DQS strobe output from the FPGA device.

As with read capture, the TimeQuest Timing Analyzer analyzes write timing paths through conventional static timing analysis and further processing steps that account for memory calibration (which may include pessimism removal) and calibration uncertainties as shown in the following figure.



**Figure 82. Write Timing Analysis**

ed_synth_ddr3w Write		
	Operation	Margin
1	Ideal Timing Window	0.469
2	ISI	0.063
3	SSO	0.047
4	Slew Rate Derating	0.118
5	tDS effect	0.053
6	tDH effect	0.055
7	Memory Calibration	-0.043
8	Jitter Effects	0.039
9	Duty Cycle Distortion	0.016
10	EOL	0.013
11	Calibration Uncertainty	0.038
12	Skew Effect	0.000
13	Final Write Margin	0.071

#### 9.6.3.2.3. Address and Command

Address and command signals are single data rate signals latched by the memory device using the FPGA output clock; some are half-rate data signals, while others, such as the chip select, are full-rate signals.

The TimeQuest Timing Analyzer analyzes the address and command timing paths through conventional static timing analysis and further processing steps that account for memory pessimism removal (as shown in the following figure). Depending on the memory protocol in use, if address command calibration is performed, calibration uncertainty is subtracted from the timing window while PVT variation and skew effects are not subtracted, and vice versa

**Figure 83. Address and Command Timing Analysis**

ed_synth_ddr3w Address/Command		
	Operation	Margin
1	Ideal Timing Window	0.937
2	ISI	0.094
3	SSO	0.023
4	Slew Rate Derating	0.118
5	tIS effect	0.060
6	tIH effect	0.095
7	Memory Calibration	-0.046
8	Jitter Effects	0.083
9	Duty Cycle Distortion	0.016
10	EOL	0.013
11	Calibration Uncertainty	0.041
12	PVT variation	0.027
13	Skew Effect	0.000
14	Final CA Margin	0.442

#### 9.6.3.2.4. DQS Gating / Postamble

Postamble timing is a setup period during which the DQS signal goes low after all the DQ data has been received from the memory device during a read operation. After postamble time, the DQS signal returns from a low-impedance to a high-impedance state to disable DQS and disallow any glitches from writing false data over valid data.

The TimeQuest Timing Analyzer analyzes the postamble timing path in DDRx memory protocols only through an equation which considers memory calibration, calibration uncertainty, and tracking uncertainties as shown in the following figure.

**Figure 84. DQS Gating Timing Analysis**

ed_synth_ddr3w DQS Gating		
	Operation	Margin
1	Ideal Timing Window	0.750
2	ISI	0.094
3	SSI	0.012
4	Slew Rate Derating	0.050
5	tDQSK	0.360
6	Memory Calibration	-0.144
7	Jitter Effects	0.167
8	Duty Cycle Distortion	0.002
9	EOL	0.002
10	Calibration Uncertainty	0.020
11	Tracking Uncertainty	0.078
12	Setup/Hold Time	0.016
13	Skew Effect	0.000
14	Final DQS Gating Margin	0.093

#### 9.6.3.2.5. Write Leveling

In DDR3 SDRAM and DDR4 SDRAM interfaces, write leveling details the margin for the DQS strobe with respect to CK/CK# at the memory side.

The TimeQuest Timing Analyzer analyzes the write leveling timing path through an equation which considers memory calibration, calibration uncertainty and PVT variation as shown in the following figure.

**Figure 85. Write Leveling Timing Analysis**

ed_synth_ddr3w Write Levelling		
	Operation	Margin
1	Ideal Timing Window	0.937
2	ISI	0.031
3	SSO	0.035
4	tDQSS/tDSS/tDSH Effect	0.431
5	Memory Calibration	-0.172
6	tWLS/tWLH effect	0.000
7	Jitter Effects	0.165
8	Duty Cycle Distortion	0.016
9	EOL	0.000
10	Calibration Uncertainty	0.030
11	PVT variation	0.862
12	Skew Effect	0.000
13	Final Write Levelling Margin	0.401

## 9.7. Timing Report DDR

The **Report DDR** task in the TimeQuest Timing Analyzer generates custom timing margin reports for all EMIF IP instances in your design. The TimeQuest Timing Analyzer generates this custom report by sourcing the wizard-generated `<variation_name>_report_timing.tcl` script.



This `<variation_name>_report_timing.tcl` script reports the following timing slacks on specific paths of the DDR SDRAM:

- Read capture
- Read resynchronization
- Mimic, address and command
- Core
- Core reset and removal
- Half-rate address and command
- DQS versus CK
- Write
- Write leveling ( $t_{DQSS}$ )
- Write leveling ( $t_{DSS}/t_{DSH}$ )
- DQS Gating (Postamble)

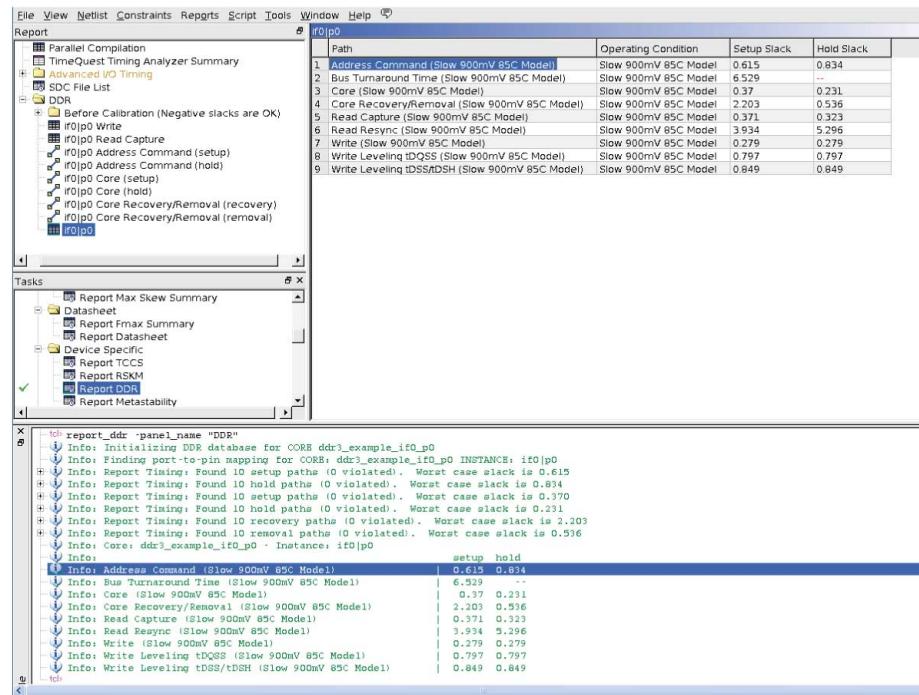
In Stratix III designs, the `<variation_name>_report_timing.tcl` script checks the design rules and assumptions as listed in "Timing Model Assumptions and Design Rules". If you do not adhere to these assumptions and rules, you receive critical warnings when the TimeQuest Timing Analyzer runs during compilation or when you run the **Report DDR** task.

To generate a timing margin report, follow these steps:

1. Compile your design in the Quartus Prime software.
  2. Launch the TimeQuest Timing Analyzer.
  3. Double-click **Report DDR** from the **Tasks** pane. This action automatically executes the **Create Timing Netlist**, **Read SDC File**, and **Update Timing Netlist** tasks for your project.
- The **.sdc** may not be applied correctly if the variation top-level file is the top-level file of the project. You must have the top-level file of the project instantiate the variation top-level file.

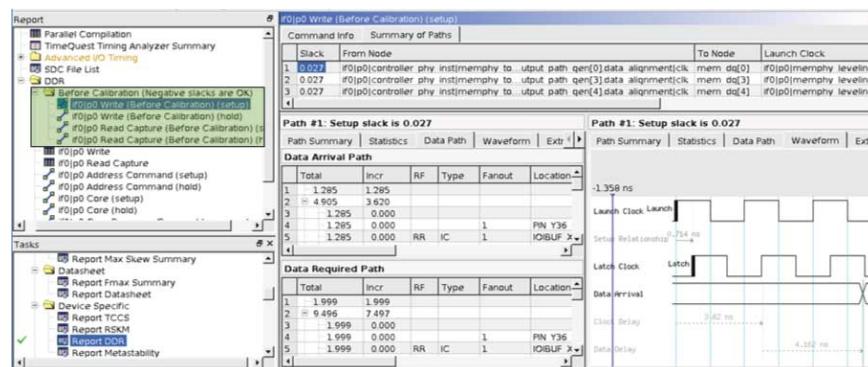
The **Report DDR** feature creates a new DDR folder in the TimeQuest Timing Analyzer **Report** pane.

Expanding the DDR folder reveals the detailed timing information for each PHY timing path, in addition to an overall timing margin summary for the UniPHY instance, as shown in the following figure.

**Figure 86.** Timing Margin Summary Window Generated by Report DDR Task

**Note:**

Bus turnaround time shown in the above figure is available in all UniPHY IPs and devices except in QDR II and QDR II+ SRAM memory protocols and Stratix III devices.

The following figure shows the timing analysis results calculated using FPGA timing model before adjustment in the **Before Calibration** panel.

**Figure 87.** Read and Write Before Calibration


The following two figures show the read capture and write margin summary window generated by the Report DDR Task for a DDR3 core. It first shows the timing results calculated using the FPGA timing model. The `<variation_name>_report_timing_core.tcl` then adjusts these numbers to account



for effects that are not modeled by either the timing model or by TimeQuest Timing Analyzer. The read and write timing margin analysis for Stratix III devices does not need any adjustments.

**Figure 88. Read Capture Margin Summary Window**

if0 p0 Read Capture			
	Operation	Setup Slack	Hold Slack
1	After Calibration Read Capture	0.371	0.323
2	Before Calibration Read Capture	0.280	0.273
3	Memory Calibration	0.078	0.150
4	Deskew Read	0.157	0.044
5	Quantization error	-0.050	-0.050
6	Calibration uncertainty	-0.093	-0.093

**Figure 89. Write Capture Margin Summary Window**

if0 p0 Write			
	Operation	Setup Slack	Hold Slack
1	After Calibration Write	0.279	0.279
2	Before Calibration Write	0.027	0.026
3	Memory Calibration	0.135	0.113
4	Deskew Write and/or more clock pessimism removal	0.216	0.240
5	Quantization error	-0.050	-0.050
6	Calibration uncertainty	-0.050	-0.050

#### Related Information

[Timing Model Assumptions and Design Rules](#) on page 529

## 9.8. Report SDC

The **Report SDC** task in the TimeQuest Timing Analyzer generates the SDC assignment reports for your design. The TimeQuest Timing Analyzer generates this constraint report by sourcing the **.sdc**. The SDC assignment reports show the constraint applied in the design.

For example, the reports may include the following constraints:

- Create Clock
- Create Generated Clock
- Set Clock Uncertainty
- Set Input Delay
- Set Output Delay
- Set False Path
- Set Multicycle Path
- Set Maximum Delay
- Set Minimum Delay

The following figure shows the SDC assignments generated by the **Report SDC** task for a DDR3 SDRAM core design. The timing analyzer uses these constraint numbers in analysis to calculate the timing margin. Refer to the **.sdc** files of each constraints number.

**Figure 90. SDC Assignments Report Window**

The screenshot shows the TimeQuest Timing Analyzer interface. On the left, there's a tree view of reports under 'Report' and 'Tasks'. Under 'Report', 'SDC Assignments' is expanded, showing various SDC command nodes like 'Create Clock', 'Set Input Delay', etc. Under 'Tasks', 'Report SDC' is checked. On the right, there's a table titled 'Set Input Delay' with columns: SDC Command, Add Delay, Flags, Clock Name, Delay, Ports, and Source. The table lists 16 rows of SDC commands with their corresponding parameters and results.

SDC Command	Add Delay	Flags	Clock Name	Delay	Ports	Source
1 set input delay	-add delay	-max	[get clocks memm dqs IN]	0.201	[get ports memm dq[0]]	
2 set input delay	-add delay	-min	[get clocks memm dqs IN]	-0.420	[get ports memm dq[0]]	
3 set input delay	-add delay	-max	[get clocks memm dqs IN]	0.201	[get ports memm dq[1]]	
4 set input delay	-add delay	-min	[get clocks memm dqs IN]	-0.420	[get ports memm dq[1]]	
5 set input delay	-add delay	-max	[get clocks memm dqs IN]	0.201	[get ports memm dq[2]]	
6 set input delay	-add delay	-min	[get clocks memm dqs IN]	-0.420	[get ports memm dq[2]]	
7 set input delay	-add delay	-max	[get clocks memm dqs IN]	0.201	[get ports memm dq[3]]	
8 set input delay	-add delay	-min	[get clocks memm dqs IN]	-0.420	[get ports memm dq[3]]	
9 set input delay	-add delay	-max	[get clocks memm dqs IN]	0.201	[get ports memm dq[4]]	
10 set input delay	-add delay	-min	[get clocks memm dqs IN]	-0.420	[get ports memm dq[4]]	
11 set input delay	-add delay	-max	[get clocks memm dqs IN]	0.201	[get ports memm dq[5]]	
12 set input delay	-add delay	-min	[get clocks memm dqs IN]	-0.420	[get ports memm dq[5]]	
13 set input delay	-add delay	-max	[get clocks memm dqs IN]	0.201	[get ports memm dq[6]]	
14 set input delay	-add delay	-min	[get clocks memm dqs IN]	-0.420	[get ports memm dq[6]]	
15 set input delay	-add delay	-max	[get clocks memm dqs IN]	0.201	[get ports memm dq[7]]	
16 set input delay	-add delay	-min	[get clocks memm dqs IN]	-0.420	[get ports memm dq[7]]	

## 9.9. Calibration Effect in Timing Analysis

Timing analysis for Arria II, Arria V, Arria V GZ, Cyclone IV, Cyclone V, Stratix IV, and Stratix V devices take into account the calibration effects to improve the timing margin.

The following topics discuss ways to include the calibration effects in timing analysis.

### 9.9.1. Calibration Emulation for Calibrated Path

In conventional static timing analysis, calibration paths do not include calibration effects. To account for the calibration effects, the timing analyzer emulates the calibration process and integrates it into the timing analysis.

Normally the calibration process involves adding or subtracting delays to a path. The analyzer uses the delay obtained through static timing analysis in the emulation algorithm to estimate the extra delay added during calibration. With these estimated delays, the timing analysis emulates hardware calibration and obtains a better estimate timing margin.

**Note:** Refer to `<phy_variation_name>_report_timing.tcl` and `<phy_variation_name>_report_timing_core.tcl` for the files that determine the timing margin after calibration.

### 9.9.2. Calibration Error or Quantization Error

Hardware devices use calibration algorithms when delay information is unknown or incomplete. If the delay information is unknown, the timing analysis of the calibrated paths has to work with incomplete data. This unknown information may cause the timing analysis calibration operations to pick topologies that are different than what would actually occur in hardware.

The differences between what can occur in hardware and what occurs in the timing analysis are quantified and included in the timing analysis of the calibrated paths as quantization error or calibration error.



### 9.9.3. Calibration Uncertainties

Calibration results may change or reduce due to one or more of the following uncertainties:

- Jitter and duty cycle distortion (DCD) effects
- Voltage and temperature variations
- Board trace delays changing due to noise on terminated supply voltages

These calibration uncertainties are accounted for in the timing analysis.

### 9.9.4. Memory Calibration

All the timing paths reported include one or more memory parameters, such as  $t_{DQSS}$  and  $t_{DQSQ}$ . These specifications indicate the amount of variation that occurs in various timing paths in the memory and abstracts them into singular values so that they can be used by others when interfacing with the memory device.

JEDEC defines these parameters in their specification for memory standards, and every memory vendor must meet this specification or improve it. However, there is no proportion of each specification due to different types of variations. Variations that are of interest are typically grouped into three different types: process variations (P), voltage variations (V), and temperature variations (T). These together compose PVT variations that typically define the JEDEC specification. You can determine the maximum P variation by comparing different dies, and you can determine the maximum V and T variations by operating a design at the endpoints of the range of voltage and temperature. P variations do not change once the chip has been fabricated, while V and T variations change over time.

The timing analysis for Stratix V FPGAs at 667 MHz of various paths (if the analysis is comprehensive and includes all the sources of noise) indicate that there is no timing margin available. However, the designs do actually work in practice with a reasonable amount of margin. The reason for this behavior is that the memory devices typically have specifications that easily beat the JEDEC specification and that our calibration algorithms calibrate out the process portion of the JEDEC specification, leaving only the V and T portions of the variations.

The memory calibration figure determination includes noting what percentage of the JEDEC specification of various memory parameters is caused by process variations for which UniPHY calibration algorithms can calibrate out, and to apply that to the full JEDEC specification. The remaining portion of the variation is caused by voltage and temperature variations which cannot be calibrated out.

You can find the percentage of the JEDEC specification that is due to process variation is set in `<variation_name>_report_timing.tcl`.

## 9.10. Timing Model Assumptions and Design Rules

External memory interfaces using Intel FPGA IP are optimized for highest performance, and use a high-performance timing model to analyze calibrated and source-synchronous, double-data rate I/O timing paths. This timing model applies to designs that adhere to a set of predefined assumptions.

These timing model assumptions include memory interface pin-placement requirements, PLL and clock network usage, I/O assignments (including I/O standard, termination, and slew rate), and many others.

For example, the read and write datapath timing analysis is based on the FPGA pin-level  $t_{TCCS}$  and  $t_{SW}$  specifications, respectively. While calculating the read and write timing margins, the Quartus Prime software analyzes the design to ensure that all read and write timing model assumptions are valid for your variation instance.

*Note:* Timing model assumptions only apply to Stratix III devices.

When the Report DDR task or **report\_timing.tcl** script is executed, the timing analysis assumptions checker is invoked with specific variation configuration information. If a particular design rule is not met, the Quartus Prime software reports the failing assumption as a Critical Warning message.

The following figure shows a sample set of messages generated when the memory interface DQ, DQS, and CK/CK# pins are not placed in the same edge of the device.

**Figure 91. Read and Write Timing Analysis Assumption Verification**

```

477 tcl> report_ddr -panel_name "DDR1"
478
479 [  ] Critical Warning: Pin mem_clk[0], mem_dq[0], mem_dq[10], mem_dq[11], mem_dq[12], mem_dq[13], mem_dq[14], mem_dq[15], mem_dq[16], mem_dq[17], mem_dq[18] was placed on the left edge of the device
480 [  ] Critical Warning: mem_dq[0] was placed on the bottom edge of the device
481 [  ] Critical Warning: mem_dq[0] was placed on the bottom edge of the device
482 [  ] Critical Warning: mem_dq[11] was placed on the bottom edge of the device
483 [  ] Critical Warning: mem_dq[12] was placed on the bottom edge of the device
484 [  ] Critical Warning: mem_dq[13] was placed on the bottom edge of the device
485 [  ] Critical Warning: mem_dq[14] was placed on the bottom edge of the device
486 [  ] Critical Warning: mem_dq[15] was placed on the bottom edge of the device
487 [  ] Critical Warning: mem_dq[16] was placed on the bottom edge of the device
488 [  ] Critical Warning: mem_dq[17] was placed on the bottom edge of the device
489 [  ] Critical Warning: mem_dq[18] was placed on the bottom edge of the device
551 [  ] Critical Warning: mem_dq[9] was placed on the bottom edge of the device
552 [  ] Critical Warning: mem_dqs[0] was placed on the bottom edge of the device
553 [  ] Critical Warning: mem_dqs[1] was placed on the bottom edge of the device
554 [  ] Critical Warning: mem_dqs[2] was placed on the bottom edge of the device
555 [  ] Critical Warning: mem_dqs[3] was placed on the bottom edge of the device
556 [  ] Critical Warning: mem_dqs[4] was placed on the bottom edge of the device
557 [  ] Critical Warning: mem_dqs[5] was placed on the bottom edge of the device
558 [  ] Critical Warning: mem_dqs[6] was placed on the bottom edge of the device
559 [  ] Critical Warning: mem_dqs[7] was placed on the bottom edge of the device
560 [  ] Critical Warning: mem_dqs[8] was placed on the bottom edge of the device
561 [  ] Critical Warning: Memory clock pin mem_clk[0], mem_clk[1], mem_clk[2] must be placed on the same edge of the device
562 [  ] Critical Warning: mem_clk[0] was placed on the left edge of the device
563 [  ] Critical Warning: mem_clk[1] was placed on the top edge of the device
564 [  ] Critical Warning: mem_clk[2] was placed on the bottom edge of the device
565 [  ] Critical Warning: Read Capture and Write Timing analyses may not be valid due to violated timing model assumptions
566 [  ] Info: Report Timing: Found 24 setup paths (0 violated). Worst case slack is 1.155
  
```

### 9.10.1. Memory Clock Output Assumptions

To verify the quality of the FPGA clock output to the memory device (CK/CK# or K/K#), which affects FPGA performance and quality of the read clock/strobe used to read data from the memory device, the following assumptions are necessary:

- The slew rate setting must be **Fast** or an on-chip termination (OCT) setting must be used.
- The output delay chains must all be **0** (the default value applied by the Quartus Prime software). These delay chains include the Stratix III D5 and D6 output delay chains.
- The output open-drain parameter on the memory clock pin **IO\_OBUF** atom must be **Off**. The **Output Open Drain** logic option must not be enabled.

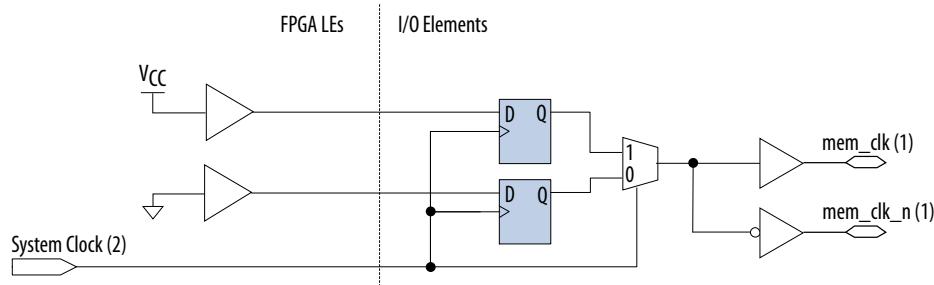
- The weak pull-up on the CK and CK# pads must be **Off**. The **Weak Pull Up Resistor** logic option must not be enabled.
- The bus hold on the CK and CK# pads must be **Off**. The **Enable Bus Hold Circuitry** logic option must not be enabled.
- All CK and CK# pins must be declared as output-only pins or bidirectional pins with the output enable set to V<sub>CC</sub>.

### 9.10.1.1. Memory Clock Assumptions for Stratix III Devices

For Stratix III devices the following additional memory clock assumptions are necessary:

- All memory clock output pins must be placed on DIFFOUT pin pairs on the same edge of the device.
- For DDR3 SDRAM interfaces:
- The CK pins must be placed on FPGA output pins marked DQ, DQS, or DQSn.
- The CK pin must be fed by an OUTPUT\_PHASE\_ALIGNMENT WYSIWYG with a 0° phase shift.
- The PLL clock driving CK pins must be the same as the clock driving the DQS pins.
- The T4 (DDIO\_MUX) delay chains setting for the memory clock pins must be the same as the settings for the DQS pins.
- For non-DDR3 interfaces, the T4 (DDIO\_MUX) delay chains setting for the memory clock pins must be greater than 0.
- The programmable rise and fall delay chain settings for all memory clock pins must be set to 0.
- The memory output clock signals must be generated with the DDIO configuration shown in the following figure, with a signal splitter to generate the n- pin pair and a regional clock network-to-clock to output DDIO block.

**Figure 92. DDIO Configuration with Signal Splitter**



### 9.10.2. Write Data Assumptions

To verify the memory interface using the FPGA TCCS output timing specifications, the following assumptions are necessary:



- For QDRII and QDRII+ memory interfaces, the write clock output pins (such as K/K# or DK/DK#) must be placed in DQS/DQSn pin pairs.
- The PLL clock used to generate the write-clock signals and the PLL clock used to generate the write-data signals must come from the same PLL.
- The slew rate for all write clocks and write data pins must be set to **Fast** or OCT must be used.
- When auto deskew is not enabled, the output delay chains and output enable delay chains must all be set to the default values applied by the Quartus Prime software. These delay chains include the Stratix III D5 and D6 delay chains.
- The output open drain for all write clocks and write data pins' IO\_OBUF atom must be set to **Off**. The **Output Open Drain** logic option must not be enabled.
- The weak pull-up for all write clocks and write data pins must be set to **Off**. The **Weak Pull Up Resistor** logic option must not be enabled.
- The Bus Hold for all write clocks and write data pins must be set to **Off**. The **Enable Bus Hold Circuitry** logic option must not be enabled.

#### 9.10.2.1. Write Data Assumptions for Stratix III Devices

For Stratix III devices the following additional write data assumptions are necessary:

- Differential write clock signals (DQS/DQSn) must be generated using the signal splitter.
- The write data pins (including the DM pins) must be placed in related DQ pins associated with the chosen DQS pin. The only exception to this rule is for QDRII and QDRII+ ×36 interfaces emulated using two ×18 DQ groups. For such interfaces, all of the write data pins must be placed on the same edge of the device (left, right, top, or bottom). Also, the write clock K/K# pin pair should be placed on one of the DQS/DQSn pin pairs on the same edge.
- All write clock pins must have similar circuit structure.
  - For DDR2 SDRAM interfaces and DDR3 SDRAM with leveling interfaces, all DQS/DQS# write strobes must be fed by DDIO output registers clocked by the write-leveling delay chain in the OUTPUT\_PHASE\_ALIGNMENT block.
  - For DDR and DDR2 SDRAM interfaces, all write clock pins must be fed by DDIO output registers clocked by a global or regional clock network.
- All write data pins must have similar circuit structure.
  - For DDR3 SDRAM interfaces, all write data pins must be fed by either DDIO output registers clocked by the OUTPUT\_PHASE\_ALIGNMENT block, V<sub>CC</sub>, or GND.
  - For DDR and DDR2 SDRAM interfaces, all write data pins must be fed by either DDIO output registers clocked by a global or regional clock network, V<sub>CC</sub>, or GND.



- The write clock output must be 72°, 90°, or 108° more than the write data output.
  - For DDR2 SDRAM and DDR3 SDRAM with leveling interfaces, the write-leveling delay chain in the OUTPUT\_PHASE\_ALIGNMENT block must implement a phase shift of 72°, 90°, or 108° to center-align write clock with write data.
  - For DDR and DDR2 SDRAM interfaces, the phase shift of the PLL clock used to clock the write clocks must be 72 to 108° more than the PLL clock used to clock the write data clocks to generated center-aligned clock and data.
- The T4 (DDIO\_MUX) delay chains must all be set to **3**. When differential DQS (using splitter) is used, T4 must be set to **2**.
- The programmable rise and fall delay chain settings for all memory clock pins must be set to **0**.

The following table lists I/O standards supported for the write clock and write data signals for each memory type and pin location.

**Table 500. I/O standards**

MemoryType	Placement	Legal I/O Standards for DQS	Legal I/O Standards for DQ
DDR3 SDRAM	Row I/O	Differential 1.5-V SSTL Class I	1.5-V SSTL Class I
DDR3 SDRAM	Column I/O	Differential 1.5-V SSTL Class I Differential 1.5-V SSTL Class II	1.5-V SSTL Class I 1.5-V SSTL Class II
DDR2 SDRAM	Any	SSTL-18 Class I SSTL-18 Class II Differential 1.8V SSTL Class I Differential 1.8V SSTL Class II	SSTL-18 Class I SSTL-18 Class II
DDR SDRAM	Any	SSTL-2 Class I SSTL-2 Class II	SSTL-2 Class I SSTL-2 Class II
QDR II and QDR II + SRAM	Any	HSTL-1.5 Class I HSTL-1.8 Class I	HSTL-1.5 Class I HSTL-1.8 Class I
RDRAM II	Any	HSTL-1.5 Class I HSTL-1.8 Class I	HSTL-1.5 Class I HSTL-1.8 Class I

### 9.10.3. Read Data Assumptions

To verify that the external memory interface can use the FPGA Sampling Window (SW) input timing specifications, the following assumptions are necessary:

- The read clocks input pins must be placed on DQS pins. DQS/DQS# inputs must be placed on differential DQS/DQSn pins on the FPGA.
- Read data pins (DQ) must be placed on the DQ pins related to the selected DQS pins.
- For QDR II and QDR II+ SRAM interfaces, the complementary read clocks must have a single-ended I/O standard setting of HSTL-18 Class I or HSTL-15 Class I.
- For RDRAM II interfaces, the differential read clocks must have a single ended I/O standard setting of HSTL 18 Class I or HSTL 15 Class I.

#### 9.10.3.1. Read Data Assumptions for Stratix III Devices

For Stratix III devices the following additional read data and mimic pin assumptions are necessary:



- For DDR3, DDR2, and DDR SDRAM interfaces, the read clock pin can only drive a DQS bus clocking a  $\times 4$  or  $\times 9$  DQ group.
- For QDR II, QDR II+ SRAM, and RLDRAM II interfaces, the read clock pin can only drive a DQS bus clocking a  $\times 9$ ,  $\times 18$ , or  $\times 36$  DQ group.
- For non-wraparound DDR, DDR2, and DDR3 interfaces, the mimic pin, all read clock, and all read data pins must be placed on the same edge of the device (top, bottom, left, or right). For wraparound interfaces, these pins can be placed on adjacent row I/O and column I/O edges and operate at reduced frequencies.
- All read data pins and the mimic pin must feed DDIO\_IN registers and their input delay chains D1, D2, and D3 set to default values.
- DQS phase-shift setting must be either  $72^\circ$  or  $90^\circ$  (supports only one phase shift for each operating band and memory standard).
- All read clock pins must have the `dqs_ctrl_latches_enable` parameter of its `DQS_DELAY_CHAIN` WYSIWYG set to false.
- The read clocks pins must have their D4 delay chain set to the Quartus Prime software default value of **0**.
- The read data pins must have their T8 delay chain set to the Quartus Prime software default value of **0**.
- When differential DQS strobes are used (DDR3 and DDR2 SDRAM), the mimic pin must feed a true differential input buffer. Placing the memory clock pin on a `DIFFIO_RX` pin pair allows the mimic path to track timing variations on the DQS input path.
- When single ended DQS strobes are used, the mimic pin must feed a single ended input buffer.

#### 9.10.4. DLL Assumptions

The following DLL assumptions are necessary:

- The DLL must directly feed its `delayctrlout[]` outputs to all DQS pins without intervening logic or inversions.
- The DLL must be in a valid frequency band of operation as defined in the corresponding device data sheet.
- The DLL must have jitter reduction mode and dual-phase comparators enabled.

#### 9.10.5. PLL and Clock Network Assumptions for Stratix III Devices

To verify that the memory interface's PLL is configured correctly, the following assumptions are necessary:



- The PLL that generates the memory output clock signals and write data and clock signals must be set to **No compensation** mode to minimize output clock jitter.
- The reference input clock signal to the PLL must be driven by the dedicated clock input pin located adjacent to the PLL, or from the clock output signal from the adjacent PLL. If the reference clock cascades from another PLL, that upstream PLL must be in **No compensation** mode and **Low bandwidth** mode.
- For DDR3 and DDR2 SDRAM interfaces, use only regional or dual regional clock networks to route PLL outputs that generate the write data, write clock, and memory output clock signals. This requirement ensures that the memory output clocks (CK/CK#) meet the memory device input clock jitter specifications, and that output timing variations or skews are minimized.
- For other memory types, the same clock tree type (global, regional, or dual regional) is recommended for PLL clocks generating the write clock, write data, and memory clock signals to minimize timing variations or skew between these outputs.

## 9.11. Common Timing Closure Issues

The following topics describe potential timing closure issues that can occur when using the UniPHY IP.

For possible timing closure issues with UniPHY variations, refer to the *Quartus Prime Software Release Notes* for the software version that you are using. You can solve some timing issues by moving registers or changing the project fitting setting to **Standard** (from **Auto**).

The *Quartus Prime Software Release Notes* list common timing issues that can be encountered in a particular version of the Quartus Prime software.

**Note:** In UniPHY-based memory controllers, the `derive_pll_clocks` command can affect timing closure if it is called before the memory controller files are loaded. Ensure that the Quartus Prime IP File (.qip) appears in the file list before any Synopsys Design Constraint Files (.sdc) files that contain `derive_pll_clocks`.

### Related Information

[Quartus Prime Software and Device Support Release Notes](#)

### 9.11.1. Missing Timing Margin Report

The UniPHY timing margin reports may not be generated during compilation if the **.sdc** does not appear in the Quartus Prime project settings.

Timing margin reports are not generated if you specify the UniPHY variation as the top-level project entity. Instantiate the UniPHY variation as a lower level module in your user design or memory controller.

### 9.11.2. Incomplete Timing Margin Report

The timing report may not include margin information for certain timing paths if certain memory interface pins are optimized away during synthesis.



Verify that all memory interface pins appear in the `<variation>_all_pins.txt` file generated during compilation, and ensure that they connect to the I/O pins of the top-level FPGA design.

### 9.11.3. Read Capture Timing

In Stratix III and Stratix IV devices, read capture timing may fail if the DQS phase shift selected is not optimal or if the board skew specified is large.

- You can adjust the effective DQS phase shift implemented by the DLL to balance setup and hold margins on the read timing path. The DQS phase shift can be adjusted in coarse PVT-compensated steps of 22.5°, 30°, 36°, or 45° by changing the number of delay buffers (range 1 to 4), or in fine steps using the DQS phase offset feature that supports uncompensated delay addition and subtraction in approximately 12 ps steps.
- To adjust the coarse phase shift selection, determine the supported DLL modes for your chosen memory interface frequency by referencing the DLL and DQS Logic Block Specifications tables in the *Switching Characteristics* section of the device data sheet. For example, a 400 MHz DDR2 interface on a -2 speed grade device can use DLL mode 5 (resolution 36°, range 290 – 450 MHz) to implement a 72° phase shift, or DLL mode 6 (resolution 45°, range 360–560 MHz) to implement a 90° phase shift.

**Note:** Ensure that you specify the appropriate board-skew parameter when you parameterize the controllers in the parameter editor. The default board trace length mismatch used is 20 ps.

### 9.11.4. Write Timing

Negative timing margins may be reported for write timing paths if the PLL phase shift used to generate the write data signals is not optimal.

Adjust the PLL phase shift selection on the write clock PLL output using the PLL parameter editor.

**Note:** Regenerating the UniPHY-based controller overwrites changes made using the PLL parameter editor.

### 9.11.5. Address and Command Timing

You can optimize the timing margins on the address and command timing path by changing the PLL phase shift used to generate these signals. In the DDR2 or DDR3 SDRAM Controllers with UniPHY IP cores, modify the **Additional CK/CK# phase and Additional Address and Command clock phase** parameters.

The DDR2 and DDR3 SDRAM memory controllers use 1T memory timing even in half-rate mode, which may affect the address command margins for DDR2 or DDR3 SDRAM designs that use memory DIMMs. DDR2 SDRAM designs have a greater impact because the address command bus fans out to all the memory devices on a DIMM increasing the loading effect on the bus. Make sure your board designs are robust enough to have the memory clock rising edge within the 1T address command window. You can also use the **Additional Address and Command clock phase** parameter to adjust the phase of the address and command if needed.



The far-end load value and board trace delay differences between address and command and memory clock pins can result in timing failures if they are not accounted for during timing analysis.

The Quartus Prime Fitter may not optimally set output delay chains on the address and command pins. To ensure that any PLL phase-shift adjustments are not negated by delay chain adjustments, create logic assignments using the Assignment Editor to set all address and command output pin D5 delay chains to 0.

For Stratix III and Stratix IV devices, some corner cases of device family and memory frequency may require an increase to the address and command clock phase to meet core timing. You can identify this situation, if the DDR timing report shows a **PHY setup** violation with the `phy_clk` launch clock, and the address and command latch clock—clock 0 (half-rate `phy_clk`) or 2 (full-rate `phy_clk`), and 6, respectively.

If you see this timing violation, you may fix it by advancing the address and command clock phase as required. For example, a 200-ps violation for a 400-MHz interface represents 8% of the clock period, or 28.8°. Therefore, advance the address and command phase from 270° to 300°. However, this action reduces the setup and hold margin at the DDR device.

### 9.11.6. PHY Reset Recovery and Removal

A common cause for reset timing violations in UniPHY designs is the selection of a global or regional clock network for a reset signal.

The UniPHY IP does not require any dedicated clock networks for reset signals. Only UniPHY PLL outputs require clock networks, and any other PHY signal using clock networks may result in timing violations.

You can correct such timing violations by:

- Setting the Global Signal logic assignment to **Off** for the problem path (using the Assignment Editor), or
- Adjusting the logic placement (using the Assignment Editor or Chip Planner)

### 9.11.7. Clock-to-Strobe (for DDR and DDR2 SDRAM Only)

Memory output clock signals and DQS strobes are generated using the same PLL output clock. Therefore, no timing optimization is possible for this path and positive timing margins are expected for interfaces running at or below the FPGA data sheet specifications.

For DDR3 interfaces, the timing margin for this path is reported as **Write Leveling**.

### 9.11.8. Read Resynchronization and Write Leveling Timing (for SDRAM Only)

These timing paths apply only to Arria II GX, Stratix III, and Stratix IV devices, and are implemented using calibrated clock signals driving dedicated IOE registers. Therefore, no timing optimization is possible for these paths, and positive timing margin is expected for interfaces running at or below the FPGA data sheet specifications.

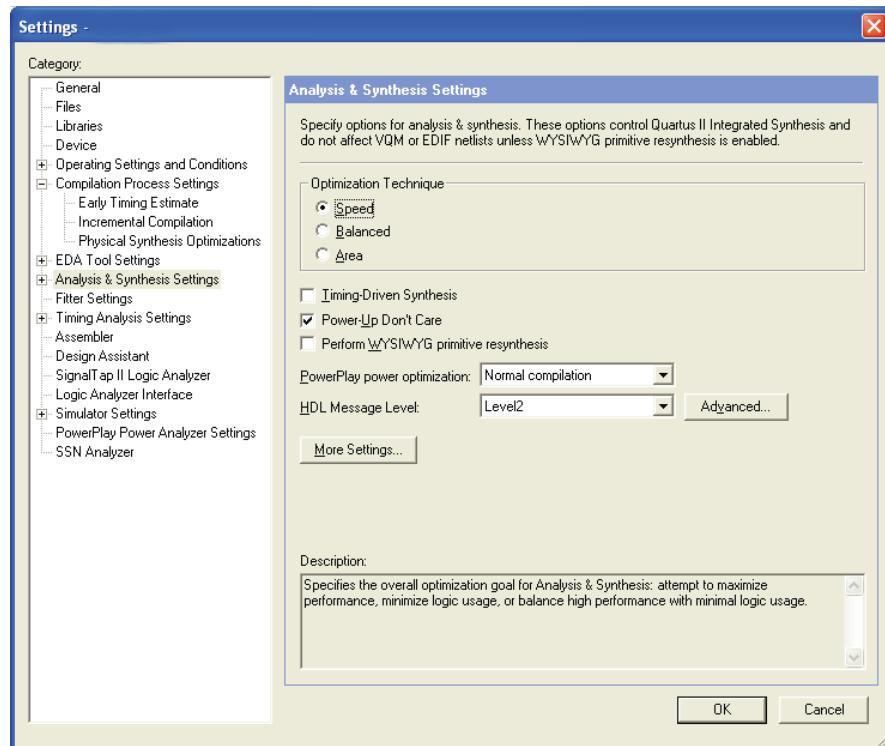
Ensure that you specify the correct memory device timing parameters ( $t_{DQSCK}$ ,  $t_{DSS}$ ,  $t_{DSH}$ ) and board skew ( $t_{EXT}$ ) in the DDR2 and DDR3 SDRAM Controllers with UniPHY parameter editor.

## 9.12. Optimizing Timing

For full-rate designs you may need to use some of the Quartus Prime advanced features, to meet core timing, by following these steps:

1. On the Assignments menu click **Settings**. In the **Category** list, click **Analysis & Synthesis Settings**. For **Optimization Technique** select **Speed**.

**Figure 93. Optimization Technique**



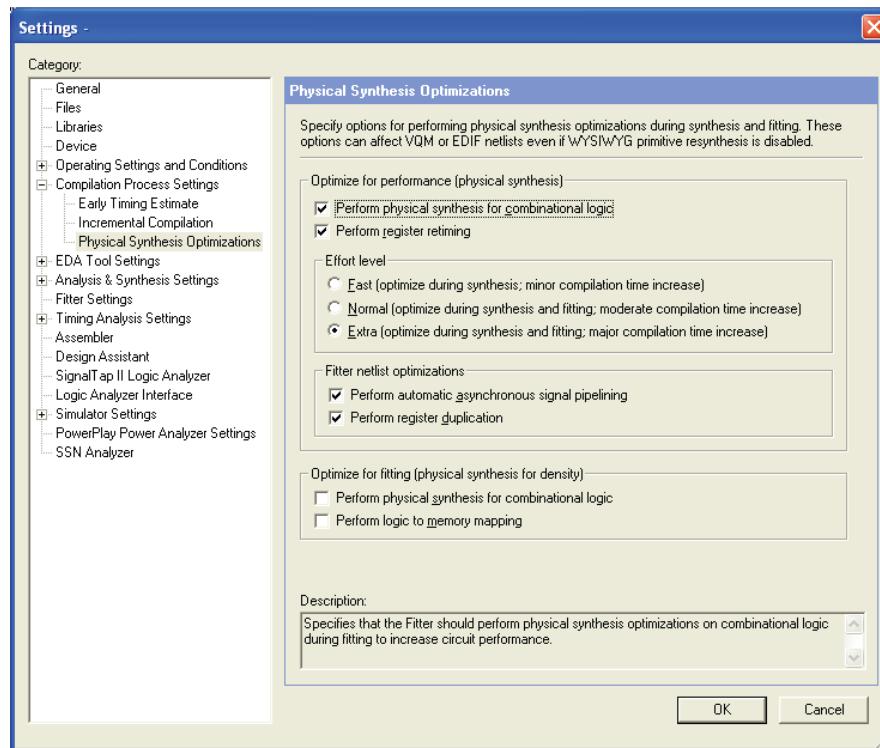
- Turn on **Perform physical synthesis for combinational logic**.

For more information about physical synthesis, refer to the *Netlist and Optimizations and Physical Synthesis* chapter in the *Quartus Prime Software Handbook*.

- Turn on **Perform register retiming**
- Turn on **Perform automatic asynchronous signal pipelining**
- Turn on **Perform register duplication**
- You can initially select **Normal** for **Effort level**, then if the core timing is still not met, select **Extra**.



Figure 94. Physical Synthesis Optimizations



#### Related Information

[Netlist Optimizations and Physical Synthesis](#)

### 9.13. Timing Derivation Methodology for Multiple Chip Select DDR2 and DDR3 SDRAM Designs

In a multiple chip select system, each individual rank has its own chip select signal. Consequently, you must change the **Total Memory chip selects**, **Number of chip select** (for discrete components) or **Number of chip select per slot** (DIMMs) in the **Preset Editor** of the UniPHY-based parameter editors.

In the **Preset Editor**, you must leave the baseline non-derated  $t_{DS}$ ,  $t_{DH}$ ,  $t_{IS}$ ,  $t_{IH}$  values, because the settings on the **Board Settings** page account for multiple chip select slew rate derivation.

The following topics explain two timing derivation methodologies for multiple chip-select DDR2 and DDR3 SDRAM designs:

- Timing Derivation using the Board Settings
- Timing Derivation Using the Excel-Based Calculator

For Arria II GX, Arria II GZ, Arria V GZ, Cyclone V, Stratix IV, and Stratix V devices, the UniPHY-based controller parameter editor has an option to select multiple chip-select deration.



**Note:** To perform multiple chip-select timing deration for other Intel devices (such as Stratix III devices), Intel provides an Excel-based

Timing deration in this section applies to either discrete components or DIMMs.

**Note:** You can derate DDR SDRAM multiple chip select designs by using the DDR2 SDRAM section of the Excel-based calculator, but Intel does not guarantee the results.

This section assumes you know how to obtain data on PCB simulations for timing deration from HyperLynx or any other PCB simulator.

#### Related Information

- [Timing Deration using the Board Settings](#) on page 542
- [Multi-Chip Select Calculator](#)

### 9.13.1. Multiple Chip Select Configuration Effects

A DIMM contains one or several RAM chips on a small PCB with pins that connect it to another system such as a motherboard or router.

Nonregistered (unbuffered) DIMMs (or UDIMMs) connect address and control buses directly from the module interface to the DRAM on the module.

Registered DIMMs (RDIMMs) and load-reduced DIMMs (LRDIMMs) improve signal integrity (and hence potential clock rates and/or overall memory size) by electrically buffering address and command signals as well as the data bus (for LRDIMMs) at a cost of additional latency. Both RDIMMs and LRDIMMs use parity on the address and command bus for increased robustness.

Multiple chip select configurations allow for one set of data pins (and address pins for UDIMMs) to be connected to two or more memory ranks. Multiple chip select configurations have a number of effects on the timing analysis including the intersymbol interference (ISI) effects, board effects, and calibration effects.

#### 9.13.1.1. ISI Effects

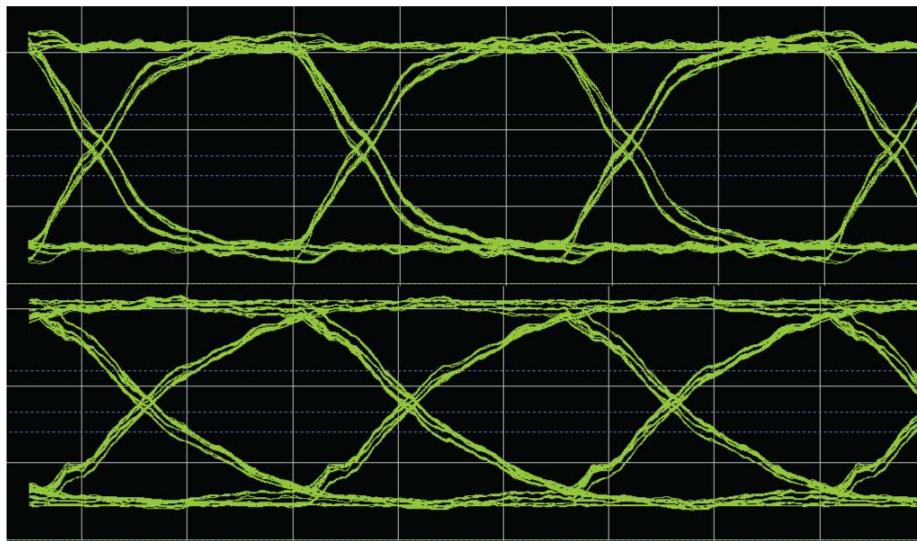
With multiple chip selects and possible slots loading the far end of the pin, there may be ISI effects on a signal causing the eye openings for DQ, DQS, and address and command signals to be smaller than for single-rank designs.

The following figure shows the eye shrinkage for DQ signal of a single rank system (top) and multiple chip select system (bottom). The ISI eye reductions reduce the timing window available for both the write path and the address and command path analysis. You must specify them as output delay constraints in the **.sdc**.

Extra loading from the additional ranks causes the slew rate of signals from the FPGA to be reduced. This reduction in slew rate affects some of the memory parameters including data, address, command and control setup and hold times ( $t_{DS}$ ,  $t_{DH}$ ,  $t_{IS}$ , and  $t_{IH}$ ).



Figure 95. Eye Shrinkage for DQ Signal



### 9.13.1.2. Calibration Effects

In addition to SI effects, multiple chip select topologies change the way that the FPGA calibrates to the memories.

In single rank without leveling situations, the calibration algorithm centers the resynchronization or capture phase such that it is optimum for the single rank. When there are two or more ranks in a system, the calibration algorithms must calibrate to the average point of the ranks.

### 9.13.1.3. Board Effects

Unequal length PCB traces result in delays reducing timing margins. Skews between different memory ranks can further reduce the timing margins in multiple chip select topologies.

Board skews can also affect the extent to which the FPGA can calibrate to the different ranks. If the skew between various signals for different ranks is large enough, the timing margin on the fully calibrated paths such as write leveling and resynchronization changes.

To account for all these board effects for Arria II GX, Arria II GZ, Arria V, Cyclone V, Stratix IV, and Stratix V devices, refer to the **Board Settings** page in the UniPHY-based controller parameter editors.

**Note:** To perform multiple chip select timing derivation for other Intel devices (for example, Stratix III devices), use the Excel-based calculator available from [www.altera.com](http://www.altera.com).

#### Related Information

[Multi-Chip Select Calculator](#)



### 9.13.2. Timing Derivation using the Board Settings

When you target Arria II GX, Arria II GZ, Arria V, Cyclone V, Stratix IV, or Stratix V devices, the UniPHY-based parameter editors include the **Board Settings** page, to automatically account for the timing deration caused by the multiple chip selects in your design.

When you target Stratix III devices, you can derate single chip-select designs using the parameter editors to account for the skews, ISI, and slew rates in the **Board Settings** page.

If you are targeting Stratix III devices you see the following warning:

"Warning: Calibration performed on all chip selects, timing analysis only performed on first chip select. Manual timing derating is required"

*Note:* You must perform manual timing deration using the Excel-based calculator.

The **Board Settings** page allows you to enter the parameters related to the board design including skews, signal integrity, and slew rates. The **Board Settings** page also includes the board skew setting parameter, **Addr/Command to CK skew**.

#### 9.13.2.1. Slew Rates

You can obtain the slew rates in one of the following ways:

- Intel performs PCB simulations on internal Intel boards to compute the output slew rate and ISI effects of various multiple chip select configurations. These simulation numbers are prepopulated in the **Slew Rates** based on the number of ranks selected. The address and command setup and hold times (tDS, tDH, tIS, tIH) are then computed from the slew rate values and the baseline nonderated tDS, tDH, tIS, tIH numbers entered in the **Preset Editor**. The parameter editor shows the computed values in **Slew Rates**. If you do not have access to a simulator to obtain accurate slew rates for your specific system, Intel recommends that you use these prepopulated numbers for an approximate estimate of your actual board parameters.
- Alternatively, you can update these default values, if dedicated board simulation results are available for the slew rates. Custom slew rates cause the tDS, tDH, tIS, tIH values to be updated. Intel recommends performing board level simulations to calculate the slew rate numbers that accounts for accurate board-level effects for your board.
- You can modify the auto-calculated tDS, tDH, tIS, tIH values with more accurate dedicated results direct from the vendor data sheets, if available.

#### 9.13.2.2. Slew Rate Setup, Hold, and Derating Calculation

Slew rate is calculated based on the nominal slew rate for setup and hold times. The total tIS (setup time) and tIH (hold time) required is calculated by adding the Micron data sheet tIS (base) and tIH (base) values to the delta tIS and delta tIH derating values, respectively.



For more information about slew rate calculation, setup, hold, and derating values, download the data sheet specifications from the following vendor websites:

- Micron ([www.micron.com](http://www.micron.com)) For example, refer to *Command and Address Setup, Hold, and Derating* section in the Micron DDR3 data sheet.
- JEDEC ([www.jedec.org](http://www.jedec.org)) For example, refer to the DDR2 SDRAM Standard data sheet.

The following section describes the timing derating algorithms and shows you where to obtain the setup, hold, and derating values in the Micron data sheet.

The slew rate derating process uses the following timing derating algorithms, which is similar to the JEDEC specification:

$$tDS = tDS(\text{base}) + \text{deltatDS} + (V_{IHAC} - V_{REF}) / (\text{DQ slew rate})$$

$$tDH = tDH(\text{base}) + \text{delta tDH} + (V_{IHDC} - V_{REF}) / (\text{DQ slew rate})$$

$$tIS = tIS(\text{base}) + \text{delta tIS} + (V_{IHAC} - V_{REF}) / (\text{Address/Command slew rate})$$

$$tIH = tIH(\text{base}) + \text{delta tIH} + (V_{IHDC} - V_{REF}) / (\text{Address/Command slew rate})$$

where:

- The setup and hold values for tDS(base), tDH(base), tIS(base), and tIH(base) are obtained from the Micron data sheet. The following figure shows an example of the values from the Micron data sheet.

**Figure 96. Setup and Hold Values from Micron Data sheet**

Parameter	Symbol	DDR3-800		DDR3-1066		DDR3-1333		DDR3-1600		Units	Notes
		Min	Max	Min	Max	Min	Max	Min	Max		
DQ Input Timing											
Data setup time to DQS, DQS#	Base (specification) $V_{REF} @ 1 \text{ V/ns}$	tDS AC175	75 250	- -	25 200	- -	- -	- -	- -	ps	18, 19
Data setup time to DQS, DQS#	Base (specification) $V_{REF} @ 1 \text{ V/ns}$	tDS AC150	125 275	- -	75 250	- -	30 180	- -	10 160	ps	18, 19, 20
Data setup time to DQS, DQS#	Base (specification) $V_{REF} @ 1 \text{ V/ns}$	tDS AC135	- -	- -	- -	- -	- -	- -	- -	ps	18, 19, 20
Data hold time from DQS, DQS#	Base (specification) $V_{REF} @ 1 \text{ V/ns}$	tDH DC100	150 250	- -	100 200	- -	65 165	- -	45 145	ps	18, 19, 20
Minimum data pulse width		tDPW	600	-	490	-	400	-	360	ps	41
Command and Address Timing											
DLL locking time		tDLLK	512	-	512	-	512	-	512	CK	28
CTRL, CMD, ADDR setup to CK, CK#	Base (specification) $V_{REF} @ 1 \text{ V/ns}$	tS AC175	200 375	- -	125 300	- -	65 240	- -	45 220	ps	29, 30
CTRL, CMD, ADDR setup to CK, CK#	Base (specification) $V_{REF} @ 1 \text{ V/ns}$	tS AC150	350 500	- -	275 425	- -	190 340	- -	170 320	ps	29, 30, 20, 30
CTRL, CMD, ADDR hold from CK, CK#	Base (specification) $V_{REF} @ 1 \text{ V/ns}$	tH DC100	275 375	- -	200 300	- -	140 240	- -	120 220	ps	29, 30, 20, 30
Minimum CTRL, CMD, ADDR pulse width		tPW	900	-	780	-	620	-	560	ps	41

- The JEDEC defined logic trip points for DDR3 SDRAM memory standard are as follow:
  - $V_{IHAC} = V_{REF} + 0.175 \text{ V}$
  - $V_{IHDC} = V_{REF} + 0.1 \text{ V}$
  - $V_{ILAC} = V_{REF} - 0.175 \text{ V}$
  - $V_{ILDC} = V_{REF} - 0.1 \text{ V}$
- The derating values for delta tIS, tIH, tDH, and tDS are obtained from the Micron data sheet.

The following figure shows an image of the derating values from the Micron data sheet.

**Figure 97. Derating Values from Micron Data sheet**

$\Delta tIS, \Delta tIH$ Derating (ps) - AC/DC-Based																
		AC175 Threshold: $V_{IH(AC)} = V_{REF(DC)} + 175\text{mV}$ , $V_{IL(AC)} = V_{REF(DC)} - 175\text{mV}$														
CMD/ ADDR Slew Rate V/ns		CK, CK# Differential Slew Rate														
		4.0 V/ns	3.0 V/ns	2.0 V/ns	1.8 V/ns	1.6 V/ns	1.4 V/ns	1.2 V/ns	1.0 V/ns	4.0 V/ns	3.0 V/ns	2.0 V/ns	1.8 V/ns			
$\Delta tIS$	$\Delta tIH$	$\Delta tIS$	$\Delta tIH$	$\Delta tIS$	$\Delta tIH$	$\Delta tIS$	$\Delta tIH$	$\Delta tIS$	$\Delta tIH$	$\Delta tIS$	$\Delta tIH$	$\Delta tIS$	$\Delta tIH$			
2.0	88	50	88	50	88	50	96	58	104	66	112	74	120	84	128	100
1.5	59	34	59	34	59	34	67	42	75	50	83	58	91	68	99	84
1.0	0	0	0	0	0	0	8	8	16	16	24	24	32	34	40	50
0.9	-2	-4	-2	-4	-2	-4	6	4	14	12	22	20	30	30	38	46
0.8	-6	-10	-6	-10	-6	-10	2	-2	10	6	18	14	26	24	34	40
0.7	-11	-16	-11	-16	-11	-16	-3	-8	5	0	13	8	21	18	29	34
0.6	-17	-26	-17	-26	-17	-26	-9	-18	-1	-10	7	-2	15	8	23	24
0.5	-35	-40	-35	-40	-35	-40	-27	-32	-19	-24	-11	-16	-2	-6	5	10
0.4	-62	-60	-62	-60	-62	-60	-54	-52	-46	-44	-38	-36	-30	-26	-22	-10

Shaded cells indicate slew rate combinations not supported

$\Delta tDS, \Delta tDH$ Derating (ps) - AC/DC-Based														
		DQS, DQS# Differential Slew Rate												
DQ Slew Rate V/ns		4.0 V/ns	3.0 V/ns	2.0 V/ns	1.8 V/ns	1.6 V/ns	1.4 V/ns	1.2 V/ns	1.0 V/ns	4.0 V/ns	3.0 V/ns	2.0 V/ns	1.8 V/ns	1.6 V/ns
		$\Delta tDS$	$\Delta tDH$	$\Delta tDS$	$\Delta tDH$	$\Delta tDS$	$\Delta tDH$	$\Delta tDS$	$\Delta tDH$	$\Delta tDS$	$\Delta tDH$	$\Delta tDS$	$\Delta tDH$	$\Delta tDS$
2.0	88	50	88	50	88	50								
1.5	59	34	59	34	59	34	67	42						
1.0	0	0	0	0	0	0	8	8	16	16				
0.9	-2	-4	-2	-4	-2	-4	6	4	14	12	22	20		
0.8					-6	-10	2	-2	10	6	18	14	26	24
0.7						-3	-8	5	0	13	8	21	18	29
0.6								-1	-10	7	-2	15	8	23
0.5										-11	-16	-2	-6	5
0.4											-30	-26	-22	-10



#### Related Information

- [www.micron.com](http://www.micron.com)
- [Micron DDR3 Data Sheet](#)
- [www.jedec.org](http://www.jedec.org)

#### 9.13.2.3. Intersymbol Interference

ISI parameters are similarly auto-populated based on the number of ranks you select with Intel's PCB simulations. You can update these autopopulated typical values, if more accurate dedicated simulation results are available.

Intel recommends performing board-level simulations to calculate the slew rate and ISI deltas that account for accurate board level effects for your board. You can use HyperLynx or similar simulators to obtain these simulation numbers. The default values have been computed using HyperLynx simulations for Intel boards with multiple DDR2 and DDR3 SDRAM slots.

The wizard writes these parameters for the slew rates and the ISI into the **.sdc** and they are used during timing analysis.

#### 9.13.2.4. Measuring Eye Reduction for Address/Command, DQ, and DQS Setup and Hold Time

This topic describes how to measure eye reduction for address/command, DQ, and DQS.

Channel signal integrity is a measure of the distortion of the eye due to intersymbol interference or crosstalk or other effects. Typically, when going from a single-rank configuration to a multi-rank configuration there is an increase in the channel loss due to reflections caused by multiple stubs. Although the Quartus Prime timing model includes some channel uncertainty, you must perform your own channel signal integrity simulations and enter the additional channel uncertainty, as compared to the reference eye, into the parameter editor.

For details about measuring channel signal integrity, refer to *Measuring Channel Signal Integrity*, on [www.alterawiki.com](http://www.alterawiki.com).

#### Related Information

[http://alterawiki.com/wiki/Measuring\\_Channel\\_Signal\\_Integrity](http://alterawiki.com/wiki/Measuring_Channel_Signal_Integrity)

### 9.14. Early I/O Timing Estimation for Arria 10 EMIF IP

Early I/O timing analysis allows you to run I/O timing analysis without first compiling your design. You can use early I/O timing analysis to quickly evaluate whether adequate timing margin exists on the I/O interface between the FPGA and external memory device.



Early I/O timing analysis performs the following analyses:

- Read analysis
- Write analysis
- Address and command analysis
- DQS gating analysis
- Write leveling analysis

Early I/O timing analysis takes into consideration the following factors:

- The timing parameters of the memory device
- The speed and topology of the memory interface
- The board timing and ISI characteristics
- The timing of the selected FPGA device

### 9.14.1. Performing Early I/O Timing Analysis for Arria 10 EMIF IP

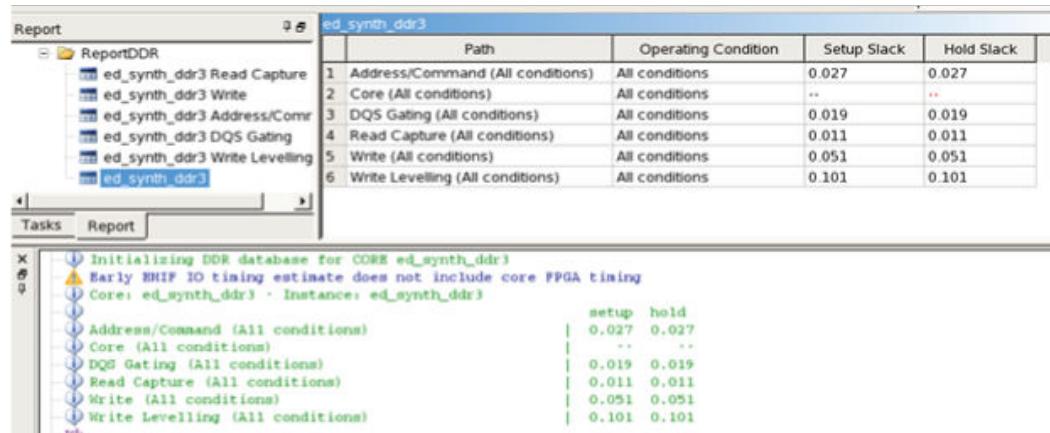
To perform early I/O timing analysis, follow these steps:

1. Instantiate an Arria 10 EMIF IP core.
  - a. On the **Memory Timing** tab, enter accurate memory parameters.
  - b. On the **Board Timing** tab, enter accurate values for Slew Rate, Intersymbol Interference, and Board and Package Skews.
2. After generating your IP core, create a Quartus Prime project and select your device from the **Available devices** list.
3. To launch the TimeQuest Timing Analyzer, select **TimeQuest Timing Analyzer** from the **Tools** menu.
4. To run early I/O timing analysis:
  - a. Select **Run Tcl Script** from the **Script** menu.
  - b. Run **submodule\<variation\_name>\_report\_io\_timing.tcl**.

The following figure shows an early I/O timing analysis from the TimeQuest Timing Analyzer using a DDR3 example design.



Figure 98. Report DDR Timing Results



Report DDR details the read capture, write, address and command, DQS gating, and write leveling timing analyses, which are identical to those obtained after a full design compilation. Core FPGA timing paths are not included in early I/O timing analysis.

## 9.15. Early I/O Timing Estimation for Stratix 10 EMIF IP

Early I/O timing analysis allows you to run I/O timing analysis without first compiling your design. You can use early I/O timing analysis to quickly evaluate whether adequate timing margin exists on the I/O interface between the FPGA and external memory device.

Early I/O timing analysis performs the following analyses:

- Read analysis
- Write analysis
- Address and command analysis
- DQS gating analysis
- Write leveling analysis

Early I/O timing analysis takes into consideration the following factors:

- The timing parameters of the memory device
- The speed and topology of the memory interface
- The board timing and ISI characteristics
- The timing of the selected FPGA device

### 9.15.1. Performing Early I/O Timing Analysis for Stratix 10 EMIF IP

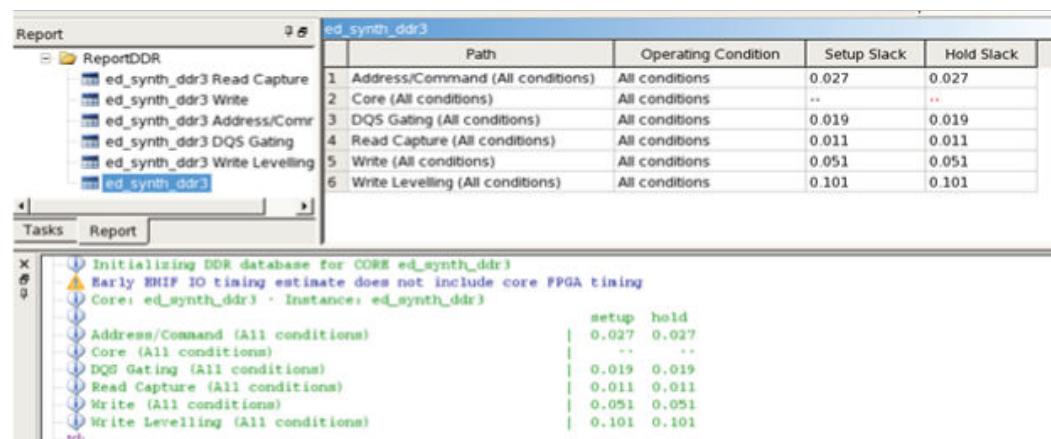
To perform early I/O timing analysis, follow these steps:

1. Instantiate a Stratix 10 EMIF IP core.
  - a. On the **Memory Timing** tab, enter accurate memory parameters.

- b. On the **Board Timing** tab, enter accurate values for Slew Rate, Intersymbol Interference, and Board and Package Skews.
2. After generating your IP core, create a Quartus Prime project and select your device from the **Available devices** list.
3. To launch the TimeQuest Timing Analyzer, select **TimeQuest Timing Analyzer** from the **Tools** menu.
4. To run early I/O timing analysis:
  - a. Select **Run Tcl Script** from the **Script** menu.
  - b. Run **submodule\<variation\_name>\_report\_io\_timing.tcl**.

The following figure shows an early I/O timing analysis from the TimeQuest Timing Analyzer using a DDR3 example design.

**Figure 99. Report DDR Timing Results**



Report DDR details the read capture, write, address and command, DQS gating, and write leveling timing analyses, which are identical to those obtained after a full design compilation. Core FPGA timing paths are not included in early I/O timing analysis.

## 9.16. Performing I/O Timing Analysis

For accurate I/O timing analysis, the Quartus Prime software must be made aware of the board trace and loading information. This information must be derived and refined during your PCB development process of pre-layout (line) and post-layout (board) simulations.

For external memory interfaces that use memory modules (DIMMs), the board trace and loading information must include the trace and loading information of the module in addition to the main and host platform, which you can obtain from your memory vendor.

You can use the following I/O timing analysis methods for your memory interface:

- Perform I/O Timing Analysis with 3rd Party Simulation Tools
- Perform Advanced I/O Timing Analysis with Board Trace Delay Model



### Related Information

- [Performing I/O Timing Analysis with Third-Party Simulation Tools](#) on page 549
- [Performing Advanced I/O Timing Analysis with Board Trace Delay Model](#) on page 549

## 9.16.1. Performing I/O Timing Analysis with Third-Party Simulation Tools

Intel recommends that you perform I/O timing analysis using the third-party simulation tool flow because this flow involves post layout simulation that can capture more accurate I/O timing. This method is also easier because it only requires you to enter the slew rates, board skews, and ISI values in the parameter editor.

To perform I/O timing analysis using third-party simulation tools, follow these steps:

1. Use a third-party simulation tool such as HyperLynx to simulate the full path for DQ, DQS, CK, Address, and Command signals.
2. Under the **Board Settings** tab of the parameter editor, enter the slowest slew rate, ISI, and board skew values.

## 9.16.2. Performing Advanced I/O Timing Analysis with Board Trace Delay Model

You should use this method only if you are unable to perform post-layout simulation on the memory interface signals to obtain the slew rate parameters, and/or when no simulation tool is available.

To perform I/O timing analysis using board trace delay model, follow these steps:

1. After the instantiation is complete, analyze and synthesize your design.
2. Add pin and DQ group assignment by running the `<variation_name>_p0_pin_assignments.tcl` script.

*Note:* The naming of the pin assignment file may vary depending on the Quartus Prime software version that you are using.
3. Enter the pin location assignments.
4. Assign the virtual pins, if necessary.
5. Enter the board trace model information. To enter board trace model information, follow these steps:
  - a. In the Pin Planner, select the pin or group of pins for which you want to enter board trace parameters.
  - b. Right-click and select **Board Trace Model**.
6. Compile your design. To compile the design, on the Processing menu, click **Start Compilation**.
7. After successfully compiling the design, perform timing analysis in the TimeQuest timing analyzer. To perform timing analysis, follow these steps:
  - a. In the Quartus Prime software, on the Tools menu, click **TimeQuest Timing Analyzer**.
  - b. On the **Tasks** pane, click **Report DDR**.
  - c. On the **Report** pane, select **Advanced I/O Timing>Signal Integrity Metrics**.



- d. In the **Signal Integrity Metrics** window, right-click and select **Regenerate** to regenerate the signal integrity metrics.
- e. In the **Signal Integrity Metrics** window, note the 10–90% rise time (or fall time if fall time is worse) at the far end for CK/CK#, address, and command, DQS/DQS#, and DQ signals.
- f. In the DDR3 SDRAM controller parameter editor, in the **Board Settings** tab, type the values you obtained from the signal integrity metrics.
- g. For the board skew parameters, set the maximum skew within DQS groups of your design. Set the other board parameters to 0 ns.
- h. Compile your design.

## 9.17. Document Revision History

Date	Version	Changes
May 2017	2017.05.08	<ul style="list-style-type: none"><li>• Added <i>Timing Constraint and Report Files for Stratix 10 EMIF IP</i>, <i>Timing Analysis Description for Stratix 10 EMIF IP</i>, and <i>Early I/O Timing Estimation for Stratix 10 EMIF IP</i> sections.</li><li>• Rebranded as Intel.</li></ul>
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	Maintenance release.
November 2015	2015.11.02	Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i> .
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	Removed occurrences of <i>MegaWizard Plug-In Manager</i> .
December 2013	2013.12.16	<ul style="list-style-type: none"><li>• Removed references to <i>ALTMEMPHY</i>.</li><li>• Removed references to <i>HardCopy</i>.</li><li>• Removed references to <i>Stratix II</i> devices.</li><li>• Removed references to <i>Cyclone III</i> and <i>Cyclone IV</i> devices.</li><li>• Added information for <i>Arria 10</i> support.</li></ul>
November 2012	6.0	Changed chapter number from 10 to 11.
June 2012	5.0	Added Feedback icon.
November 2011	4.0	<ul style="list-style-type: none"><li>• Added <i>Arria V</i> and <i>Cyclone V</i> information.</li><li>• Added <i>Performing I/O Timing Analysis</i> section.</li><li>• Added <i>Measuring Eye Reduction for Address/Command, DQ, and DQS Setup and Hold Time</i> section.</li></ul>
June 2011	3.0	Updated for 11.0 release.
December 2010	2.1	Added <i>Arria II GZ</i> and <i>Stratix V</i> , updated board skews table.
July 2010	2.0	Added information about <i>UniPHY-based IP</i> and controllers.
January 2010	1.2	Corrected typos.
December 2009	1.1	Added <i>Timing Derivation</i> section.
November 2009	1.0	Initial release.

## 10. Debugging Memory IP

---

The following topics describe the tools and processes for debugging external memory interfaces.

The discussion focuses on issues pertaining to the Intel DDR, DDR2, DDR3, DDR4, LPDDR2, LPDDR3, QDRII, QDRII+, QDRII+ Xtreme, QDR-IV, RLDRAM II, and RLDRAM 3 IP.

In general, memory debugging issues can be categorized as follows:

- Resource and planning issues
- Interface configuration issues
- Functional issues
- Timing issues

Some issues may not be directly related to interface operation; problems can also occur at the Quartus Prime Fitter stage, or in timing analysis.

### 10.1. Resource and Planning Issues

Typically, single stand-alone interfaces should not present Quartus Prime Fitter or timing problems.

You may find that fitter, timing, and hardware operation can sometimes become a challenge, as multiple interfaces are combined into a single project, or as the device utilization increases. In such cases, interface configuration is not the issue; rather, the placement and total device resource requirements can create problems.

#### Resource Issue Evaluation

External memory interfaces typically require the following resource types, which you must consider when trying to place logic manually. You might also use additional constraints to force the placement or location of external memory interface IP:

- Dedicated IOE DQS group resources and pins
- Dedicated DLL resources
- Specific PLL resources
- Specific global, regional, and dual-regional clock net resources

#### 10.1.1. Dedicated IOE DQS Group Resources and Pins

Fitter issues can occur with even a single interface, if you do not size the interface to fit within the specified constraints and requirements. A typical requirement includes containing assignments for the interface within a single bank or possibly side of the chosen device.

Intel Corporation. All rights reserved. Intel, the Intel logo, Altera, Arria, Cyclone, Empirion, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.



Such a constraint requires that the chosen device meets the following conditions:

- Sufficient DQS groups and sizes to support the required number of common I/O (CIO) or separate I/O (SIO) data groups.
- Sufficient remaining pins to support the required number of address, command, and control pins.

Failure to evaluate these fundamental requirements can result in suboptimal interface design, if the chosen device cannot be modified. The resulting wraparound interfaces or suboptimal pseudo read and write data groups artificially limit the maximum operating frequency.

Multiple blocks of IP further complicate the issue, if other IP has either no specified location constraints or incompatible location constraints.

The Quartus Prime fitter may first place other components in a location required by your memory IP, then error at a later stage because of an I/O assignment conflict between the unconstrained IP and the constrained memory IP.

Your design may require that one instance of IP is placed anywhere on one side of the device, and that another instance of IP is placed at a specific location on the same side.

While the two individual instances may compile in isolation, and the physical number of pins may appear sufficient for both instances, issues can occur if the instance without placement constraints is placed before the instance with placement constraints.

In such circumstances, Intel recommends manually placing each individual pin, or at least try using more granular placement constraints.

For more information about the pin number and DQS group capabilities of your chosen device, refer to device data sheets or the Quartus Prime Pin Planner.

### 10.1.2. Dedicated DLL Resources

Intel devices typically use DLLs to enhance data capture at the FPGA. While multiple external memory interfaces can usually share DLL resources, fitter issues can occur when there is insufficient planning before HDL coding.

If DLL sharing is required, Intel gives the following recommendations for each instance of the IP that shares the DLL resources:

- Must have compatible DLL requirements—same frequency and mode.
- Exports its autogenerated DLL instance out of its own dedicated PHY hierarchy and into the top-level design file. This procedure allows easy comparison of the generated DLL's mode, and allows you to explicitly show the required DLL sharing between two IP blocks in the HDL

*Note:* The Quartus Prime fitter does not dynamically merge DLL instances.



### 10.1.3. Specific PLL Resources

When only a single interface resides on one side or one quadrant of a device, PLL resources are typically not an issue. However if multiple interfaces or IP are required on a single side or quadrant, consider the specific PLL used by each IP, and the sharing of any PLL resources.

The Quartus Prime software automerges PLL resources, but not for any dynamically controlled PLL components. Use the following PLL resource rules:

- Ensure that the PLL located in the same bank or side of the device is available for your memory controller.
- If multiple PLLs are required for multiple controllers that cannot be shared, ensure that enough PLL resources are available within each quadrant to support your interface number requirements.
- Try to limit multiple interfaces to a single quadrant. For example, if two complete same size interfaces can fit on a single side of the device, constrain one interface entirely in one bank of that side, and the other controller in the other bank.

For more information about using multiple PHYs or controllers, refer to the design tutorials on the *List of designs using Intel External Memory IP* page of [www.alterawiki.com](http://www.alterawiki.com).

#### Related Information

[List of designs using Intel FPGA External Memory IP](#)

### 10.1.4. Specific Global, Regional and Dual-Regional Clock Net Resources

Memory PHYs typically have specific clock resource requirements for each PLL clock output.

For example because of characterization data, the PHY may require that the `phy_clk` is routed on a global clock net. The remaining clocks may all be routed on a global or a regional clock net. However, they must all be routed on the same type. Otherwise, the operating frequency of the interface is lowered, because of the increased uncertainty between two different types of clock nets. The design may still fit, but not meet timing.

### 10.1.5. Planning Your Design

It is important to understand your design and to plan its resource usage and layout. Include the following steps in your design planning:



1. Plan the total number of DQS groups and total number of other pins required in your shared area. Use the Pin Planner to assist with this activity.
2. Decide which PLLs or clock networks can be shared between IP blocks, then ensure that sufficient resources are available. For example, if an IP core requires a regional clock network, a PLL located on the opposite side of the device cannot be used.
3. Calculate the number of total clock networks and types required when trying to combine multiple instances of IP.
4. You must understand the number of quadrants that the IP uses and if this number can be reduced. For example, an interface may be autoplated across an entire side of the device, but may actually be constrained to fit in a single bank.

By optimizing physical placement, you ensure that the system uses regional clock networks wherever possible. The use of regional clock networks instead of dual-regional clock networks can help maintain clock net resources and simplify routing.

#### 10.1.6. Optimizing Design Utilization

As device utilization increases, the Quartus Prime software may have difficulty placing the core. To optimize design utilization, follow these steps:

1. Review any fitter warning messages in multiple IP designs to ensure that clock networks or PLL modes are not modified to achieve the desired fit.
2. Use the Quartus Prime Fitter resource section to compare the types of resources used in a successful standalone IP implementation to those used in an unreliable multiple IP implementation.
3. Use this information to better constrain the project to achieve the same results as the standalone project.
4. Use the Chip Planner (Floorplan and Chip Editor) to compare the placement of the working stand-alone design to the multiple IP project. Then use LogicLock™ or Design Partitions to better guide the Quartus Prime software to the required results.
5. When creating LogicLock regions, ensure that they encompass all required resources. For example, if constraining the read and write datapath hierarchy, ensure that your LogicLock region includes the IOE blocks used for your datapath pin out.

#### 10.2. Interface Configuration Performance Issues

There are a large number of interface combinations and configurations possible in an Intel design, therefore it is impractical for Intel to explicitly state the achievable  $f_{MAX}$  for every combination.

Intel seeks to provide guidance on typical performance, but this data is subject to memory component timing characteristics, interface widths, depths directly affecting timing deration requirements, and the achieved skew and timing numbers for a specific PCB.

FPGA timing issues should generally not be affected by interface loading or layout characteristics. In general, the Intel performance figures for any given device family and speed-grade combination should usually be achievable.



To resolve FPGA (PHY and PHY reset) timing issues, refer to the *Analyzing Timing of Memory IP* chapter.

Achievable interface timing (address and command, half-rate address and command, read and write capture) is directly affected by any layout issues (skew), loading issues (derivation), signal integrity issues (crosstalk timing derivation), and component speed grades (memory timing size and tolerance). Intel performance figures are typically stated for the default (single rank, unbuffered DIMM) case. Intel provides additional expected performance data where possible, but the  $f_{MAX}$  is not achievable in all configurations. Intel recommends that you optimize the following items whenever interface timing issues occur:

- Improve PCB layout tolerances
- Use a faster speed grade of memory component
- Ensure that the interface is fully and correctly terminated
- Reduce the loading (reduce the derivation factor)

#### Related Information

[Analyzing Timing of Memory IP](#) on page 494

### 10.2.1. Interface Configuration Bottleneck and Efficiency Issues

Depending on the transaction types, efficiency issues can exist where the achieved data rate is lower than expected. Ideally, these issues should be assessed and resolved during the simulation stage because they are sometimes impossible to solve later without rearchitecting the product.

Any interface has a maximum theoretical data rate derived from the clock frequency, however, in practice this theoretical data rate can never be achieved continuously due to protocol overhead and bus turnaround times.

Simulate your desired configuration to ensure that you have specified a suitable external memory family and that your chosen controller configuration can achieve your required bandwidth.

Efficiency can be assessed in several different ways, and the primary requirement is an achievable continuous data rate. The local interface signals combined with the memory interface signals and a command decode trace should provide adequate visibility of the operation of the IP to understand whether your required data rate is sufficient and the cause of the efficiency issue.

To show if under ideal conditions the required data rate is possible in the chosen technology, follow these steps:

1. Use the memory vendor's own testbench and your own transaction engine.
2. Use either your own driver, or modify the provided example driver, to replicate the transaction types typical of your system.
3. Simulate this performance using your chosen memory controller and decide if the achieved performance is still acceptable.



Observe the following points that may cause efficiency or bottleneck issues at this stage:

- Identify the memory controller rate (full, half, or quarter) and commands, which may take two or four times longer than necessary
- Determine whether the memory controller is starved for data by observing the appropriate request signals.
- Determine whether the memory controller processor transactions at a rate sufficient to meet throughput requirements by observing appropriate signals, including the local ready signal.

Intel has several versions and types of memory controller, and where possible you can evaluate different configurations based on the results of the first tests.

Consider using either a faster interface, or a different memory type to better align your data rate requirements to the IP available directly from Intel.

Intel also provides stand-alone PHY configurations so that you may develop custom controllers or use third-party controllers designed specifically for your requirements.

### 10.3. Functional Issue Evaluation

Functional issues occur at all frequencies (using the same conditions) and are not altered by speed grade, temperature, or PCB changes. You should use functional simulation to evaluate functional issues.

The Intel FPGA IP includes the option to autogenerate a testbench specific to your IP configuration, which provides an easy route to functional verification.

The following issues should be considered when trying to debug functional issues in a simulation environment.

#### 10.3.1. Correct Combination of the Quartus Prime Software and ModelSim - Intel FPGA Edition Device Models

When running any simulations, ensure that you are using the correct combination of the Quartus Prime software and device models.

Intel only tests each release of software and IP with the aligned release of device models. Failure to use the correct RTL and model combination may result in unstable simulation environments.

The ModelSim - Intel FPGA Edition comes precompiled with the Intel device family libraries included. You must always install the correct release of ModelSim - Intel FPGA Edition to align with your Quartus Prime software and IP release.

If you are using a full version of ModelSim, or any other supported simulation environment, ensure that you are compiling the current Quartus Prime supplied libraries. These libraries are located in the *<Quartus Prime install path>/quartus/eda/sim\_lib/* directory.



### 10.3.2. Intel IP Memory Model

Intel memory IP autogenerates a generic simplified memory model that works in all cases. This simple read and write model is not designed or intended to verify all entered IP parameters or transaction requirements.

The Intel-generated memory model may be suitable to evaluate some limited functional issues, but it does not provide comprehensive functional simulation.

### 10.3.3. Vendor Memory Model

Contact the memory vendor directly, because many additional models are available from the vendor's support system.

When using memory vendor models, ensure that the model is correctly defined for the following characteristics:

- Speed grade
- Organization
- Memory allocation
- Maximum memory usage
- Number of ranks on a DIMM
- Buffering on the DIMM
- ECC

**Note:** Refer to the **readme.txt** file supplied with the memory vendor model, for more information about how to define this information for your configuration. Also refer to Transcript Window Messages, for more information.

**Note:** Intel does not provide support for vendor-specific memory models.

During simulation vendor models output a wealth of information regarding any device violations that may occur because of incorrectly parameterized IP.

### 10.3.4. Insufficient Memory in Your PC

If you are running the ModelSim - Intel FPGA Edition simulator, the limitation on memory size may mean that you have insufficient memory to run your simulation. Or, if you are using a 32-bit operating system, your PC may have insufficient memory.

Typical simulation tool errors include: "Iteration limit reached" or "Error out of memory".

When using either the Intel generic memory model, or a vendor specific model quite large memory depths can be required if you do not specify your simulation carefully.

For example, if you simulate an entire 4-GB DIMM interface, the hardware platform that performs that simulation requires at least this amount of memory just for the simulation contents storage.

**Note:** Refer to Memory Allocation and Max Memory Usage in the vendor's **readme.txt** files for more information.



### 10.3.5. Transcript Window Messages

When you are debugging a functional issue in simulation, vendor models typically provide much more detailed checks and feedback regarding the interface and their operational requirements than the Intel generic model.

In general, you should use a vendor-supplied model whenever one is available. Consider using second-source vendor models in preference to the Intel generic model.

Many issues can be traced to incorrectly configured IP for the specified memory components. Component data sheets usually contain settings information for several different speed grades of memory. Be aware data sheet specify parameters in fixed units of time, frequencies, or clock cycles.

The Intel generic memory model always matches the parameters specified in the IP, as it is generated using the same engine. Because vendor models are independent of the IP generation process, they offer a more robust IP parameterization check.

During simulation, review the transcript window messages and do not rely on the Simulation Passed message at the end of simulation. This message only indicates that the example driver successfully wrote and then read the correct data for a single test cycle.

Even if the interface functionally passes in simulation, the vendor model may report operational violations in the transcript window. These reported violations often specifically explain why an interface appears to pass in simulation, but fails in hardware.

Vendor models typically perform checks to ensure that the following types of parameters are correct:

- Burst length
- Burst order
- tMRD
- tMOD
- tRFC
- tREFPDEN
- tRP
- tRAS
- tRC
- tACTPDEN
- tWR
- tWRPDEN
- tRTP
- tRDPDEN
- tINIT
- tXPDLL
- tCKE
- tRRD



- tCCD
- tWTR
- tXPR
- PRECHARGE
- CAS length
- Drive strength
- AL
- tDQS
- CAS\_WL
- Refresh
- Initialization
- tIH
- tIS
- tDH
- tDS

If a vendor model can verify all these parameters are compatible with your chosen component values and transactions, it provides a specific insight into hardware interface failures.

### 10.3.6. Passing Simulation

Passing simulation means that the interface calibrates and successfully completes a single test complete cycle without asserting pass not fail (pnf).

It does not take into account any warning messages that you may receive during simulation. If you are debugging an interface issue, review and, if necessary, correct any warning messages from the transcript window before continuing.

### 10.3.7. Modifying the Example Driver to Replicate the Failure

Often during debugging, you may discover that the example driver design works successfully, but that your custom logic encounters data errors.

When the example design works but your custom design doesn't, the underlying problem may be either of the following:

- Related to the way that the local interface transactions are occurring. You should probe and compare using the Signal Tap II analyzer.
- Related to the types or format of transactions on the external memory interface. You should try modifying the example design to replicate the problem.



Typical issues on the local interface side include:

- Incorrect local-address-to-memory-address translation causing the word order to be different than expected. Refer to *Burst Definition* in your memory vendor data sheet.
- Incorrect timing on the local interface. When your design requests a transaction, the local side must be ready to service that transaction as soon as it is accepted without any pause.
- For more information, refer to the *Avalon® Interface Specification*.

The default example driver performs only a limited set of transaction types, consequently potential bus contention or preamble and postamble issues can often be masked in its default operation. For successful debugging, isolate the custom logic transaction types that are causing the read and write failures and modify the example driver to demonstrate the same issue. Then, you can try to replicate the failure in RTL simulation with the modified driver.

For Arria 10 and Stratix 10 interfaces, you can enable the Traffic Generator 2.0 in the example design, allowing you to use the EMIF Debug Toolkit to configure different traffic pattern for debug purposes.

A problem that you can replicate in RTL simulation indicates a potential bug in the IP. You should recheck the IP parameters. A problem that you can not replicate in RTL simulation indicates a timing issue on the PCB. You can try to replicate the issue on an Intel development platform to rule out a board issue.

**Note:** Ensure that all PCB timing, loading, skew, and deration information is correctly defined in the Quartus Prime software. The timing report is inaccurate if this initial data is not correct.

Functional simulation allows you to identify any issues with the configuration of either the memory controller or the PHY. You can then check the operation against both the memory vendor data sheet and the respective JEDEC specification. After you resolve functional issues, you can start testing hardware.

For more information about simulation, refer to the *Simulating Memory IP* chapter.

#### Related Information

- [Avalon Interface Specification](#)
- [Simulating Memory IP](#) on page 468

## 10.4. Timing Issue Characteristics

The Altera PHY and controller combinations autogenerate timing constraint files to ensure that the PHY and external interface are fully constrained and that timing is analyzed during compilation. However, timing issues can still occur. This topic discusses how to identify and resolve any timing issues that you may encounter.

Timing issues typically fall into two distinct categories:

- FPGA core timing reported issues
- External memory interface timing issues in a specific mode of operation or on a specific PCB



TimeQuest reports timing issues in two categories: core to core and core to IOE transfers. These timing issues include the PHY and PHY reset sections in the TimeQuest Report DDR subsection of timing analysis. External memory interface timing issues are specifically reported in the TimeQuest Report DDR subsection, excluding the PHY and PHY reset. The Report DDR PHY and PHY reset sections only include the PHY, and specifically exclude the controller, core, PHY-to-controller and local interface. Quartus Prime timing issues should always be evaluated and corrected before proceeding to any hardware testing.

PCB timing issues are usually Quartus Prime timing issues, which are not reported in the Quartus Prime software, if incorrect or insufficient PCB topology and layout information is not supplied. PCB timing issues are typically characterized by calibration failure, or failures during user mode when the hardware is heated or cooled. Further PCB timing issues are typically hidden if the interface frequency is lowered.

#### 10.4.1. Evaluating FPGA Timing Issues

Usually, you should not encounter timing issues with Intel-provided IP unless your design exceeds Intel's published performance range or you are using a device for which the Quartus Prime software offers only preliminary timing model support. Nevertheless, timing issues can occur in the following circumstances:

- The **.sdc** files are incorrectly added to the Quartus Prime project
- Quartus Prime analysis and synthesis settings are not correct
- Quartus Prime Fitter settings are not correct

For all of these issues, refer to the correct user guide for more information about recommended settings and follow these steps:

1. Ensure that the IP generated **.sdc** files are listed in the Quartus Prime TimeQuest Timing Analyzer files to include in the project window.
2. Ensure that **Analysis and Synthesis Settings** are set to **Optimization Technique Speed**.
3. Ensure that **Fitter Settings** are set to **Fitter Effort Standard Fit**.
4. Use **TimeQuest Report Ignored Constraints**, to ensure that **.sdc** files are successfully applied.
5. Use **TimeQuest Report Unconstrained Paths**, to ensure that all critical paths are correctly constrained.

More complex timing problems can occur if any of the following conditions are true:

- The design includes multiple PHY or core projects
- Devices where the resources are heavily used
- The design includes wide, distributed, maximum performance interfaces in large die sizes

Any of the above conditions can lead to suboptimal placement results when the PHY or controller are distributed around the FPGA. To evaluate such issues, simplify the design to just the autogenerated example top-level file and determine if the core meets timing and you see a working interface. Failure implies that a more fundamental timing issue exists. If the standalone design passes core timing, evaluate how this placement and fit is different than your complete design.



Use LogicLock regions, or design partitions to better define the placement of your memory controllers. When you have your interface standalone placement, repeat for additional interfaces, combine, and finally add the rest of your design.

Additionally, use fitter seeds and increase the placement and router effort multiplier.

#### 10.4.2. Evaluating External Memory Interface Timing Issues

External memory interface timing issues usually relate to the FPGA input and output characteristics, PCB timing, and the memory component characteristics.

The FPGA input and output characteristics are usually fixed values, because the IOE structure of the devices is fixed. Optimal PLL characteristics and clock routing characteristics do have an effect. Assuming the IP is correctly constrained with autogenerated assignments, and you follow implementation rules, the design should reach the stated performance figures.

Memory component characteristics are fixed for any given component or DIMM. Consider using faster components or DIMMs in marginal cases when PCB skew may be suboptimal, or your design includes multiple ranks when deration may cause read capture or write timing challenges. Using faster memory components often reduces the memory data output skew and uncertainty easing read capture, and lowering the memory's input setup and hold requirement, which eases write timing.

Increased PCB skew reduces margins on address, command, read capture and write timing. If you are narrowly failing timing on these paths, consider reducing the board skew (if possible), or using faster memory. Address and command timing typically requires you to manually balance the reported setup and hold values with the dedicated address and command phase in the IP.

Refer to the respective IP user guide for more information.

Multiple-slot multiple-rank UDIMM interfaces can place considerable loading on the FPGA driver. Typically a quad rank interface can have thirty-six loads. In multiple-rank configurations, Intel's stated maximum data rates are not likely to be achievable because of loading deration. Consider using different topologies, for example registered DIMMs, so that the loading is reduced.

Deration because of increased loading, or suboptimal layout may result in a lower than desired operating frequency meeting timing. You should close timing in the Quartus Prime software using your expected loading and layout rules before committing to PCB fabrication.

Ensure that any design with an Altera PHY is correctly constrained and meets timing in the Quartus Prime software. You must address any constraint or timing failures before testing hardware.

For more information about timing constraints, refer to the *Analyzing Timing of Memory IP* chapter.

#### Related Information

[Analyzing Timing of Memory IP](#) on page 494

### 10.5. Verifying Memory IP Using the Signal Tap II Logic Analyzer

The Signal Tap II logic analyzer shows read and write activity in the system.



For more information about using the Signal Tap II logic analyzer, refer to the *Design Debugging Using the Signal Tap II Embedded Logic Analyzer* chapter in volume 3 of the *Quartus Prime Handbook*.

To add the Signal Tap II logic analyzer, follow these steps:

1. On the Tools menu click **Signal Tap II Logic Analyzer**.
2. In the **Signal Configuration** window next to the **Clock** box, click ... (Browse Node Finder).
3. Type the memory interface system clock (typically \* phy\_clk) in the **Named** box, for **Filter** select **Signal Tap II: presynthesis** and click **List**.
4. Select the memory interface clock that is exposed to the user logic.
5. Click **OK**.
6. Under Signal Configuration, specify the following settings:
  - For **Sample depth**, select **512**
  - For **RAM type**, select **Auto**
  - For **Trigger flow control**, select **Sequential**
  - For **Trigger position**, select **Center trigger position**
  - For **Trigger conditions**, select **1**
7. On the Edit menu, click **Add Nodes**.
8. Search for specific nodes that you want to monitor, and click **Add**.

*Note:* Signal Tap can probe only nodes that are exposed to FPGA core logic. Refer to pin descriptions in the External Memory Interface Handbook for help in deciding which signals to monitor.
9. Decide which signal and event you want to trigger on, and set the corresponding trigger condition.
10. On the File menu, click **Save**, to save the Signal Tap II .stp file to your project.

*Note:* If you see the message **Do you want to enable Signal Tap II file "stp1.stp" for the current project**, click **Yes**.
11. After you add signals to the Signal Tap II logic analyzer, recompile your design by clicking **Start Compilation** on the **Processing** menu.
12. Following compilation, verify that TimeQuest timing analysis passes successfully.
13. Connect the development board to your computer.
14. On the Tools menu, click **Signal Tap II Logic Analyzer**.
15. Add the correct <project\_name>.sof file to the SOF Manager:
  - a. Click ... to open the **Select Program Files** dialog box.
  - b. Select <your\_project\_name>.sof.
  - c. Click **Open**.
  - d. To download the file, click the **Program Device** button.
16. When the example design including Signal Tap II successfully downloads to your development board, click **Run Analysis** to run once, or click **Autorun Analysis** to run continuously.



### Related Information

[Design Debugging Using the Signal Tap Logic Analyzer](#)

#### 10.5.1. Signals to Monitor with the Signal Tap II Logic Analyzer

This topic lists the memory controller signals you should consider analyzing for different memory interfaces. This list is not exhaustive, but is a starting point.

For a description of each signal, refer to *Volume 3: Reference Material of the External Memory Interface Handbook*.

Monitor the following signals for UniPHY designs:

- avl\_addr
- avl\_rdata
- avl\_rdata\_valid
- avl\_read\_req
- avl\_ready
- avl\_wdata
- avl\_write\_req
- fail
- pass
- afi\_cal\_fail
- afi\_cal\_success
- test\_complete
- be\_reg (QDRII only)
- pnf\_per\_bit
- rdata\_reg
- rdata\_valid\_reg
- data\_out
- data\_in
- written\_data\_fifo|data\_out
- usequencer|state \*
- usequencer|phy\_seq\_rdata\_valid
- usequencer|phy\_seq\_read\_fifo\_q
- usequencer|phy\_read\_increment\_vfifo \*
- usequencer|phy\_read\_latency\_counter
- uread\_datapath|afi\_rdata\_en
- uread\_datapath|afi\_rdata\_valid



- uread\_datapath|ddio\_phy\_dq
- qvld\_wr\_address \*
- qvld\_rd\_address \*

#### Related Information

Volume 3: Reference Material

## 10.6. Hardware Debugging Guidelines

Before debugging your design, confirm that it follows the recommended design flow. Refer to the *Design Flow* chapter in volume 1 of the *External Memory Interface Handbook*.

Always keep a record of tests, to avoid repeating the same tests later. To start debugging the design, perform the following initial steps.

#### Related Information

Recommended Design Flow

### 10.6.1. Create a Simplified Design that Demonstrates the Same Issue

To help debugging, create a simple design that replicates the problem.

A simple design should compile quickly and be easy to understand. The EMIF IP generates an example top-level file that is ideal for debugging. The example top-level file uses all the same parameters, pin-outs, and so on.

If you are using the Arria 10 or Stratix 10 example design and the Traffic Generator 2.0, you can configure the traffic pattern using the EMIF Debug Toolkit.

### 10.6.2. Measure Power Distribution Network

Measure voltages of the various power supplies on their hardware development platform over a suitable time base and with a suitable trigger.

Ensure that you use an appropriate probe and grounding scheme. In addition, take the measurements directly on the pins or vias of the devices in question, and with the hardware operational.

### 10.6.3. Measure Signal Integrity and Setup and Hold Margin

Measure the signals on the PCB. When measuring any signal, consider the edge rate of the signal, not just its frequency. Modern FPGA devices have very fast edge rates, therefore you must use a suitable oscilloscope, probe, and grounding scheme when you measure the signals.

You can take measurements to capture the setup and hold time of key signal classes with respect to their clock or strobe. Ensure that the measured setup and hold margin is at least better than that reported in the Quartus Prime software. A worse margin indicates a timing discrepancy somewhere in the project; however, this issue may not be the cause of your problem.



#### 10.6.4. Vary Voltage

Vary the voltage of your system, if you suspect a marginality issue.

Increasing the voltage usually causes devices to operate faster and also usually provides increased noise margin.

#### 10.6.5. Use Freezer Spray and Heat Gun

If you have an intermittent marginal issue, apply cold or heat to the interface to stress the components.

Cooling ICs causes them to run faster, which makes timing easier. Conversely, heating ICs causes them to run slower, which makes timing more difficult.

If cooling or heating corrects the problem, it is probably a timing issue.

#### 10.6.6. Operate at a Lower Speed

Test the interface at a lower speed. If the interface works at a lower speed, the interface is correctly pinned out and functional.

If the interface fails at a lower speed, determine if the test is valid. Many high-speed memory components have a minimal operating frequency, or require subtly different configurations when operating at a lower speeds.

For example, DDR, DDR2, or DDR3 SDRAM typically requires modification to the following parameters if you want to operate the interface at lower speeds:

- $t_{MRD}$
- $t_{WTR}$
- CAS latency and CAS write latency

#### 10.6.7. Determine Whether the Issue Exists in Previous Versions of Software

Hardware that works before an update to either the Quartus Prime software or the memory IP indicates that the development platform is not the issue.

However, the previous generation IP may be less susceptible to a PCB issue, masking the issue.

#### 10.6.8. Determine Whether the Issue Exists in the Current Version of Software

Designs are often tested using previous generations of Intel software or IP.



Projects may not be upgraded for various reasons:

- Multiple engineers are on the same project. To ensure compatibility, a common release of Intel software is used by all engineers for the duration of the product development. The design may be several releases behind the current Quartus Prime software version.
- Many companies delay before adopting a new release of software so that they can first monitor Internet forums to get a feel for how successful other users say the software is.
- Many companies never use the latest version of any software, preferring to wait until the first service pack is released that fixes the primary issues.
- Some users may only have a license for the older version of the software and can only use that version until their company makes the financial decision to upgrade.
- The local interface specification from Intel FPGA IP to the customer's logic sometimes changes from software release to software release. If you have already spent resources designing interface logic, you may be reluctant to repeat this exercise. If a block of code is already signed off, you may be reluctant to modify it to upgrade to newer IP from Intel.

In all of the above scenarios, you must determine if the issue still exists in the latest version of the Intel software. Bug fixes and enhancements are added to the Intel FPGA IP every release. Depending on the nature of the bug or enhancement, it may not always be clearly documented in the release notes.

Finally, if the latest version of the software resolves the issue, it may be easier to debug the version of software that you are using.

### 10.6.9. Try A Different PCB

If you are using the same Intel FPGA IP on several different hardware platforms, determine whether the problem occurs on all platforms or just on one.

Multiple instances of the same PCB, or multiple instances of the same interface, on physically different hardware platforms may exhibit different behavior. You can determine if the configuration is fundamentally not working, or if some form of marginality is involved in the issue.

Issues are often reported on the alpha build of a development platform. These are produced in very limited numbers and often have received limited bare-board testing, or functional testing. These early boards are often more unreliable than production quality PCBs.

Additionally, if the IP is from a previous project to help save development resources, determine whether the specific IP configuration works on a previous platform.

### 10.6.10. Try Other Configurations

Designs are often quite large, using multiple blocks of IP in many different combinations. Determine whether any other configurations work correctly on the development platform.

The full project may have multiple external memory controllers in the same device, or may have configurations where only half the memory width or frequency is required. Find out what does and does not work to help the debugging of the issue.

### 10.6.11. Debugging Checklist

The following checklist is a good starting point when debugging an external memory interface.

**Table 501. Checklist**

Check	Item
<input type="checkbox"/>	Try a different fit.
<input type="checkbox"/>	Check IP parameters at the operating frequency ( $t_{MRD}$ , $t_{WTR}$ for example).
<input type="checkbox"/>	Ensure you have constrained your design with proper timing deration and have closed timing.
<input type="checkbox"/>	Simulate the design. If it fails in simulation, it will fail in hardware.
<input type="checkbox"/>	Analyze timing.
<input type="checkbox"/>	Place and assign $R_{UP}$ and $R_{DN}$ (OCT).
<input type="checkbox"/>	Measure the power distribution network (PDN).
<input type="checkbox"/>	Measure signal integrity.
<input type="checkbox"/>	Measure setup and hold timing.
<input type="checkbox"/>	Measure FPGA voltages.
<input type="checkbox"/>	Vary voltages.
<input type="checkbox"/>	Heat and cool the PCB.
<input type="checkbox"/>	Operate at a lower or higher frequency.
<input type="checkbox"/>	Check board timing and trace Information.
<input type="checkbox"/>	Check LVDS and clock sources, I/O voltages and termination.
<input type="checkbox"/>	Check PLL clock source, specification, and jitter.
<input type="checkbox"/>	Ensure the correct number of PLL phase steps take place during calibration. If the number stated in the IP does not match the number, you may have manually altered the PLL.
<input type="checkbox"/>	Retarget to a smaller interface width or a single bank.

### 10.7. Catagorizing Hardware Issues

The following topics divide issues into categories. By determining which category (or categories) an issue belongs in, you may be able to better focus on the cause of the issue.

Hardware issues fall into three categories:

- Signal integrity issues
- Hardware and calibration issues
- Intermittent issues



## 10.7.1. Signal Integrity Issues

Many design issues, including some at the protocol layer, can be traced back to signal integrity problems. You should check circuit board construction, power systems, command, and data signaling to determine if they meet specifications.

If infrequent, random errors exist in the memory subsystem, product reliability suffers. Check the bare circuit board or PCB design file. Circuit board errors can cause poor signal integrity, signal loss, signal timing skew, and trace impedance mismatches. Differential traces with unbalanced lengths or signals that are routed too closely together can cause crosstalk.

### 10.7.1.1. Characteristics of Signal Integrity Issues

Signal integrity problems often appear when the performance of the hardware design is marginal.

The design may not always initialize and calibrate correctly, or may exhibit occasional bit errors in user mode. Severe signal integrity issues can result in total failure of an interface at certain data rates, and sporadic component failure because of electrical stress. PCB component variance and signal integrity issues often show up as failures on one PCB, but not on another identical board. Timing issues can have a similar characteristic. Multiple calibration windows or significant differences in the calibration results from one calibration to another can also indicate signal integrity issues.

### 10.7.1.2. Evaluating Signal Integrity Issues

Signal integrity problems can only really be evaluated in two ways:

- direct measurement using suitable test equipment like an oscilloscope and probe
- simulation using a tool like HyperLynx or Allegro PCB SI

Compare signals to the respective electrical specification. You should look for overshoot and undershoot, non-monotonicity, eye height and width, and crosstalk.

#### 10.7.1.2.1. Skew

Ensure that all clocked signals, commands, addresses, and control signals arrive at the memory inputs at the same time.

Trace length variations cause data valid window variations between the signals, reducing margin. For example, DDR2-800 at 400 MHz has a data valid window that is smaller than 1,250 ps. Trace length skew or crosstalk can reduce this data valid window further, making it difficult to design a reliably operating memory interface. Ensure that the skew figure previously entered into the Intel FPGA IP matches that actually achieved on the PCB, otherwise Quartus Prime timing analysis of the interface is accurate.

#### 10.7.1.2.2. Crosstalk

Crosstalk is best evaluated early in the memory design phase.

Check the clock-to-data strobes, because they are bidirectional. Measure the crosstalk at both ends of the line. Check the data strobes to clock, because the clocks are unidirectional, these only need checking at the memory end of the line.



#### 10.7.1.2.3. Power System

Some memory interfaces draw current in spikes from their power delivery system as SDRAMs are based on capacitive memory cells.

Rows are read and refreshed one at a time, which causes dynamic currents that can stress any power distribution network (PDN). The various power rails should be checked either at or as close as possible to the SDRAM power pins. Ideally, you should use a real-time oscilloscope set to fast glitch triggering to check the power rails.

#### 10.7.1.2.4. Clock Signals

The clock signal quality is important for any external memory system.

Measurements include frequency, digital core design (DCD), high width, low width, amplitude, jitter, rise, and fall times.

#### 10.7.1.2.5. Read Data Valid Window and Eye Diagram

The memory generates the read signals. Take measurements at the FPGA end of the line.

To ease read diagram capture, modify the example driver to mask writes or modify the PHY to include a signal that you can trigger on when performing reads.

#### 10.7.1.2.6. Write Data Valid Window and Eye Diagram

The FPGA generates the write signals. Take measurements at the memory device end of the line.

To ease write diagram capture, modify the example driver to mask reads or modify the PHY export a signal that is asserted when performing writes.

#### 10.7.1.2.7. OCT and ODT Usage

Modern external memory interface designs typically use OCT for the FPGA end of the line, and ODT for the memory component end of the line. If either the OCT or ODT are incorrectly configured or enabled, signal integrity problems occur.

If the design uses OCT,  $R_{UP}$  or  $R_{DN}$  pins must be placed correctly for the OCT to work. If you do not place these pins, the Quartus Prime software allocates them automatically with the following warning:

```
Warning: No exact pin location assignment(s) for 2 pins of 110 total pins
Info: Pin termination_blk0~_rup_pad not assigned to an exact location on the
      device
Info: Pin termination_blk0~_rdn_pad not assigned to an exact location on the
      device
```

If you see these warnings, the  $R_{UP}$  and  $R_{DN}$  pins may have been allocated to a pin that does not have the required external resistor present on the board. This allocation renders the OCT circuit faulty, resulting in unreliable UniPHY calibration and/or interface behavior. The pins with the required external resistor must be specified in the Quartus Prime software.



For the FPGA, ensure that you perform the following:

- Specify the  $R_{UP}$  and  $R_{DN}$  pins in either the projects HDL port list, or in the assignment editor (termination\_blk0~\_rup\_pad/termination\_blk0~\_rdn\_pad).
- Connect the  $R_{UP}$  and  $R_{DN}$  pins to the correct resistors and pull-up and pull-down voltage in the schematic or PCB.
- Contain the  $R_{UP}$  and  $R_{DN}$  pins within a bank of the device that is operating at the same VCCIO voltage as the interface that is terminated.
- Check that only the expected number of  $R_{UP}$  and  $R_{DN}$  pins exists in the project pin-out file. Look for Info: Created on-chip termination messages at the fitter stage for any calibration blocks not expected in your design.
- Review the Fitter Pin-Out file for  $R_{UP}$  and  $R_{DN}$  pins to ensure that they are on the correct pins, and that only the correct number of calibration blocks exists in your design.
- Check in the fitter report that the input, output, and bidirectional signals with calibrated OCT all have the termination control block applicable to the associated  $R_{UP}$  and  $R_{DN}$  pins.

For the memory components, ensure that you perform the following:

- Connect the required resistor to the correct pin on each and every component, and ensure that it is pulled to the correct voltage.
- Place the required resistor close to the memory component.
- Correctly configure the IP to enable the desired termination at initialization time.
- Check that the speed grade of memory component supports the selected ODT setting.
- Check that the second source part that may have been fitted to the PCB, supports the same ODT settings as the original

## 10.7.2. Hardware and Calibration Issues

Hardware and calibration issues have the following definitions:

- Calibration issues result in calibration failure, which usually causes the `ctl_cal_fail` signal to be asserted.
- Hardware issues result in read and write failures, which usually causes the `pass not fail (pnf)` signal to be asserted.

**Note:** Ensure that functional, timing, and signal integrity issues are not the direct cause of your hardware issue, as functional, timing or signal integrity issues are usually the cause of any hardware issue.

### 10.7.2.1. Evaluating Hardware and Calibration Issues

Evaluate hardware and calibration issues using the Signal Tap II logic analyzer, as follows:



- To evaluate hardware issues, monitor the local side read and write interface with the Signal Tap II logic analyzer, with the pass or fail or error signals as triggers
- To evaluate calibration issues, monitor the various calibration signals with the Signal Tap II logic analyzer, with the pass or fail or error signals as triggers. Also use the EMIF debug toolkit and system consoles when available
- For more information about the EMIF debug toolkit and the type of signals for debugging external memory interfaces, refer to the *External Memory Interface Debug Toolkit* chapter in volume 3 of the *External Memory Interface Handbook*.

Consider adding core noise to your design to aggravate margin timing and signal integrity issues. Steadily increasing the stress on the external memory interface is an ideal way to assess and understand the cause of any previously intermittent failures that you may observe in your system. Using the Signal Tap II probe tool can provide insights into the source or cause of operational failure in the system.

Steadily increasing stress on the external memory interface allows you to assess and understand the impact that such factors have on the amount of timing margin and resynchronization window. Take measurements with and without the additional stress factor to allow evaluation of the overall effect.

Steadily increase the stress on the interface in the following order:

1. Increase the interface utilization by modifying the example driver to focus on the types of transactions that exhibit the issue. (For Arria 10 and Stratix 10 interfaces, you can implement an example design with the Traffic Generator 2.0 enabled, and then employ the EMIF Debug Toolkit to configure the data transaction and traffic pattern.)
2. Increase the SSN or aggressiveness of the data pattern by modifying the example driver to output in synchronization PRBS data patterns, or hammer patterns.
3. Increase the stress on the PDN by adding more and more core noise to your system. Try sweeping the fundamental frequency of the core noise to help identify resonances in your power system.

#### 10.7.2.1.1. Write Timing Margin

Determine the write timing margin by phase sweeping the write clock from the PLL.

Use sources and probes to dynamically control the PLL phase offset control, to increase and decrease the write clock phase adjustment so that the write window size may be ascertained.

Remember that when sweeping PLL clock phases, the following two factors may cause operational failure:

- The available write margin.
- The PLL phase in a multi-clock system.

The following code achieves this adjustment. You should use sources and probes to modify the respective output of the PLL. Ensure that the example driver is writing and reading from the memory while observing the pnf\_per\_byte signals to see when write failures occur:

```
///////////
wire [7:0] Probe_sig;
wire [5:0] Source_sig;
PhaseCount PhaseCounter (
    .resetn (1'b1),
```



```
.clock (pll_ref_clk),
.step (Source_sig[5]),
.updown (Source_sig[4]),
.offset (Probe_sig)
);
CheckoutPands freq_Pands (
.probe (Probe_sig),
.source (Source_sig)
);
ddr2_dimm_phy_alt_mem_phy_pll_siii pll (
.inclk0 (pll_ref_clk),
.areset (pll_reset),
.c0 (phy_clk_1x), // hR
.c1 (mem_clk_2x), // FR
.c2 (aux_clk), // FR
.c3 (write_clk_2x), // FR
.c4 (resync_clk_2x), // FR
.c5 (measure_clk_1x), // hR
.c6 (ac_clk_1x), // hR
.phasecounterselect (Source_sig[3:0]),
.phasesstep (Source_sig[5]),
.phaseupdown (Source_sig[4]),
.scanclk (scan_clk),
.locked (pll_locked_src),
.phasedone (pll_phase_done)
);
```

#### 10.7.2.1.2. Read Timing Margin

Assess the read timing margin by using sources and probes to manually control the DLL phase offset feature.

Open the autogenerated DLL using ALT\_DLL and add the additionally required offset control ports. This action allows control and observation of the following signals:

```
dll_delayctrlout[5:0], // Phase output control from DLL to DQS pins (Gray
Coded)
dll_offset_ctrl_a_addnsub, // Input add or subtract the phase offset value
dll_offset_ctrl_a_offset[5:0], // User Input controlled DLL offset value
(Gray Coded)
dll_aload, // User Input DLL load command
dll_dqsupdate, // DLL Output update required signal.
```

In examples where the applied offset applied results in the maximum or minimum `dll_delayctrlout[5:0]` setting without reaching the end of the read capture window, regenerate the DLL in the next available phase setting, so that the full capture window is assessed.

Modify the example driver to constantly perform reads (mask writes). Observe the `pnf_per_byte` signals while the DLL capture phase is manually modified to see when failures begin, which indicates the edge of the window.

A resynchronization timing failure can indicate failure at that capture phase, and not a capture failure. You should recalibrate the PHY with the calculated phase offset to ensure that you are using the true read-capture margin.

#### 10.7.2.1.3. Address and Command Timing Margin

You set the address and command clock phase directly in the IP. Assuming you enter the correct board trace model information into the Quartus Prime software, the timing analysis should be correct.



If you want to evaluate the address and command timing margin, use the same process as in "Write Timing Margin", only phase step the address and command PLL output (c6 ac\_clk\_1x). You can achieve this effect using the debug toolkit or system console.

Refer to the *External Memory Interface Debug Toolkit* chapter in volume 3 of the *External Memory Interface Handbook*.

#### Related Information

[Write Timing Margin](#) on page 572

#### 10.7.2.1.4. Resynchronization Timing Margin

Observe the size and margins available for resynchronization using the debug toolkit or system console.

Refer to *External Memory Interface Debug Toolkit* chapter in volume 3 of the *External Memory Interface Handbook*.

Additionally for PHY configurations that use a dedicated PLL clock phase (as opposed to a resynchronization FIFO buffer), use the same process as described in "Write Timing Margin", to dynamically sweep resynchronization margin (c4 resynch\_clk\_2x).

#### Related Information

[Write Timing Margin](#) on page 572

#### 10.7.2.1.5. Postamble Timing Issues and Margin

The postamble timing is set by the PHY during calibration.

You can diagnose postamble issues by viewing the pnf\_per\_byte signal from the example driver. Postamble timing issues mean only read data is corrupted during the last beat of any read request.

#### 10.7.2.1.6. Intermittent Issue Evaluation

Intermittent issues are typically the hardest type of issue to debug—they appear randomly and are hard to replicate.

Errors that occur during run-time indicate a data-related issue, which you can identify by the following actions:

- Add the Signal Tap II logic analyzer and trigger on the post-trigger pnf
- Use a stress pattern of data or transactions, to increase the probability of the issue
- Heat up or cool down the system
- Run the system at a slightly faster frequency

If adding the Signal Tap II logic analyzer or modifying the project causes the issue to go away, the issue is likely to be placement or timing related.



Errors that occur at start-up indicate that the issue is related to calibration, which you can identify by the following actions:

- Modify the design to continually calibrate and reset in a loop until the error is observed
- Where possible, evaluate the calibration margin either from the debug toolkit or system console.

*Note:* Refer to the *External Memory Interface Debug Toolkit* chapter in volume 3 of the *External Memory Interface Handbook*.

- Capture the calibration error stage or error code, and use this information with whatever specifically occurs at that stage of calibration to assist with your debugging of the issue.

## 10.8. EMIF Debug Toolkit Overview

The EMIF Debug Toolkit is a Tcl-based interface that runs on your PC and communicates via a JTAG connection to enable you to debug your external memory interface on the circuit board, retrieve calibration status, and perform margining activities. The EMIF Debug Toolkit does not support MAX 10 devices.

*Note:* For more information about the EMIF Debug Toolkit, refer to the *External Memory Interface Debug Toolkit* in volume 3 of the *External Memory Interface Handbook*.

## 10.9. Document Revision History

Date	Version	Changes
May 2017	2017.05.08	<ul style="list-style-type: none"><li>• Added Stratix 10 to <i>Modifying the Example Driver to Replicate the Failure, Create a Simplified Design that Demonstrates the Same Issue, and Evaluating Hardware and Calibration Issues</i> topics.</li><li>• Rebranded as Intel.</li></ul>
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	<ul style="list-style-type: none"><li>• Added DDR4, LPDDR2, LPDDR3, QDRII+ Xtreme, QDR-IV, and RLDRAM 3 to the list of IP in <i>Debugging Memory IP</i>.</li><li>• Added a paragraph about using the Traffic Generator 2.0 with Arria 10 interfaces, to <i>Modifying the Example Driver to Replicate the Failure</i>.</li><li>• Added a paragraph about using the Traffic Generator 2.0 with Arria 10 interfaces, to <i>Create a Simplified Design that Demonstrates the Same Issue</i>.</li><li>• Added a comment about using the EMIF Debug Toolkit with Arria 10 example designs using the Traffic Generator 2.0, to item 1 in <i>Evaluating Hardware and Calibration Issues</i>.</li><li>• Added DDR3, DDR4, and LPDDR3 controllers for Arria 10 EMIF IP to the list of toolkit components in <i>EMIF Debug Toolkit Overview and Usage Flow</i>.</li></ul>
November 2015	2015.11.02	Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i> .
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	Maintenance release.
December 2013	2013.12.16	<ul style="list-style-type: none"><li>• Removed references to ALTMEMPHY.</li><li>• Removed local_wdata_req from step 9 of <i>Verifying Memory IP Using Signal Tap II Logic Analyzer</i>.</li></ul>

*continued...*



Date	Version	Changes
November 2012	4.2	Changed chapter number from 11 to 12.
June 2012	4.1	Added Feedback icon.
November 2011	4.0	Added <i>Debug Toolkit</i> section.
June 2011	3.0	Removed leveling information from <i>ALTMEMPHY Calibration Stages</i> and <i>UniPHY Calibration Stages</i> chapter.
December 2010	2.1	<ul style="list-style-type: none"><li>Added new chapter: <i>UniPHY Calibration Stages</i>.</li><li>Added new chapter: <i>DDR2 and DDR3 SDRAM Controllers with UniPHY EMIF Toolkit</i>.</li></ul>
July 2010	2.0	Updated for 10.0 release.
January 2010	1.2	Corrected typos.
December 2009	1.1	Added <i>Debug Toolkit for DDR2 and DDR3 SDRAM High-Performance Controllers</i> chapter and <i>ALTMEMPHY Calibration Stages</i> chapter.
November 2009	1.0	Initial release.

## 11. Optimizing the Controller

---

It is important that you understand how to increase the efficiency and bandwidth of the memory controller when you design any external memory interface.

The following topics discuss factors that affect controller efficiency and ways to increase the efficiency of the controller.

### Controller Efficiency

Controller efficiency varies depending on data transaction. The best way to determine the efficiency of the controller is to simulate the memory controller for your specific design.

Controller efficiency is expressed as:

Efficiency = number of active cycles of data transfer/total number of cycles

The total number of cycles includes the number of cycles required to issue commands or other requests.

**Note:** You calculate the number of active cycles of data transfer in terms of local clock cycles. For example, if the number of active cycles of data transfer is 2 memory clock cycles, you convert that to the local clock cycle which is 1.

The following cases are based on a DDR2 SDRAM high-performance controller design targeting a Stratix® IV device that has a CAS latency of 3, and burst length of 4 on the memory side (2 cycles of data transfer), with accessed bank and row in the memory device already open. The Stratix IV device has a command latency of 9 cycles in half-rate mode. The local\_ready signal is high.

- Case 1: The controller performs individual reads.

$$\text{Efficiency} = 1/(1 + \text{CAS} + \text{command latency}) = 1/(1+1.5+9) = 1/11.5 = 8.6\%$$

- Case 2: The controller performs 4 back to back reads.

In this case, the number of data transfer active cycles is 8. The CAS latency is only counted once because the data coming back after the first read is continuous. Only the CAS latency for the first read has an impact on efficiency. The command latency is also counted once because the back to back read commands use the same bank and row.

$$\text{Efficiency} = 4/(4 + \text{CAS} + \text{command latency}) = 4/(4+1.5+9) = 1/14.5 = 27.5\%$$

### 11.1. Factors Affecting Efficiency

The two main factors that affect controller efficiency are the interface standard specified by the memory vendor, and the way that you transfer data.

The following sections discuss these two factors in detail.

Intel Corporation. All rights reserved. Intel, the Intel logo, Altera, Arria, Cyclone, Empirion, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.

### 11.1.1. Interface Standard

Complying with certain interface standard specifications affects controller efficiency.

When interfacing the memory with the DDR2 or DDR3 SDRAM controllers, you must follow certain timing specifications and perform the following bank management operations:

- **Activate**

Before you issue any read (RD) or write (WR) commands to a bank within a DDR2 SDRAM device, you must open a row in that bank using the activate (ACT) command. After you open a row, you can issue a read or write command to that row based on the  $t_{RCD}$  specification. Reading or writing to a closed row has negative impact on the efficiency as the controller has to first activate that row and then wait until  $t_{RCD}$  time to perform a read or write.

- **Precharge**

To open a different row in the same bank, you must issue a precharge (PCH) command. The precharge command deactivates the open row in a particular bank or the open row in all banks. Switching a row has a negative impact on the efficiency as you must first precharge the open row, then activate the next row and wait  $t_{RCD}$  time to perform any read or write operation to the row.

- **Device CAS latency**

The higher the CAS latency, the less efficient an individual access. The memory device has its own read latency, which is about 12 ns to 20 ns regardless of the actual frequency of the operation. The higher the operating frequency, the longer the CAS latency is in number of cycles.

- **Refresh**

A refresh, in terms of cycles, consists of the precharge command and the waiting period for the auto refresh. Based on the memory data sheet, these components require the following values:

- $t_{RP} = 12$  ns, 3 clock cycles for a 200-MHz operation (5 ns period for 200 MHz)
- $t_{RFC} = 75$  ns, 15 clock cycles for a 200-MHz operation.

Based on this calculation, a refresh pauses read or write operations for 18 clock cycles. So, at 200 MHz, you lose 1.15% ( $18 \times 5$  ns/7.8 us) of the total efficiency.

### 11.1.2. Bank Management Efficiency

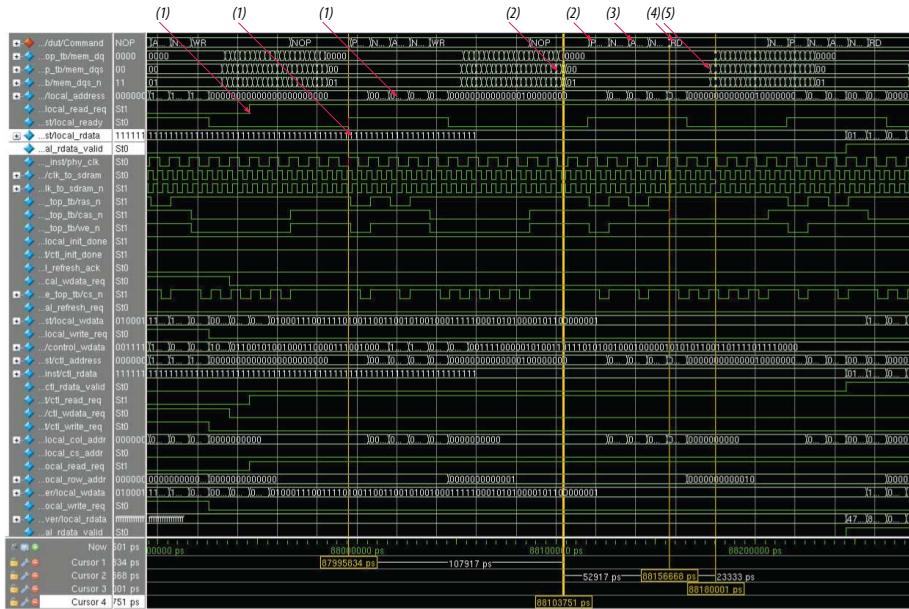
The following figures show examples of how the bank management operations affect controller efficiency.

The first figure shows a read operation in which you have to change a row in a bank. This figure shows how CAS latency and precharge and activate commands affect efficiency.

The following figure illustrates a read-after-write operation. The controller changes the row address after the write-to-read from a different row.



**Figure 100. Read Operation—Changing A Row in A Bank**

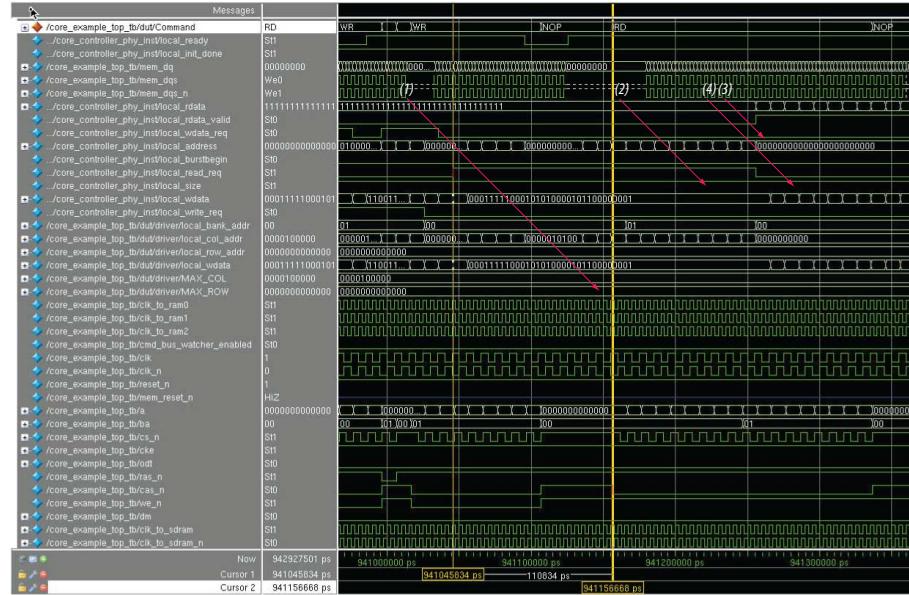


The following sequence of events describes the above figure:

1. The local\_read\_req signal goes high, and when the local\_ready signal goes high, the controller accepts the read request along with the address.
2. After the memory receives the last write data, the row changes for read. Now you require a precharge command to close the row opened for write. The controller waits for  $t_{WTR}$  time (3 memory clock cycles) to give the precharge command after the memory receives the last write data.
3. After the controller issues the precharge command, it must wait for  $t_{RP}$  time to issue an activate command to open a row.
4. After the controller gives the activate command to activate the row, it needs to wait  $t_{RCD}$  time to issue a read command.
5. After the memory receives the read command, it takes the memory some time to provide the data on the pin. This time is known as CAS latency, which is 3 memory clock cycles in this case.

For this particular case, you need approximately 17 local clock cycles to issue a read command to the memory. Because the row in the bank changes, the read operation takes a longer time, as the controller has to issue the precharge and activate commands first. You do not have to take into account  $t_{WTR}$  for this case because the precharge and activate operations already exceeded  $t_{WTR}$  time.

The following figure shows the case where you use the same the row and bank address when the controller switches from write to read. In this case, the read command latency is reduced.

**Figure 101. Changing From Write to Read—Same Row and Bank Address**


The following sequence of events describes the above figure:

1. The `local_read_req` signal goes high and the `local_ready` signal is high already. The controller accepts the read request along with the address.
2. When switching from write to read, the controller has to wait  $t_{WTR}$  time before it gives a read command to the memory.
3. The SDRAM device receives the read command.
4. After the SDRAM device receives the read command, it takes some time to give the data on the pin. This time is called CAS latency, which is 3 memory clock cycles in this case.

For the case illustrated in the second figure above, you need approximately 11 local clock cycles to issue a read command to the memory. Because the row in the bank remains the same, the controller does not have to issue the precharge and activate commands, which speeds up the read operation and in turn results in a better efficiency compared to the case in the first figure above.

Similarly, if you do not switch between read and write often, the efficiency of your controller improves significantly.

### 11.1.3. Data Transfer

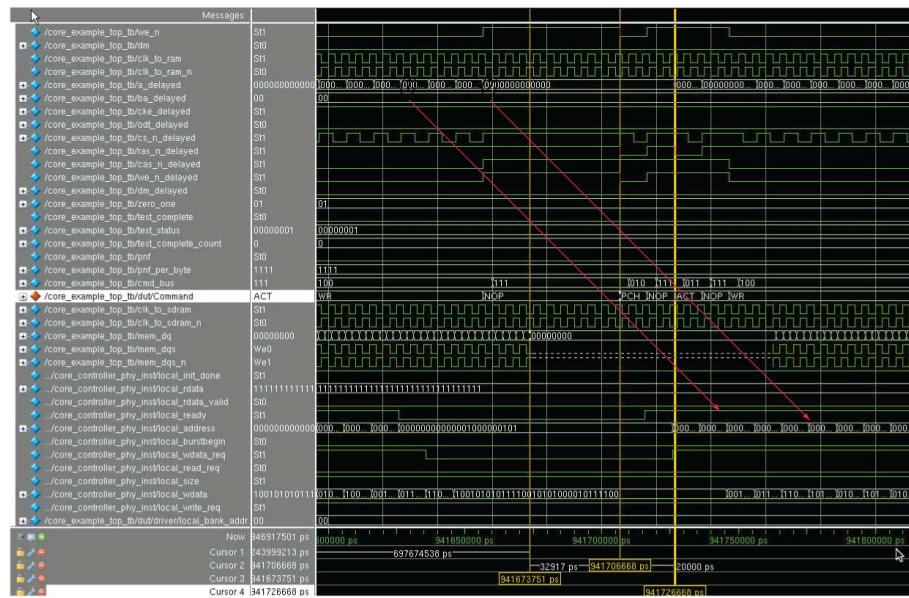
The following methods of data transfer reduce the efficiency of your controller:

- Performing individual read or write accesses is less efficient.
- Switching between read and write operation has a negative impact on the efficiency of the controller.
- Performing read or write operations from different rows within a bank or in a different bank—if the bank and a row you are accessing is not already open—also affects the efficiency of your controller.



The following figure shows an example of changing the row in the same bank.

**Figure 102. Changing Row in the Same Bank**



The following sequence of events describes the above figure:

1. You have to wait  $t_{WR}$  time before giving the precharge command
2. You then wait  $t_{RP}$  time to give the activate command.

## 11.2. Ways to Improve Efficiency

To improve the efficiency of your controller, you can use the following tools and methods:

- DDR2 SDRAM Controller
- Auto-Precharge Commands
- Additive Latency
- Bank Interleaving
- Command Queue Look-Ahead Depth
- Additive Latency and Bank Interleaving
- User-Controlled Refresh
- Frequency of Operation
- Burst Length
- Series of Reads or Writes

The following sections discuss these methods in detail.



### 11.2.1. DDR2 SDRAM Controller

The DDR2 SDRAM controller maintains up to eight open banks; one row in each bank is open at a time.

Maintaining more banks at one time helps avoid bank management commands. Ensure that you do not change a row in a bank frequently, because changing the row in a bank causes the bank to close and reopen to open another row in that bank.

### 11.2.2. Auto-Precharge Commands

The auto-precharge read and write commands allow you to indicate to the memory device that a given read or write command is the last access to the currently opened row.

The memory device automatically closes or auto-precharges the page that is currently being accessed, so that the next access to the same bank is faster. The Auto-Precharge command is useful when you want to perform fast random memory accesses.

The Timer Bank Pool (TBP) block supports the dynamic page policy, where depending on user input on local autoprecharge input would keep a page open or close. In a closed-page policy, a page is always closed after it is accessed with auto-precharge command. When the data pattern consists of repeated reads or writes to addresses not within the same page, the optimal system achieves the maximum efficiency allowed by continuous page miss limited access. Efficiency losses are limited to those associated with activating and refreshing. An efficiency of 10-20% should be expected for this closed-page policy.

In an open-page policy, the page remains open after it is accessed for incoming commands. When the data pattern consists of repeated reads or writes to sequential addresses within the same page, the optimal system can achieve 100% efficiency for page-open transactions (ignoring the effects of periodic refreshes, which typically consume around 2-3% of total efficiency), with minimum latency for highest priority single transactions.

If you turn on **Enable Auto-Precharge Control**, you can instruct the controller to issue an autoprecharge read or write command. The next time you access that bank, the access will be faster because the controller does not have to precharge the bank before activating the row that you want to access.

The controller-derived autoprecharge logic evaluates the pending commands in the command buffer and determines the most efficient autoprecharge operation to perform. The autoprecharge logic can reorder commands if necessary. When all TBP are occupied due to tracking an open page, TBP uses a scheme called on-demand flush, where it stops tracking a page to create space for an incoming command.

The following figure compares auto-precharge with and without look-ahead support.



**Figure 103. Comparison With and Without Look-ahead Auto-Precharge**

Without Look-ahead Auto-Precharge			Look-ahead Auto-Precharge		
Cycle	Command	Data	Cycle	Command	Data
1	WRITE		1	WRITE with AP	
2	NOP	DATA0 (Burst 0, Burst 1)	2	NOP	DATA0 (Burst 0, Burst 1)
3	ACT	DATA0 (Burst 2, Burst 3)	3	ACT	DATA0 (Burst 2, Burst 3)
4	NOP	DATA0 (Burst 4, Burst 5)	4	NOP	DATA0 (Burst 4, Burst 5)
5	WRITE	DATA0 (Burst 6, Burst 7)	5	WRITE	DATA0 (Burst 6, Burst 7)
6	NOP	DATA1 (Burst 0, Burst 1)	6	NOP	DATA1 (Burst 0, Burst 1)
7	ACT	DATA1 (Burst 2, Burst 3)	7	ACT	DATA1 (Burst 2, Burst 3)
8	NOP	DATA1 (Burst 4, Burst 5)	8	NOP	DATA1 (Burst 4, Burst 5)
9	WRITE	DATA1 (Burst 6, Burst 7)	9	WRITE	DATA1 (Burst 6, Burst 7)
10	NOP	DATA2 (Burst 0, Burst 1)	10	NOP	DATA2 (Burst 0, Burst 1)
11	PCH	DATA2 (Burst 2, Burst 3)	11	ACT	DATA2 (Burst 2, Burst 3)
12	NOP	DATA2 (Burst 4, Burst 5)	12	NOP	DATA2 (Burst 4, Burst 5)
13	ACT	DATA2 (Burst 6, Burst 7)	13	WRITE	DATA2 (Burst 6, Burst 7)
14	NOP	Wasted Cycle	14	NOP	DATA3 (Burst 0, Burst 1)
15	WRITE	Wasted Cycle	15	NOP	DATA3 (Burst 2, Burst 3)
16	NOP	DATA3 (Burst 0, Burst 1)	16	NOP	DATA3 (Burst 4, Burst 5)
17	NOP	DATA3 (Burst 2, Burst 3)	17	NOP	DATA3 (Burst 6, Burst 7)
18	NOP	DATA3 (Burst 4, Burst 5)			
19	NOP	DATA3 (Burst 6, Burst 7)			

Command	Bank	Row	Condition
Write	Bank 0	Row 0	
Write	Bank 1	Row 0	Activate required
Write	Bank 2	Row 0	Activate required
Write	Bank 0	Row 1	Precharge required

Without using the look-ahead auto-precharge feature, the controller must precharge to close and then open the row before the write or read burst for every row change. When using the look-ahead precharge feature, the controller decides whether to do auto-precharge read/write by evaluating the incoming command; subsequent reads or writes to same bank/different row will require only an activate command.

As shown in the preceding figure, the controller performs an auto-precharge for the write command to bank 0 at cycle 1. The controller detects that the next write at cycle 13 is to a different row in bank 0, and hence saves 2 data cycles.

The following efficiency results apply to the above figure:

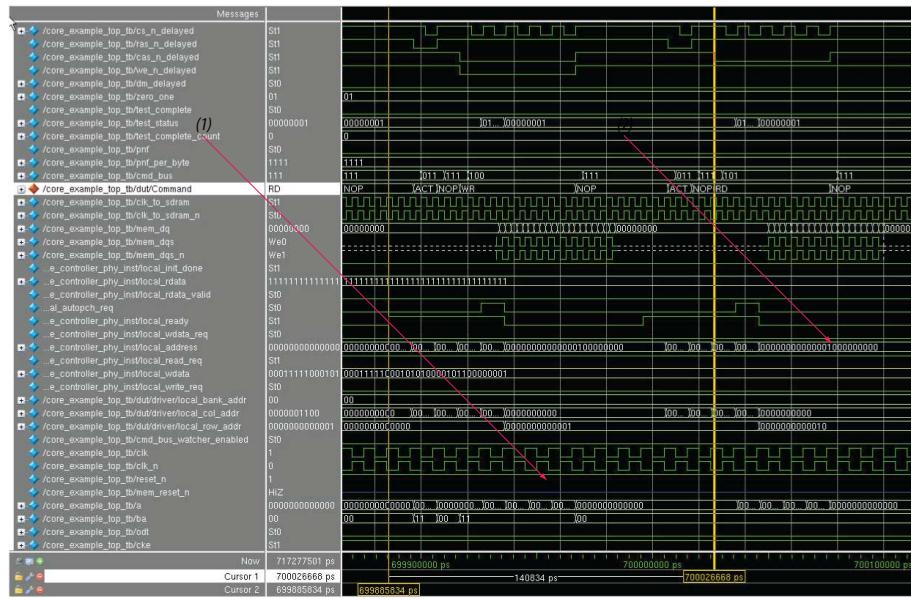
**Table 502. Comparative Efficiencies With and Without Look-Ahead Auto-Precharge Feature**

	Without Look-ahead Auto-precharge	With Look-ahead Auto-precharge
Active cycles of data transfer	16	16
Total number of cycles	19	17
Approximate efficiency	84%	94%

The look-ahead auto-precharge used increases efficiency by approximately 10%.

The following figure shows how you can improve controller efficiency using the auto-precharge command.

**Figure 104. Improving Efficiency Using Auto-Precharge Command**



The following sequence of events describes the above figure:

1. The controller accepts a read request from the local side as soon as the `local_ready` signal goes high.
2. The controller gives the activate command and then gives the read command. The read command latency is approximately 14 clock cycles for this case as compared to the similar case with no auto precharge which had approximately 17 clock cycles of latency (described in the "data Transfer" topic).

When using the auto-precharge option, note the following guidelines:

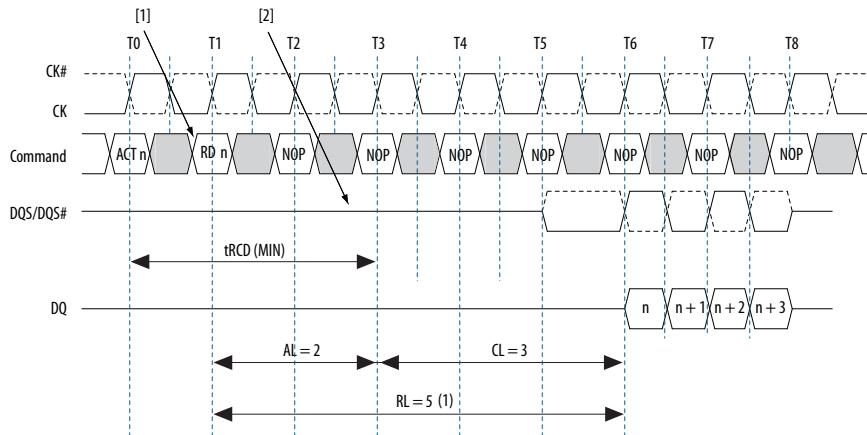
- Use the auto-precharge command if you know the controller is issuing the next read or write to a particular bank and a different row.
- Auto-precharge does not improve efficiency if you auto-precharge a row and immediately reopen it.

### 11.2.3. Additive Latency

Additive latency increases the efficiency of the command and data bus for sustainable bandwidths.

You may issue the commands externally but the device holds the commands internally for the duration of additive latency before executing, to improve the system scheduling. The delay helps to avoid collision on the command bus and gaps in data input or output bursts. Additive latency allows the controller to issue the row and column address commands—activate, and read or write—in consecutive clock cycles, so that the controller need not hold the column address for several ( $t_{RCD}$ ) cycles. This gap between the activate and the read or write command can cause bubbles in the data stream.

The following figure shows an example of additive latency.

**Figure 105. Additive Latency—Read**

The following sequence of events describes the above figure:

1. The controller issues a read or write command before the  $t_{RCD}$  (MIN) requirement — additive latency less than or equal to  $t_{RCD}$  (MIN).
2. The controller holds the read or write command for the time defined by additive latency before issuing it internally to the SDRAM device.

$$\text{Read latency} = \text{additive latency} + \text{CAS latency}$$

$$\text{Write latency} = \text{additive latency} + \text{CAS latency} - t_{CK}$$

#### 11.2.4. Bank Interleaving

You can use bank interleaving to sustain bus efficiency when the controller misses a page, and that page is in a different bank.

*Note:*

Page size refers to the minimum number of column locations on any row that you access with a single activate command. For example: For a 512Mb x8 DDR3 SDRAM with 1024 column locations (column address A[9:0]), page size = 1024 columns x 8 = 8192 bits = 8192/8 bytes = 1024 bytes (1 KB)

Without interleaving, the controller sends the address to the SDRAM device, receives the data requested, and then waits for the SDRAM device to precharge and reactivate before initiating the next data transaction, thus wasting several clock cycles.

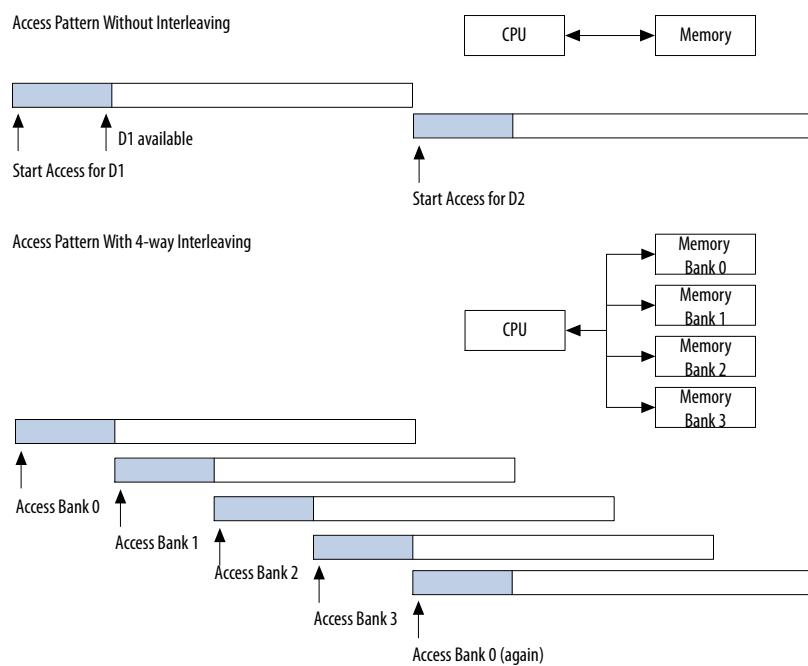
Interleaving allows banks of the SDRAM device to alternate their background operations and access cycles. One bank undergoes its precharge/activate cycle while another is being accessed. By alternating banks, the controller improves its performance by masking the precharge/activate time of each bank. If there are four banks in the system, the controller can ideally send one data request to each of the banks in consecutive clock cycles.

For example, in the first clock cycle, the CPU sends an address to Bank 0, and then sends the next address to Bank 1 in the second clock cycle, before sending the third and fourth addresses to Banks 2 and 3 in the third and fourth clock cycles respectively. The sequence is as follows:

1. Controller sends address 0 to Bank 0.
2. Controller sends address 1 to Bank 1 and receives data 0 from Bank 0.
3. Controller sends address 2 to Bank 2 and receives data 1 from Bank 1.
4. Controller sends address 3 to Bank 3 and receives data 2 from Bank 2.
5. Controller receives data 3 from Bank 3.

The following figure shows how you can use interleaving to increase bandwidth.

**Figure 106. Using Interleaving to Increase Bandwidth**



The controller supports three interleaving options:

**Chip-Bank-Row-Col** – This is a noninterleaved option. Select this option to improve efficiency with random traffic

**Chip-Row-Bank-Col** – This option uses bank interleaving without chip select interleaving. Select this option to improve efficiency with sequential traffic, by spreading smaller data structures across all banks in a chip.

**Row-Chip-Bank-Col** - This option uses bank interleaving with chip select interleaving. Select this option to improve efficiency with sequential traffic and multiple chip selects. This option allows smaller data structures to spread across multiple banks and chips.



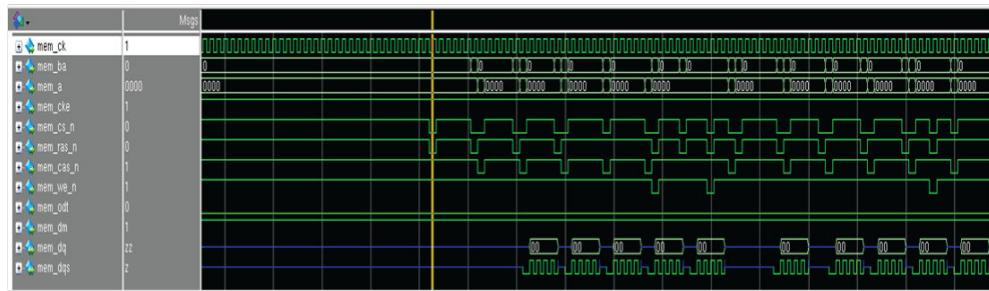
Bank interleaving is a fixed pattern of data transactions, enabling best-case bandwidth and latency, and allowing for sufficient interleaved transactions between opening banks to completely hide  $t_{RC}$ . An optimal system can achieve 100% efficiency for bank interleave transactions with 8 banks. A system with less than 8 banks is unlikely to achieve 100%.

### 11.2.5. Command Queue Look-Ahead Depth

The command queue look-ahead depth value determines the number of read or write requests that the look-ahead bank management logic examines. The command queue look-ahead depth value also determines how many open pages the High-Performance Controller II (HPC II) can track.

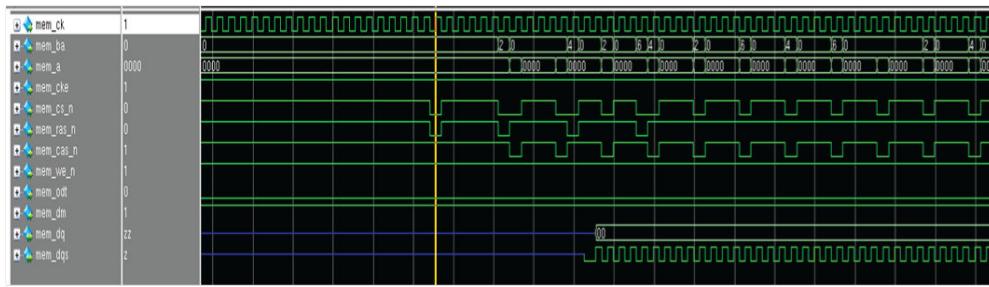
For example, if you set the command queue look-ahead depth value to 4, the HPC II controller can track 4 open pages. In a 4-bank interleaving case, HPCII will receive repeated commands with addresses of bank A, bank B, bank C, and bank D. To receive the next set of commands, the controller issues a precharge command to exit the current page and then issues an activate command to track the new incoming page, leading to a drop in efficiency.

**Figure 107. Simulation with Command Queue Look-ahead Depth of 4**



With the command queue look-ahead depth set to 8, the controller can track 8 open pages and overall efficiency is much improved relative to a command queue look-ahead value of 4.

**Figure 108. Simulation with Command Queue Look-ahead Depth of 8**



There is a trade-off between efficiency and resource usage. Higher command queue look-ahead values are likely to increase bank management efficiency, but at the cost of higher resource usage. Smaller command queue look-ahead values may be less efficient, but also consume fewer resources. Also, a command queue look-ahead value greater than 4 may cause timing violations for interfaces approaching their maximum frequency.

**Note:** If you set Command Queue Look-ahead depth to a value greater than 4, you may not be able to run the interface at maximum frequency.

To achieve an optimized balance of controller efficiency versus resource usage and frequency, you must understand your traffic patterns. You should simulate your design with a variety of controller settings to observe the results of different settings.

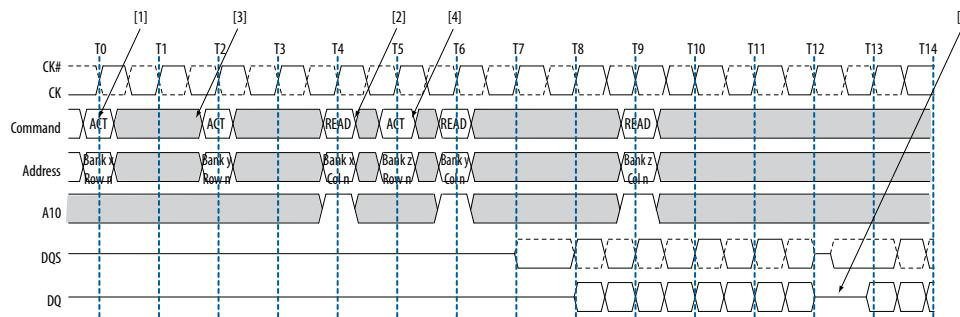
**Note:** User-selectable Command Queue Look-ahead depth is available only when using the soft memory controller. For the hard memory controller, the Command Queue Look-ahead depth value is hard-coded to 8.

### 11.2.6. Additive Latency and Bank Interleaving

Using additive latency together with bank interleaving increases the bandwidth of the controller.

The following figure shows an example of bank interleaving in a read operation without additive latency. The example uses DDR2 SDRAM bank interleave reads with CAS latency of 4, and burst length of 4.

**Figure 109. Bank Interleaving—Without Additive Latency**



The following sequence of events describes the above figure:

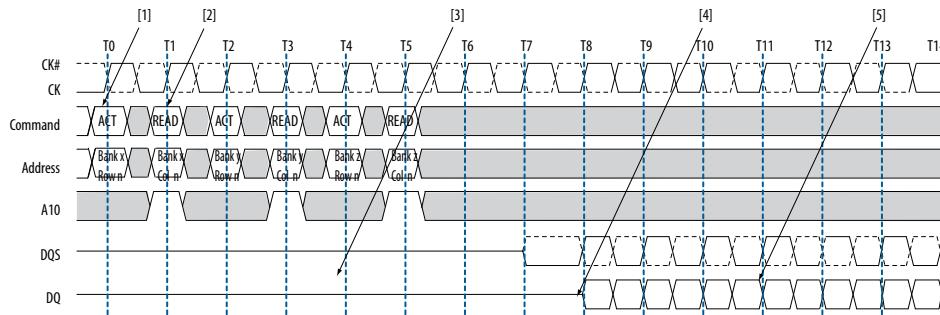
1. The controller issues an activate command to open the bank, which activates bank x and the row in it.
2. After  $t_{RCD}$  time, the controller issues a read with auto-precharge command to the specified bank.
3. Bank y receives an activate command after  $t_{RRD}$  time.
4. The controller cannot issue an activate command to bank z at its optimal location because it must wait for bank x to receive the read with auto-precharge command, thus delaying the activate command for one clock cycle.
5. The delay in activate command causes a gap in the output data from the DDR2 SDRAM device.

**Note:** If you use additive latency of 1, the latency affects only read commands and not the timing for write commands.

The following figure shows an example of bank interleaving in a read operation with additive latency. The example uses DDR2 SDRAM bank interleave reads with additive latency of 3, CAS latency of 4, and burst length of 4. In this configuration, the controller issues back-to-back activate and read with auto-precharge commands.



Figure 110. Bank Interleaving—With Additive Latency



The following sequence of events describes the above figure:

1. The controller issues an activate command to bank x.
2. The controller issues a read with auto precharge command to bank x right after the activate command, before waiting for the  $t_{RCD}$  time.
3. The controller executes the read with auto-precharge command  $t_{RCD}$  time later on the rising edge T4.
4. 4 cycles of CAS latency later, the SDRAM device issues the data on the data bus.
5. For burst length of 4, you need 2 cycles for data transfer. With 2 clocks of giving activate and read with auto-precharge commands, you get a continuous flow of output data.

Compare the efficiency results in the two preceding figures:

- DDR2 SDRAM bank interleave reads with no additive latency, CAS latency of 4, and burst length of 4 (first figure),
   
Number of active cycles of data transfer = 6.
   
Total number of cycles = 15
   
Efficiency = 40%
- DDR2 SDRAM bank interleave reads with additive latency of 3, CAS latency of 4, and burst length of 4 (second figure),
   
Number of active cycles of data transfer = 6.
   
Total number of cycles = 14
   
Efficiency = approximately 43%

The interleaving reads used with additive latency increases efficiency by approximately 3%.

**Note:** Additive latency improves the efficiency of back-to-back interleaved reads or writes, but not individual random reads or writes.

### 11.2.7. User-Controlled Refresh

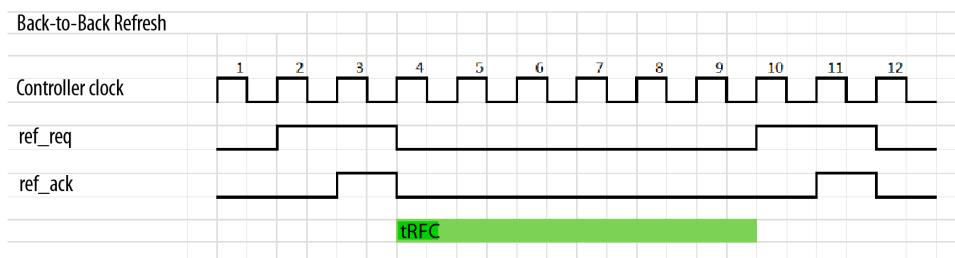
The requirement to periodically refresh memory contents is normally handled by the memory controller; however, the **User Controlled Refresh** option allows you to determine when memory refresh occurs.

With specific knowledge of traffic patterns, you can time the refresh operations so that they do not interrupt read or write operations, thus improving efficiency.

**Note:** If you enable the auto-precharge control, you must ensure that the average periodic refresh requirement is met, because the controller does not issue any refreshes until you instruct it to.

### 11.2.7.1. Back-to-Back User-Controlled Refresh Usage in Arria 10

The following diagram illustrates the back-to-back refresh model for optimized hard memory controller (HMC) performance in Arria 10 devices.



For optimal performance, ensure that you deassert the Refresh request after receiving the acknowledgement pulse. You can implement a timer to track  $t_{RFC}$  before asserting the next Refresh request. Failure to deassert the Refresh request can delay memory access to the rank not in refresh.

### 11.2.8. Frequency of Operation

Certain frequencies of operation give you the best possible latency based on the memory parameters. The memory parameters you specify through the parameter editor are converted to clock cycles and rounded up.

If you are using a memory device that has  $t_{RCD} = 20$  ns and running the interface at 100 MHz, you get the following results:

- For full-rate implementation ( $t_{CK} = 10$  ns):
 
$$t_{RCD} \text{ convert to clock cycle} = 20/10 = 2.$$
- For half rate implementation ( $t_{CK} = 20$  ns):
 
$$t_{RCD} \text{ convert to clock cycle} = 20/20 = 1$$

This frequency and parameter combination is not easy to find because there are many memory parameters and frequencies for the memory device and the controller to run. Memory device parameters are optimal for the speed at which the device is designed to run, so you should run the device at that speed.

In most cases, the frequency and parameter combination is not optimal. If you are using a memory device that has  $t_{RCD} = 20$  ns and running the interface at 133 MHz, you get the following results:

- For full-rate implementation ( $t_{CK} = 7.5$  ns):
 
$$t_{RCD} \text{ convert to clock cycle} = 20/7.5 = 2.66, \text{ rounded up to 3 clock cycles or } 22.5 \text{ ns.}$$



- For half rate implementation ( $t_{CK} = 15$  ns):
  $t_{RCD}$  convert to clock cycle =  $20/15 = 1.33$ , rounded up to 2 clock cycles or 30 ns.  
 There is no latency difference for this frequency and parameter combination.

### 11.2.9. Burst Length

Burst length affects the efficiency of the controller. A burst length of 8 provides more cycles of data transfer, compared to a burst length of 4.

For a half-rate design that has a command latency of 9 half-rate clock cycles, and a CAS latency of 3 memory clock cycles or 1.5 half rate local clock cycles, the efficiency is 9% for burst length of 4, and 16% for burst length of 8.

- Burst length of 4 (2 memory clock cycles of data transfer or 1 half-rate local clock cycle)  
 Efficiency = number of active cycles of data transfer/total number of cycles  

$$\text{Efficiency} = 1/(1 + \text{CAS} + \text{command latency}) = 1/(1 + 1.5 + 9) = 1/11.5 = 8.6\%$$
 or approximately 9%
- Burst length of 8 (4 memory clock cycles of data transfer or 2 half-rate local clock cycles)  
 Efficiency = number of active cycles of data transfer/total number of cycles  

$$\text{Efficiency} = 2/(2 + \text{CAS} + \text{command latency}) = 2/(2 + 1.5 + 9) = 2/12.5 = 16\%$$

### 11.2.10. Series of Reads or Writes

Performing a series of reads or writes from the same bank and row increases controller efficiency.

The case shown in the second figure in the "Bank Management Efficiency" topic demonstrates that a read performed from the same row takes only 14.5 clock cycles to transfer data, making the controller 27% efficient.

Do not perform random reads or random writes. When you perform reads and writes to random locations, the operations require row and bank changes. To change banks, the controller must precharge the previous bank and activate the row in the new bank. Even if you change the row in the same bank, the controller has to close the bank (precharge) and reopen it again just to open a new row (activate). Because of the precharge and activate commands, efficiency can decrease by as much as 3–15%, as the controller needs more time to issue a read or write.

If you must perform a random read or write, use additive latency and bank interleaving to increase efficiency.

Controller efficiency depends on the method of data transfer between the memory device and the FPGA, the memory standards specified by the memory device vendor, and the type of memory controller.

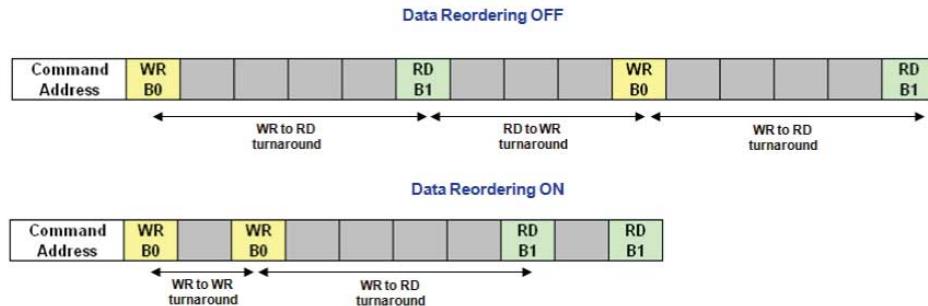
### 11.2.11. Data Reordering

Data reordering and command reordering can both contribute towards achieving controller efficiency.

The Data Reordering feature allows the single-port memory controller to change the order of read and write commands to achieve highest efficiency. You can enable data reordering by turning on **Enable Reordering** on the **Controller Settings** tab of the parameter editor.

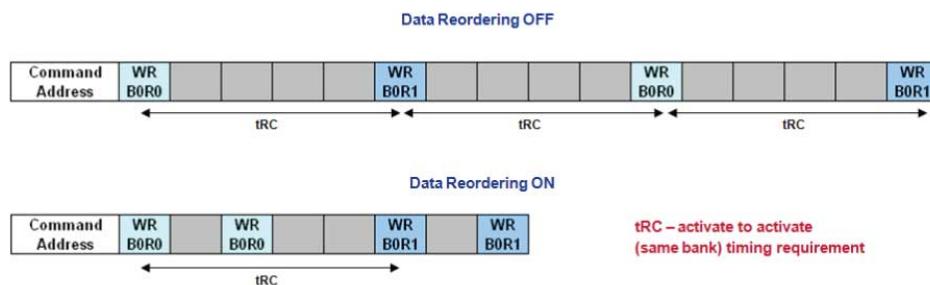
In the soft memory controller, inter-bank data reordering serves to minimize bus turnaround time by optimizing the ordering of read and write commands going to different banks; commands going to the same bank address are not reordered.

**Figure 111. Data Reordering for Minimum Bus Turnaround**



In the hard memory controller, inter-row data reordering serves to minimize  $t_{RC}$  by reordering commands going to different bank and row addresses; command going to the same bank and row address are not reordered. Inter-row data reordering inherits the minimum bus turnaround time benefit from inter-bank data reordering.

**Figure 112. Data Reordering for Minimum  $t_{RC}$**



### 11.2.12. Starvation Control

The controller implements a starvation counter to ensure that lower-priority requests are not forgotten as higher-priority requests are reordered for efficiency.

In starvation control, a counter is incremented for every command served. You can set a starvation limit, to ensure that a waiting command is served immediately upon the starvation counter reaching the specified limit.

For example, if you set a starvation limit of 10, a lower-priority command will be treated as high priority and served immediately, after ten other commands are served before it.



### 11.2.13. Command Reordering

Data reordering and command reordering can both contribute towards achieving controller efficiency.

DDR protocols are naturally inefficient, because commands are fetched and processed sequentially. The DDRx command and DQ bus are not fully utilized as few potential cycles are wasted and degrading the efficiency

The command reordering feature, or look-ahead bank management feature, allows the controller to issue bank management commands early based on incoming patterns, so that when the command reaches the memory interface, the desired page in memory is already open.

The command cycles during the t<sub>RCD</sub> period are idle and the bank-management commands are issued to next access banks. When the controller is serving the next command, the bank is already precharged. The command queue look-ahead depth is configurable from 1-16, to specify how many read or write requests the look-ahead bank management logic examines. With the look-ahead command queue, if consecutive write or read requests are to a sequential address with same row, same bank, and column incremental by 1, the controller merges the write or read requests at the memory transaction into a single burst.

**Figure 113. Comparison With and Without Look-Ahead Bank Management Feature**

Without Lookahead			With Lookahead		
Cycle	Command	Data	Cycle	Command	Data
1	ACT		1	ACT	
2	NOP		2		
3	NOP		3		
4	READ		4	READ	
5		DATA0 (Burst 0, Burst 1)	5	ACT	DATA0 (Burst 0, Burst 1)
6	NOP	DATA0 (Burst 2, Burst 3)	6	NOP	DATA0 (Burst 2, Burst 3)
7	ACT	DATA0 (Burst 4, Burst 5)	7	ACT	DATA0 (Burst 4, Burst 5)
8	NOP	DATA0 (Burst 6, Burst 7)	8	READ	DATA0 (Burst 6, Burst 7)
9	NOP	Wasted Cycle	9	NOP	DATA1 (Burst 0, Burst 1)
10	READ	Wasted Cycle	10	NOP	DATA1 (Burst 2, Burst 3)
11		DATA1 (Burst 0, Burst 1)	11	NOP	DATA1 (Burst 4, Burst 5)
12	NOP	DATA1 (Burst 2, Burst 3)	12	READ	DATA1 (Burst 6, Burst 7)
13	ACT	DATA1 (Burst 4, Burst 5)	13	NOP	DATA2 (Burst 0, Burst 1)
14	NOP	DATA1 (Burst 6, Burst 7)	14	NOP	DATA2 (Burst 2, Burst 3)
15	NOP	Wasted Cycle	15	NOP	DATA2 (Burst 4, Burst 5)
16	READ	Wasted Cycle	16	NOP	DATA2 (Burst 6, Burst 7)
17	NOP	DATA2 (Burst 0, Burst 1)			
18	NOP	DATA2 (Burst 2, Burst 3)			
19	NOP	DATA2 (Burst 4, Burst 5)			
20	NOP	DATA2 (Burst 6, Burst 7)			

Command	Address	Condition
Read	Bank 0	Activate required
Read	Bank 1	Precharge required
Read	Bank 2	Precharge required

Compare the following efficiency results for the above figure:

**Table 503. Efficiency Results for Above Figure**

	Without Look-ahead Bank Management	With Look-ahead Bank Management
Active cycles of data transfer	12	12
Total number of cycles	20	16
Approximate efficiency	60%	75%

In the above table, the use of look-ahead bank management increases efficiency by 15%. The bank look-ahead pattern verifies that the system is able to completely hide the bank precharge and activation for specific sequences in which the minimum number of page-open transactions are placed between transactions to closed pages to allow bank look-ahead to occur just in time for the closed pages. An optimal system would completely hide bank activation and precharge performance penalties for the bank look-ahead traffic pattern and achieve 100% efficiency, ignoring refresh.

### 11.2.14. Bandwidth

Bandwidth depends on the efficiency of the memory controller controlling the data transfer to and from the memory device.

You can express bandwidth as follows:

$$\text{Bandwidth} = \text{data width (bits)} \times \text{data transfer rate (1/s)} \times \text{efficiency}$$

$$\text{Data rate transfer (1/s)} = 2 \times \text{frequency of operation (4 for QDR SRAM interfaces)}$$

The following example shows the bandwidth calculation for a 16-bit interface that has 70% efficiency and runs at 200 MHz frequency:

$$\text{Bandwidth} = 16 \text{ bits} \times 2 \text{ clock edges} \times 200 \text{ MHz} \times 70\% = 4.48 \text{ Gbps.}$$

DRAM typically has an efficiency of around 70%, but when you use the memory controller, efficiency can vary from 10 to 92%.

In QDR II+ or QDR II SRAM the IP implements two separate unidirectional write and read data buses, so the data transfer rate is four times the clock rate. The data transfer rate for a 400-MHz interface is 1,600 Mbps. The efficiency is the percentage of time the data bus is transferring data. It is dependent on the type of memory. For example, in a QDR II+ or QDR II SRAM interface with separate write and read ports, the efficiency is 100% when there is an equal number of read and write operations on these memory interfaces.

For information on best-case and worst-case efficiency scenarios, refer to the white paper, *The Efficiency of the DDR & DDR2 SDRAM Controller Compiler*.

#### Related Information

[The Efficiency of the DDR & DDR2 SDRAM Controller Compiler](#)



## 11.2.15. Efficiency Monitor

The Efficiency Monitor and Protocol Checker measures traffic efficiency on the Avalon interface between the traffic generator and the controller, and checks that the Avalon protocol is not violated.

The Protocol Checker monitors the controller's Avalon slave interface for any illegal commands presented to it by any master; it does not monitor the legality of the controller's Avalon outputs.

**Figure 114. Efficiency Monitor and Protocol Checker Statistics**

The screenshot shows a software interface with a tree view on the left and a table on the right. The tree view includes 'External Memory Interface Toolkit Summary', 'Connections', 'External Memory Interface: core1\_example\_if0:if0', 'Avalon-MM Efficiency Monitor: core1\_example\_if0:', and sub-options like 'Summary', 'Interface Details', 'Efficiency Monitor Statistics' (which is selected), and 'Protocol Checker Summary Statistics'. The table on the right is titled 'Efficiency Monitor Statistics' and contains the following data:

Counter	Value
Efficiency Monitor Cycle Count	1621015524
Transfer Write Count	422114606
Write Count	244767146
Read Count	119528668
Read Burst Count	24809931
Non-Transfer Cycle Waitrequest Count	837988587
Non-Transfer Cycle No readdatavalid Count	1115317247
Non-Transfer Cycle Master Write Idle Count	0
Non-Transfer Cycle Master Idle Count	529331033
System Efficiency	26.04 %
System Efficiency (During user access)	38.67 %
Minimum Read Latency	16
Maximum Read Latency	111
Average Read Latency	28

**Note:** To enable efficiency measurements to be performed on the controller Avalon interface through UniPHY External Memory Interface Toolkit, turn on **Enable the Efficiency Monitor and Protocol Checker on the Controller Avalon Interface**.

**Note:** For UniPHY-based designs, the efficiency monitor is not available for QDR II and QDR II+ SRAM interfaces, or for the MAX 10 device family, or for Arria V or Cyclone V designs using the Hard Memory Controller.

**Note:** The efficiency monitor does not take refreshes into account.

The Efficiency Monitor counts the number of cycles of command transfers and wait times for the controller interface and provides an Avalon slave port to allow access to this data. The efficiency monitor has an internal 32-bit counter for accessing transactions; its status can be any of the following:

- Not Running
- Not Running: Waiting for pattern start
- Running
- Not Running: Counter Saturation

For example, once the counter saturates the efficiency monitor stops because it can no longer track transactions. In the summary panel, this appears as Not Running: Counter Saturation.



The debug toolkit summarizes efficiency monitor statistics as follows:

- **Efficiency Monitor Cycle Count** – counts cycles from first command/start until  $2^{32}$  or a stop request
- **Transfer Count** – counts any data transfer cycle, read or write
- **Write Count** – counts how many writes requested, including those during a burst
- **Read Count** – counts how many reads requested (just commands)
- **Read Burst counter** – counts how many reads requested (total burst requests)
- **Non-Transfer Cycle Waitrequest Count** – counts Non Transfer Cycles due to slave wait request high. A Non-Transfer Cycle is a cycle during which no read data is received and no write data is accepted on the Avalon interface.
- **Non-Transfer Cycle No readdatavalid Count** – counts Non Transfer Cycles due to slave not having read data
- **Non-Transfer Cycle Master Write Idle Count** – counts Non Transfer Cycles due master not issuing command or pause in write burst
- **Non-Transfer Cycle Master Idle Count** – counts Non Transfer Cycles due master not issuing command anytime
- **System Efficiency** – The percentage of all Avalon-MM cycles where the interface is transferring data. Refreshes and idle time are not taken into consideration when calculating efficiency.
- **System Efficiency (During user access)** – Tracks the efficiency when transactions are occurring, which is a reflection on waitrequest. It is defined as: Transfer Count/(Efficiency Monitor Cycle Count - Non-Transfer Cycle Master Idle Count)
- **Minimum Read Latency** – The lowest of all read latencies, which is measured by time between a read command is being accepted by the Controller till the first beat of read data is presented to the driver.
- **Maximum Read Latency** – The highest of all read latencies, which is measured by time between a read command is being accepted by the Controller till the first beat of read data is presented to the driver.
- **Average Read Latency** – The average of all read latencies, which is measured by time between a read command is being accepted by the Controller till the first beat of read data is presented to the driver.

### 11.3. Document Revision History

Date	Version	Changes
May 2017	2017.5.08	Rebranded as Intel.
October 2016	2016.10.31	Added <i>Back-to-Back User-Controlled Refresh Usage in Arria 10</i> topic.
May 2016	2016.05.02	Added note about efficiency monitor protocol and device support, to <i>Efficiency Monitor</i> .
November 2015	2015.11.02	Maintenance release.
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	Removed occurrence of MegaWizard Plug-In Manager.

*continued...*



Date	Version	Changes
December 2013	2013.12.16	<ul style="list-style-type: none"><li>Removed references to ALTMEMPHY.</li><li>Changed chapter number from 14 to 13.</li></ul>
November 2012	3.1	Changed chapter number from 13 to 14.
June 2012	3.0	<ul style="list-style-type: none"><li>Expanded <i>Auto-Precharge Commands</i> section.</li><li>Expanded <i>Bank Interleaving</i> section.</li><li>Added <i>Command Queue Look-Ahead Depth</i> section.</li><li>Added <i>Reordering</i> section.</li><li>Added <i>Efficiency Monitor</i> section.</li><li>Added Feedback icon.</li></ul>
November 2011	2.0	Reorganized optimizing the controller information into an individual chapter.
June 2011	1.0	Initial release.

## 12. PHY Considerations

The performance of your external memory interface is affected by several factors, including the data rate at which core logic and the memory interface operate, the PHY block in use, the sequencer in use, shared resources, and pin placement.

The following topics describe design considerations that affect the external memory interface performance and the device resource usage for UniPHY-based designs.

### 12.1. Core Logic and User Interface Data Rate

The clocking operation in the PHY consists of the following two domains:

- PHY-memory domain—the PHY interfaces with the external memory device and is always at full-rate.
- PHY-AFI domain—the PHY interfaces with the memory controller and can either be at full, half or quarter rate of the memory clock depending on your choice of controller and PHY.

For the memory controller to operate at full, half and quarter data rate, the UniPHY IP supports full, half and quarter data rate. The data rate defines the ratio between the frequency of the Altera PHY Interface (AFI) clock and the frequency of the memory device clock.

the following table compares the clock cycles, data bus width and address/command bus width between the full, half, and quarter-rate designs.

**Table 504. Ratio between Clock Cycles, Data Bus Width, and Address/Command Bus Width**

Data Rate	Controller Clock Cycles	Bus Width	
		AFI Data	AFI Address/Command
Full	1	2	1
Half	2	4	2
Quarter	4	8	4

In general, full-rate designs require smaller data and address/command bus width. However, because the core logic runs at a high frequency, full rate designs might have difficulty in closing timing. Consequently, for high frequency memory interface designs, Intel recommends that you use half-rate or quarter-rate UniPHY IP and controllers.

DDR3 SDRAM interfaces can run at much higher frequencies than DDR, DDR2 SDRAM, QDRII, QDRII+ SRAM, and RLDRAM II interfaces. For this reason, Altera High-Performance Controller II and UniPHY IPs do not support full rate designs using the



DDR3 SDRAM interface. However, the DDR3 hard controller in Arria® V devices supports only full rate. Quarter-rate design support is for DDR3 SDRAM interfaces targeting frequencies higher than 667 MHz.

Although it is easier to close timing for half-rate and quarter-rate designs due to the lower frequency required on the core logic, full-rate interfaces offer better efficiency for low burst-length designs. This is because of 1T addressing mode where the address and command signals are asserted for one memory clock cycle. Typically half-rate and quarter-rate designs operate in 2T and 4T mode, respectively, in which the address and command signals in 2T and 4T mode must be asserted for two and four memory clock cycles, respectively. To improve efficiency, the controller can operate in Quasi-1T half-rate and Quasi-2T quarter-rate modes. In Quasi-1T half-rate mode, two commands are issued to the memory on two memory clock cycles. In Quasi-2T quarter-rate mode, two commands are issued to the memory on four memory clock cycles. The controller is constrained to issue a row command on the first clock phase and a column command on the second clock phase, or vice versa. Row commands include activate and precharge commands; column commands include read and write commands.

## 12.2. Hard and Soft Memory PHY

The Arria V and Cyclone® V device families support hard and soft memory interfaces. Hard memory interfaces use the hard memory controllers and hard memory PHY blocks in the devices.

The hard memory PHY is instantiated together with the hard memory controller. In addition to the PHY data path that uses the hard IP blocks in the devices (similar to how the soft PHY is implemented for device families supported by UniPHY), the hard memory PHY also uses the dedicated hardware circuitries in the devices for certain component managers in the sequencer, including the read write (RW) and PHY managers.

In soft memory PHY, the UniPHY sequencer implements the Nios® II processor and all the component managers in the core logic. The hard memory PHY uses dedicated hard IP blocks in the Arria V and Cyclone V devices to implement the RW and PHY managers to save LE resources, and to allow better performance and lower latency.

Each Arria V and Cyclone V device has a fixed number of hard PHYs. Dedicated I/O pins with specific functions for data, strobe, address, command, control, and clock must be used together with each hard PHY.

For the list of hard PHY dedicated pins, refer to the device pin-out files for your target device on the *Pin-Out Files for Intel FPGAs* page of [www.altera.com](http://www.altera.com).

The soft memory PHY gives you the flexibility to choose the pins to be used for the memory interface. Soft memory PHY also supports wider interfaces than hard memory PHY.

### Related Information

[Pin-Out Files for Intel FPGAs](#)

## 12.3. Sequencer

The UniPHY IP soft memory PHY supports the following two types of sequencer used for QDRII and QDRII+ SRAM, and RLDRAM II calibration:

- RTL-based sequencer
- Nios II-based sequencer

The RTL-based sequencer performs FIFO calibration that includes adjusting the valid-prediction FIFO (VFIFO) and latency FIFO (LFIFO) length. In addition to the FIFO calibration, the Nios II-based sequencer also performs I/O calibration that includes adjusting delay chains and phase settings to center-align the data pins with respect to the strobes that sample them. I/O calibration is required for memory interfaces running at higher frequencies to increase read and write margins.

Because the RTL-based sequencer performs a relatively simpler calibration process, it does not require a Nios II processor. For this reason, utilization of resources such as LEs and RAMs is lower than with the Nios II-based sequencer.

For more information about the RTL-based sequencer and Nios II-based sequencer, refer to the *Functional Description—UniPHY* chapter in volume 3 of the *External Memory Interface Handbook*.

For more information about the calibration process, refer to the “UniPHY Calibration Stages” section in the *Functional Description—UniPHY* chapter of the *External Memory Interface Handbook*.

#### Related Information

[Functional Description - UniPHY](#)

### 12.4. PLL, DLL and OCT Resource Sharing

By default, each external memory interface in a device needs one PLL, one DLL, and one OCT control block. The number of PLL, DLL and OCT resources in a device is fixed; however, these resources can be shared by two or more memory interfaces when certain criteria are met. This method allows more memory interfaces to fit into a device and allows the remaining resources to be used for other purposes.

By sharing PLLs, fewer PLLs are used, and the number of clock networks and clock input pins required is also reduced. To share PLLs, the memory interfaces must meet the following criteria:

- Run the same memory protocol (for example, DDR3 SDRAM)
- Run at the same frequency
- The controllers or PHYs run at the same rate (for example, half rate)
- Use the same phase requirements (for example, additional core-to-periphery clock phase of 90°)
- The memory interfaces are located on the same side of the device, or adjacent sides of the device if the PLL is able to drive both sides.

Intel devices have up to four DLLs available to perform phase shift on the DQS signal for capturing the read data. The DLLs are located at the device corners and some of the DLLs can access two adjacent sides of the device. To share DLLs, the memory interfaces must meet the following criteria:

- Run at the same frequency
- The memory interfaces are located on the same side of the device, or adjacent sides of the device accessible by the DLL.



Memory interface pins with OCT calibration require the OCT control block to calibrate the OCT resistance value. Depending on the device family, the OCT control block uses either the RUP and RDN, or RZQ pins for OCT calibration. Each OCT control block can only be shared by pins powered by the same VCCIO level. Sharing of the OCT control block by interfaces operating at the same VCCIO level allows other OCT control blocks in the device to support other VCCIO levels. The unused RUP/RDN or RZQ pins can also be used for other purposes. For example, the RUP/RDN pins can be used as DQ or DQS pins. To share an OCT control block, the memory interfaces must operate at the same VCCIO level.

For more information about the resources required for memory interfaces in various device families, refer to the *Planning Pin and FPGA Resources* chapter.

For more information about how to share PLL, DLL and OCT control blocks, refer to the *Functional Description—UniPHY* chapter in volume 3 of the *External Memory Interface Handbook*.

For more information about the DLL, refer to the external memory interface chapters in the respective device handbooks.

For more information about the OCT control block, refer to the I/O features chapters in the respective device handbooks.

#### Related Information

- [Functional Description - UniPHY](#)
- [Planning Pin and FPGA Resources](#) on page 9

### 12.5. Pin Placement Consideration

The Stratix V, Arria V, Cyclone® V, and MAX® 10 device families use the PHY clock (PHYCLK) networks to clock the external memory interface pins for better performance.

Each PHYCLK network is driven by a PLL.

- In Cyclone V and Stratix V devices, the PHYCLK network spans across two I/O banks on the same side of the device.
- For Arria V devices, each PHYCLK network spans across one I/O bank.
- For MAX 10 devices, the PHYCLK network is available only for the I/O banks on the right side of the device.

All pins for a memory interface must be placed on the same side of the device.

For more information about pin placement guidelines related to the PHYCLK network, refer to the following documents:

- *External Memory Interfaces in Stratix V Devices* chapter in volume 2 of the *Stratix V Device Handbook*
- *External Memory Interfaces in Arria V Devices* chapter in volume 2 of the *Arria V Device Handbook*
- *External Memory Interfaces in Cyclone V Devices* chapter in volume 2 of the *Cyclone V Device Handbook*
- *MAX 10 External Memory Interface Architecture and Features* chapter of the *MAX 10 External Memory Interface User Guide*

Certain device families do not use the PHYCLK network to allow greater flexibility in pin placement. These device families support the following interface types:

- Wraparound interfaces, in which data pins from a memory interface are placed on two adjacent sides of a device.
- Split interfaces, in which data pins are placed on two opposite I/O banks.

The x36 emulated mode is supported in certain device families that do not use the PHY clock network for QDRII and QDRII+ SRAM x36 interfaces. In x36 emulated mode, two x18 DQS groups or four x9 DQS groups can be combined to form a 36-bit wide write data bus. Also, two x18 DQS groups can be combined to form a 36-bit wide read data bus. This method allows a device to support x36 QDRII and QDRII+ SRAM interfaces even if the device does not have the required number of x36 DQS groups.

Some device families might support wraparound or x36 emulated mode interfaces at slightly lower frequencies.

For information about the devices that support wraparound and x36 emulated mode interfaces, and the supported frequency for your design, refer to the External Memory Interface Spec Estimator page on [www.altera.com](http://www.altera.com).

For more information about x36 emulated mode support for QDRII and QDRII+ SRAM interfaces, refer to the *Planning Pin and FPGA Resources* chapter.

#### Related Information

- [Planning Pin and FPGA Resources](#) on page 9
- [External Memory Interfaces in Stratix V Devices](#)
- [External Memory Interfaces in Arria V Devices](#)
- [External Memory Interfaces in Cyclone V Devices](#)
- [MAX 10 PHY Clock \(PHYCLK\) Network, MAX 10 External Memory Interface Architecture](#)
- [External Memory Interface Spec Estimator](#)

## 12.6. Document Revision History

Date	Version	Changes
May 2017	2017.5.08	Rebranded as Intel.
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.01	Maintenance release.
November 2015	2015.11.02	Maintenance release.
May 2015	2015.05.04	Updated the <i>Pin Placement Consideration</i> topic to include MAX 10 devices.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	Maintenance release.
December 2013	2013.12.16	Changed chapter number from 15 to 14.
November 2012	3.1	Changed chapter number from 14 to 15.
June 2012	3.0	Added Feedback icon.
November 2011	1.0	Initial release.

## 13. Power Estimation Methods for External Memory Interfaces

The following table lists the supported power estimation methods for external memory interfaces.

**Table 505. Power Estimation Methods for External Memory Interfaces**

Method	Vector Source	UniPHY Support	Accuracy	Estimation Time <sup>(1)</sup>
Early power estimator (EPE)	Not applicable	v	Lowest	Fastest
Vector-less power analysis (PPPA)	Not applicable	v		
Vector-based power analysis	RTL simulation	v		
	Zero-delay simulation <sup>(2)</sup>	v		
	Timing simulation	<sup>(2)</sup>	Highest	Slowest

Notes to Table:

1. To decrease the estimation time, you can skip power estimation during calibration. Power consumption during calibration is typically equivalent to power consumption during user mode.
2. Power analysis using timing simulation vectors is not supported.

When using Intel FPGA IP, you can use the zero-delay simulation method to analyze the power required for the external memory interface. Zero-delay simulation is as accurate as timing simulation for 95% designs (designs with no glitching). For a design with glitching, power may be under estimated.

For more information about zero-delay simulation, refer to the *Power Estimation and Analysis* section in the *Quartus® Prime Handbook*.

**Note:** The size of the vector file (**.vcd**) generated by zero-delay simulation of an DDR3 SDRAM High-Performance Controller example design is 400 GB. The **.vcd** includes calibration and user mode activities. When vector generation of calibration phase is skipped, the vector size decreases to 1 GB.



### Related Information

Quartus Prime Handbook

## 13.1. Performing Vector-Based Power Analysis with the Power Analyzer

To perform vector-based power analysis with the Power Analyzer using zero-delay simulation, follow these steps:

1. Compile your design in the Quartus Prime software to generate a Netlist `<project_name>.vo` file for your design.  
*Note:* The `<project_name>.vo` is generated in the last stage of a compile EDA Netlist Writer.
2. Open the `<project_name>.vo` file in a text editor.
3. In the `<project_name>.vo` file, locate the include statement for `<project_name>.sdo`, and comment-out that include statement. Save the `<project_name>.vo` file.
4. Create a simulation script containing device model files and libraries and design specific files:
  - Netlist file for the design, `<project_name>.vo`
  - RTL or netlist file for the memory device
  - Testbench RTL file
5. Compile all the files.
6. Invoke the simulator with commands to generate `.vcd` files.
7. Generate `.vcd` files for the parts of the design that contribute the most to power dissipation.
8. Run simulation.
9. Use the generated `.vcd` files in the Power Analyzer as the signal activity input file.
10. Run the Power Analyzer.

*Note:* For more information about estimating power, refer to the *Power Estimation and Analysis* section in the *Quartus Prime Handbook*.

### Related Information

Quartus Prime Handbook

## 13.2. Document Revision History

Date	Version	Changes
May 2017	2017.5.08	Rebranded as Intel.
October 2016	2016.10.31	Maintenance release.
May 2016	2016.05.02	Maintenance release.
November 2015	2015.11.02	Changed instances of <i>Quartus II</i> to <i>Quartus Prime</i> .

*continued...*



Date	Version	Changes
May 2015	2015.05.04	Maintenance release.
December 2014	2014.12.15	Maintenance release.
August 2014	2014.08.15	Maintenance release.
December 2013	2013.12.16	<ul style="list-style-type: none"><li>• Removed references to ALTMEMPHY.</li><li>• Changed chapter number from 16 to 15.</li></ul>
November 2012	2.2	Changed chapter number from 15 to 16.
June 2012	2.1	Added Feedback icon.
November 2011	2.0	Reorganized power estimation methods section into an individual chapter.
April 2010	1.0	Initial release.