ECP5 and ECP5-5G High-Speed I/O Interface

Technical Note

FPGA-TN-02035-1.2

May 2019

Disclaimers

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

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
CLKDIV	Edge Clock Dividers
DDR	Double Data Rate
DLL	Delay-Locked Loops
DM	Data Mask
DSP	Digital Signal Processing
IDDR	Input DDR
ECLK	Edge Clock
ODDR	Output DDR
PCB	Printed Circuit Board
PCLK	Primary Clock
PIO	Programmable I/O
SDR	Single Data Rate
SSN	Simultaneous Switching Noise

1. Introduction

ECP5™ and ECP5-5G™ devices support high-speed I/O interfaces, including Double Data Rate (DDR) and Single Data Rate (SDR) interfaces, using the logic built into the Programmable I/O (PIO). SDR applications capture data on one edge of a clock while DDR interfaces capture data on both the rising and falling edges of the clock, thus doubling the performance. ECP5 and ECP5-5G device I/O also have dedicated circuitry that is used along with the DDR I/O to support DDR2, DDR3, DDR3L, LPDDR2, and LPDDR3 SDRAM memory interfaces.

This document discusses how to utilize the capabilities of the ECP5 and ECP5-5G devices to implement highspeed generic DDR interface and the DDR memory interfaces. Refer to the Implementing DDR Memory Interfaces section of this document for more information.

2. External Interface Description

This technical note uses two types of external interface definitions, centered and aligned. A centered external interface means that, at the device pins, the clock is centered in the data opening. An aligned external interface means that, at the device pins, the clock and data transition are aligned. This is also sometimes called edge-on-edge.

Figure 2.1 shows the external interface waveform for SDR and DDR.

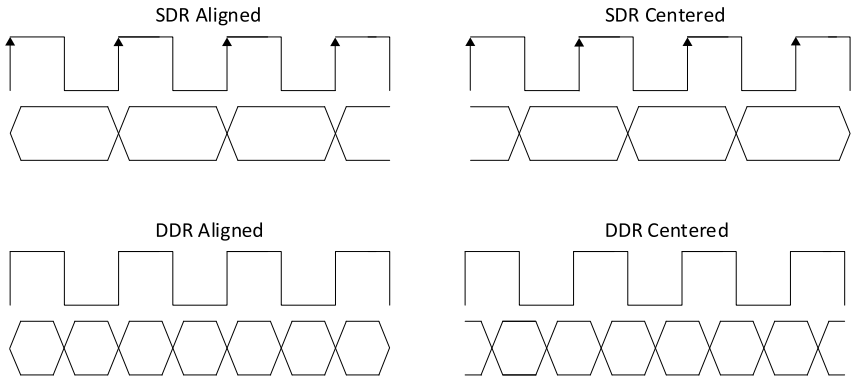

Figure 2.1. External Interface Definitions

The interfaces described are referenced as centered or aligned interfaces. An aligned interface needs to adjust the clock location to satisfy the capture flip-flop setup and hold times. A centered interface needs to balance the clock and data delay to the first flip-flop to maintain the setup and hold already provided.

3. High-Speed I/O Interface Building Blocks

ECP5 and ECP5-5G devices contain dedicated functions for building high-speed interfaces. This section describes when and how to use these functions. A complete description of the library elements, including descriptions and attributes, is provided at the end of this document.

Figure 3.1 shows a high-level diagram of the clocking resources available in the ECP5 and ECP5-5G devices for building high-speed I/O interfaces.

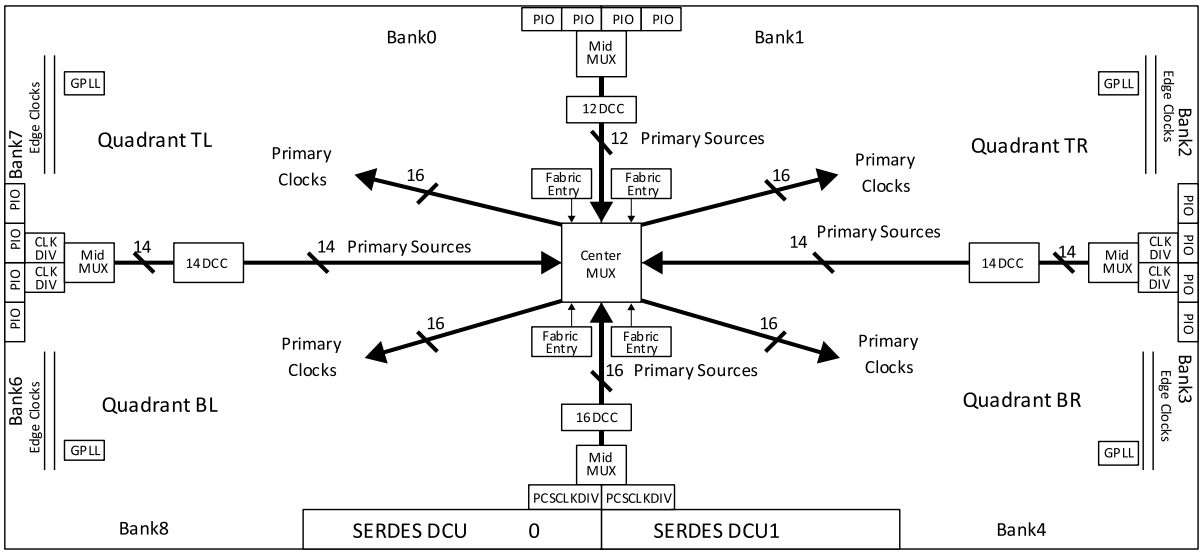

Figure 3.1. ECP5 and ECP5-5G Device Clocking Diagram

A complete description of the ECP5 and ECP5-5G device family clocking resources and clock routing restrictions are available in ECP5 and ECP5-5G sysClock PLL/DLL Design and Usage Guide (TN1263).

Below is a brief description of each of the major elements used for building various high-speed interfaces. The DDR Software Primitives and Attributes section of this document describes the library elements for these components.

3.1. Edge Clocks

Edge Clocks (ECLK) are high-speed, low-skew I/O dedicated clocks. They are arranged in groups of two per I/O bank on the left and right sides of the device. Each of these Edge Clocks can be used to implement a high speed interface. There is an Edge Clock Bridge (ECLKBRIDGECS) that allows you to build large interfaces by bridging the Edge Clocks from one bank to the other on the same side or from one side to the other side.

3.2. Primary Clocks

Primary Clocks (PCLK) refer to the system clock of the design. The SCLK ports of the DDR primitives are connected to the system clock of the design.

3.3. DQS Lane

A DQS Lane uses the embedded circuit for memory interfaces. Each DQS Lane provides a clock pair (DQSP and DQSN) for the DQS strobe and up to 12 to 16 ports for DQ data and DM data mask signals. The number of DQS Lanes on the device is different for each device size. ECP5 and ECP5-5G devices support DQS lanes on the left and right sides of the device.

3.4. PLL

The PLL provides frequency synthesis, with additional static and dynamic phase adjustment, as well. Four output ports are provided, CLKOP, CLKOS, CLKOS2, and CLKOS3. All four outputs have the same set of dividers. There is one PLL per corner on the biggest device, totaling to four PLLs on each device.

3.5. DDRDLL

The DDRDLL is a dedicated DLL for creating the 90° clock delay. The DDRDLL outputs delay codes that are used in the DQSBUF elements to delay the DQS input or in the DLLDEL module to delay the input clock. There is one DDRDLL at each corner of the device, totaling to four DDRDLLs on each device. The DDRDLL on the top corners of the device can drive delay codes to two adjacent edges of the device, providing a possible two DDRDLL codes for an edge.

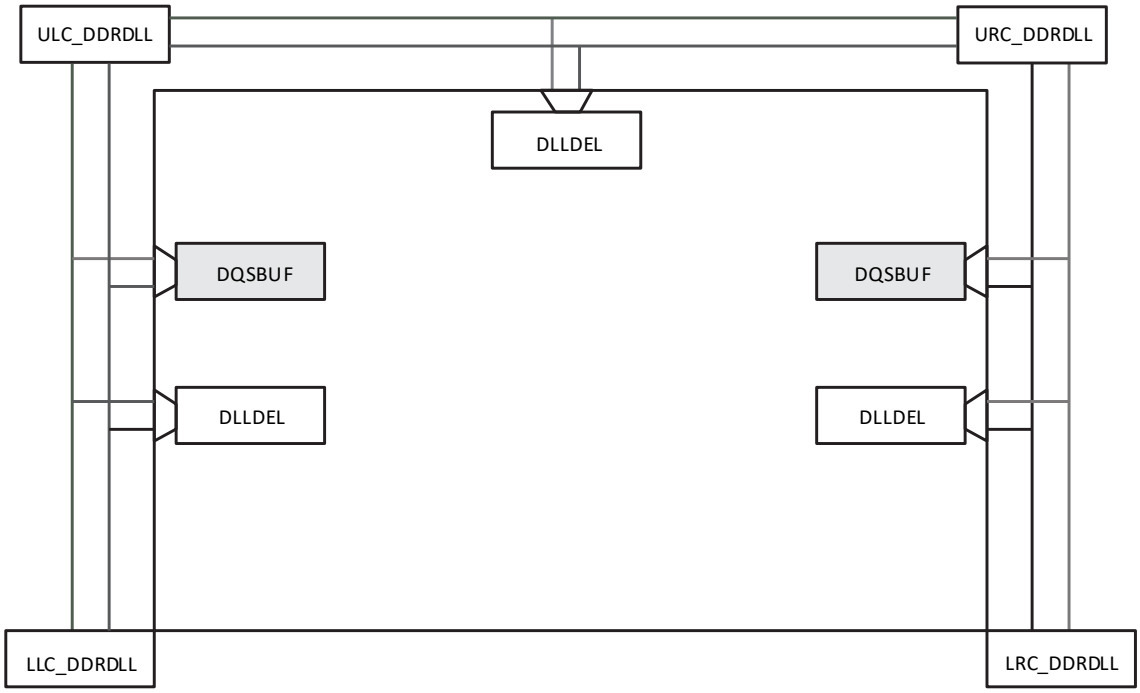

Figure 3.2. DDRDLL Connectivity

3.6. DQSBUF

There is one DQSBUF for each DQS lane (every 12 to 16 I/O depending on the selected device). The DQS input is used when interfacing to DDR memories. It generates the delay on the DQS pin of the DQS lane, to provide a 90o phase shift on DQS to clock the DDR data at the center. The delay is set by a delay code generated in the DDRDLL component. Each DQSBUF can receive delay codes from two different DDRDLs hence two different DDR memory interfaces can be built on one side of the device.

Each of the DQSBUF modes has an additional feature that allows you to adjust the delay from the delay set by the DDRDLL code, by using the MOVE and DIRECTION inputs controlled by the user logic. The LOADN resets the delay back to the DDRDLL code.

3.7. DLLDEL

DLLDEL provides phase shift on the receive side clocks to each ECLK. It functions similar to the DQSBUF, shifting the clock input by delay set by the DDRDLL delay code, before the clock drives the clock tree. The DLLDEL element has the ability to further adjust the delay from the delay set by the DDRDLL code, by using the MOVE and DIRECTION inputs controlled by the user logic. The LOADN resets the delay back to the DDRDLL code.

3.8. Input DDR (IDDR)

The input DDR function can be used in either 1x (2:1), 2x (4:1), or 7:1 gearing modes. In the 1x mode, the IDDR module inputs a single DDR data input and SCLK (primary clock) and provides a 2-bit wide data synchronized to the SCLK (primary clock) to the FPGA fabric.

The 2x gearing is used for interfaces with data rate higher than 400Mbps which would require higher than 200 MHz system clock. There the IDDR element inputs a single DDR data input and DQS clock (for DDR memory interface) or ECLK (for all other high speed interfaces) and provides a 4-bit wide parallel data synchronized to SCLK (primary clock) to the FPGA fabric.

In the 7:1 mode, mostly used in video applications required 7:1 interface, the IDDR element inputs a single DDR data input and ECLK and output a 7-bit wide parallel data synchronized to SCLK (primary clock) to the FPGA fabric.

3.9. Output DDR (ODDR)

The output DDR function can also be supported in 1x (2:1), 2x (4:1), or 7:1 gearing modes. In the 1x mode, the ODDR element receives 2-bit wide data from the FPGA fabric and generates a single DDR data output and Clock output.

Similar to input interfaces the 2x gearing is used for data rate higher than 400 Mbps which would require higher than 200 MHz system clock. Here the ODDR element receives 4-bit wide data from the FPGA fabric and generates a single DDR data output and Clock output. The 2x element uses high speed edge clock (ECLK) to clock the data out for generic high speed interfaces and DQS clock for DDR memory interfaces.

In 7:1 mode, the ODDR element receives 7-bit wide data from FPGA fabric and generates a single DDR data output and Clock output. The 7:1 element sends out data using high speed edge clock.

3.10. Edge Clock Dividers (CLKDIV)

Clock dividers are provided to create the divided down clocks used with the I/O Mux/DeMux gearing logic (SCLK inputs to the DDR) and drives to the Primary Clock routing to the fabric. There are two clock dividers on each side of the device.

3.11. Input/Output DELAY

There are two different types of input/output data delay available. Both DELAYF and DELAYG provide a fixed value of delay to compensate for clock injection delay. The DELAYF element also allows you to set the delay value using 128 steps of delay. Each delay step generates ~25 ps of delay. In DELAYF, you can overwrite the DELAY setting dynamically using the MOVE and DIRECTION control inputs. The LOADN resets the delay back to the default value.

4. Building Generic High Speed Interfaces

This section describes in detail on how the high speed interfaces that can be built using the building blocks described in the section above. The Clarity Designer tool in Lattice Diamond design software builds these interfaces based on external interface requirements.

4.1. Types of High-Speed DDR Interfaces

This section describes the different types of high-speed DDR interfaces available in ECP5 and ECP5-5G devices.

Table 4.1 lists these interfaces. The naming conventions use for each interface are provided below the table.

Table 4.1. Generic High-Speed I/O DDR Interfaces

Mode	Interface Name	Description
Receive SDR	GIREG_RX.SCLK	SDR Input register using SCLK.
Receive DDRX1 Aligned	GDDRX1_RX.SCLK.Aligned	DDR 1x Input using SCLK. Data is edge-to-edge with incoming clock. DLLDEL is be used to shift the incoming clock.
Receive DDRX1 Centered	GDDRX1_RX.SCLK.Centered	DDR x1 Input using SCLK. Clock is already centered in data window.
Receive DDRX2 Aligned	GDDRX2_RX.ECLK.Aligned	DDR x2 Input using ECLK. Data is edge-to-edge with incoming clock. Generic DDR X2 using Edge Clock. DLLDEL is be used to shift the incoming clock.
Receive DDRX2 Centered	GDDRX2_RX.ECLK.Centered	DDR x2 Input using ECLK. Clock is already centered in data window.
Receive DDRX2 MIPI	GDDRX2_RX.MIPI	DDRx2 Input using ECLK interfaces to MIPI interface. This uses additional IMIPI module for the interface.
Receive DDRX71	GDDRX71_RX.ECLK	DDR 7:1 input using ECLK.
Transmit SDR	GOREG_TX.SCLK	SDR Output using SCLK. Clock is forwarded through ODDR.
TX DDRX1 Aligned	GDDRX1_TX.SCLK.Aligned	DDR x1 Output using SCLK. Data is edge-on-edge using same clock through ODDR.
TX DDRX1 Centered	GDDRX1_TX.SCLK.Centered	DDR x1 Output using SCLK. Clock is centered using PLL with different SCLK.
TX DDRX1 Centered	GDDRX2_TX.ECLK.Aligned	DDR x2 Output that is edge-on-edge using ECLK.
TX DDRX1 Centered	GDDRX2_TX.ECLK.Centered	DDR x2 Output that is pre-centered PLL generated 90o phase, and output on ECLKs.
TX DDRX71	GDDRX71_TX.ECLK	DDR 7:1 output using ECLK. Data and CLK are aligned on first of 7 bits.

Note: The following describes the naming conventions used for each of the interfaces:

- G – Generic
- IREG – SDR Input I/O Register
- OREG – SDR Output I/O Register
- DDRX1 – DDR 1x gearing I/O Register
- DDRX2 – DDR 2x gearing I/O Registers
- _RX – Receive Interface
- _TX – Transmit Interface
- .ECLK – Uses ECLK (Edge Clock) clocking resource
- .SCLK – Uses SCLK (Primary Clock) clocking resource
- .Centered – Clock is centered to the data when coming into the device

5. High-Speed DDR Interface Details

This section describes each of the generic high-speed interfaces in detail, including the clocking to be used for each interface. For detailed information about the ECP5 and ECP5-5G device clocking structure, refer to ECP5 and ECP5-5G sysClock PLL/DLL Design and Usage Guide (TN1263). The various interface rules listed under each interface should be followed to build these interfaces successfully. Refer to the Timing Analysis for High Speed DDR Interfaces section of this document for more information about the timing analysis on these interfaces.

Some of these interfaces may require a soft IP in order utilize all the features available in the hardware. These soft IP cores are available in Clarity Designer and are described in this section. Some of the soft IPs are optional and can be selected in the Clarity Designer. Some of these are mandatory for the module to function as expected and are automatically generated when building the interface through Clarity Designer.

5.1. GDDRX1_RX.SCLK.Centered

This a Generic 1x gearing Receive interface using SCLK. The clock is coming in centered to the Data. This interface must be used for speeds below 250 MHz.

This DDR interface uses the following modules:

- IDDRX1F element to capture the data
- The incoming clock is routed through the Primary (SCLK) clock tree
- Static data delay element DELAYG is used to delay the incoming data enough to remove the clock injection time.
- Optionally, you can choose to use Dynamic Data delay adjustment using DELAYF element to control the delay on the DATA dynamically. DELAYF also allows you to override the input delay set. The type of delay required can be selected through Clarity Designer.
- DEL_MODE attribute is used with DELAYG and DELAYF element to indicate the interface type so that the correct delay value can be set in the delay element.

The following figures show the static delay and dynamic delay options for this interface.

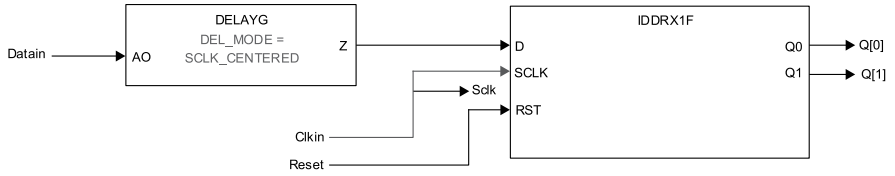

Figure 5.1. GDDRX1_RX.SCLK.Centered Interface (Static Delay)

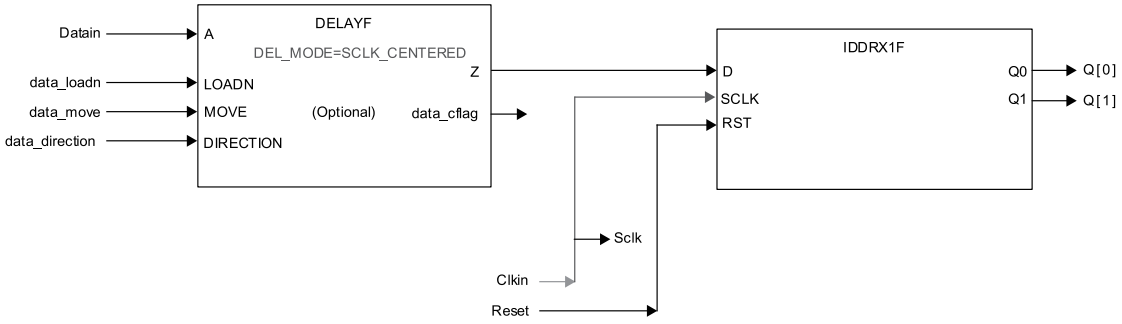

Figure 5.2. GDDRX1_RX.SCLK.Centered Interface (Dynamic Data delay)

Interface Requirements

- The clock input must use a PCLK input so that it can be routed directly to the primary clock tree.
- You must set the timing preferences as indicated in the Timing Analysis for High Speed DDR Interfaces section.

5.2. GDDRX1_RX.SCLK.Aligned

This a Generic 1x gearing Receive interface using SCLK. The clock is coming in edge aligned to the Data. This interface must be used for speeds below 250 MHz.

This DDR interface uses the following modules:

- IDDRX1F element to capture the data
- DDRDLLA/DLLDELD blocks are used to phase shift the incoming clock going to primary clock tree (SCLK).
- Static data delay element DELAYG is used to delay the incoming data enough to remove the clock injection time.
- Optionally, you can choose to use Dynamic Data delay adjustment using DELAYF element to control the delay on the DATA dynamically. DELAYF also allows you to override the input delay set. The type of delay required can be selected through Clarity Designer.
- DEL_MODE attribute is used with DELAYG and DELAYF element to indicate the interface type so that the correct delay value can be set in the delay element.
- Dynamic Margin adjustment in the DDRDLLA module can be optionally used to adjust the DDRDLLA delay dynamically.

The output of the DLLDELD module is also used as the clock input to the DDRDLLA which sends the delay values to the DLLDELD module. The Receiver Synchronization (RX_SYNC) soft IP is required for the aligned interfaces to prevent stability issues that may occur due to this loop at startup. The soft IP prevents any updates to the DLLDELD at start until the DDRDLLA is locked. Once locked the DLLDELD is updated and FREEZE on the DDRDLL is removed. This soft IP is automatically generated by Clarity Designer.

The following figures show the static delay and dynamic delay options for this interface.

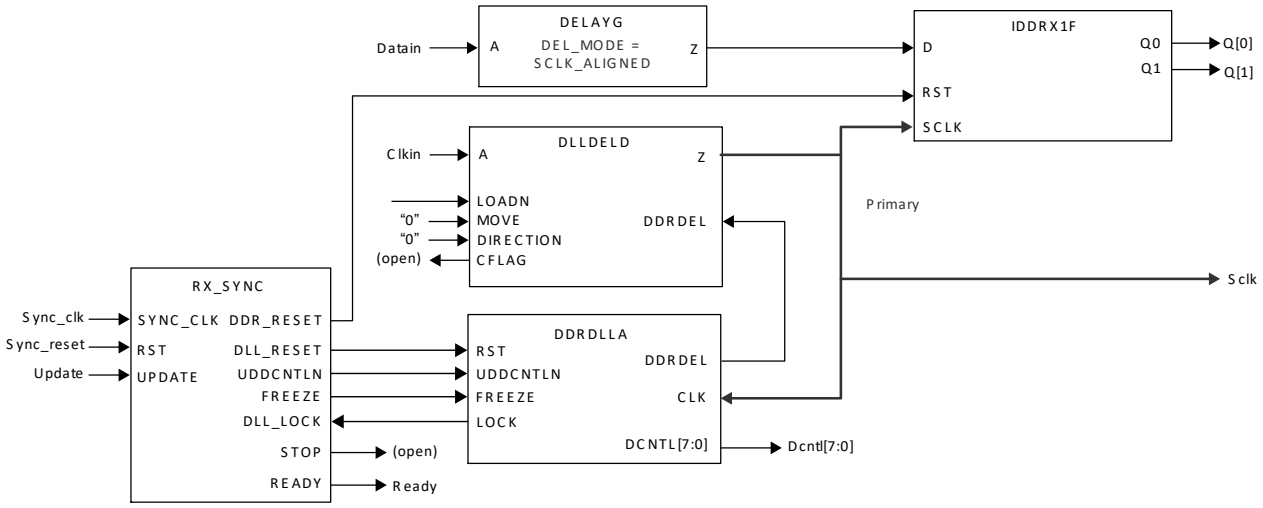

Figure 5.3. GDDRX1_RX.SCLK.Aligned Interface (Static Delay)

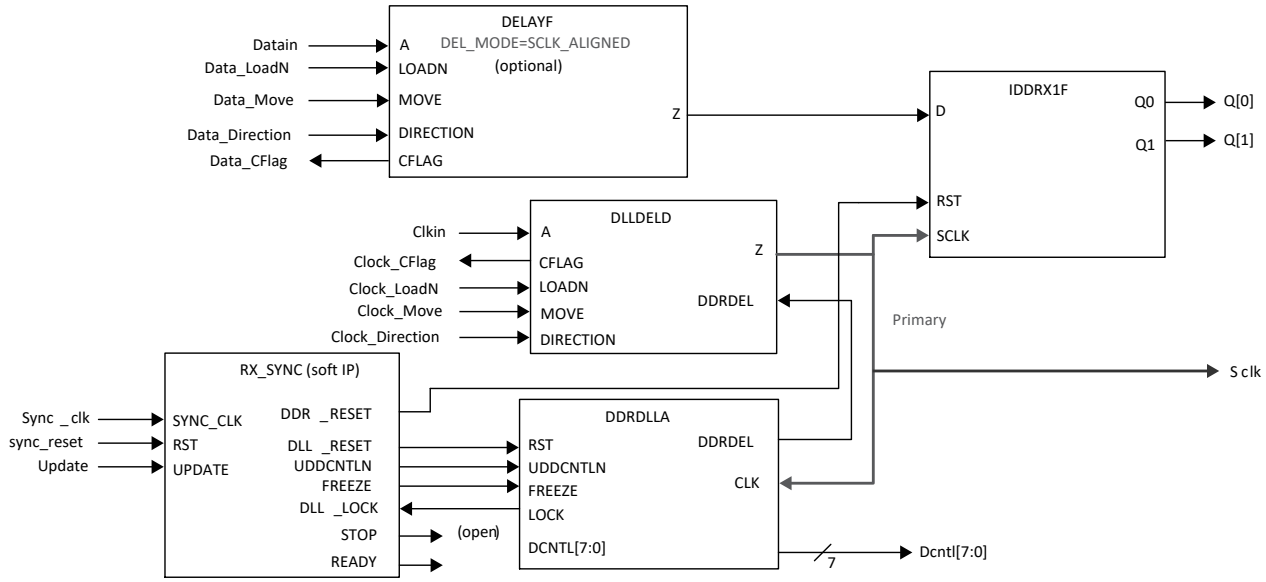

Figure 5.4. GDDRX1_RX.SCLK.Aligned Interface (Dynamic Data/Clock Delay)

Interface Requirements

- The clock input must use a PCLK input so that it can be routed directly to the DLLDELD input
- You must set the timing preferences as indicated in the Timing Analysis for High Speed DDR Interfaces section.

5.3. GDDRX2_RX.ECLK.Centered

Generic Receive DDR with the 2x gearing using Edge Clock Tree (ECLK). Input clock is centered to the input Data. This interface must be used for speeds above 400 MHz.

This DDR interface uses the following modules:

- IDDRX2F element for X2 mode to capture the data
- The incoming clock is routed to the Edge Clock (ECLK) clock tree through the ECLKSYNCB module.
- CLKDIVF module is used to divide the incoming clock by 2 to generate the SCLK.
- Static data delay element DELAYG to delay the incoming data enough to remove the clock injection time
- Optionally, you can choose to use Dynamic Data delay adjustment using DELAYF element to control the delay on the DATA dynamically. DELAYF also allows you to override the input delay set. The type of delay required can be selected through Clarity Designer.
- DEL_MODE attribute is used with DELAYG and DELAYF element to indicate the interface type so that the correct delay value can be set in the delay element.
- The ECLKBRIDGE can be optionally enabled if the data bus is crossing over between the left and right sides of the device. If ECLKBRIDGE is enabled then the ECLKBRIDGECS element should be used in the interface before the ECLKSYNCB element. This element can be enabled through Clarity Designer.
- The startup synchronization soft IP (GDDRX_SYNC) is required for this interface to tolerate the skew between the ECLKSYNCB Stop input and the Reset to the DDR and CLKDIV modules.

The following figures show the static delay and dynamic delay options for this interface.

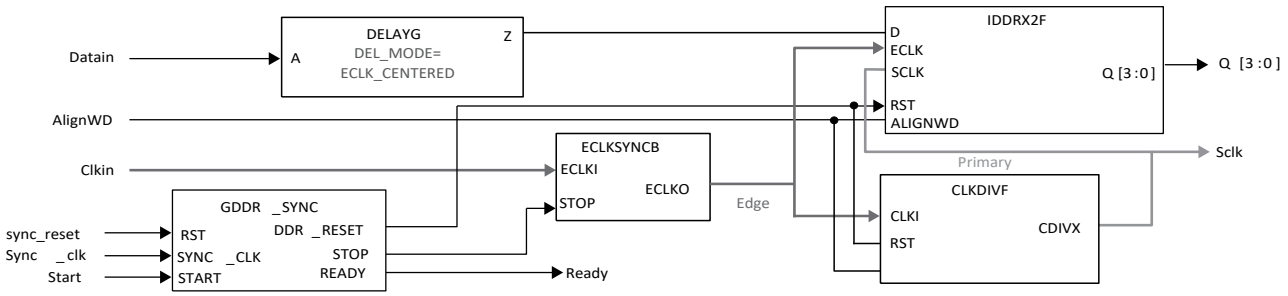

Figure 5.5. GDDRX2_RX.ECLK.Centered Interface (Static Delay)

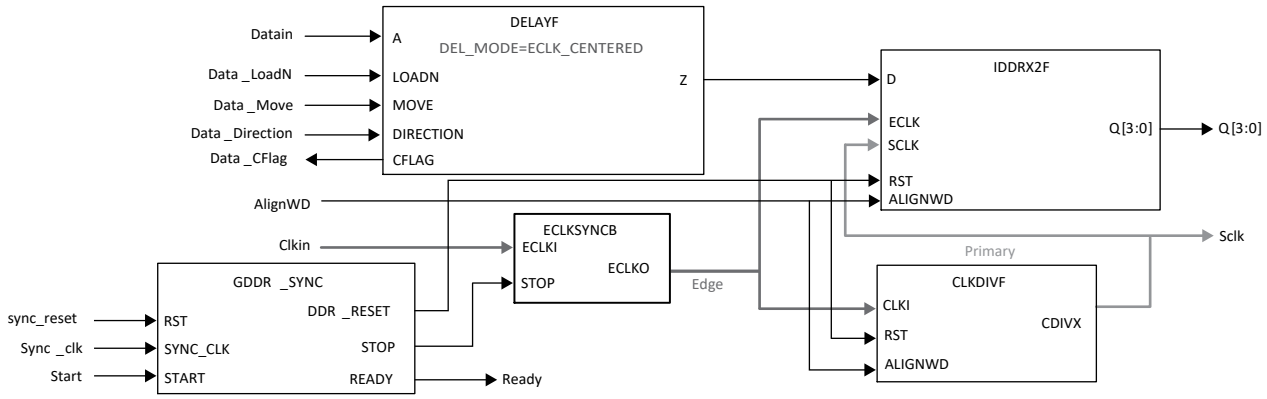

Figure 5.6. GDDRX2_RX.ECLK.Centered Interface (Dynamic Data Delay)

Interface Requirements

- The clock input must use a PCLK input so that it can be routed directly to the Edge Clock tree.
- ECLK must use the Edge Clock tree and the SCLK out of the CLKDIVF must use the Primary Clock tree, software errors out if these dedicated clock routes are not used.
- USE PRIMARY preference may be assigned to the SCLK net.
- You must set the timing preferences as indicated in the Timing Analysis for High Speed DDR Interfaces section.

5.4. GDDRX2_RX.ECLK.Aligned

Generic Receive DDR with the 2x gearing with ECLK. Input Clock is coming in edge aligned to the Data. This interface must be used for speeds above 400 MHz.

This DDR interface uses the following modules:

- IDDRX2F element for 2x mode to capture the data
- DDRDLLA/DLLDELD blocks are used to phase shift the incoming clock routed to the Edge Clock (ECLK) clock tree through the ECLKSYNCB module.
- CLKDIVF module is used to divide the incoming clock by 2.
- Static data delay element DELAYG to delay the incoming data enough to remove the clock injection time
- Optionally, you can choose to use Dynamic Data delay adjustment using DELAYF element to control the delay on the DATA dynamically. DELAYF also allows you to override the input delay set. The type of delay required can be selected through Clarity Designer.
- DEL_MODE attribute is used with DELAYG and DELAYF element to indicate the interface type so that the correct delay value can be set in the delay element.

- The ECLKBRIDGE can be optionally enabled if the data bus is crossing over between the left and right sides of the device. If ECLKBRIDGE is enabled then the ECLKBRIDGECS element should be used in the interface before the ECLKSYNCB element. This element can be enabled through Clarity Designer.
- Dynamic Margin adjustment in the DDRDLLA module can be optionally used to adjust the DDRDLLA delay dynamically.

The output of the DLLDELD module is also used as the clock input to the DDRDLLA which sends the delay values to the DLLDELD module. The Receiver Synchronization (RX_SYNC) soft IP is required for the aligned interfaces to prevent stability issues that may occur due to this loop at startup. The soft IP prevents any updates to the DLLDELD at start until the DDRDLLA is locked. Once locked the DLLDELD is updated and FREEZE on the DDRDLL is removed. This soft IP is automatically generated by Clarity Designer.

The following figures show the static delay and dynamic delay options for this interface.

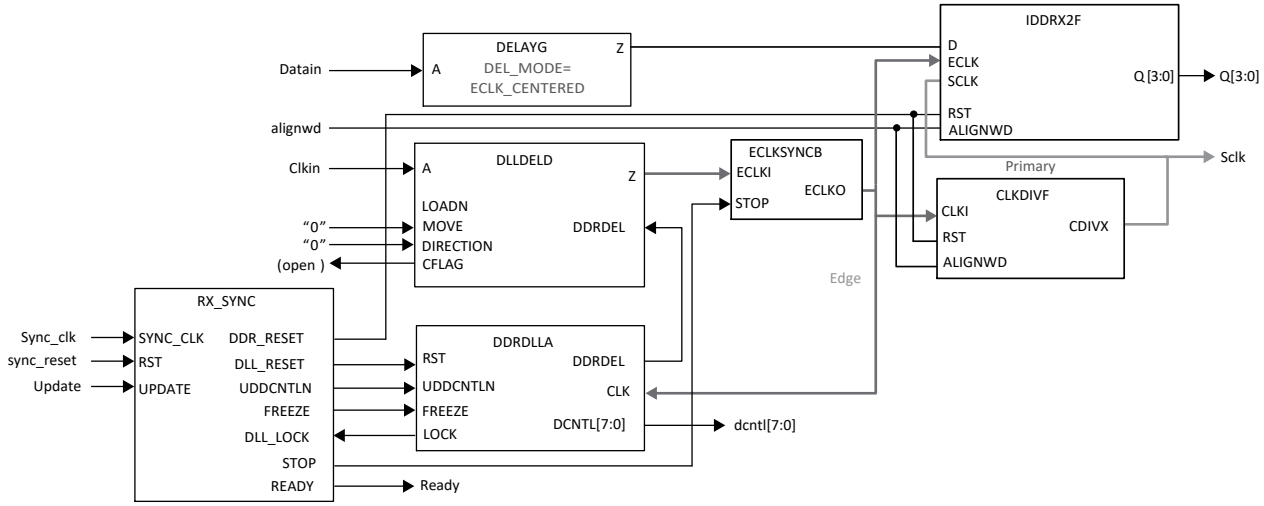

Figure 5.7. GDDRX2_RX.ECLK.Aligned Interface (Static Delay)

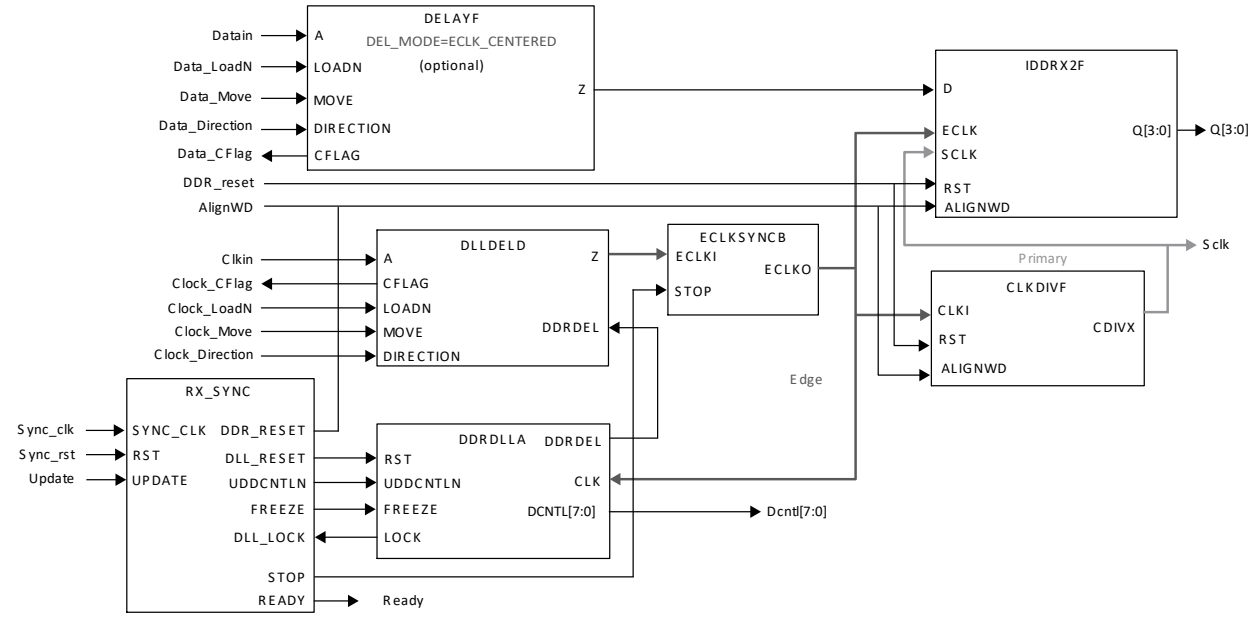

Figure 5.8. GDDRX2_RX.ECLK.Aligned Interface (Dynamic Data/Clock Delay)

Interface Requirements

- The clock input must use a dedicated PCLK input so that it can be routed directly to the DLLDEL module.
- ECLK must use the Edge Cock tree and the SCLK out of the CLKDIVD must use the Primary Clock tree, software errors out if these dedicated clock routes are not used.
- USE PRIMARY preference may be assigned to the SCLK net.
- You must set the timing preferences as indicated in the Timing Analysis for High Speed DDR Interfaces section.

5.5. GDDRX2_RX.MIPI

Generic Receive DDR for MIPI interfaces using the X2 gearing with ECLK. Clock is coming in centered to the Data.

This interface must be used for speeds above 400 MHz.

This DDR interface uses the following modules:

- IMIPI element use to receive the MIPI data and clock
- The HSSEL of the IMIPI is used to switch between the High speed and Low Speed modes.
- The HSSEL of IMIPI should be driven by a soft IP.
- When in high speed mode
  - The OHSOLS1 of the element is active.
  - The OHSOLS1 of the data IMIPI element is connected to the Data input GDDRX2_RX.ECLK.Centered Interface.
  - The OHSOLS1 of the clock IMIPI is connected to the ECLK input GDDRX2_RX.ECLK.Centered Interface.
  - This is then treated similar to the GDDRX2_RX.ECLK.Centered Interface.
- When in low speed mode:
  - Both the outputs of IMIPI are active since it is not a 2-bit interface.
  - The OHSLS1 is the bit 1 and OLS0 is the bit 0 of the interface.
  - Each of the data input and clock input is connected through a 20 ns filter soft IP to the core.
  - The ECLKBRIDGE can be optionally enabled if the data bus is crossing over between the left and right sides of the device. If ECLKBRIDGE is enabled then the ECLKBRIDGECS element should be used in the interface before the ECLKSYNCB element. This element can be enabled through Clarity Designer.

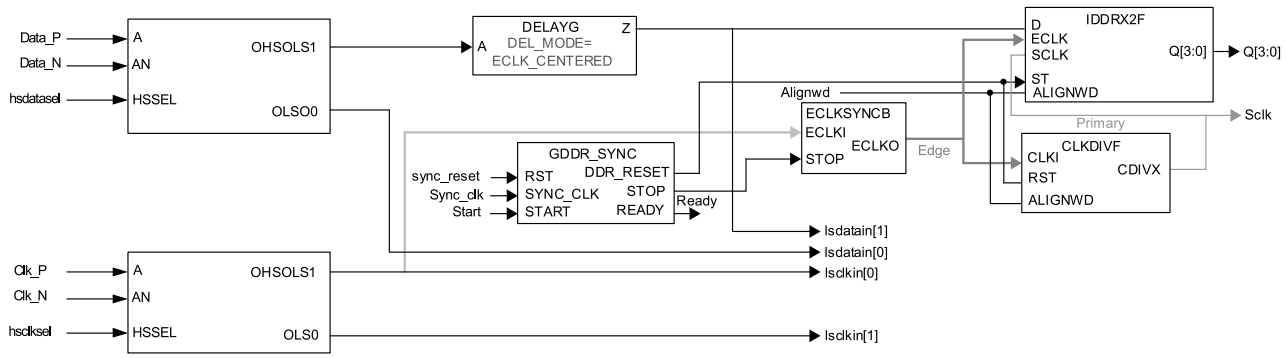

Figure 5.9. GDDRX2_RX.MIPI

Interface Requirements

- The clock input must use a PCLK input so that it can be routed directly to the Edge Clock tree.
- ECLK must use the Edge Clock tree and the SCLK out of the CLKDIVF must use the Primary Clock tree, software errors out if these dedicated clock routes are not used.
- USE PRIMARY preference may be assigned to the SCLK net.
- You must set the timing preferences as per section as indicated in the Timing Analysis for High Speed DDR Interfaces section.

5.6. GDDRX71_RX.ECLK

This interface is used to implement 7:1 LVDS Receiver interface using the 1 to 7 gearing with ECLK. Slow speed clock coming in is multiplied 3.5X using a PLL. This clock is used to capture the data at the receiver IDDRX71 module.

This DDR interface uses the following modules:

- IDDRX71B element is used to capture the data.
- EHXPLLK multiplies the input clock by 3.5 and phase shift the incoming clock based on the dynamic phase shift input.
- This clock is routed to the Edge Clock (ECLK) clock tree through the ECLKSYNCB module.
- CLKDIVF module is used to divide the ECLK by 3.5 and is routed to the primary clock tree used as the SCLK input
- A second IDDRX71B element is used with data connected to clock input to generate 7 bit clock phase that can be used for word alignment.
- The startup synchronization soft IP (GDDRX_SYNC) is required for this interface to tolerate the skew between the ECLKSYNCB Stop input and the Reset to the DDR and CLKDIV modules.
- An optional Bit and Word alignment soft IP(BW_ALIGN) can be enabled in Clarity Designer. The Bit alignment module rotates PLL's 16 phases to center Edge Clock to middle of data eye and the word alignment module uses ALIGNWD function of CLKDIVD and IDDRX71B to achieve 7-bit word alignment.
- The ECLKBRIDGE can be optionally enabled if the data bus is crossing over between the left and right sides of the device. If ECLKBRIDGE is enabled then the ECLKBRIDGECS element should be used in the interface before the ECLKSYNCB element. This element can be enabled through Clarity Designer.

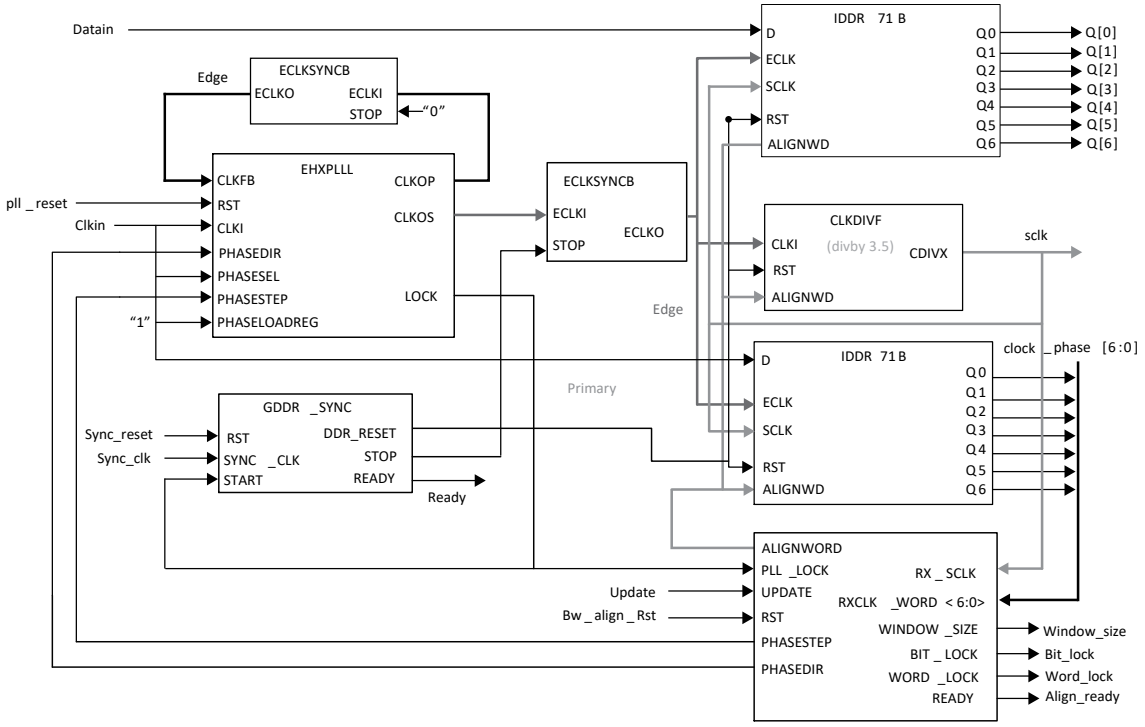

Figure 5.10. GDDRX71_RX.ECLK Interface

Interface Requirements

The clock input must use a dedicated PLL input pin so it is routed directly to the PLL.

- CLKOP output of the PLL must be used as feedback using another Edge Clock tree to compensate for ECLK tree delay used by CLKOS. Hence this interface uses two ECLK trees.
- ECLK must use the Edge Clock tree and the SCLK out of the CLKDIVF must use the Primary Clock tree.
- USE PRIMARY preference may be assigned to the SCLK out of the CLKDIVF module.

5.7. GDDRX1_TX.SCLK.Aligned

This interface is used to implement Generic Transmit DDR with 1x gearing using primary clock (SCLK). The Clock output is aligned to the Data output.

This DDR interface uses the following modules:

- ODDRX1F element is used to generate the data output
- The primary clock (SCLK) is used as the clock for both data and clock generation
- Optionally, you can choose to use the DELAYG or DELAYF element to delay the output data
- The output data can be optionally tristated using either a Tristate input going through an I/O register.

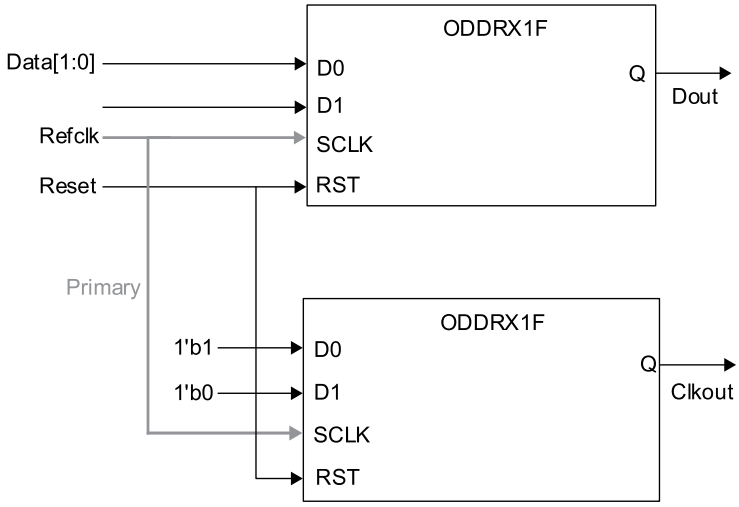

Figure 5.11. GDDRX1_TX.SCLK.Aligned Interface

Interface Requirement

- The clock to the output DDR modules must be routed on the primary clock tree.

5.8. GDDRX1_TX.SCLK.Centered

Generic Transmit DDR using X1 gearing with SCLK. Clock output is centered to the Data output.

This DDR interface uses the following modules:

- ODDRX1F element is used to generate the data output
- The EHXPLLL element is used to generate the clocks for the data and clock ODDRX1F modules. The clock used to generate the clock output is delayed 90 to center to data at the output.
- Both these clocks are routed on primary clock tree
- Optionally, you can choose to use the DELAYG or DELAYF element to delay the output data
- The output data can be optionally tristated using either a Tristate input going through an I/O register.

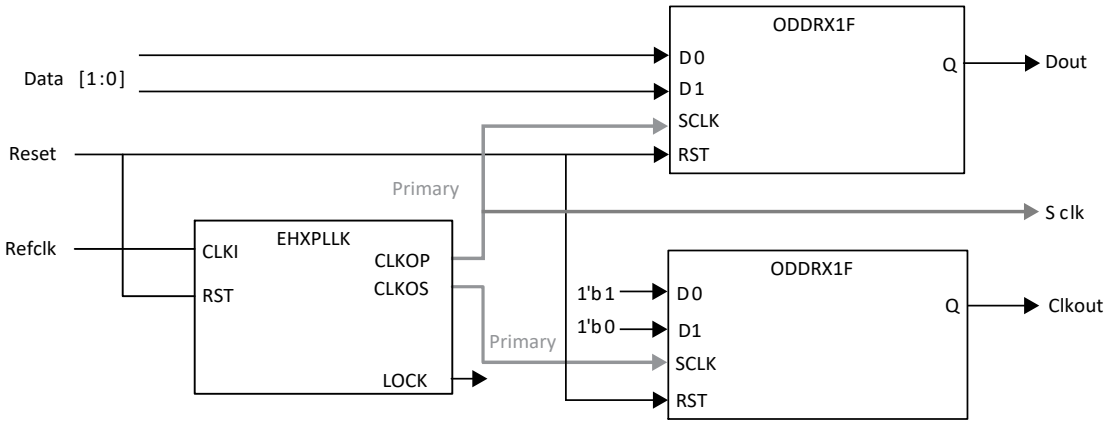

Figure 5.12. GDDRX1_TX.SCLK.Centered Interface

Interface Requirement

- The clock to the output DDR modules must be routed on the primary clock tree

5.9. GDDRX2_TX.ECLK.Aligned

The interface is used to generate Generic Transmit DDR with 2x gearing using high speed Edge Clock (ECLK). The Clock output is edge aligned to the Data output.

This DDR interface uses the following modules:

- ODDRX2Ffor 2x gearing is used to generate the output data.
- The high speed ECLK is routed to the Edge Clock tree through the ECLKSYNCB module.
- The SCLK is routed on the primary clock tree and is generated from the ECLK using the CLKDIVF module.
- The same ECLK and SCLK are used for both Data and Clock generation.
- The startup synchronization soft IP (GDDRX_SYNC) is required for this interface to tolerate the skew between the ECLKSYNCB Stop input and the Reset to the DDR and CLKDIV modules.
- The ECLKBRIDGE can be optionally enabled if the data bus is crossing over between the left and right sides of the device. If ECLKBRIDGE is enabled then the ECLKBRIDGECS element should be used in the interface.
- before the ECLKSYNCB element. This element can be enabled through Clarity Designer.
- Optionally, you can choose to use the DELAYG or DELAYF element to delay the data output.
- The output data can be optionally tristated using either a Tristate input going through an I/O register.

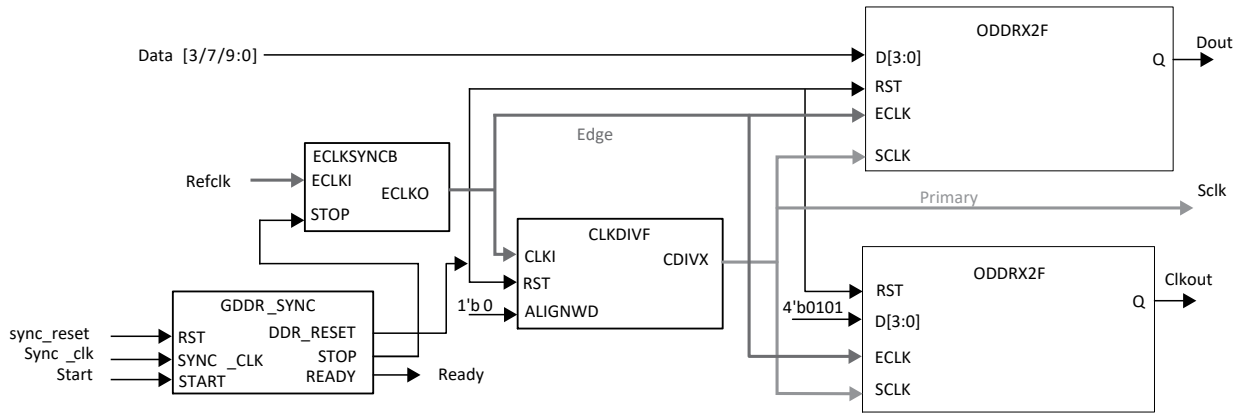

Figure 5.13. GDDRX2_TX.ECLK.Aligned Interface

Interface Requirements

- The SCLK input to the output DDR modules must be routed on the primary clock tree and the ECLK input is routed on the Edge Clock tree.
- USE PRIMARY preference may be assigned to the SCLK net.
- You must set the timing preferences as indicated in the Timing Analysis for High Speed DDR Interfaces section.

5.10.GDDRX2_TX.ECLK.Centered

This interface is used to implement Generic Transmit DDR with 2x gearing using Edge Clock (ECLK). The Clock output is centered to the Data output.

This DDR interface uses the following modules:

- ODDRX2F for X2 gearing is used to generate the data output.
- The high speed ECLK is routed to the Edge Clock tree through the ECLKSYNCB module.
- The SCLK is routed on the primary clock tree and is generated from the ECLK using the CLKDIVF module.
- The same ECLK and SCLK are used for both Data and Clock generation.
- The EHXPLLL element is used to generate the clocks for the data and clock ODDR modules. The clock used to generate the clock output is delayed 90 to center to data at the output.

- The startup synchronization soft IP (GDDRX_SYNC) is required for this interface to tolerate the skew between the ECLKSYNCB Stop input and the Reset to the DDR and CLKDIV modules.
- Optionally, you can choose to use the DELAYG or DELAYF element to delay the data output.
- The output data can be optionally tristated using either a Tristate input going through an I/O register.
- The ECLKBRIDGE can be optionally enabled if the data bus is crossing over between the left and right sides of the device. If ECLKBRIDGE is enabled then the ECLKBRIDGECS element should be used in the interface before the ECLKSYNCB element. This element can be enabled through Clarity Designer.

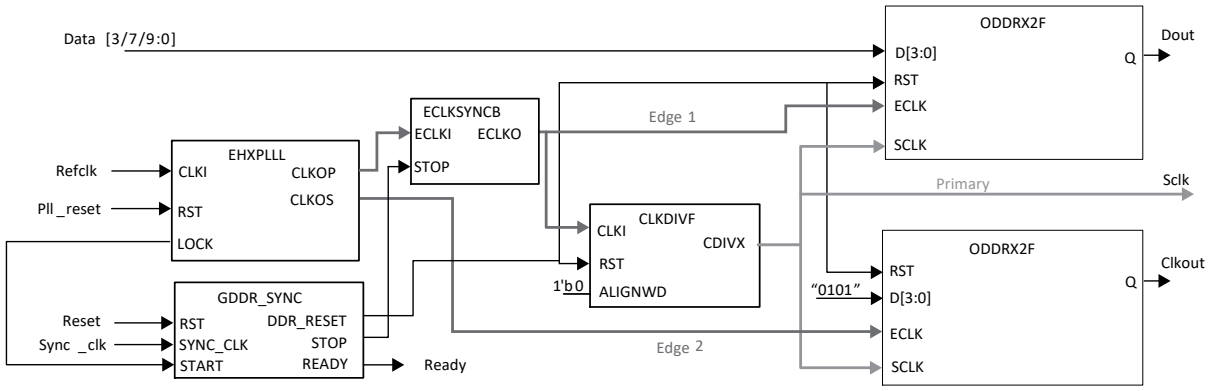

Figure 5.14. GDDRX2_TX.ECLK.Centered Interface

Interface Requirements

- The SCLK input to the output DDR modules must be routed on the primary clock tree and the ECLK input is routed on the Edge Clock tree.
- USE PRIMARY preference may be assigned to the SCLK net.
- You must set the timing preferences as indicated in the Timing Analysis for High Speed DDR Interfaces section.

5.11. GDDRX71_TX.ECLK

This interface is used to implement transmit side of the 7:1 LVDS interface DDR using the 7 to 1 gearing with ECLK. The clock output is aligned to the data output.

This DDR interface uses the following modules:

- ODDRX71B is used to generate the data output.
- The high speed ECLK is routed to the Edge Clock tree through the ECLKSYNCB module.
- The SCLK is routed on the primary clock tree and is generated from the ECLK using the CLKDIVD module.
- The same ECLK and SCLK are used for both Data and Clock generation.
- The startup synchronization soft IP (GDDRX_SYNC) is required for this interface to tolerate the skew between the ECLKSYNCB Stop input and the Reset to the DDR and CLKDIV modules.
- The ECLKBRIDGE can be optionally enabled if the data bus is crossing over between the left and right sides of the device. If ECLKBRIDGE is enabled then the ECLKBRIDGECS element should be used in the interface before the ECLKSYNCB element. This element can be enabled through Clarity Designer.

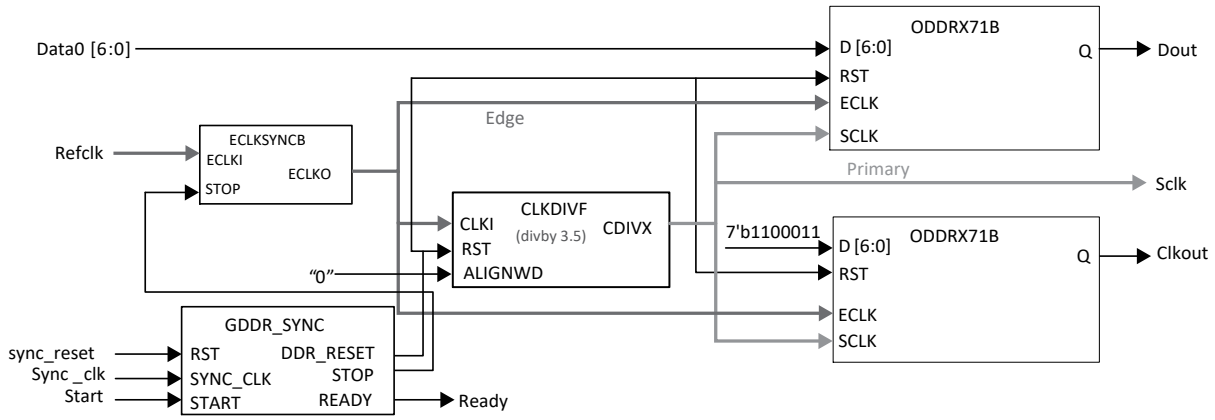

Figure 5.15. GDDRX71_TX.ECLK Interface

Interface Requirements

- The SCLK input to the output DDR modules must be routed on the primary clock tree and the ECLK input is routed on the Edge Clock tree.
- USE PRIMARY preference may be assigned to the SCLK net.
- You must set the timing preferences as indicated in the Timing Analysis for High Speed DDR Interfaces section.

5.12. Generic DDR Design Guidelines

This section describes the various design guidelines used for building generic high speed DDR interfaces in ECP5 and ECP5-5G devices. In additional to these guidelines, it is also required to follow the Interface Rules described for each type of interface, you need to find the interface you are building in the High-Speed DDR Interface Details section.

5.12.1. Using the High Speed Edge Clock Bridge

The High Speed Edge Clock Bridge is available to wide data busses to bridge the Edge Clock from one side to the other. To enable this bridge, instantiate the ECLKBRIDGECS element in the HDL design. When using the ECLKBRIDGE, both the ECLK1 or ECLK0 on that side (spanning both the banks are used). This reduces the number of interfaces that can be built on a given side. See ECP5 and ECP5-5G sysClock PLL/DLL Design and Usage Guide (TN1263) for details.

5.12.2. Receive Interface Guidelines

- Differential DDR interface can be implemented on the Left and Right sides of the device.
- There are 4 different Edge Clocks available per side (two per bank).
- Each of the Edge Clocks can be used to generate either a centered of aligned interface.
- Each side has two CLKDIV modules which would mean you can implement two different GDDRX2 RX interface per side since each 2x gearing would require CLKDIV module to generate a slower SCLK.
- There is DDRDLLA located on each corner of the device, total of 4 in a device. LLC and LRC DDRDLLs only drive code to one side whereas the ULC and URC DDRDLLs drive code to two sides turning corners.
- Each DQSBUF/DLLDEL has access to two DDRDLLs hence two different RX rates are available per side, 4 are available on the entire device.
- The Receive clock input should be placed on a dedicated PCLK input pin. The PCLK pin has direct access to the Edge Clock tree for centered interface and it also has direct connection to the DLLDELD when implementing an aligned interface.
- When implementing IDDRX71 interface, the complementary PAD is not available for other functions since the IDDRX71 used the I/O registers of the complementary PAD as well.
- It is recommend that clock input be located on the same side as data pins.

- The top side of the device does not have Edge Clocks hence can only be used to receive lower speed interfaces (<200 MHz) that use 1x gearing. Top side of the device can be used for single ended interfaces only.
- Interfaces using the x1 gearing uses the primary clock resource. You can use as many interfaces as the number of primary clocks supported in the device.
- In addition to dedicate PCLK pins, ECP5 and ECP5-5G devices have GR_PCLK pins, these use shortest general route path to get to the primary clock tree. These pins are not recommended for use with DDR interfaces. They can be used for SDR or other generic FPGA designs.

5.12.3. Transmit interface Guidelines

- Use PADA and PADB for all TX using true LVDS interfaces.
- When implementing Transmit Centered interface, two ECLKs are required. Once to generate the Data Output and the other to generate the CLK Output.
- When implementing Transmit Aligned interface only one ECLK is required for both Data output and Clock output.
- Each side has two CLKDIV modules which would mean you can implement two different GDDRX2 TX interface per side since each 2x gearing would require CLKDIV module to generate a slower SCLK.
- The top side of the device does not have Edge Clocks hence can only be used to receive lower speed interfaces (<200 MHz) that use 1x gearing. Top side of the device can be used for single ended interfaces only.
- Interfaces using the x1 gearing uses the primary clock resource. You can use as many interfaces as the number of primary clocks supported in the device.

5.12.4. Clocking Guidelines for Generic DDR Interface

- The Edge Clock and Primary Clock resources are used when implementing a 2x receive or transmit interface.
- Only the Primary Clock (PCLK) resources are used when implementing x1 receive or transmit interfaces.
- Each Edge Clock can only span up to one side (Left or Right) of the device, hence all the data bits of the in the x2 interface must be locked to one side of the device. If wide bus implementation is required then the ECLKBRIDGE element must be used to bridge the Edge Clock to the other side. ECLKBRIDGE can be used to bridge the Left and Right Side ECLKs.
- When implementing x1 interfaces, the bus can span Left, Right or Top sides as primary clocks can access DDR registers on all sides.
- Bottom side only supports SERDES function hence does not have any Edge Clocks or DDR registers except on the LFE-85 device, some I/O support 1x DDR registers similar to the Top side.
- The ECLK to DDR registers can be accessed through dedicated PCLK pins, GPLL outputs, DDRDLL outputs. See ECP5 and ECP5-5G sysClock PLL/DLL Design and Usage Guide (TN1263) for details.
- Primary Clock to DDR registers can be accessed through dedicated PCLK pins, GPLL outputs and CLKDIV outputs. See ECP5 and ECP5-5G sysClock PLL/DLL Design and Usage Guide (TN1263) for details.
- None of the clocks going to the DDR registers can come from internal general routing.
- DQS clocking is used for DDR memory interface implementation. DQS clock spans every 12 to 16 I/O's include the DQS pins. Refer to the DQS Grouping section for pinout assignment rules when using DQS clocking.

5.13. Timing Analysis for High Speed DDR Interfaces

It is recommended that you run Static Timing Analysis in the software for each of the high speed interfaces. This section describes the timing preferences to used for each type of interface and the expected trace results. The preferences can either be entered directly in the .lpf file or through the Design Planner graphical user interface.

The External Switching Characteristics section of ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) should be used along with this section. The data sheet specifies the actual values for these constraints for each of the interfaces.

5.13.1. Frequency Constraints

It is required that you explicitly specify FREQUENCY (or PERIOD) PORT preferences to all input clocks in the design. This preference may not be required if the clock is generated out of a PLL or DLL or is input to a PLL or DLL.

5.13.2. DDR Input Setup and Hold Time Constraints

All of the Receive (RX) interfaces, both x1 and x2 can be constrained with setup and hold preference.

5.13.2.1. Receive Centered Interface

Figure 5.16 below shows the Data and Clock relationship for a Receive Centered Interface. The clock is centered to the data, so it comes into the devices with a setup and hold time.

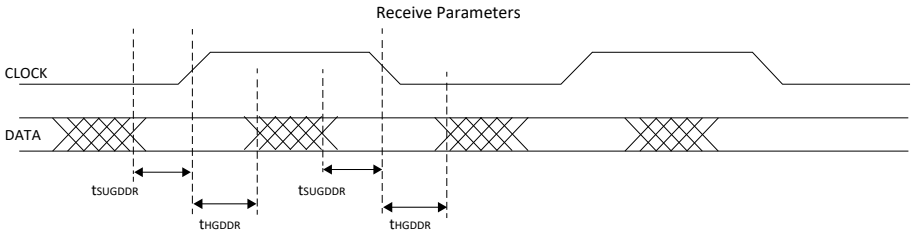

Figure 5.16. RX Centered Interface Timing

Note: tSUGDDR = Setup Time, tHOGDDR = Hold Time

In this case, you must specify in the software preference the amount of setup and hold time available. These parameters are listed in Figure 5.16 as tSU_GDDRX1/2 and tHO_GDDRX1/2. These can be directly provided using the INPUT_SETUP and HOLD preference as –

INPUT_SETUP PORT “DATA” <tSU_GDDRX1/2> ns HOLD <tHO_GDDRX1/2> ns CLKPORT “CLOCK”;

where:

Data = Input Data Port

Clock = Input Clock Port

The external Switching Characteristics section of ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) specifies the MIN setup and hold time required for each of the high speed interfaces running at MAX speed. These values can be picked up from the data sheet if the interface is running at MAX speed.

Example:

For GDDRX2_RX.ECLK.Centered Interface running at max speed of 400 MHz, the preference would be –

INPUT_SETUP PORT "datain" 0.320000 ns HOLD 0.320000 ns CLKPORT "clk”;

Note: Refer to ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) for the latest tSUDDR and tHOGDDR numbers.

5.13.2.2. Receive Aligned Interface

Figure 5.17 below shows the Data and Clock relationship for a Receive Aligned Interface. The clock is aligned edge to edge the data.

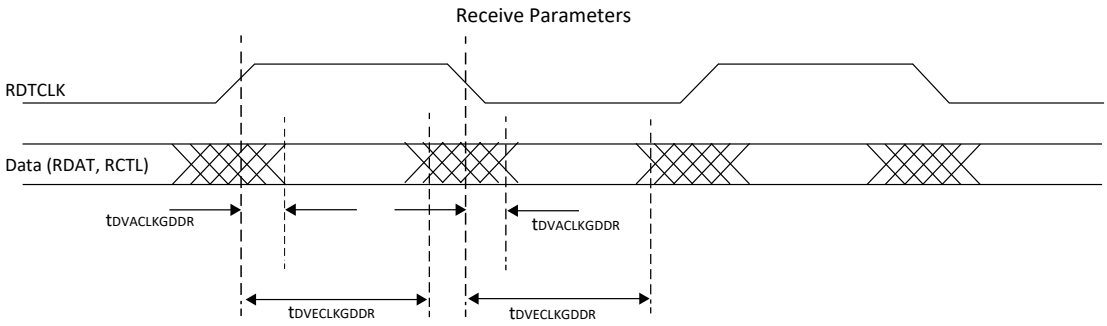

Figure 5.17. RX Aligned Interface Timing

Note: tDVA_GDDRX1/2 = Data Valid after CLK, tDVE_GDDRX1/2 = Data Hold After CLK

In this case, the worst case data may occur after the clock edge hence has a negative setup time when entering the device. In this case, the worst case setup is specified by the tDVACLKGDDR after the clock edge and the worst case hold time is specified as tDVECLKGDDR. For this case the setup and hold time can be specified as –

INPUT_SETUP PORT “din” <-tDVA_GDDRX1/2 > ns HOLD < tDVE_GDDRX1/2> ns CLKPORT “clk”;

Note: Negative number is used for SETUP time as the data occurs after the clock edge in this case.

The External Switching Characteristics section of ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) specifies the MIN tDVA_GDDRX1/2 and tDVE_GDDRX1/2 values required for each of the high speed interfaces running at MAX speed. These values can be picked up from the data sheet if the interface is running at MAX speed. The data sheet numbers for this preference is listed in ns + ½ UI (Unit Interface). 1 UI is equal to ½ the Clock Period. Hence these numbers need to be calculated from the CLK Period used.

Preference Example:

For GDDRX2_RX.ECLK.Aligned interface running at max speed of 400 MHz (UI = 1.25 ns)

tDVA_GDDRX2 = - 0.344 ns + ½ UI = 0.281 ns, tDVE_GDDRX2 = 0.344 ns + ½ UI =0.969 ns

The preference for this case would be –

INPUT_SETUP PORT "datain" -0.2810000 ns HOLD 0.969 ns CLKPORT "clk”;

Note: Please check ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) for the latest tDVA_GDDRX1/X2 and tDVE_GDDRX1/X2 numbers.

5.13.2.3. Receive Dynamic Interfaces

Static Timing Analysis does not show timing for all the Dynamic interfaces cases as either the Clock or the Data delay is dynamically updated at run time.

5.13.3. DDR Clock to Out Constraints for Transmit Interfaces

All of the Transmit (TX) interfaces both x1 and x2 can be constrained with Clock to out constraint to detect the relationship between the Clock and Data when leaving the device.

Figure 5.18 shows how the clock to out is constrained in the software. Min tCO is the minimum time after the clock edge transition that the data does not transition. Max tCO is the maximum time after clock transition before which the data transitions. So any data transition must occur between the tCO Min and tCO Max values.

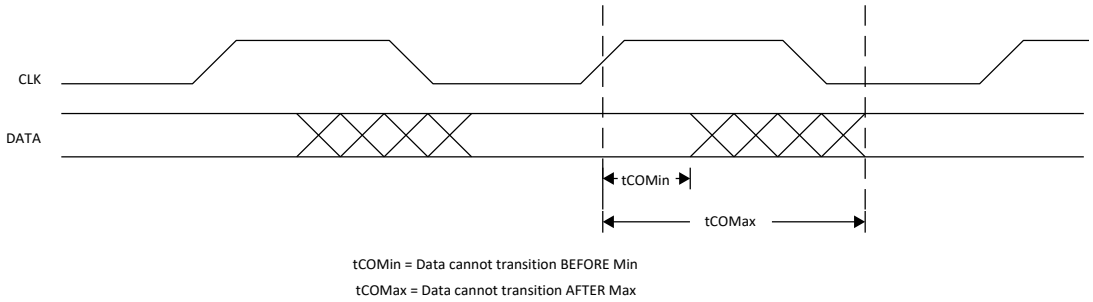

Figure 5.18. tCO Min and Max Timing Analysis

5.13.3.1. Transmit Centered Interfaces

In this case, the transmit clock is expected to be centered to the data when leaving the device. Figure 5.19 shows the timing for a centered transmit interface.

tDVBGDDR = Data valid before clock

tDVAGDDR = Data valid after clock

tU = Data transition

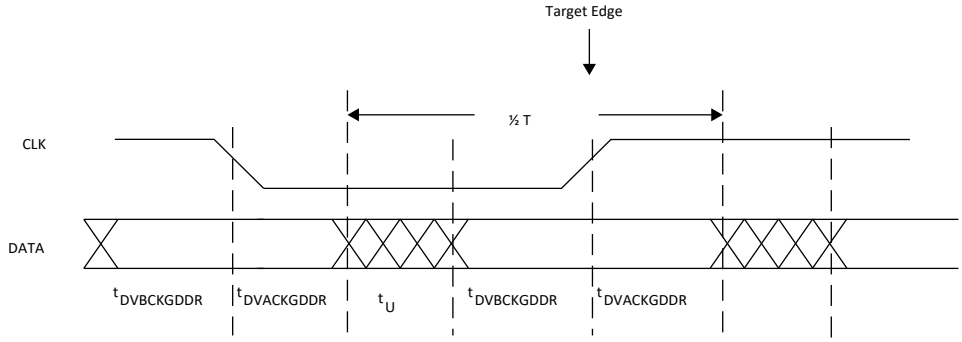

Figure 5.19. Transmit Centered Interface Timing

Figure 5.19 shows that max value after which the data cannot transition is – tVB_GDDR. The min value before which the data cannot transition is – (tU+tVB_GDDR). Negative sign is used because in this particular case where clock is forwarded centered aligned to the data these two conditions occurs before the clock edge.

The ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) specifies the tDVB_GDDRX1/X2 and tDVA_GDDRX1/X2 values at maximum speed. However, we do not have the tU value. Hence, min tCO can be calculated using the following equation.

tCO Min = - (tVB_GDDRX1/X2 + tU)

½ T = tDVA_GDDRX1/X2 + tVB_GDDRX1/X2 + tU

- (tVB_GDDRX1/X2 + tU) = 1/2T - tDVA_GDDRX1/X2

tCO Min = 1/2T - tDVA_GDDRX1/X2

The clock to out time in the software can be specified as –

CLOCK_TO_OUT PORT “dataout” MAX <-tDVB_GDDRX1/X2> MIN <tDVA_GDDRX1/X2 -1/2 Clock Period> CLKPORT “clk” CLKOUT PORT “clkout”;

where:

Data = Data Output Port

Clock = Forwarded Clock Output Port

clk = Input Clock Port

The values for tDVBCKGDDR and tDVACKGDDR can be picked up from the External Switching Characteristics section of ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) for the MAX speed.

Preference Example:

For GDDRX1_TX.SCLK.Centered interface running at 250 MHz, tDVB_GDDRX1 = tDVA_GDDRX1 = 0.67 ns, the preference would be –

CLOCK_TO_OUT PORT "dataout" MAX -0.670000 ns MIN -1.330000 ns CLKPORT "clk" CLKOUT PORT "clkout";

Note: Refer to ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) for the latest tDVAGDDR and Tdvbgddr numbers.

5.13.3.2. Transmit Aligned Interfaces

In this case, the clock and data are aligned when leaving the device. Figure 5.20 below shows the timing diagram for this interface.

tDIAGDDR = Data valid after clock.

tDIBGDDR = Data valid before clock.

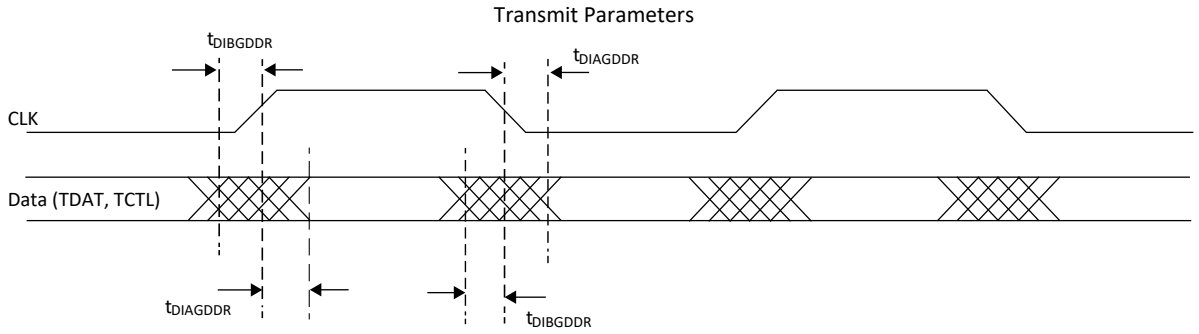

Figure 5.20. Transmit Aligned Interface Timing

Figure 5.20 shows that max value after which the data cannot transition is tDIA_GDDRX1/X2. The min value before which the data cannot transition is – tDIB_GDDRX1/X2. Negative sign is used for the min value is because in this particular case the min condition occurs before the clock edge.

The clock to out time in the software can be specified as –

CLOCK_TO_OUT PORT “dataout” MAX <tDIA_GDDRX1/X2> MIN <-tDIB_GDDRX1/X2> CLKPORT “clk” CLKOUT PORT “clk”;

where:

Data = Data Output Port

Clock = Forwarded Clock Output Port

clk = Input Clock Port

Both tDIA_GDDRX1/X2 and tDIB_GDDRX1/X2 numbers are available in the External Switching Characteristics section of ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) for maximum speed.

Preference Example:

For GDDRX2_TX.Aligned case running at 400 MHz, tDIA_GDDRX2= tDIB_GDDRX2=0.16 ns. The preference would be –

CLOCK_TO_OUT PORT "dataout" MAX 0.16 ns MIN -0.16 ns CLKPORT "clk" CLKOUT PORT "clkout";

Note: Refer to ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) for the latest tDIA_GDDX1/X2 and tDIB_GDDRX1/X2 numbers.

6. ECP5 and ECP5-5G Memory Interfaces

All of the DDR SDRAM interface transfers data at both the rising and falling edges of the clock. The I/O DDR registers in the ECP5 and ECP5-5G device can be used to support DDR2, DDR3, DDR3L, LPDDR2, and LPDDR3 memory interfaces.

These memory interfaces rely on the use of a data strobe signal, called DQS, for high-speed operation. The DQS strobe is a differential signal except for DDR2 you can choose between single-ended or differential DQS strobe.

Figure 6.1 shows typical DDR memory signals. DDR2, DDR3, and DDR3L memory interfaces are typically implemented with either four or eight DQ data bits per DQS. So, a 16-bit DDR memory interface uses two or four DQS signals, and each DQS is associated with four or eight DQ bits, respectively. Both the DQ and DQS are bi-directional ports and are used to read and write to the memory. LPDDR2 and LPDDR3 memory are the same but only supports 8 DQ data bits per DQS strobe.

When reading data from the external memory device, data coming into the FPGA controller is edge-aligned with respect to the DQS signal. This DQS strobe signal needs to be phase shifted 90° before the FPGA logic can sample the read data. When writing to a DDR memory, the memory controller (FPGA) must shift the DQS by 90° to center-align with the data signals (DQ). A clock signal is also provided to the memory. This clock is provided as differential clock (CK and CK#) to minimize duty cycle variations. The memory also uses these clock signals to generate the DQS signal during a read via a DLL inside the memory. The figures below show DQ and DQS timing relationships for read and write cycles.

During read, the DQS signal is low for some duration after it comes out of tristate. This state is called Preamble.

The state when the DQS is low before it goes into tristate is the Postamble state. This is the state after the last valid data transition.

DDR memories also require a Data Mask (DM) signal to mask data bits during write cycles. Note that the ratio of DQS to data bits is independent of the overall width of the memory. Figure 6.1 shows a typical 8-bit interface that has eight associated DQ data bits per DQS strobe signal.

The DDR3 memory module uses fly-by routing topology for the address, command, control, and clock signals. This requires the memory controller to support read and write leveling to adjust for leveled delay on read and write data transfers. LPDDR3 does not use fly-by routing but write leveling may be supported if you emulate the fly-by routing using board traces. You can see more information in the DDR pin placement and layout guidelines section of this document.

One major difference between DDR2/DDR3 and LPDDR2/LPDDR3 is lack of DLL in LPDDR2/LPDDR3 memory device. This would mean in LPDDR2 and LPDDR3, the clock to data output delay from memory device is not compensated by DLL as in traditional DDR2/DDR3, thus the delay is much larger and has larger spread. Theoretically, there is no low frequency limitation on LPDDR2/LPDDR3, although most manufactures place a low limit of 10 MHz. Please note FPGA memory controller side, DLL is still needed to manage write and read phase shift, for all memory interfaces including LPDDR2 and LPDDR3.

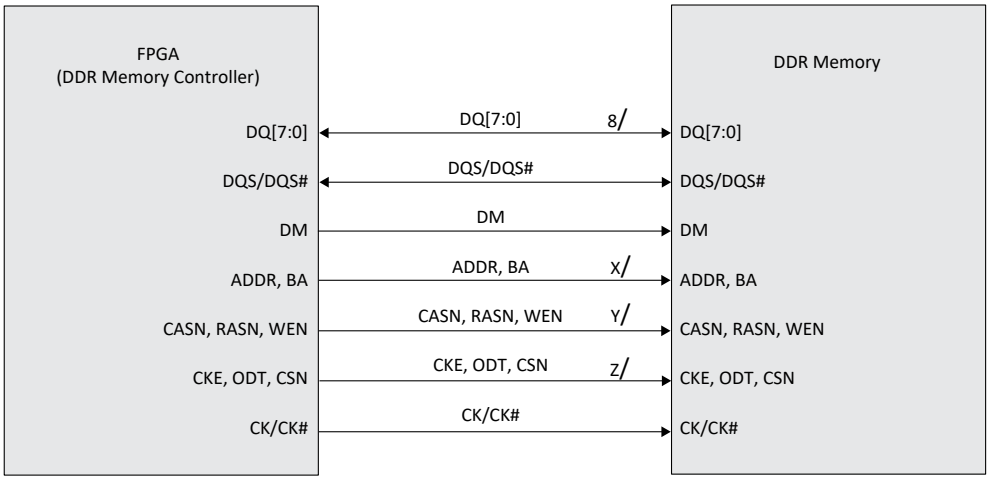

Figure 6.1. Typical DDR2/DDR3/DDR3L Memory Interface

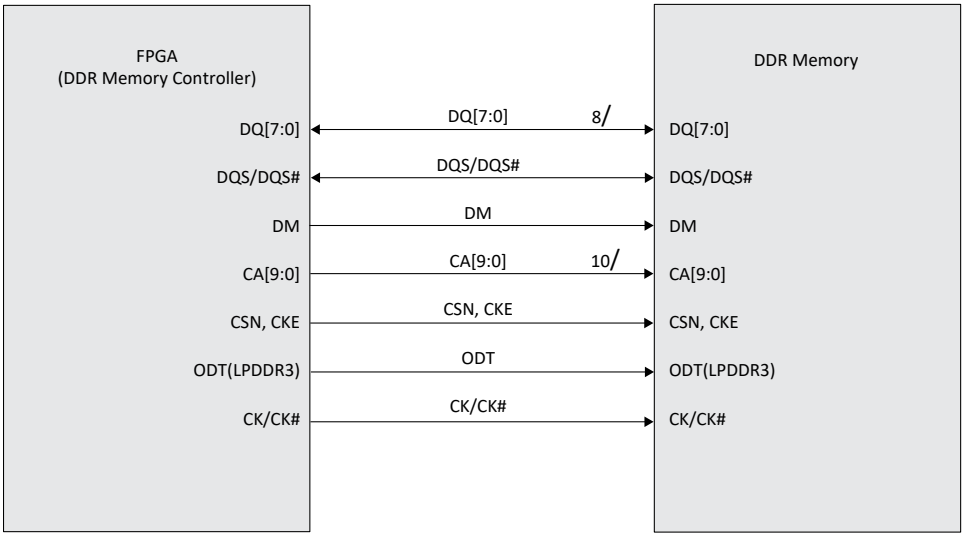

Figure 6.2. Typical LPDDR2/LPDDR3 Memory Interface

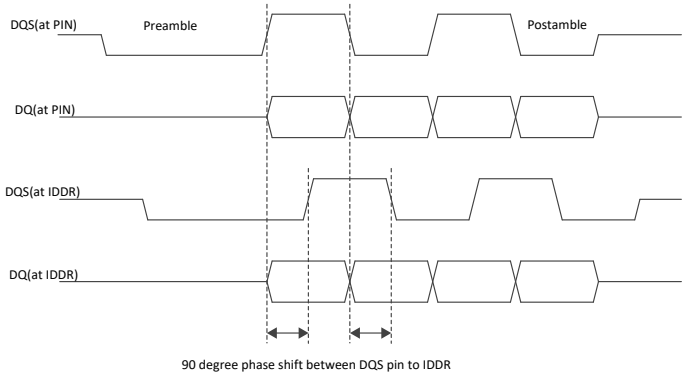

Figure 6.3. DQ-DQS During Read

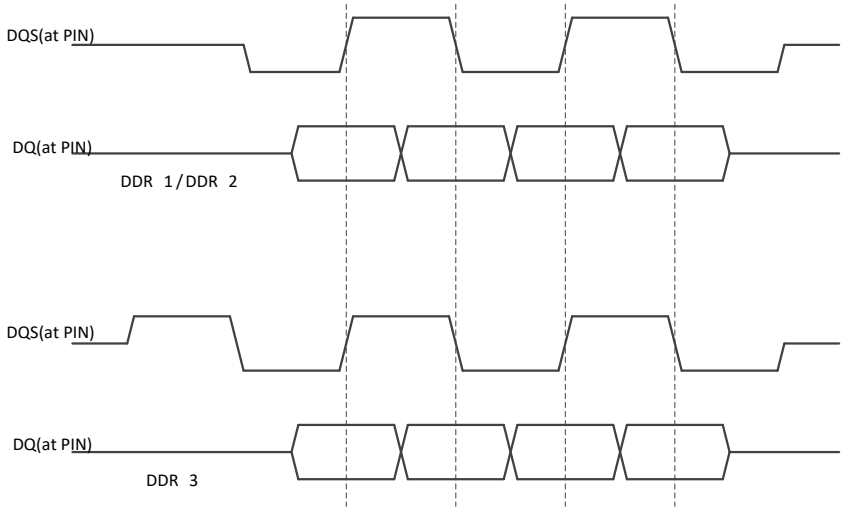

Figure 6.4. DQ-DQS During Write

Table 6.1 below shows the different DDR memory configurations and features supported by ECP5 and ECP5-5G device.

Table 6.1. DDR Memory Configurations Support

DDR Memory	Data Width	VCCIO	DQ	DQS	Modules Types	Rank	Chip Selects	Write Leveling	CMD/ADDR Timing	Fmax
DDR2	8 to 72 bits	1.8V	SSTL18	SSTL18D or SSTL18	UDIMM, SODIMM, RDIMM	Single, Dual	1, 2	No	2T	400 MHz
	8 to 72 bits	1.8V	SSTL18	SSTL18D or SSTL18	Embedded	Single, Dual	1, 2	No	2T[3]	400 MHz
DDR3	8 to 72 bits	1.5V	SSTL15	SSTL15D	UDIMM, SODIMM, RDIMM	Single, Dual	1, 2	Yes	2T[3]	400 MHz
	8 to 72 bits	1.5V	SSTL15	SSTL15D	Embedded	Single, Dual	1, 2	Yes[1]	2T[3]	400 MHz
DDR3L	8 to 72 bits	1.35V	SSTL135	SSTL135D	UDIMM, SODIMM, RDIMM	Single, Dual	1, 2	Yes	2T[3]	400 MHz
	8 to 72 bits	1.35V	SSTL135	SSTL135D	Embedded	Single, Dual	1, 2	Yes[1]	2T[3]	400 MHz
LPDDR2	16 and 32 bits	1.2V	HSUL12	HSUL12D	Embedded (Single Channel)	Single	1	No	ODDRX2	400 MHz
LPDDR3	16 and 32 bits	1.2V	HSUL12	HSUL12D	Embedded (Single Channel)	Single	1	Yes[2]	ODDRZ2	400 MHz

Notes:
1. If Fly-by Wiring is implemented
2. Fly- by wiring is emulated using board traces (Guidelines are in the section below) CSN uses 1T timing

6.1. DDR Memory Interface Requirements

As described in the overview section, all the DDR memory interfaces rely on the use of a data strobe signal, called DQS, for high-speed operation. When reading data from the external memory device, data coming into the ECP5 and ECP5-5G device is edge-aligned with respect to the DQS signal. Therefore, the ECP5 and ECP5-5G device needs to shift the incoming DQS (90° phase shift) before using it to sample the read data.

To implement the write portion of a DDR memory interface, parallel single data rate data must be multiplexed depending on the IDDR and ODDR register gearing mode together with data transitioning on both edges of the clock. In addition, during a write cycle, the ECP5 and ECP5-5G devices generate a DQS signal that is center aligned with the DQ, the data signal. This is accomplished by ensuring a DQS strobe is 90° shifted relative to the DQ data. The ECP5 and ECP5-5G devices provide the solutions to achieve the following design challenges to implement DDR memory write functions:

- DQ/DM needs to be center-aligned to DQS.
- DQS needs to be edge-aligned to CK. In DDR3 interfaces where fly-by routing is used, write leveling should be used to compensate for skews between CK and DQS.
- The DDR output data must be multiplexed into a single outgoing DDR data stream.
- Generate ADDR/CMD signal edge-aligned to CK falling edge to maximize the tIS and tIH timing parameters.
- Differential CK signals (CK and CK#) need to be generated.
- The controller must meet the DDR interface specification for the tDSS and tDSH parameters, defined as DQS falling edge setup and hold time to/from CK rising edge, respectively. The skews, if caused by the fly-by topology, are compensated by write-leveling.
- In case of LPDDR2 and LPDDR3, the CA[9:0] bus needs to be 90 degree from the CLKP/CLKN signal.
- For DDR3 memory, the memory controller also needs to handle the write leveling required by the interface when the fly-by topology is applied.

6.2. Features for Memory Interface Implementation

The ECP5 and ECP5-5G devices contain a variety of features to simplify implementation of the read and write operations of a DDR interface:

- DQS Clock Tree spanning the DQS group
- DDRDLL used to generate the 90 degree delay codes
- DLL-compensated DQS delay elements
- Input FIFO for read data clock domain transfer
- Dedicated DDR Memory input and output registers
- Dynamic Margin Control Circuit to adjust Read and Write delays
- Input/Output Data Delay used to compensate for DQS clock tree delay

6.2.1. DQS Grouping

In DDR interfaces with eight DQ pads associated to one DQS pad, each DQS group generally consists of at least 10 I/O (one DQS, eight DQ, and one DM) for an 8-bit DDR2 memory interface or 11 I/O (two DQS, eight DQ, one DM) to implement a complete 8-bit DDR3/DDR3L/LPDDR2/LPDDR3 memory interface. In case of LPDDR2/3, two additional DQS groups are required to generate the CA[9:0] (with 10 I/O) and Control/CLKP/CLKN (with 5 I/O for LPDDR3 and 4 I/O on LPDDR2) outputs.

In ECP5 and ECP5-5G devices, a DQS group consists of 12 to 16 I/O depending on the device and package selected to accommodate these DDR interface needs. ECP5 and ECP5-5G devices support DQS signals on the left and right sides of the device.

Each DQS signal spans across 12 to 16 I/O. Any 10 (for DDR2) or 11 (for DDR2/DDR3/LPDDR2/LPDDR3) of these 16 I/O spanned by the DQS can be used to implement an 8-bit data side interface. For LPDDR2/LPDDR23, any group with ten I/O is required for CA[9:0] bus and another group with five I/O for LPDDR3 and four I/O for LPDDR2 is required to generate the Control and CLKP/CLKN outputs. In addition to the DQS grouping, you must also assign the reference voltage (VREF) input to an I/O in that bank required to implement the referenced I/O standard.

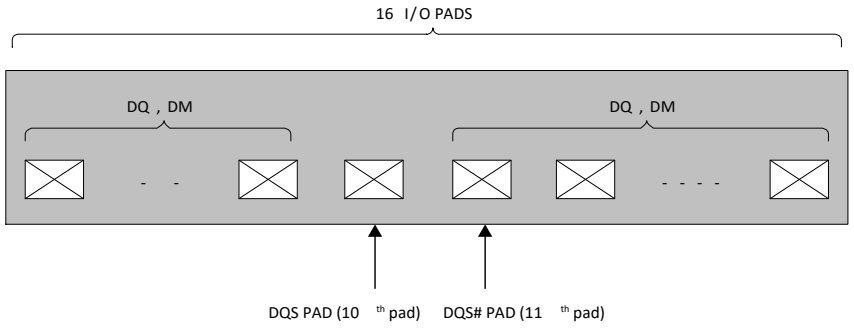

Figure 6.5. DQ-DQS Grouping

Figure 6.5 shows a typical DQ-DQS group for ECP5 and ECP5-5G devices. The 10th I/O Pad of this 16 I/O group is the dedicated DQS pin. All the nine pads before the DQS and six pads after the DQS are covered by this DQS bus span. If a differential DQS pair is required then the 11th pad is used by the DQS# signal. You can assign any other I/O pads to be DQ data or DM pins. Therefore, for example, to implement a 32-bit wide memory interface you would need to use four such DQ-DQS groups when eight-to-one DQ-DQS association is used.

In case of LPDDR2 and LPDDR2, two additional DQS groups are required to assign the CA[9:0] and the control signals.

In case of DDR2/DDR3/DDR3L, additional I/O pads are required to implement address, command, and control. They do not have to be I/O in DQS groups but need to use 1x gearing generic output DDR capable pads.

Each of the dedicated DQS pins is internally connected to the DQS phase shift circuitry. The pin out sheets included as part of ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012) shows pin locations for each of the DQS groups.

6.2.2. DLL-Compensated DQS Delay Elements

The DQS to and from the memory is connected to the DQS delay element inside the ECP5 and ECP5-5G device. The DQS delay block receives the delay control code, DDRDEL, from the on-chip DDRDLL. The code generated by DDRDLL is connected to the DQSBUF circuit to perform 90° read phase shift and 90° write phase shift. DDRDLL requires the frequency reference from PLL, normally going through the Edge Clock tree.

ECP5 and ECP5-5G devices support one DDRDLL modules in each corner of the device. The DQSBUF modules that receive the DDRDEL code from DDRDLL can either receive the code from the top or bottom DDRDLL on that side. Hence each side can support up to two different rate DDR memory interfaces..

Table 6.2. DDRDLL Connectivity

DDRDLL Location	Left DQSBUFs	Right DQSBUFs
DDRDLL_TR	—	X
DDRDLL_TL	X	—
DDRDLL_BR	—	X
DDRDLL_BL	X	—

The DQS received from the memory is delayed in the DQS delay element in the DQSBUF block, and this delayed DQS is used to clock the first set stage DDR input registers.

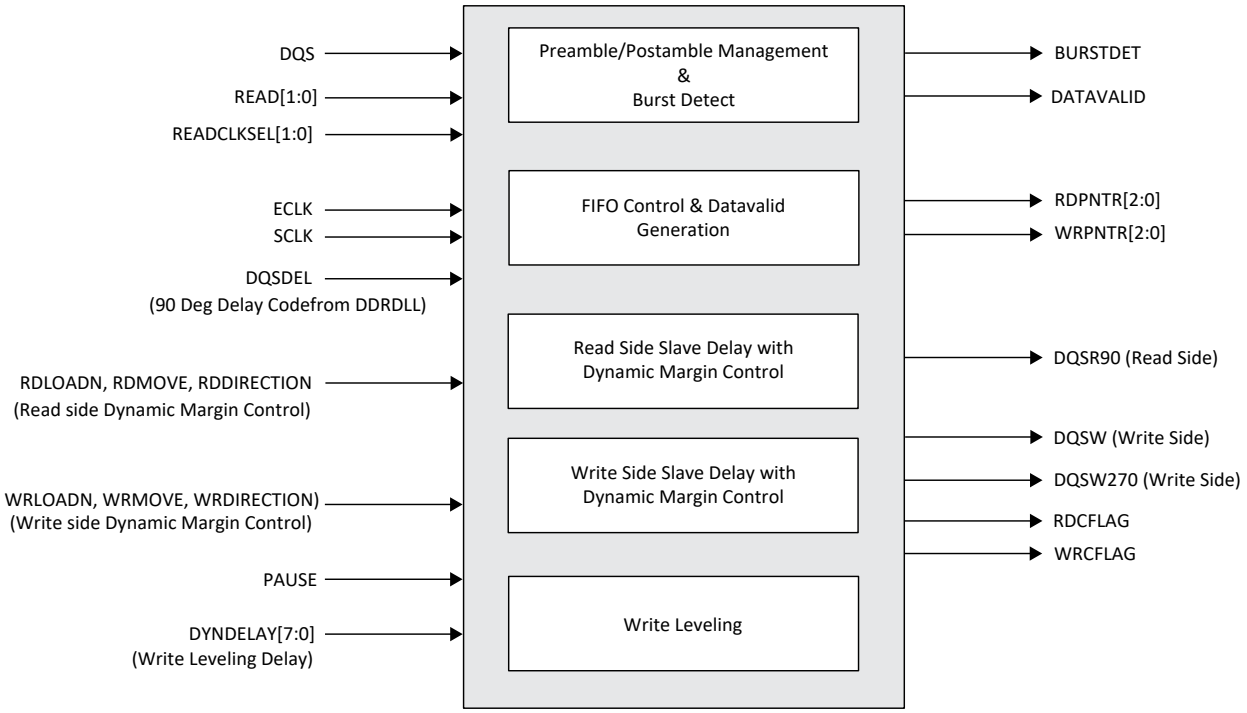

Figure 6.6. DQSBUF Block Functions

6.2.3. Data Valid Module

The DQSBUF block generates a DATAVALID signal. This signal indicates the timing that the IDDR module drives the valid read data transmitting to the FPGA fabric. DATAVALID is a level-sensitive active high signal and indicates that the read data output from the IDDR module is valid while it is asserted high. The DATAVALID signal can stay asserted during the IDDR module outputs back-to-back valid read data until all consecutive read operations are completed.

6.2.4. READ Pulse Positioning Optimization

The memory controller is required to provide the READ1/0 signal to the DQSBUF block to position the internal READ pulse and generate the DATAVALID output that indicates the proper timing window of the valid read data. The internal READ pulse is also used to get a clean internal DQS signal between the preamble and postamble periods. The clean DQS is 90° shifted internally to be used to capture the read data.

Due to the DQS round trip delay that includes PCB routing and I/O pad delays, proper positioning of the READ pulse is crucial for successful read operations. The ECP5 and ECP5-5G device DQSBUF block provides the dynamic READ pulse positioning function which allows the memory controller to locate the READ pulse to an appropriate timing window for the read operations by monitoring the positioning result.

The READCLKSEL2/1/0 and BURSTDET signals are used to accomplish the READ pulse positioning function for a corresponding DQSBUF block. The READ1/0, READCLKSEL2/1/0 signals are driven by the user logic and are part of the DDR memory controller.

The READ1/0 signal needs to be asserted high a certain amount of time before the read preamble starts. The suggested READ1/0 signal assertion timing and the required duration of assertion are listed in Table 6.3. When the internal READ pulse is properly positioned, BURSTDET is asserted high and guarantee that the generated DATAVALID signal properly indicates the valid read data time window. The READ1/0 signal must stay asserted as long as the number of SCLK cycles that is equal to one fourth of the total burst length as listed in the Table 6.3.

Table 6.3. DDRDLL Connectivity

Gearing Mode	READ Control	DQSBUF Block	Initial READ Assertion Position*	READ Width in SCLK
X2 Gearing (All DDR Memory Interfaces)	READ1 READ0 READCLKSEL2 READCLKSEL1 READCLKSEL0	DQSBUFM	At least 5.5T before preamble	Total Burst Length/4

***Note**: Subject to change after validation tests. The number shown does not include DQS round trip delay.
1T = 1 tCK memory clock cycle

Once the controller initially positions the internal READ pulse using the READ1/0 and READCLKSEL2/1/0 signals, BURSTDET can be used to monitor the positioning result to optimize the READ pulse position. The BURSTDET signal provides a feedback mechanism to inform the memory controller whether the READ pulse has reached to the optimal position for the read operations or not with the current READ1/0 and READCLKSEL2/1/0 values. When it reaches to the optimal position, BURSTDET is asserted High after a read operation. Otherwise, BURSTDET remains Low. A minimum burst length of eight on the memory bus must be used in the training process. This can be done either with two consecutive BL4 (BL=4) read accesses or one BL8 read access. Any even number of BL4, or any multiplication of BL8(BL=8) can also be used. This read pulse training process must be performed during the initial training and can also be periodically calibrated during the normal operations.

The BURSTDET signal is asserted after the last DQS transition is completed during a read operation and lasts until the next read cycle is started. Once a read operation is started, the memory controller should wait until the DATAVALID signal from DQSBUFM is asserted and then sample the BURSTDET signal at the next cycle to monitor the READ pulse positioning result. If there is no assertion on BURSTDET, it means that the READ pulse has not been located to the optimal position yet. Then, the memory controller needs to shift the READ1/0 signal and/or increase the READCLKSEL2/1/0 value until it detects a BURSTDET assertion. It is recommended that at least 128 read operations be performed repetitively at a READ pulse position during the initialization for getting jitter immunity. 16 read operations can be performed in a periodic calibration if used during the normal operation. The memory controller can determine the proper position alignment when there is no failure on BURSTDET assertions during these multiple trials.

To reposition the internal READ pulse:

1. The memory controller sets READ1/0 to an initial position before starting the read pulse training. READ1/0 must be asserted for the number of SCLK cycles that is equal to one-fourth of the current read burst length as listed in Table 6.3. Each READ bit (READ1 or READ0) in the system clock domain is translated to an 1T time slot of the memory clock domain as shown in Figure 6.7.

2. Once READ1/0 positions the READ pulse, READCLKSEL2/1/0 can be used to shift the READ pulse by 1/4T per step. With the total eight possible combinations from 000 to 111, READCLKSEL2/1/0 covers the READ pulse shift up to a whole 2T timing window. If BURSTDET is asserted with a certain READCLKSEL2/1/0 value, it indicates that the READ pulse has been located to the optimal position. If no BURSTDET is asserted during this step, the READ pulse needs to be moved to the next timing window.
3. To shift the READ pulse timing window, READ1/0 can be moved to the next cycle. If a READ bit is asserted in the next cycle while the other READ bit remains in the current cycle, only the corresponding time slot on the READ pulse moves to the next allotted slot. Therefore, only READ0 must be moved to the next cycle if the READ pulse needs to be shifted only by 1T. However, if only READ1 is moved to the next SCLK cycle, then the READ pulse gets two short pulses in wrong timing. Since READCLKSEL2/1/0 covers a whole 2T timing, it is recommended that both READ1 and READ0 are moved to the next cycle together to shift the READ pulse to the next 2T window as the example shown in Figure 6.7.

Repeat *step 2* and *step 3* until BURSTDET is asserted.

Figure 6.7 shows an example of a burst length 8 (BL8) read operation. The bottom side of the diagram indicates the case that the incoming DQS (DQSI) gets slightly more than 2T delay after a round trip. Due to this round trip delay, both READ1 and READ0 need to be shifted to the next SCLK cycle so that the internal READ pulse gets a 2T shift. Then, the READCLKSEL2/1/0 signals can be used to fine tune the READ signal position throughout the training process. When any of READCLKSEL2/1/0 is changed at any time after a system reset, the PAUSE input to DQSBUFM must be asserted before 4T of the change and remain asserted for another 4T after the change to avoid glitches and malfunction.

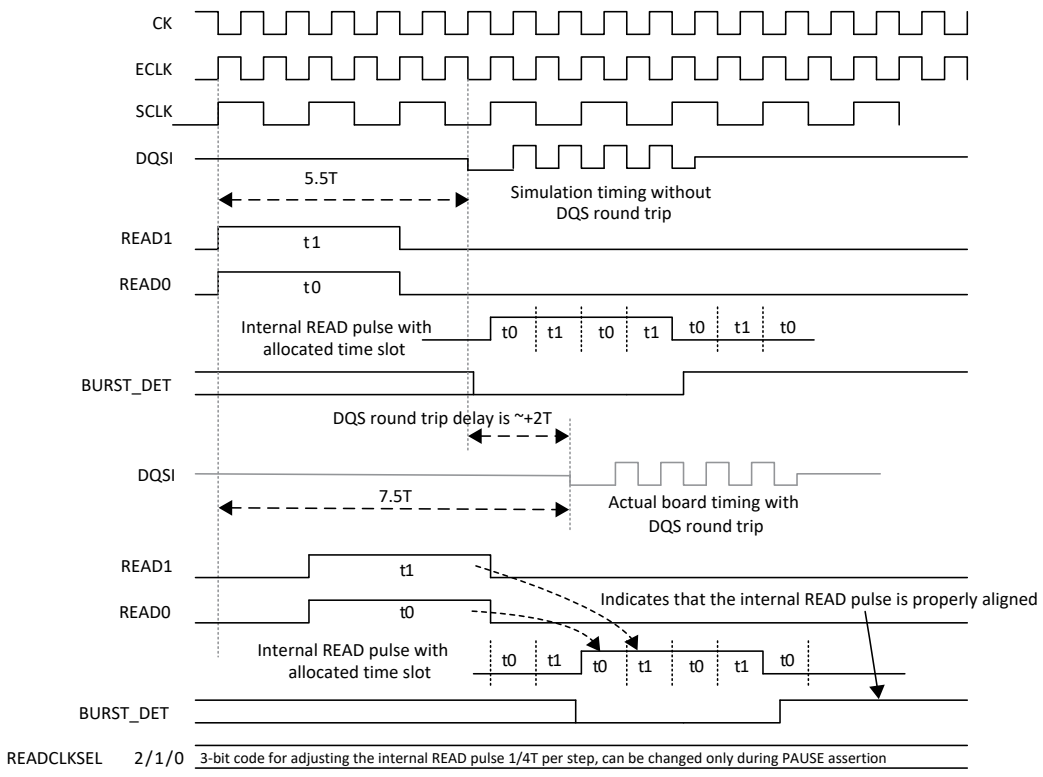

Figure 6.7. READ Signal Training Process

Note that the DYNDELAY[7:0] signal, margin control signals (WR/RDMOVE, WR/RDLOADN) and DDRDLL update signal (UDDCNTLN) also have the same PAUSE requirement.

6.2.5. Dynamic Margin Control on DQSBUF

The ECP5 and ECP5-5G family includes dynamic margin control signals in the DQSBUF module allows you to dynamically adjust the read or write side DQS delays generated in the DDRDLL.

Once the margin control mode is enabled by de-asserting WRLOADN (=1) or RDLOADN (=1), the DQSBUF's phase shift control to make a center aligned interface is no longer controlled by the DDRDLL component. It becomes your responsibility to complete the margin control training to maximize the valid window and then continuously monitor the DDRDLL delay code (DCNTL7~DCNTL0) and controls the DQSBUF delays accordingly using the WR/RDMOVE and WR/RDDIRECTION signals to compensate the PVT variations.

6.2.6. Read Data Clock Domain Transfer Using Input FIFO

Each IDDR module in the ECP5 and ECP5-5G device has a dedicated input FIFO to provide a safe clock domain transfer from the DQS domain to the ECLK or SCLK domain. The input FIFO is 8-level deep with 3-bit write and read pointers. It transfers the read data from the non-continuous DQS domain to the continuous ECLK. The FIFO is written by the DQS strobe and read back by ECLK which has the identical frequency rate as DQS.

The input FIFO also performs the read leveling function. When each DQS strobe signal and its associated DQ data signals arrive at slightly different time with others to the FPGA, the input FIFO allows the skewed read data to be captured and transferred properly.

Each DQS group has one FIFO control in the DQSBUF block. It distributes the FIFO read/write pointers, WRPNTR [2:0] and RDPNTR [2:0], to each memory IDDR module in the same DQS group. Safe domain crossing between ECLK and SCLK is guaranteed by the ECP5 and ECP5-5G device hardware design.

6.2.7. DDR Input and Output Registers (IDDR/ODDR)

ECP5 and ECP5-5G devices provide dedicated input DDR (IDDR) and output DDR (ODDR) functions supporting 4:1(X2) gearing modes that are used to implement the DDR memory functions. These automatically handle the transfer of data from ECLK domain to the SCLK (FPGA clock) domain.

6.3. Memory Interface Implementation

The following sections explain the DDR2, DDR3/DDR3L, LPDDR2, LPDDR3 memory interfaces implementation using the X2 gearing mode. ECP5 and ECP5-5G devices support these memory interfaces generation through the Clarity Designer tool. Clarity Designer generates one module that includes the Read and Write side implementation shown below.

All of the memory interfaces use DQS clock, one ECLK and one SCLK to implement the read and write side operations. ECLK must always be routed on the Edge Clock tree and SCLK on the primary clock tree.

6.3.1. Read Implementation

The read side implementation is shown in Figure 6.8.

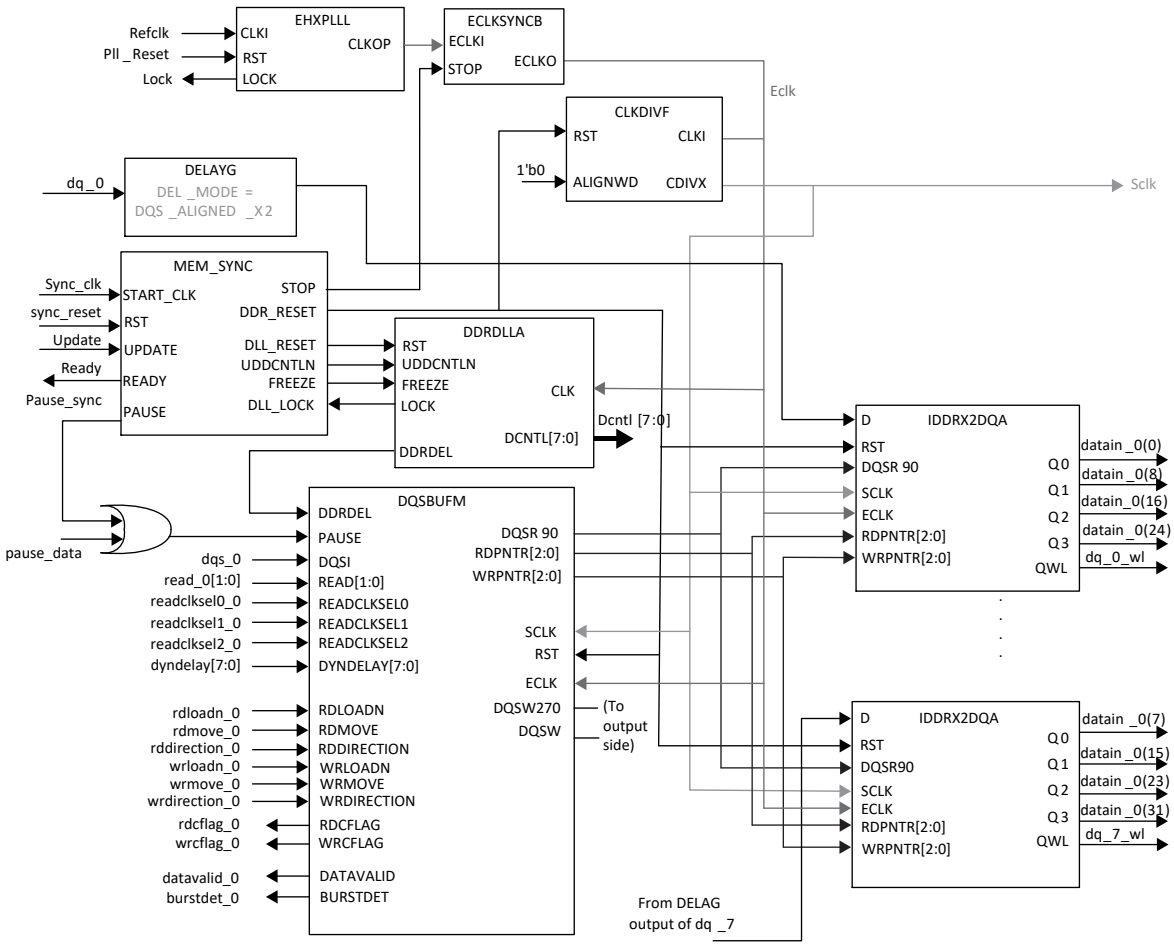

Figure 6.8. DDR2, DDR3/DDR3L, LPDDR2, and LPDDR3 Read side Implementation

The read side is implemented using the following software elements.

- DDRX2DQA element to capture the data
- DDRDLLA is used to generate the delay code for DQSBUFM to get the 90 degree phase shift on the DQS input (DQSR90).
- The incoming DQS clock (DQSI) is routed through the DQSBUFM module to the DQS clock tree.
- The DQSBUFM receives the delay code from DDRDLLA and generates the delayed DQS signal to IDDRX2DQA.
- The DQSBUFM is used to generate the Read and Write pointers that is used to transfer data from the DQS to ECLK inside the IDDRX2DQA module.
- Read 1, 0 and Readclksel_2, 1, 0 signals of DQSBUFM are used by the user logic to obtain the optimal READ pulse position and driven by the user logic to generate a clean DQS output signal based on the trained READ pulse with respect to preamble and postamble.
- The dynamic delay control ports are available on the DQSBUFM module when you select the *enable dynamic margin control* option.
- DYNDELAY[7:0] of DQSBUFM is used to perform write leveling. If write leveling is not used, it is connected to 0.
- Port QWL of IDDRX2DQA is used for DDR3/DDR3L and LPDDR3 to support write leveling. It is used to deliver the write leveling monitor signals from the memory device to the FPGA user logic.
- MEM_SYNC soft IP must always be included in the interface. It is required to avoid issues on DDR memory bus and update code in operation without interrupting interface operation. When a DDR memory interface IP is generated from Clarity Designer, the MEM_SYNC soft IP block is also generated and included.

- The Pause_sync output of the MEM_SYNC soft IP is used to request DQSBUFM pause for the DDRDLL update and goes to an output port of the Clarity Designer module. The input port Pause_data goes to DQSBUFM. It is required by user logic, to OR the Pause_sync output of the MEM_SYNC module with the user pause to drive the Pause_data input of the DQSBUFM. This OR would need to be implemented outside of the Clarity Designer module in your design.
- CLKDIVF set to divide by 2 function is used to generate the SCLK from the ECLK.
- When DDR data bus is required to cross two sides, an ECLKBRIDGECS should be enabled in Clarity Designer. When using ECLKBRIDGECS, there are two DDRDLLs in the design one for each side. Also the DQSBUFMs used on the second side should be connected to its DDRDLL. Clarity Designer automatically generates all the required DDRDLLs and DQSBUFMs.

6.3.2. Write Implementation (DQ, DQS, and DM)

Figure 6.9 shows the DDR2, DDR3/DDR3L, LPDDR2, and LPDDR3 memory interface write side implementation to generate DQ, DQS, and DM outputs.

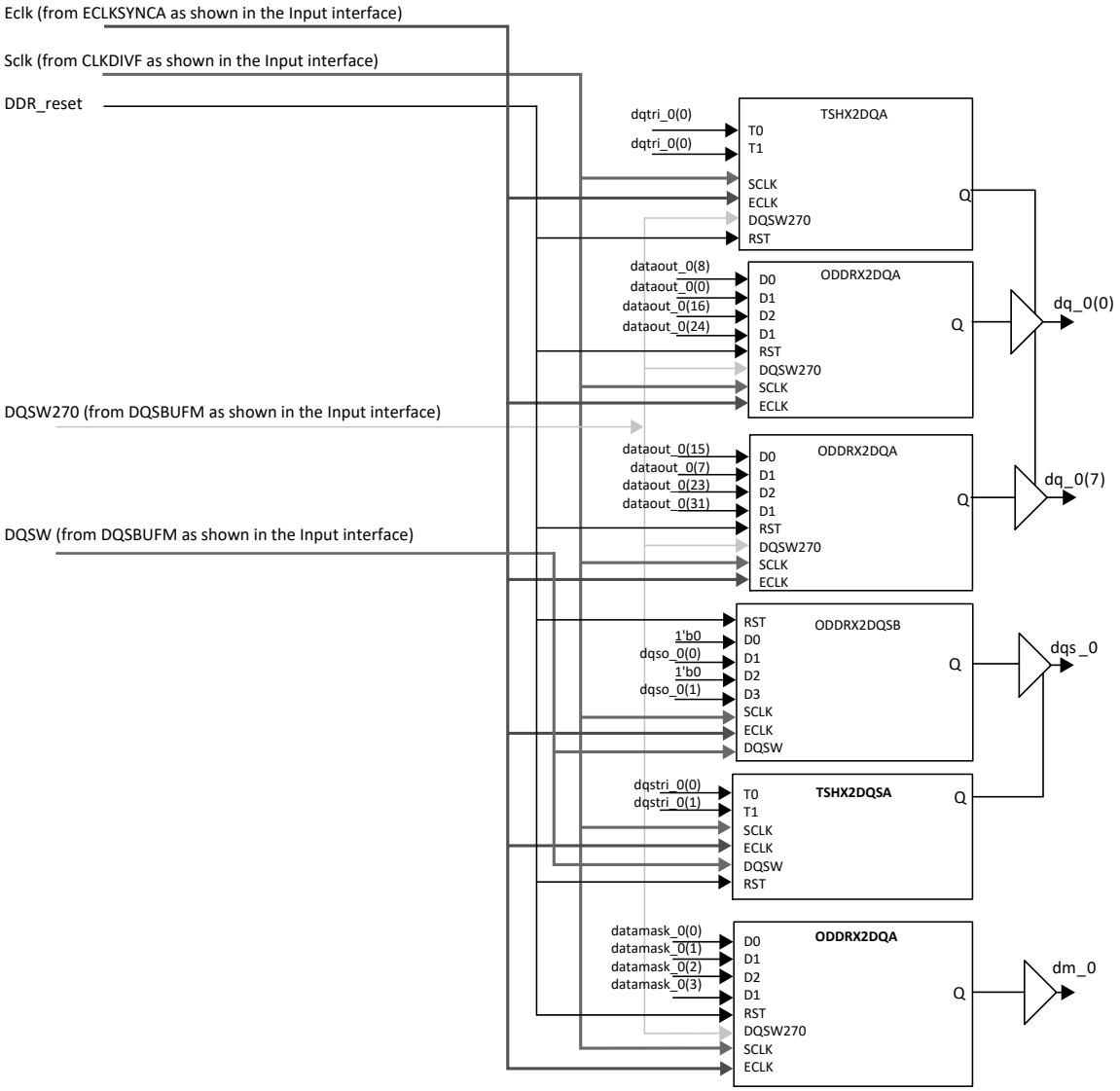

Figure 6.9. DDR2, DDR3/DDR3L, LPDDR2, and LPDDR3 Write Side (DQ, DQS, and DM)

This interface uses the following modules:

- ODDRX2DQA to generate the data DQ and DM signals. TSHX2DQA is used to generate the tristate control for the DQ output.
- ODDRX2DQSB to generate DQS output. TSHX2DQSA is used to generate the DQS tristate control.
- DQSW270 which is the 270 degree delayed DQS signal is used to generate the DQ and DM outputs.
- DQSW is 90 degrees shifted from the DQSW270 is used to generate DQS output.
- The DQSW270 and DQSW clocks are generated in the DQSBUFM module shown on the Read side Implementation figure.
- When write leveling is enabled, the dynamic delay for write leveling (DYNDELAY[7:0]) is applied both to DQSW and DQSW270 so that the DQ and DQS phase relationship is maintained.
- ECLK and SCLK are used inside the ODDRX2 module before data is transferred to the DQSW270 and DQSW clocks. The ECLK is generated by the EHXPLLL module and the SCLK is generated by the CLKDIVF module, both shown in the Read side implementation.
- Figure 6.9 shows one tristate. The software generates one tristate element for each DQ port.

6.3.3. Write Implementation (DDR2, DDR3/DDR3L Address, Command, and Clock)

DDR2, DDR3, DDR3L write side interface side to generate the Clock, Address and Command uses the following modules:

- ODDRX2F with inputs tied to constants to generate the DDRCLK output.
- The ADDR, BA, CASN, RASN, WEN, CKE, and ODT command and address signals are generated using the ODDRX1F. CSN output is generated using OSHX2A.
- Both ECLK and SCLK is used in these elements. This is same ECLK and SLCK generated in the Read side.

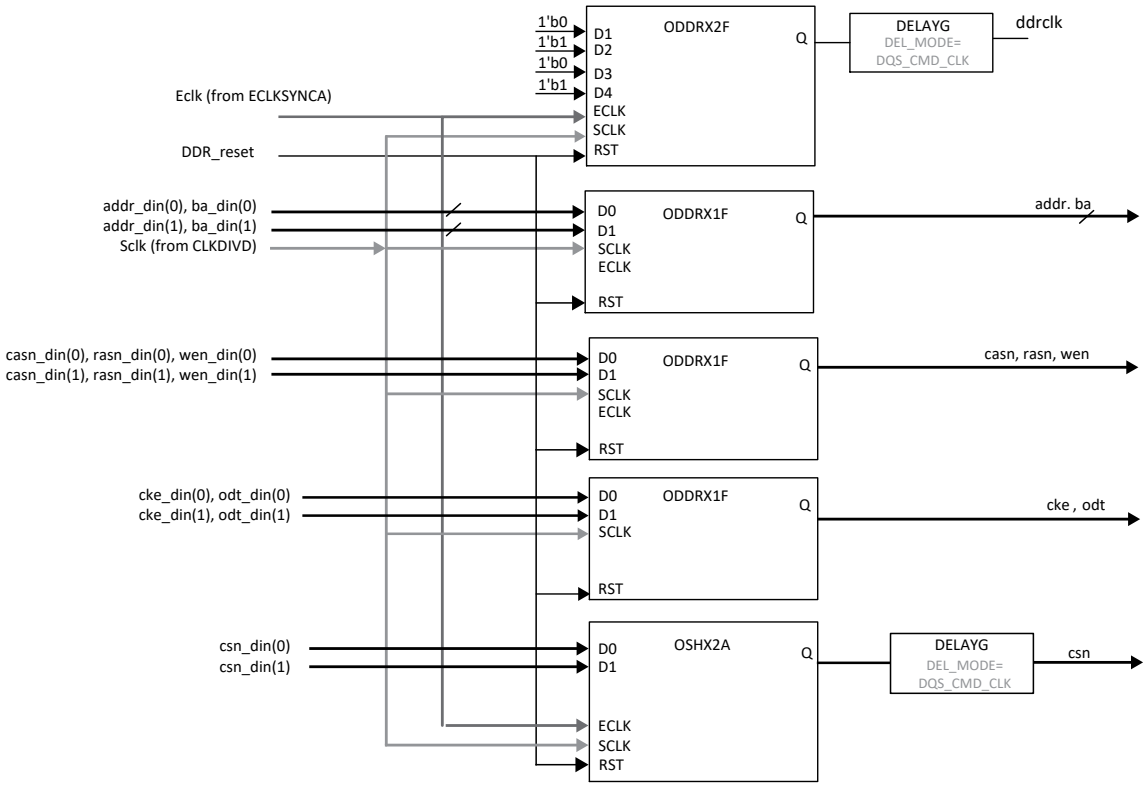

Figure 6.10. DDR2, DDR3/DDR3L Address, Command, and Clock Generation

6.3.4. Write Implementation (LPDDR2 and LPDDR3 Address, Command, and Clock)

LPDDR2 and LPDDR3 output side interface side to generate the Clock, Address, and Command uses the following modules:

- Two DQSBUFM are required generate the LPDDR2 and LPDDR2 address/command and clock signals. One of the DQSBUFM is used for CA output and the other for CKE, CSN, ODT, and CLKP/CLKN (note that ODT is only for LPDDR3).
- ODDRX2DQA module to generate the CA[9:0] outputs
- ODDRX2DQSB with D0 and D1 tied together and D2 and D3 tied together to generate CSN, CKE signals
- ODDRX2DQSB with inputs tied to 0 and 1 is used to generate the CLKP/CLKN outputs.
- On LPDDR3, even the ODT has D0 and D1 tied together and D2 and D3 tied together and uses same DQSBUFM.
- The DQSBUFM used for CA requires a separate input. Ideally, you must take Pause_sync output of the MEM_SYNC module and make an OR gate with user CA pause to drive the PAUSE input port of DQSBUFM. The Pause_CA pause is connected to the user CA training logic PAUSE required. If CA training is not used, then user CA pause should be tied to GND.
- Both ECLK and SCLK is used in these elements. This is same ECLK and SLCK generated in the Input Read side module shown above.

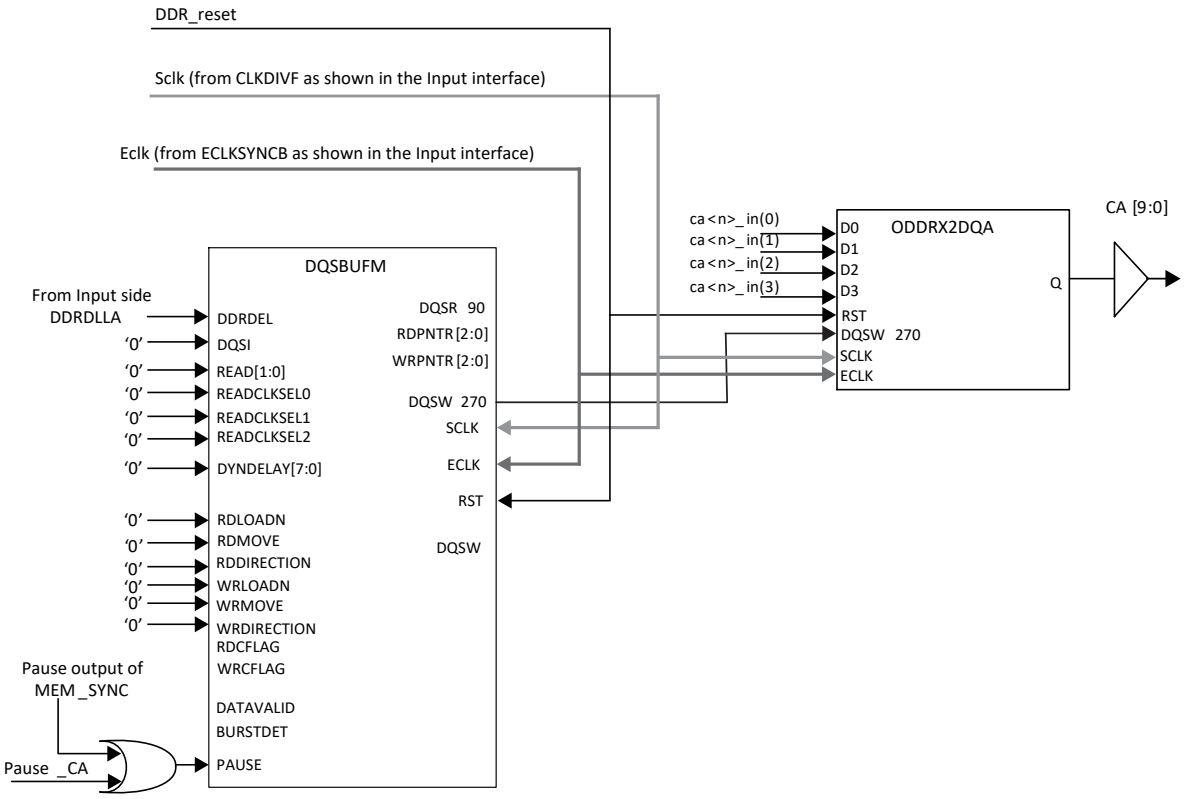

Figure 6.11. LPDDR2 Output for CA Generation

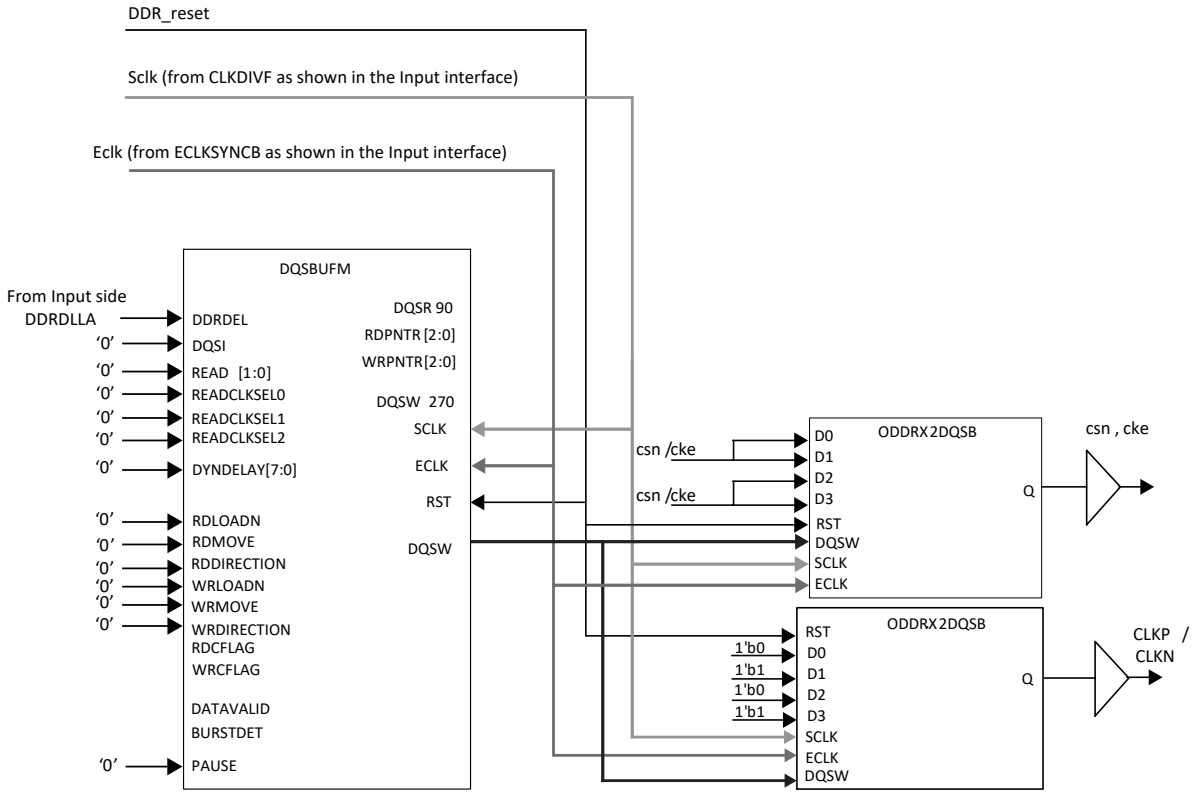

Figure 6.12. LPDDR2 Output for CSN, CKE, and CLOCK Generation

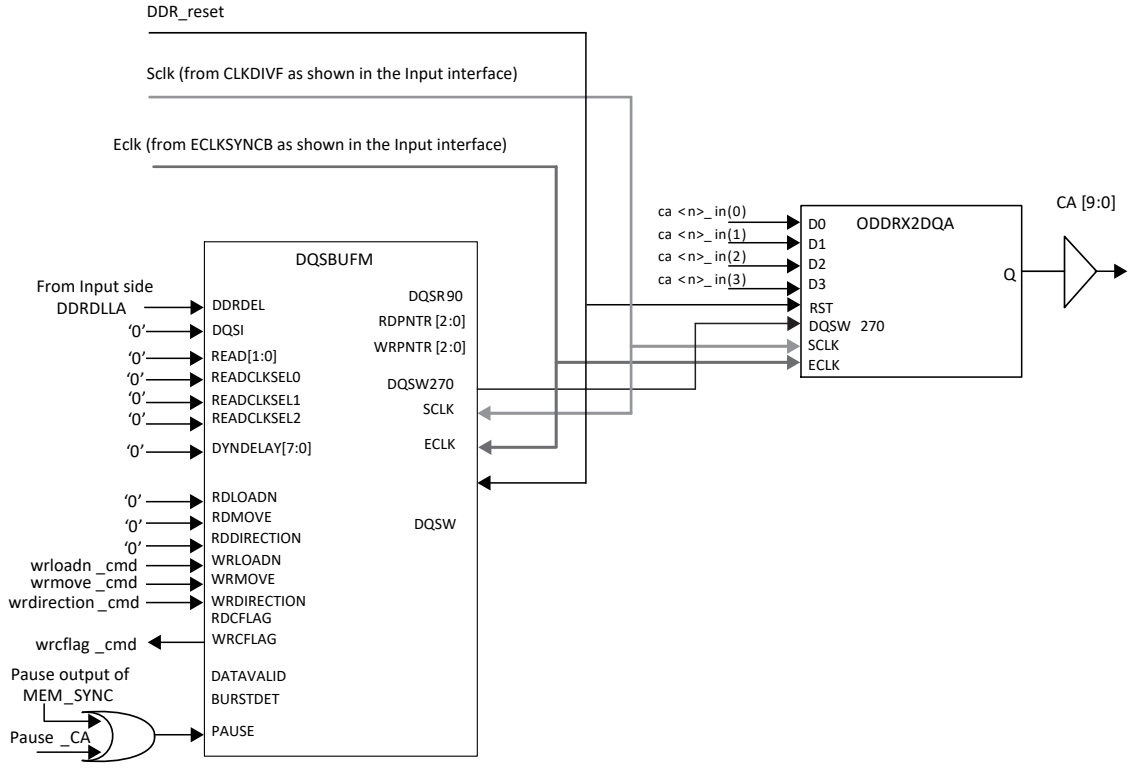

Figure 6.13. LPDDR3 Output Side for CA Generation

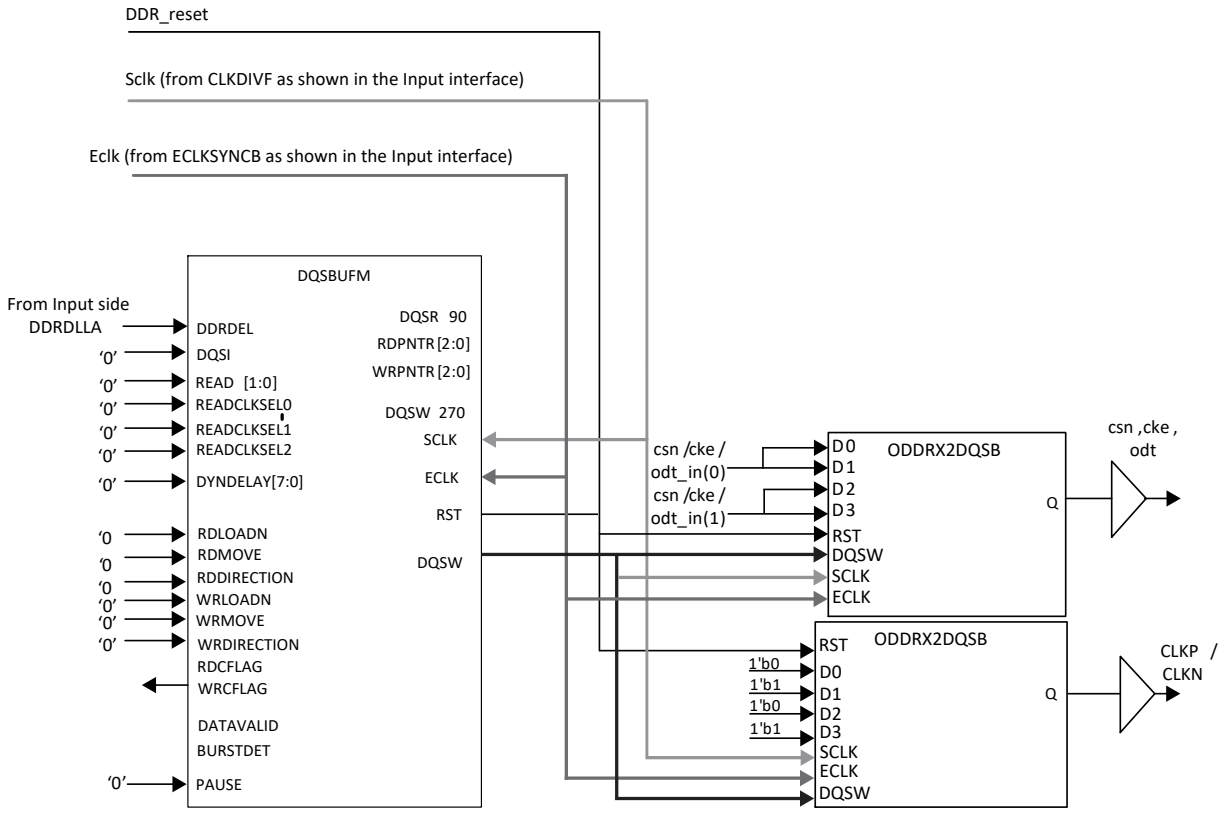

Figure 6.14. LPDDR3 Output Side for CSN, CKE, ODT, and CLOCK Generation

6.4. DDR Memory Interface Design Rules and Guidelines

Listed below are some rules and guidelines to keep in mind when implementing DDR memory interfaces in ECP5 and ECP5-5G devices. ECP5 and ECP5-5G devices have dedicated DQS banks with the associated DQ pads.

- The left and right sides of an ECP5 and ECP5-5G device share an identical I/O structure. All of the memory interfaces can be implemented on these sides.
- The Top side of the device does not support DQSBUF blocks. As such, it does not support DDR memory interfaces. Although some of the ADDR/CMD generation that uses ODDRX1 modules can be placed on the top side of the device for DDR2, DDR3, and DDR3L.
- DDRDLLA primitive should be instantiated for all DDR memory interfaces. Each DDRDLLA generates 90° digital delay code for all the connected DQS delay blocks based on the reference clock input to the DDRDLLA. Therefore, all the DDR memory interfaces under the same DDRDLLA coverage must run at the same frequency.
- There are four DDRDLLs on each device, one DDRDLL in each corner of the device. Each DQSBUF module can receive delay codes from either of the DDRDLLs in each top and bottom corner of the device. This would be an exception for the smallest device where there are only two DDRDLLs on the device.
- When a DDR memory interface is added to the side where another DDR memory interface is running at a different frequency, another available DDRDLLA for the side must be instantiated and used for the new interface.
- The reference clock input to the PLL used in the DDR memory interface implementation must be located to the dedicated PLL pin or a PCLK pin. The dedicated PLL input pin is preferred due to less skew.
- Each DDR memory interface must use its corresponding I/O standard.
  - For the DDR2 memory interface, the interface signal should use the SSTL18 I/O standards.
  - For the DDR3 memory interface, these signals should be connected to the SSTL15 standards.
  - For the DDR3L memory interface, these signals should be connected to the SSTL135 standards.

- For the LPDDR2 and LPDDR3 memory interface, the interface signal should use the HSUL12 standard.
- DDR3, DDR3L, LPDDR2, and LPDDR3 memory interfaces also requires differential DQS signals. The use of differential DQS is optional for DDR2.

Table 6.4 shows the IO_TYPE setup for each of the DDR memory interfaces.

Table 6.4. I/O Standards for DDR Memory

	DDR2	DDR3	DDR3L	LPDDR2	LPDDR3
DQ	SSTL18_I, SSTL18_II	SSTL15_I, SSTL15_II	SSTL135_I, SSTL135_II	HSUL12	HSUL12
DQS	SSTL18_I, SSTL18_II, SSTL18D_I, SSTL18D_II	SSTL15D_I, SSTL15D_II	SSTL135D_I, SSTL135D_II	HSUL12D	HSUL12D
Cmd/Addr	SSTL18_I, SSTL18_II	SSTL15_I, SSTL15_II	SSTL135_I, SSTL135_II	HSUL12	HSUL12
CK	SSTL18D_I, SSTL18D_II	SSTL15D_I, SSTL15D_II	SSTL135D_I, SSTL135D_II	HSUL12D	HSUL12D

Note: When implementing the DDR interface, the VREF1 of the bank is used to provide the reference voltage for the interface pins.

6.5. DDR2/DDR3 Memory Interface Termination Guidelines

(These updates are still preliminary. Another update is needed once the whole validation processes are completed.)

Proper termination of a DDR memory interface is an important part of implementation that ensures reliable data transactions at high speed. Below is the general termination guideline for the ECP5 and ECP5-5G device DDR memory interface.

6.5.1. Termination for DQ, DQS, and DM

- Do not locate any termination on the memory side. The memory side termination on DQ, DQS and DM is dynamically controlled by the DDR3 SDRAM's ODT function.
- Do not locate any termination on the FPGA side. The ECP5 and ECP5-5G device has internal termination on DQ and DQS, which is dynamically controlled. Use the TERMINATION preference for DQ and DQS pads to enable the internal parallel termination to VCCIO/2. The TERMINATION preference has the OFF, 50-, 60-, and 75-Ω options. (Recommended setting for each interface TBD)

6.5.2. Termination for CK

DDR memory clocks require differential termination because they use a differential signaling. Use SSTL15D in DDR3 or SSTL18D in DDR2 to drive the clock signals. You can locate an effective 100-Ω termination resistance on the memory side to achieve the differential termination using the following guideline:

- Locate a 100-Ω resistor between the positive and negative clock signal, OR
- Connect one end of an Rtt resistor to the positive pin and one end of another Rtt to the negative pin of a CK pair, then connect the other ends of two Rtt resistors together and return to VDD or GND through a Ctt capacitance. Note that the JEDEC CK termination scheme defined in the DIMM specifications uses 36-Ω for Rtt with 0.1 µF Ctt for DIMM for DDR3 DIMMs returning to VDD. 50-Ω Rtt can also be used for non-DIMM applications.
- Use of series termination resistors at the FPGA side is not recommended.

When fly-by wiring is used in DDR3, the CK termination resistor should be located after the last DDR3 SDRAM device.

6.5.3. Termination for Address, Commands, and Controls

- Parallel termination to VTT on address, command and control lines is typically required at the DDR2/DDR3 and DDR3L memory side.
- Locate a 50-Ω parallel-to-VTT resistor (or a best known resistance obtained from your SI simulation) to each address, command, and control line on the memory side.
- Series termination resistors can be optionally used on the address, command and control signals to suppress overshoot/undershoot and to help decrease overall SSO noise level. 22-Ω or 15-Ωseries termination is recommended when used.
- When fly-by wiring is used in DDR3, the address, command, and control termination resistors should be located after the last DDR3 SDRAM device.

No termination is required on the LPDDR2 and LPDDR3 CA bus and control lines. They use point-to-point connections.

6.5.4. Termination for DDR3/DDR3L DIMM

The DDR3 DIMMs incorporate internal termination following the requirements defined by the JEDEC DIMM specification. For this reason, the user termination requirement for the DDR3 DIMM is slightly different from that of DDR3 SDRAM devices:

- Do not locate any termination on the memory side. The memory side termination on DQ, DQS, and DM is dynamically controlled by the DDR3 SDRAM's ODT function.
- Do not locate differential termination on CK at the memory side because the DIMM already has termination on the module.
- Do not locate parallel termination to VTT on address, command, and control signals at the memory side because the DIMM already has termination on the module.
- Follow the termination for DQ, DQS, and DM guideline above for the FPGA side termination.

6.6. DDR Memory Interface Pinout Guidelines

The ECP5 and ECP5-5G device contains dedicated I/O functions for supporting DDR memory interfaces. The following pinout rules must be followed to properly use the dedicated I/O functions.

- The DQS-DQ association rule must be followed.
- All associated DQs (8 or 4) to a DQS must be in the same DQS group.
- A data mask (DM) must be part of the corresponding DQS group.
  Example: DM[0] must be in the DQS-16 group that has DQ[7:0], DQS[0].
- A DQS pad must be allocated to a dedicated DQS True (+) pad.
- A DQS# pad is auto-placed when a differential SSTL type (SSTL15D in DDR3, SSTL18D in DDR2, SSTL135D in DDR3L, and HSUL12D in LPDDR2/LPDDR3) is selected.
- Do not assign any signal to a DQS# pad if used as differential strobe. The software automatically places DQS# when a differential I/O type is applied.
- DQS/DQS# pads can be used for other DDR functions. For example, DQS# can be used as a DQ pad for a nondifferential DQS interface such as DDR2 with single ended strobe. However, a DQS signal must use the DQS/DQS# pads only.
- Data group signals (DQ, DQS, DM) can use any of the left and right sides of the ECP5 and ECP5-5G device as long as they keep the DQS-DQ association rule.
- It is recommended that the CK/CK# outputs be located on the same side where the DQ and DQS pads are located to minimize the skew.
- Place the address, command, and control signals either on the same side as where the DQ and DQS pads are located or they can be placed on the top side for DDR2, DDR3, and DDR3L memory interfaces.
- In DDR3 interface the RST# can be located anywhere an output is available as long as the same I/O Standard as the memory interface is applicable.

- The input reference clock to the PLL must be assigned to use dedicated clock routing. The dedicated PLL input pads are recommended while PCLK inputs can also be used.
- VREF1 of the bank where the DQ, DQS, and DM pads are located must be available to be used as a reference voltage input.
  - Do not assign an I/O signal to VREF1 in the preference file. VREF1 for the bank has to be available for the DDR memory interface.
  - Unused VREF1 can be taken as a general purpose I/O in the bank where no DQ/DQS pad is located.

6.7. Pin Placement Considerations for Improved Noise Immunity

In addition to the general pinout guidelines, you need to pay attention to additional pinout considerations to minimize simultaneous switching noise (SSN) impact. The following considerations are generally necessary to control SSN within the required level:

- Properly terminated interface
- SSN optimized PCB layout
- SSN considered I/O pad assignment
- Use of pseudo power pads

The guidelines listed below address the I/O pad assignment and pseudo power pad usage. Unlike the pinout guidelines, they are not absolute requirements. However, it is recommended that the pin placement follow the guidelines as much as possible to increase the SSO/SSI immunity.

- Place the DQS groups for data implementation starting from the middle of the (right or left) edge of the ECP5 and ECP5-5G device. Allow a corner DQS group to be used as a data group only when necessary to implement the required width.
- Locate a spacer DQS group between the data DQS groups if possible. A DQS group becomes a spacer DQS group if the I/O pads inside the group are not used as data pads (DQ, DQS, DM).
  - In DDR2, DDR3, and DDR3, the pads in a spacer group can be used for address, command, control or CK pads as well as for user logic or the pseudo power pads.
  - It would provide better noise immunity if no more than two data DQS groups are consecutively placed. If more data DQS groups need to be placed consecutively, use the pseudo power pads as many as possible to isolate each DQS group more effectively from others.
- It is recommended that you locate a few pseudo VCCIO/ground (GND) pads inside a spacer DQS group and at least one pseudo VCCIO in the data DQS group. An I/O pad becomes a pseudo power pad when it is configured to OUTPUT with its maximum driving strength (that is, SSTL15, 10 mA for DDR3) and connected to the external VCCIO or ground power source on the PCB.
  - Your design needs to drive the pseudo power I/O pads according to the external connection. (that is, you assign them as OUTPUT and let your design drive *1* for pseudo VCCIO pads and *0* for pseudo GND pads in your RTL coding.)
  - Locating two to four pseudo power pads in a spacer DQS group should be sufficient to provide suppressing the SSN impact.
  - Locate a pseudo power pad in a location where it can provide the best balanced and isolated separation.
- You may have one or more remaining pads in a data DQS group which are not assigned as a data pad in a DDR memory interface. Assign them to pseudo VCCIO or pseudo GND. Preferred location is in the middle of the group (right next to a DQS pad pair) if the DQS group is isolated by a spacer DQS group. If consecutively placed, locating the pseudo power pads to the edge of the group may be more effective. Note that you may not have this extra pad if the DQS group has 12 pins only and includes a VREF pad for the bank.

The additional guidelines below are not as effective as the ones listed above. However, following them is still recommended to improve the SSN immunity further:

- Assign the DM (data mask) pad in a data DQS group close to the other side of DQS pads where a pseudo powerm pad is located. If the data DQS group includes VREF1, locate DM to the other side of VREF with respect to DQS. It can be used as an isolator due to its almost static nature in most applications.
- Other DQS groups (neither data nor spacer group) can be used for accommodating DDR memory interface's address, command, control, and clock pads.
- You can assign more unused I/O pads to pseudo power if you want to increase the SSN immunity. Note that the SSN immunity does not get increased at the same rate as the increased number of pseudo power pads. The first few pseudo power pad placements described above are more crucial. Keep the total pseudo power pad ratio (VCCIO vs. GND) between 2:1 to 3:1.
- It is a good idea to shield the VREF pad by locating pseudo power pads around it if extra pins are available in the bank where the VREF1 pad is not located.

7. Using Clarity Designer to Build and Plan High Speed DDR Interfaces

The Clarity Designer tool is used to configure, build, and plan placement all DDR interfaces. This section covers how Clarity Designer is used to configure and plan placement for the DDR interfaces. In addition to building and planning Clarity Designer can be used to build top level modules by connecting the modules together in the Builder.

For step by step assistance with Clarity, please refer to the Clarity Designer User Manual. Clarity Designer can be opened from the Tools menu in Project Navigator. Figure 7.1 shows a Clarity Design Project which includes following three tabs:

- **Catalog** – Used to Configure the DDR Modules.
- **Builder** – Used to Build the HDL file that includes all the DDR modules configured.
- **Planner** – Used to Plan the Placement of the various DDR Interfaces.

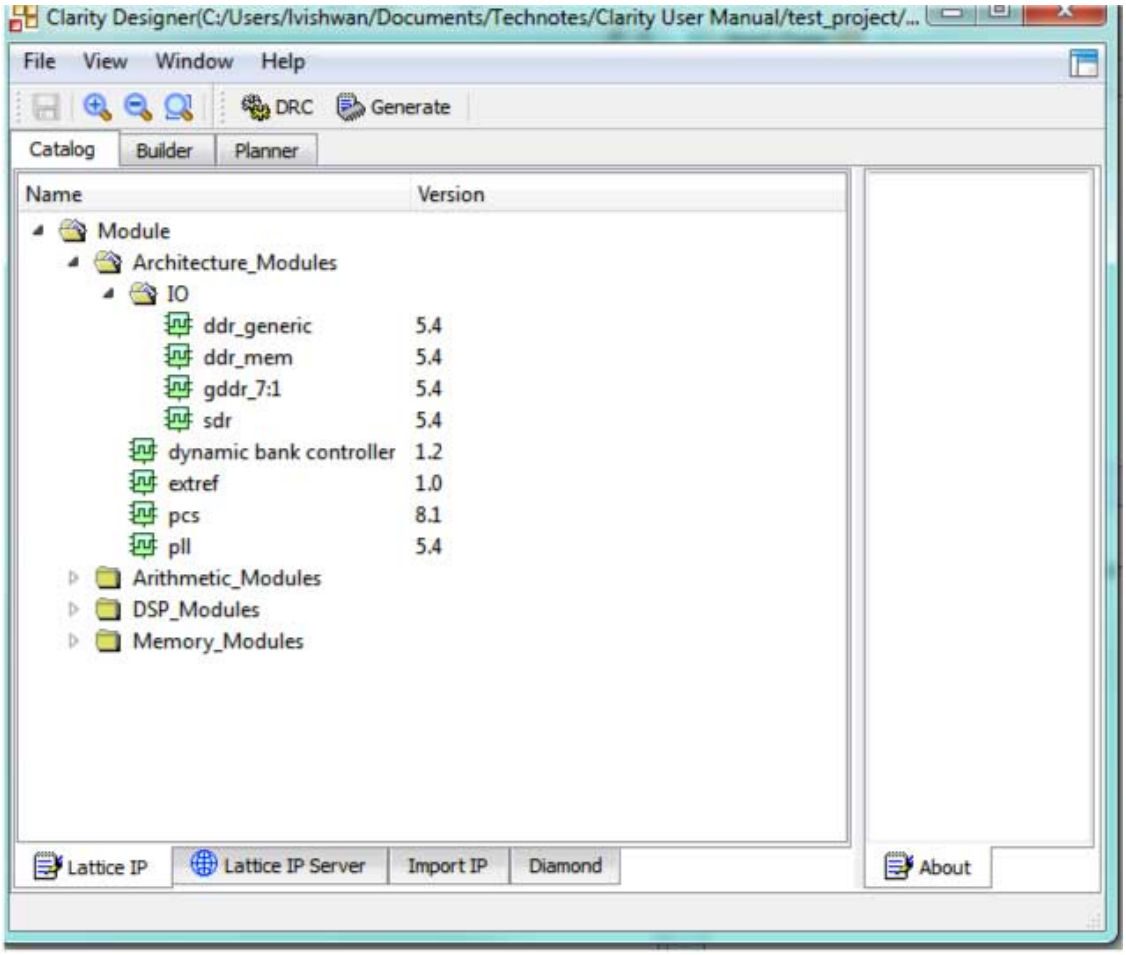

Figure 7.1. Clarity Design Main Window

Note: It is recommend that all the DDR modules required for the current design be generated in the same Clarity Design project. This allows for the Design Rule Checking as well as Resource Conflict Checking for all the modules at the same time.

7.1. Configuring DDR Modules in Clarity Designer

The catalog section of Clarity Design lists all the DDR architecture modules available on ECP5 and ECP5-5G.

All the DDR modules are located under Architecture Modules – I/O. This includes:

- **SDR** – Select to build SDR Modules.
- **DDR_GENERIC** – Select to build any DDR Generic Receive and Transmit Interfaces.
- **GDDR_7:1** – Select to build 7:1 LVDS Receiver and Transmit Interface.
- **DDR_MEM** – Select to build DDR Memory Interfaces.

To see the detailed block diagram for each interface generated by Clarity Designer see the High-Speed DDR Interface Details section.

7.2. Configuring SDR Modules

To build and SDR interface, select **SDR** option under Architecture Modules – I/O in the Catalog tab of Clarity Designer. Enter the name of the module. Figure 7.2 shows the type of interface selected as SDR and module name entered. This module can then be configured by clicking the **Customize** button.

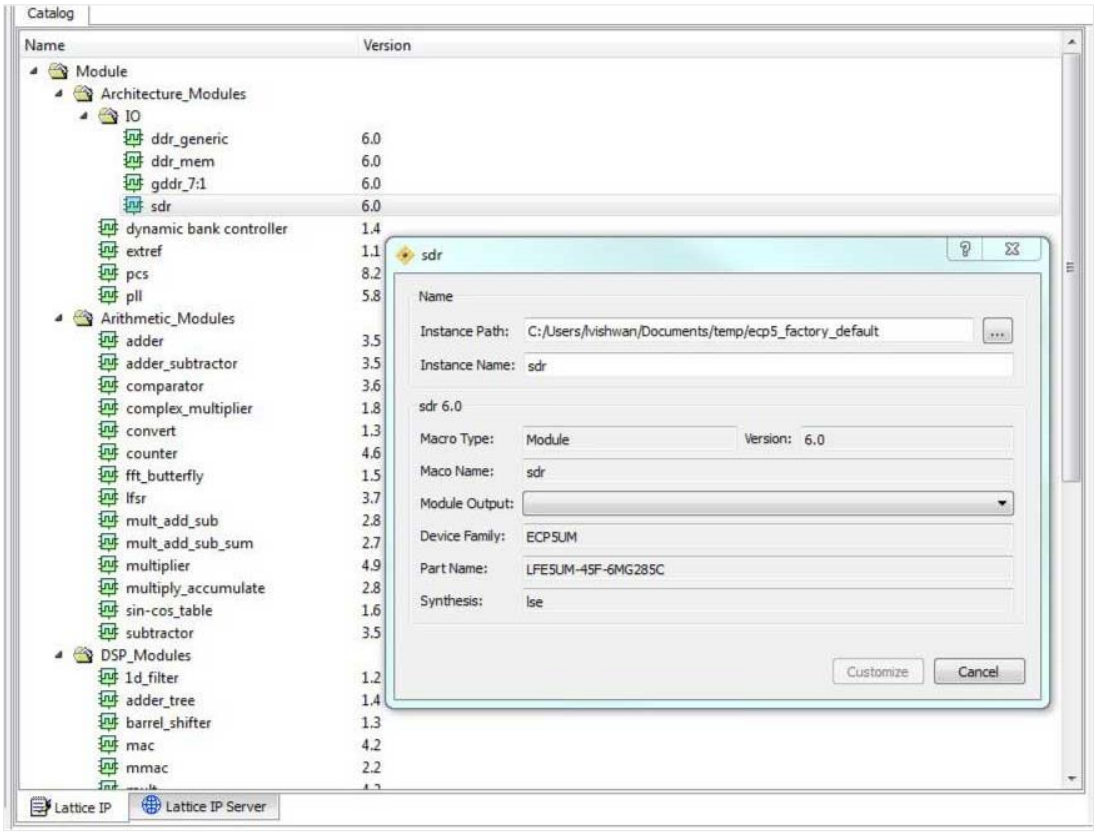

Figure 7.2. SDR Option Selected in the Catalog Tab of Clarity Designer

Figure 7.3 shows the Configuration tab for SDR module. You can make selections and then click **Configure**.

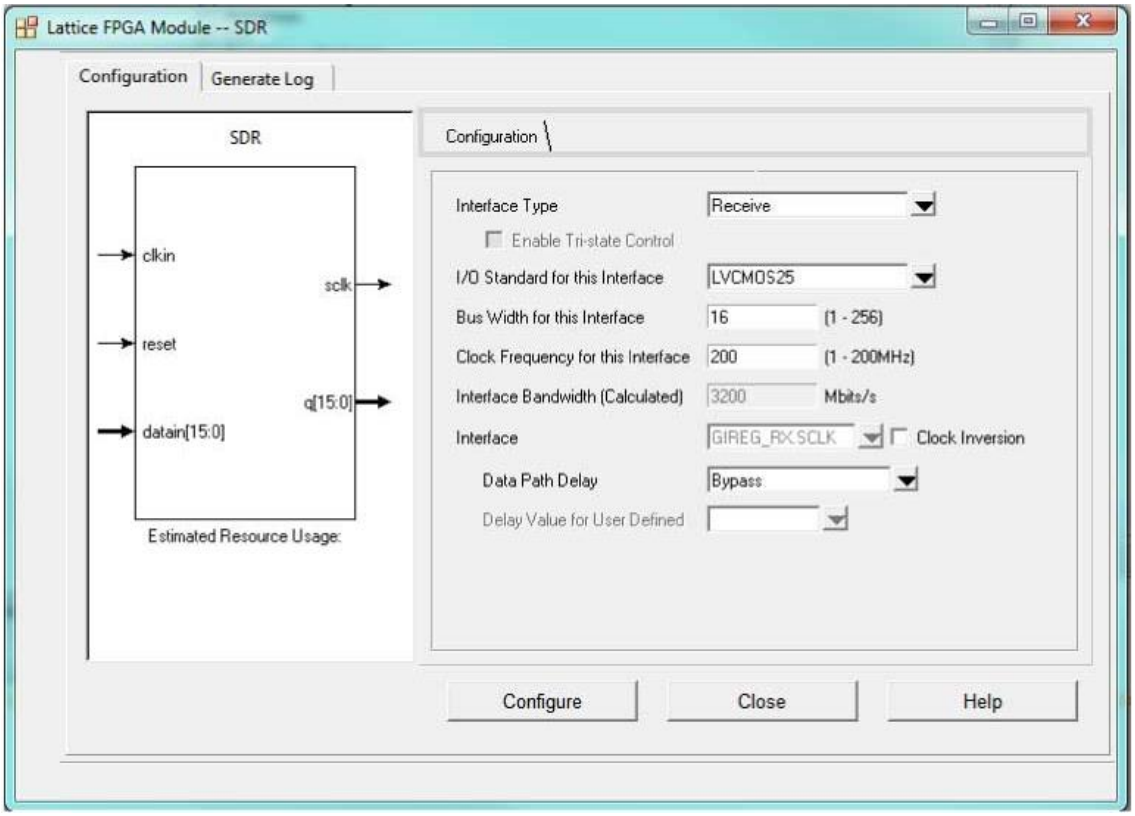

Figure 7.3. SDR Configuration Tab

Table 7.1 explains the various configurations options available for SDR modules.

Table 7.1. SDR Configuration Parameters

GUI Option	Description	Values	Default
Interface Type	Type of Interface (Transmit or Receive)	Transmit, Receive	Receive
I/O Standard for this Interface	I/O Standard to be used for the interface	All Valid IO_TYPES	LVCMOS25
Bus Width for this Interface	Bus size for the interface	1 — 256	16
Clock Frequency for this Interface	Interface Speed	1 — 200	200
Bandwidth (Calculated)	This is the calculated from the Clock frequency entered.	(Calculated)	(Calculated)
Interface	Interface selected based on previous entries	Transmit: GOREG_TX.SCLK Receive: GIREG_RX.SCLK (default)	GIREG_RX.SCLK
Clock Inversion	Option to invert the clock Input to the I/O Register	DISABLED, ENABLED	DISABLED
Data Path Delay*	Data input can be optionally delayed using the DELAY block.	If Interface Type = Receive then: Bypass, Static Default Dynamic Default Static User Defined Dynamic User Defined If Interface Type = Transmit then: Bypass, Static User Defined Dynamic User Defined	Bypass
FDEL for User Defined	If Delay type selected above is *user defined*, delay values can be entered with this parameter	0 to 127	0

***Note**: When Data Path Delay value is:

- Bypass: No delay cell is used.
- Static Default: Static delay element DELAYG is used with attribute DEL_MODE set to SCLK_ZEROHOLD.
- Static User Defined: Static delay element DELAYG is used with attribute DEL_MODE set to USER_DEFINED.
- Dynamic Default: Dynamic delay element DELAYF is used with attribute DEL_MODE set to SCLK_ZEROHOLD.
- Dynamic User Defined: Dynamic delay element DELAYF is used with attribute DEL_MODE = USER_DEFINED.

7.3. Configuring DDR Generic modules

To build a DDR Generic interface, select the DDR_Generic option under Architecture Modules – I/O in the Catalog tab of Clarity Designer. Enter the name of the module. Figure 7.4 shows the type of interface selected as DDR_Generic and module name entered. This module can then be configured by clicking the **Customize** button.

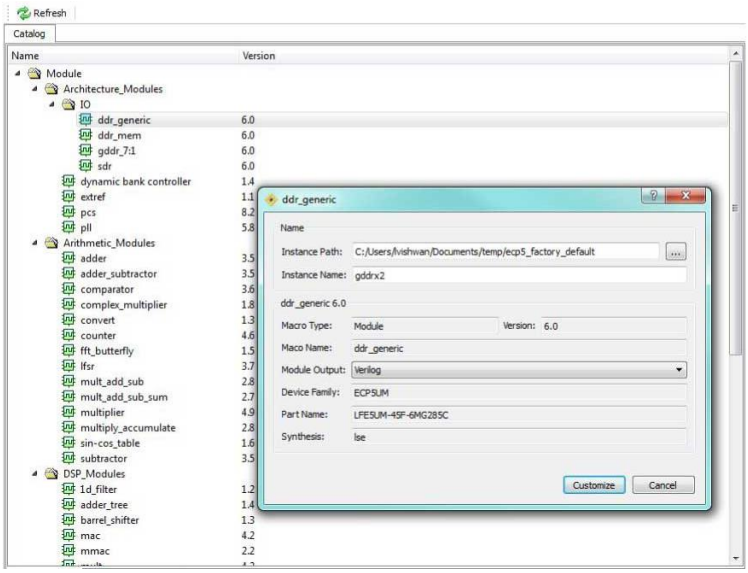

Figure 7.4. DDR_Generic Option Selected in the Catalog Tab of Clarity Designer

When clicking Customize, DDR modules have a Pre-Configuration tab and a Configuration tab. The Pre-Configuration tab allows you to enter information about the type of interface to be built. Based on the entries in the Pre- Configuration tab, the Configuration tab is populated with the best interface selection. You can also, if needed, override the selection made for the interface in the Configuration tab and customize the interface based on the design requirement.

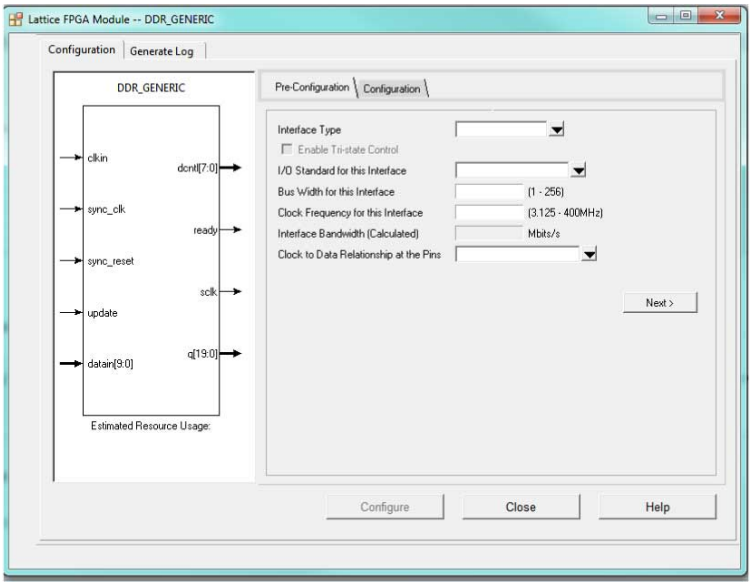

Figure 7.5. DDR_Generic Pre-Configuration Tab

Figure 7.5 shows the Pre-Configuration tab for DDR generic interfaces. The Table 7.2 explains the various parameters in this tab.

Table 7.2. DDR_Generic Pre-Configuration Parameters

GUI Option	Range
Interface Type	Transmit, Receive, Receive MIPI
I/O Standard for this Interface	List of Legal Input or Output Standards
Enable Tristate Control	Enabled, Disabled
Bus Width for this Interface	1 — 256
Clock Frequency for this Interface	3.125 — 400 MHz
	200 — 400 MHz (or Receive MIPI)
Interface Bandwidth (Calculated)	Clock Frequency for *2* Bus Width
Clock to Data Relationship at the Pins	Edge-to-Edge, Centered
	Centered (for Receive MIPI)

Based on the selections made in the Pre-Configuration tab, the Configuration tab is populated with the selections.

Figure 7.6 shows the Configuration tab for the selection made in Pre-Configuration tab.

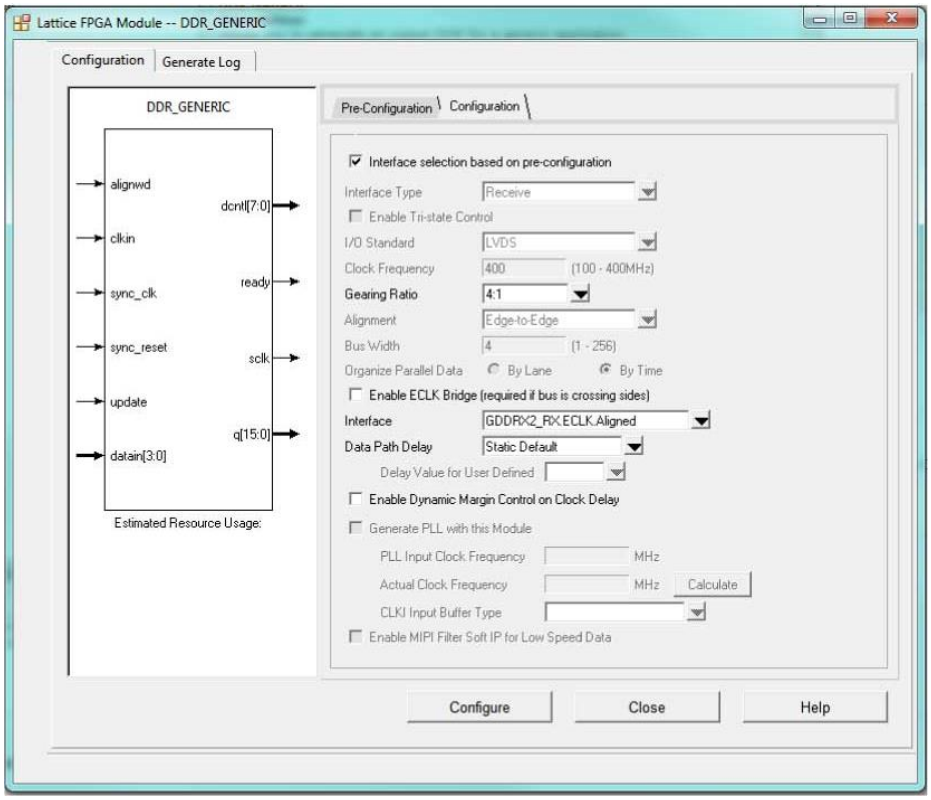

Figure 7.6. DDR_Generic Configuration Tab

The check box on the top of this tab indicates that the interface is selected based on entries in the Pre-Configuration tab. You can choose to change these values by disabling this entry. The best suitable interface is picked based on the selections made in the Pre-Configuration tab.

Table 7.3 explains the various parameters in the Configuration tab.

Table 7.3. DDR_Generic Configuration Tab Parameters

User Interface Option	Description	Values	Default
Interface selection based on preconfiguration	Indicates interface is selected based on selection made in the Pre-configuration tab. Disabling this checkbox allows you to make changes if needed.	ENABLED, DISABLED	ENABLED
Interface Type	Type of Interface (Transmit or Receive)	Transmit, Receive, Receive MIPI	Receive
Enable Tristate Control	Generate Tristate control for Transmit Interfaces	ENABLED, DISABLED	DISABLED
I/O Standard	I/O Standard used for the interface	All Legal Input & Output standards	LVCMOS25
Clock Frequency	Speed of the Interface	100 MHz — 400 MHz (Transmit) 3.125 MHz — 400 MHz (Receive) 200 MHz — 400 MHz (Receive MIPI)	200 MHz
Gearing Ratio	DDR register gearing ratio	2:1, 4:1	2:1
Alignment	Clock to Data alignment	Edge-to-Edge or Centered	Centered
Bus Width	Bus width for each interface	1 — 256	10
Organize Parallel Data	Allows you to select how the data bits of the parallel bus should be arranged. You can choose to set it By Lane where all the parallel data bits from each lane are organized together in the data output. If *By Time* is chosen instead, a single bit from each of the data lanes is put together in the data output.	By Lane, By Time	By Time
Enable ECLK Bridge (required if bus is crossing sides)	This is required if the data bus is wide and is crossing sides. Enabling this option instantiates the ECLK Bridge module in the HDL.	Enable, Disable	Disable
Interface	Shows list of all valid high speed Interfaces for the given configuration.	Refer to the table below to see interfaces available for a given configuration.	—
Data Path Delay	Data input can be optionally delayed using the DELAY block. Default Value is selected based on Interface Type.	If Interface Type = Receive: Static Default Dynamic Default Static User Defined Dynamic User Defined If interface type = Receive MIPI Static Default Static User Defined If Interface Type = Transmit: Bypass Static User Defined Dynamic User Defined	If Interface Type = Receive: Static Default If Interface Type = Transmit: Bypass
Delay Value for User Defined	When Data Path Delay of user defined is selected, you also need to set the number of delay steps to be used.	1 — 127	1
Enable Dynamic Margin Control on Clock Delay	Allows dynamic user control on clock phase shift for Receiver edge to edge aligned interfaces.	Enable, Disable	Disable
Generate PLL with this module	When is option is enabled for Transmit interfaces, the PLL used to generate the clocks is included in the generated module.	Enable, Disable	Disable
PLL Input Clock Frequency	Frequency of the clock used as PLL Input	10 MHz — 400 MHz	—

User Interface Option	Description	Values	Default
Actual Clock Frequency	Displays the achieved PLL output clock frequency	Actual PLL output Frequency achieved based on interface requirement	—
CLKI Input Buffer Type	The I/O Standard for the PLL Reference Clock	List of Legal Input Standards, None (if coming from fabric)	LVCMOS25
Enable MIPI Filter Soft IP for Low Speed Data	Generates the MIPI Filter soft IP in module for Interface = Receiver MIPI	Enable, Disable	Disable

Table 7.4 shows how the interfaces are selected by Clarity Designer based on the selections made in the Pre-Configuration tab.

Table 7.4. Clarity Designer DDR_Generic Interface Selection

Interface Type	Gearing Ratio	Alignment	Default Interface
Receive	2:1	Edge-to-Edge	GDDRX1_RX.SCLK.Aligned
Receive	2:1	Centered	GDDRX1_RX.SCLK.Centered
Receive	4:1	Edge-to-Edge	GDDRX2_RX.ECLK.Aligned
Receive	4:1	Centered	GDDRX2_RX.ECLK.Centered
Receive MIPI	4:1	Centered	GDDRX2_RX.ECLK.MIPI
Transmit	2:1	Edge-to-Edge	GDDRX1_TX.SCLK.Aligned
Transmit	2:1	Centered	GDDRX1_TX.SCLK.Centered
Transmit	4:1	Edge-to-Edge	GDDRX2_TX.ECLK.Aligned
Transmit	4:1	Centered	GDDRX2_TX.ECLK.Centered

Refer to the High-Speed DDR Interface Details section to see implementation details for each of these interfaces.

7.4. Configuring 7:1 LVDS Interface Modules

To build a 7:1 LVDS DDR interface, select GDDR_7:1 option under Architecture Modules – I/O in the Catalog tab of Clarity Designer. Enter the name of the module.

Figure 7.7 shows the type of interface selected as GDDR_7:1 and module name entered. This module can then be configured by clicking the **Customize** button.

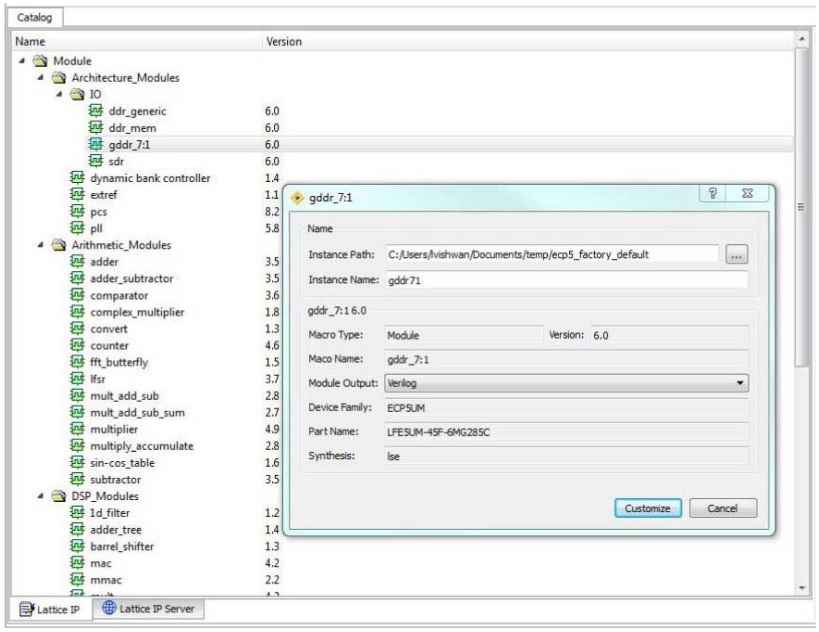

Figure 7.7. GDDR_7:1 Option Selected in the Catalog Tab of Clarity Designer

Clicking **Customize** displays the Configuration tab where the 7:1 LVDS interface can be configured. Figure 7.8 shows the Configuration tab for 7:1 LVDS interfaces.

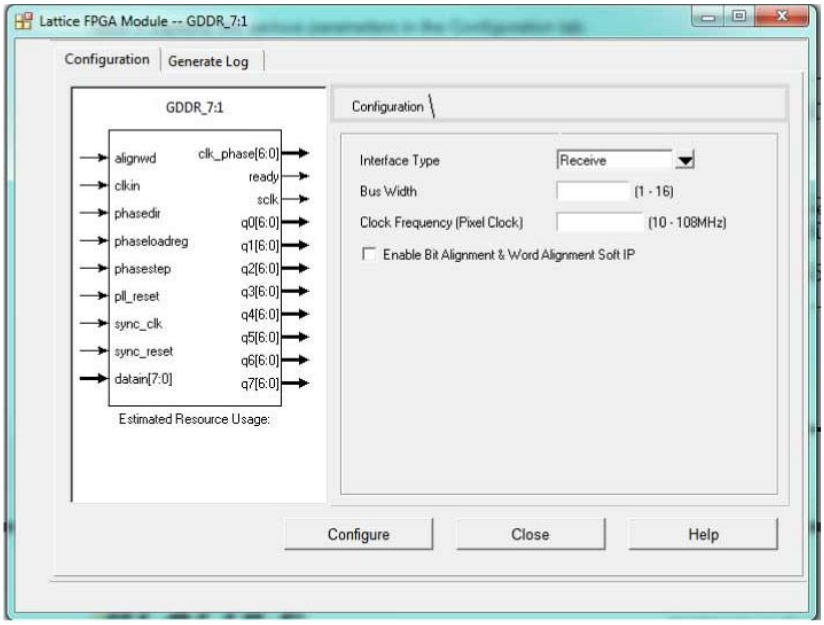

Figure 7.8. GDDR_7:1 LVDS Configuration Tab

Table 7.5 explains the various parameters in this tab.

Table 7.5. GDDR_7:1 LVDS Configuration Parameters

User Interface Option	Description	Values
Interface Type	Type of interface (Receive or Transmit)	Transmit, Receive
Bus Width	Bus width for 1 channel of 7:1 LVDS interface	1 — 16
Clock Frequency	Pixel clock speed	3.125 MHz — 108 MHz
Enable Bit Alignment & Word Alignment	Soft IP included with the module to implement Bit and Word alignment fo	Enable, Disable

7.5. Configuring DDR Memory Interfaces

Clarity Designer is used to configure the PHY portion of the DDR2, DDR3, DDR3L, LPDDR2, and LPDDR3 memory interfaces. For the detailed block diagram for each interface, see the Memory Interface Implementation section.

To build a DDR Memory interface, select DDR_MEM option under Architecture Modules – I/O in the Catalog tab of Clarity Designer. Enter the name of the module.

Figure 7.9 shows the type of interface selected as GDDR_MEM and module name entered. This module can then be configured by clicking the **Customize** button.

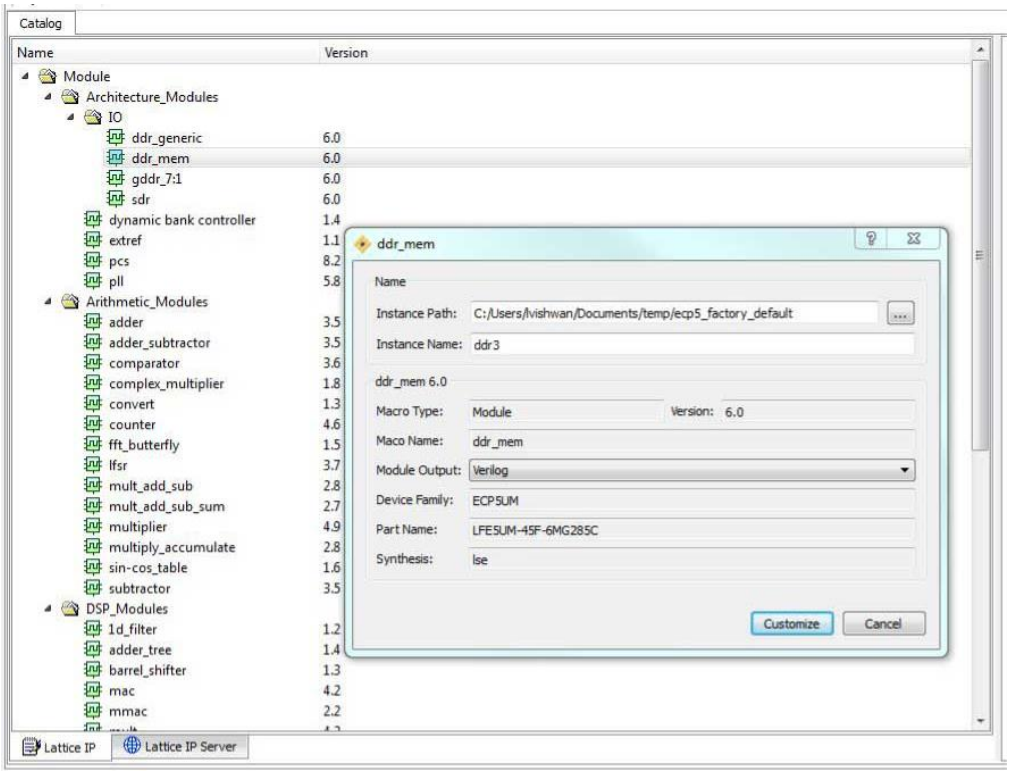

Figure 7.9. DDR_MEM Option Selected in the Catalog Tab of Clarity Designer

Figure 7.10 shows the Configuration tab for the DDR_MEM interface.

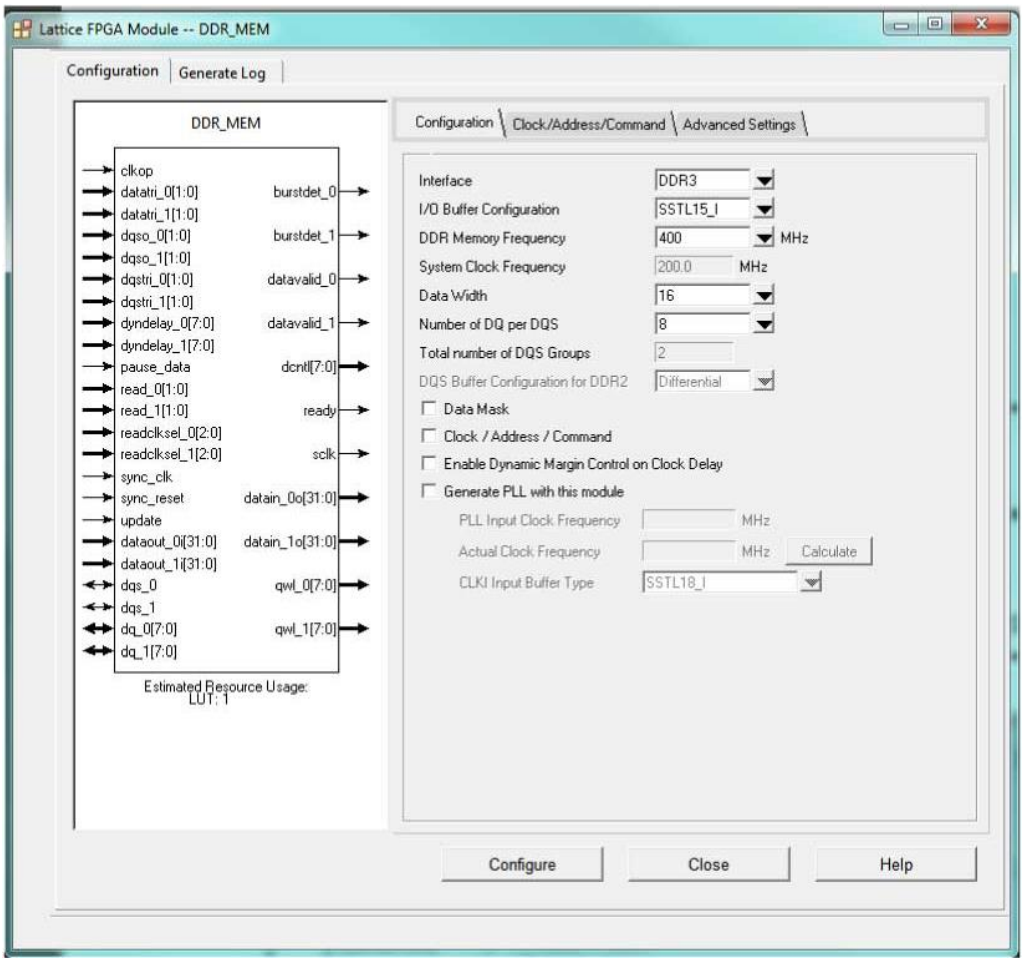

Figure 7.10. DDR_MEM Configuration Tab

Table 7.6 below describes the various settings shown in the Configuration tab.

Table 7.6. DDR_MEM Configuration Tab Parameters

User Interface Option	Description	Values	Default
Interface	DDR memory interface type	DDR2, DDR3, DDR3L, LPDDR2, LPDDR3	DDR2
I/O Buffer Configuration	I/O type configuration for DDR pins	DDR2: SSTL18_I, STL18_II DDR3: SSTL15_I, SSTL15_II DDR3L: SSTL135_I, SSTL135_II LPDDR2: HSUL12 LPDDR3: HSUL12	DDR2: SSTL18_I DDR3: SSTL15_I DDR3L: SSTL135_I LPDDR2: HSUL2 LPDDR3: HSUL12
DDR Memory Frequency	Target DDR memory interface frequency	DDR2: 125, 200, 267, 333, 400 (MHz) DDR3: 300, 400 DDR3L: 300, 400 LPDDR2: 125, 200, 267, 333, 400 (MHz) LPDDR3: 125, 200, 267, 333, 400 (MHz)	DDR2: 267 MHz DDR3: 400 MHz DDR3L: 400 MHz LPDDR2: 400 MHz LPDDR3: 400 MHz
System Clock Frequency	Calculated system clock frequency. Not user selectable, display only.	SCLK Frequency Value	DDR Memory Frequency/2
Data Width	DDR memory interface data width	DDR2, DDR3, DDR3L: 8, 16, 24, 32, 40, 48, 56, 64, 72 LPDDR2, LPDDR3: 16, 32	16
Number of DQ per DQS	Number of associated DQ per DQS pin	DDR2, DDR3, DDR3L: 4, 8 LPDDR2, LPDDR3: 8	8
Total number of DQS Groups	Total number of DQS groups. Not user selectable, display only.	Data width/number of DQ per DQS group	2
DQS Buffer Configuration for DDR2	DDR2 DQS I/O buffer type selection	Single-ended, Differential	Single-ended
Clock/Address/Command	Clock/address/command pins added with this option checked	ENABLED, DISABLED	DISABLED
Data Mask	Data mask pins added with this option checked	ENABLED, DISABLED	DISABLED
Enable Dynamic Margin Control on Clock Delay	Dynamic margin control ports added with this option checked	ENABLED, DISABLED	DISABLED
Generate PLL with this module	PLL included with this option checked	ENABLED, DISABLED	DISABLED
PLL Input Clock Frequency	Input reference clock frequency	10 MHz — 400 MHz	—
CLKI Input Buffer Type	The I/O Standard for the PLL Reference Clock	List of Legal Input Standards, None (if coming from fabric)	LVCMOS25
Actual DDR Memory Frequency	Calculated actual memory bus frequency. Not user selectable, display only.	—	—

If you choose to generate the Clock/Address/Command signals, then the settings in the Clock/Address/Command tab are active and can be set up as required.

Figure 7.11 shows the Clock/Address/Command tab of the DDR memory Catalog.

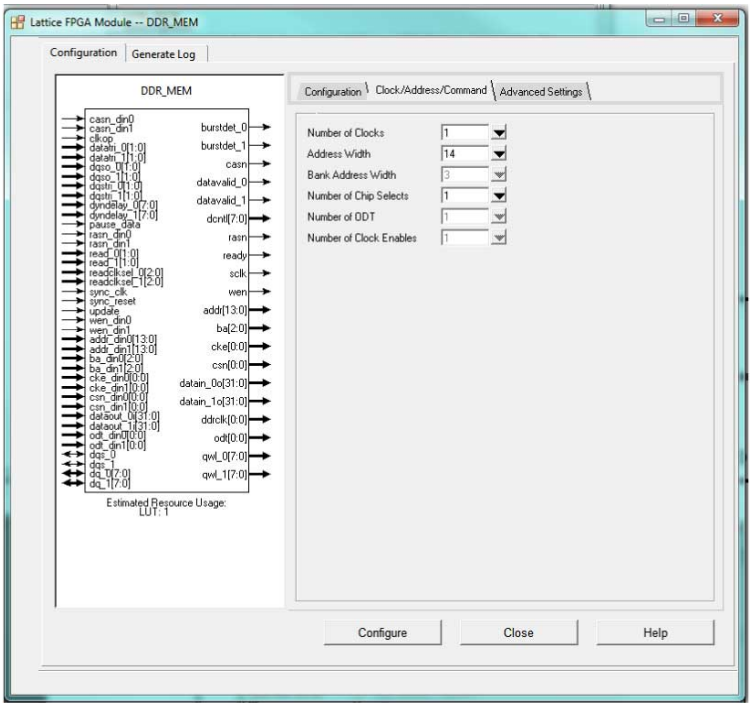

Figure 7.11. DDR_MEM Clock/Address/Command Tab

Table 7.7 lists the values that can be used for the Clock/Address/Command settings.

Table 7.7. DDR_MEM Clock/Address/Command Parameters

GUI Option	Range	Default Value
Number of Clocks	DDR2: 1, 2, 4 DDR3: 1, 2, 4 DDR3L: 1, 2, 4 LPDDR2: 1 LPDDR3: 1	DDR3: 1 DDR2: 1 DDR3L: 1 LPDDR2: 1 LPDDR3: 1
Address Width	DDR2: 13 – 16 DDR3: 13 – 16 DDR3L: 13 – 16 LPDDR2: Blank LPDDR3: Blank	DDR2: 13 DDR3: 14 DDR3L: 14 LPDDR2: Blank LPDDR3: Blank
Bank Address Width	DDR2: 2, 3 DDR3: 3 DDR3L: 3 LPDDR2: Blank LPDDR3: Blank	DDR2: 2 DDR3: 3 DDR3L: 3 LPDDR2: Blank LPDDR3: Blank
Number of Chip Selects	DDR2: 1, 2, 4 DDR3: 1, 2, 4 DDR3L: 1, 2, 4 LPDDR2: 1 LPDDR3: 1	DDR2: 1 DDR3: 1 DDR3L: 1 LPDDR2: 1 LPDDR3: 1
Number of Clock Enables	= Number of Chip Selects	= Number of Chip Selects
Number of ODT	DDR2, DDR3, DDR3L = Number of Chip Selects LPDDR2: Blank LPDDR3 = Number of Chip Selects	DDR2, DDR3, DDR3L = Number of Chip Selects LPDDR2: Blank LPDDR3 = Number of chip Selects

There is an additional tab called Advanced Settings for the ECP5 and ECP5-5G device that can be used to adjust the default DQS Read and Write Delay settings.

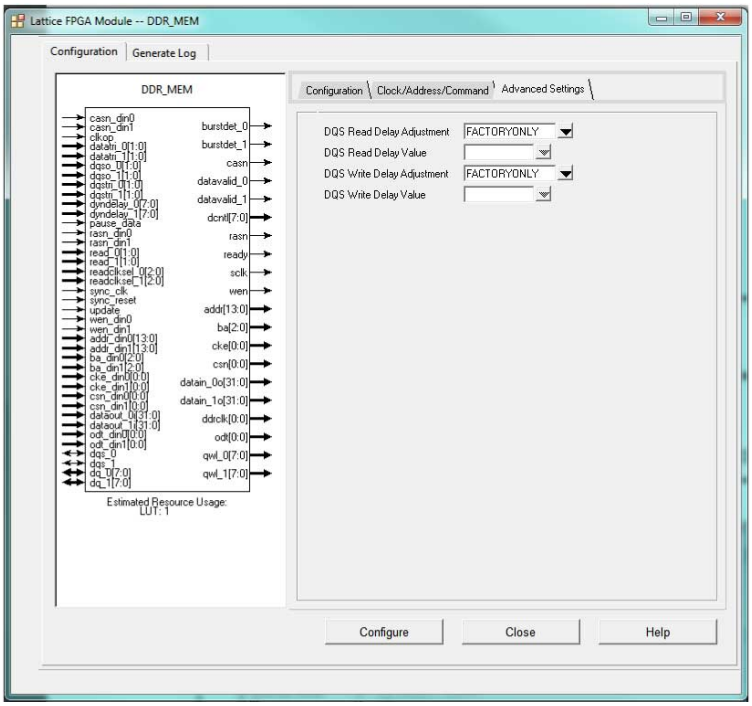

Figure 7.12. DDR_MEM Advanced Settings Tab

Figure 7.8 shows the available values in this tab.

Table 7.8. DDR_MEM Advanced Settings Tab Parameters

GUI Option	Range	Default Value
DQS Read Delay Adjustment	FACTORYONLY, PLUS, MINUS	FACTORYONLY
DQS Read Delay Value	Grey out (if DQS Delay Adjustment = FACTORYONLY) 0 — 255 (if DQS delay adjustment = PLUS) 1 — 256 (If DQS delay Adjustment = MINUS)	—
DQS Write Delay Adjustment	FACTORYONLY, PLUS, MINUS	FACTORYONLY
DQS Write Delay Value	Grey out (if DQS Delay Adjustment = FACTORYONLY) 0 — 255 (if DQS delay adjustment = PLUS) 1 — 256 (If DQS delay Adjustment = MINUS)	—

7.6. Building DDR Interfaces in Clarity Designer

After all the DDR Modules are configured, they can be connected up together in the Builder tab of Clarity Designer. The connections made in the *Builder* are carried over to the combined HDL file that contains all the DDR module instances. If the DDR modules were to share resources, the connections for the sharing can be done here.

For example, if a single PLL was shared among the different modules, then the clocks can be connected together in the Builder tab.

For step by step instructions on using the *Builder*, refer to the Clarity Designer User Manual.

7.7. Planning DDR Interfaces in Clarity Designer

Once the interface are configured and connected, the placement of these modules can be planned in the Planner tab of Clarity Designer.

The Planner allows you to drag and drop each DDR interface into the Chip View. This automatically locks the pins for that DDR interface at the selected location. The planner takes into account all the clocking and placement requirements, any architecture limitations for each type DDR interface. If any of the placement rules are violated, the planner does not place the module if there are not enough resources or it places it at the next available location.

This capability allows you to plan and place all the DDR interfaces before Synthesis. The placement constraints are carried over through the rest of the flow. Since all the design constraints are taken into account, it saves you a lot of time to not have to run multiple iterations through the tool.

For step by step instructions on using the Planner, refer to the Clarity Designer User Manual.

Figure 7.13 shows DDR modules that are placed using Clarity Design Planner.

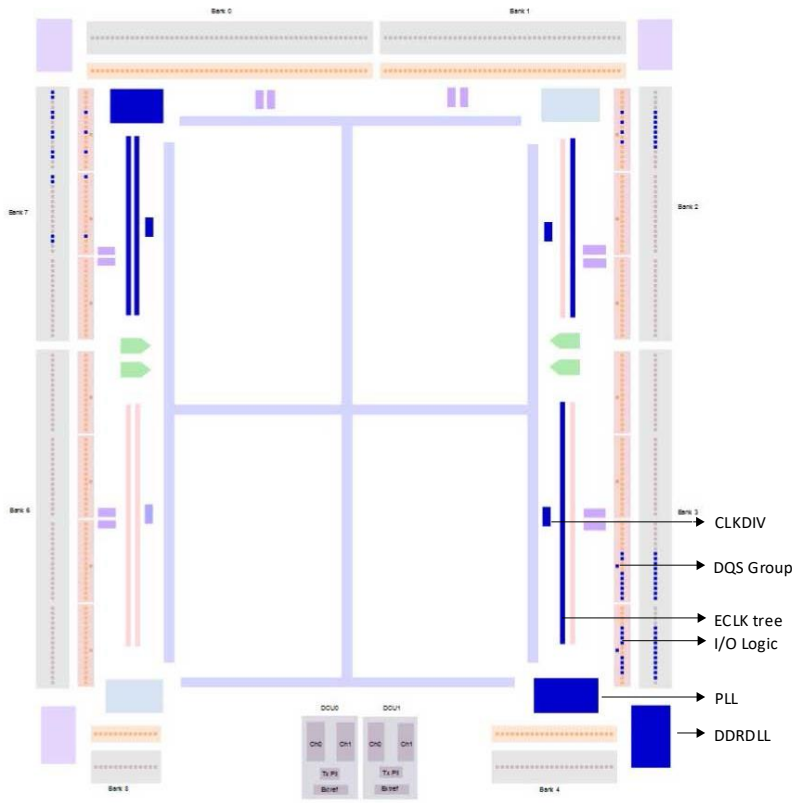

Figure 7.13. DDR Modules Paced Using Clarity Design Planner

8. DDR Software Primitives and Attributes

This section describes the software primitives that can be used to implement all the DDR interfaces. These primitives are divided into ones that are used to implement the DDR data and ones for DDR Strobe signal or the Source Synchronous clock. The DQSBUF primitives are used to generate the signals required to correctly capture the data from the DDR memory.

Table 8.1. Software Primitives

Type	Primitive	Description
Input or Output Delay	DELAYF	Dynamic Delay module
	DELAYG	Static Delay module, used to remove clock injection time
DDRDLL	DDRDLLA	DLL used to generate 90 degree phase shift
DLL Delay	DLLDELD	Slave Delay module used for Generic DDR
Generic DDR Data Input	IDDRX1F	Generic DDR 1x gearing registers
	IDDRX2F	Generic DDR 2x gearing registers
	IDDR71B	Generic DDR 7:1 gearing registers, shared by two I/O LOGIC blocks
Generic DDR Data Output	ODDRX1F	Generic DDR 1x gearing registers
	ODDRX2F	Generic DDR 2x gearing registers
	ODDR71B	Generic DDR 7:1 gearing registers, shared by two I/O LOGIC blocks
DDR Memory DQSBUF Control	DQSBUFM	Used to phase shift DQS Strobe signal and generate control signals for DDR memory
DDR Memory DQ Input	IDDRX2DQA	Used to receive DQ input
DDR Memory DQ Output	ODDRX2DQA	Used to transmit DQ output
DDR Memory DQS Output	ODDRX2DQSB	Used to transmit DQS output
DDR Memory Data Tristate	TSHX2DQA	Used for DQ Tristate output
DDR Memory DQS Tristate	TSHX2DQSA	Used for DQS Tristate output
DDR Memory Address/Command	OSHX2A	Used to generate the Address/Command output

8.1. Input/Output DELAY

The DELAY block can be used to delay the input data from the input pin to the IDDR or IREG or FPGA OR to delay the output data from the ODDR, OREG or FPGA fabric to the output pin. It is useful to adjust for any skews amongst the input or output data bus. It can also be used to generate skew between the bits of output bus to reduce SSO noise.

The DELAY block can be used with IDDR or ODDR modules, SDR module, as well as on the direct input to the FPGA. The DELAY is shared by the input and output paths and hence can only be used either to delay the input data or the output data on a given pin.

The data input to this block can be delayed using:

- Pre-determined delay value (for Zero Hold time, delay based on Interface Type)
- Fixed Delay values thet you enter

The data input to this block can also be dynamically updated using counter up and down controls. You can optionally bypass the DELAY block completely as well.

8.2. DELAYF

By default, the DELAYF is configured to factory delay settings based on the clocking structure. You can overwrite the DELAY setting using the MOVE and DIRECTION control inputs. The LOADN resets the delay back to the default value.

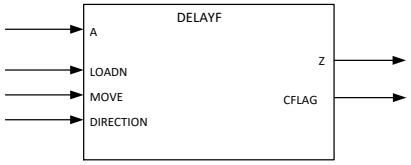

Figure 8.1. DELAYF Primitive

Table 8.2. DELAYF Port List

Port	I/O	Description
A	I	Data input from pin or output register block
LOADN	I	*0* on LOADN resets to default delay setting
MOVE	I	*Pulse* on MOVE changes delay setting. DIRECTION is sampled at falling edge of MOVE.
DIRECTION	I	*1* to decrease delay and '0' to increase delay
Z	O	Delayed data to input register block or to pin
CFLAG	O	Flag indicating the delay counter has reached the max (when moving up) or min (when moving down) value

8.3. DELAYG

By default, the DELAYG is configured to factory delay settings based on the clocking structure. You cannot change the delay when using this module.

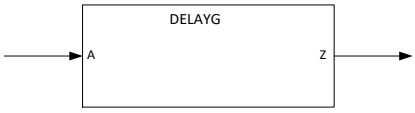

Figure 8.2. DELAYG Primitive

Table 8.3. DELAYG Port List

Port	I/O	Description
A	I	Data input from pin or output register block
Z	O	Delayed data to input register block or to pin

8.4. DELAY Attribute Description

Table 8.4 describes the attributes available for the DELAYF and DELAYG elements. The value of DEL_MODE is selected based on the interface that is generated. These values are used to compensate for the clock injection time, hence should be selected based on clocking used. IP Express automatically assigns the correct values for this attribute when Clarity Designer is used to build the interface.

Table 8.4. DELAYF and DELAYG Attributes

Attribute	Description	Values[1, 2, 3, 4]	Default	DELAY
DEL_MODE	Sets the delay mode to be used	USER_DEFINED SCLK_ZEROHOLD ECLK_ALIGNED ECLK_CENTERED SCLK_ALIGNED SCLK_CENTERED ECLKBRIDGE_ALIGNED ECLKBRIDGE_CENTER ED DQS_CMD_CLK DQS_ALIGNED_X2	USER_DEFINED	DELAYG DELAYF
DEL_VALUE	Sets delay value when DEL_MODE is set to USER_DEFINED	0..127	0	DELAYG DELAYF

Notes:

1. DEL_MODE must be ECLKBRIDGE_ALIGNED or ECLKBRIDGE_CENTERED when it is required to place the data pins of the same hispeed interface on the other side of the device. In addition to setting the DEL_MODE attribute it is also require to instantiate the ECLKBRIDGECS element in the HDL design.
2. DQS_CMD_CLK is only for the DDR Memory CMD and CLK Outputs.
3. DQS_ALIGNED_X2 is shared by DQS Generic and the DDR Memory Inputs.

8.5. DDRDLL (Master DLL)

The DDRDLL is used to generate a 90° delay for the DQS Strobe Input during a memory interface or for the clock input for a generic DDR interface.

There is one DDRDLL module on each corner of the device. The DDRDLL outputs delay codes that are used in the DQSBUF elements to delay the DQS input, or in the DLLDEL module to delay the input clock. DDRDLL, by default, generates 90° phase shift.

8.5.1. DDRDLLA

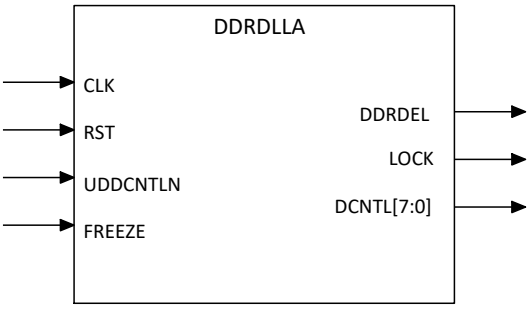

Figure 8.3. DDRDLLA Primitive

Table 8.5. DDRDLLA Port List

Port	I/O	Description
CLK	I	Reference clock input to the DDRDLL. Should run at the same frequency as the clock to be delayed.
RST	I	Reset input to the DDRDLL
UDDCNTLN	I	Update control to update the delay code. When low, the delay code out of the DDRDLL is updated. Should not be active during a read or a write cycle.
FREEZE	I	Releases the DDRDLL input clock
DDRDEL	O	The delay codes from the DDRDLL to be used in DQSBUF or DLLDEL
LOCK	O	Lock output to indicate the DDRDLL has valid delay output
DCNTL [7:0]	O	The delay codes from the DDRDLL available for the user IP.

Table 8.6. DDRDLL Attributes

Attribute	Description	Values	Default
FORCE_MAX_DELAY*	Bypass DLL locking procedure at low frequency	YES, NO	NO

***Note**: When Fin is =<30 MHz. The software sets Force_max_delay to YES. DDRDLL does not go through the locking process but is locked to maximum delay steps under such conditions.

8.6. DLL Delay (DLLDEL)

The DLLDEL receive delay codes from the DDRDLL and generates a delayed clock output.

The DLLDEL receives delay from the DDRDLL. The delayed clock output of the DLLDEL can be connected to the IDDR module.

The delay from the DLLDEL can be dynamically adjusted using counter margin control signals that can shift the delay up or down.DLLDELD

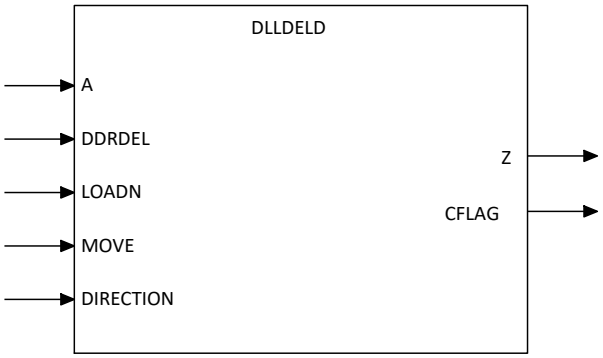

Figure 8.4. DLLDELD Primitive

Table 8.7. DLLDELD Port List

Port	I/O	Description
A	I	Clock input
DDRDEL	I	Delay inputs from DDRDLL
LOADN	I	Used to reset back to 90° delay.
MOVE	I	Pulse is required to change delay settings. The value on Direction is sampled at the falling edge of MOVE.
DIRECTION	I	Indicates delay direction. ‘1’ to decrease delay and ‘0’ to increase delay
CFLAG	O	Indicates the delay counter has reached its maximum value when moving up or minimum value when moving down.
Z	O	Delayed clock output

Table 8.8. DLLDELD Attributes

Attribute	Description	Values	Default
DEL_ADJ[2]	Sign bit for READ delay adjustment, DDR input	PLUS, MINUS	PLUS
DEL_VAL[2]	Value of delay for input DDR.	0 to 255 (PLUS) 1 to 256 (MINUS)	Note[2]

Notes:
1. Attributes are only available through EPIC and ECL Editor. It is recommended that values of this attribute are not updated without consulting Lattice Semiconductor Technical Support.
2. Default value is set based on device characterization to achieve the 90 degree phase shift.

8.7. Generic DDR Input and Output Primitives

The ECP5 and ECP5-5G device IDDR/ODDR modules support 2:1, 4:1 and 7:1 gearing modes on the left and right sides only. IDDR/ODDR modules on the top (and bottom for non-SERDES parts) only supports 2:1 due to lack of Edge Clocks. The 2:1 is available on each pin. The 4:1 gearing IDDR/ODDR is available on each pin on the left and right. 7:1 gearing mode is only available per pin pair on the left and right. This means the DDR register of the N side pin is used to implement 7:1 mode and is not available to you. It is assumed that Generic DDR applications using 7:1 model uses a differential input so it would not require the DDR registers of the N side.

8.8. Input DDR Primitives

The following are the primitives used to implement various Generic DDR Input and Output data.

8.8.1. IDDRX1F

This primitive is used to receive Generic DDR with 1x gearing.

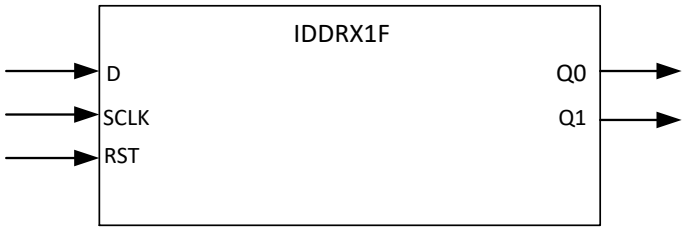

Figure 8.5. IDDRX1F Primitive

Table 8.9. IDDRX1F Port List

Port	I/O	Description
D	I	DDR data input
SCLK	I	Primary Clock input
RST	I	Reset to DDR registers
Q0	O	Data at the positive edge of the clock
Q1	O	Data at the negative edge of the clock

8.8.2. IDDRX2F

This primitive is used to receive Generic DDR with 2x gearing.

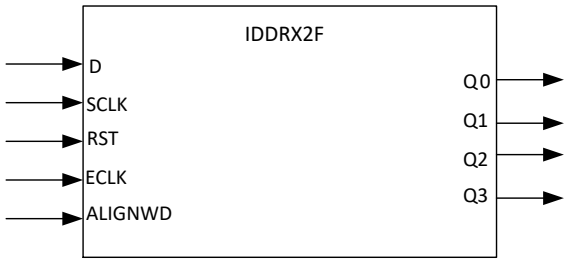

Figure 8.6. IDDRX2F Primitive

Table 8.10. IDDRX2F Port List

Port	I/O	Description
D	I	DDR data input
ECLK	I	Fast Edge Clock
SCLK	I	Primary Clock input (divide-by-2 of ECLK)
RST	I	Reset to DDR registers
ALIGNWD	I	This signal is used for word alignment. It shifts the word by one bit.
Q0, Q2	O	Data at positive edge of input ECLK
Q1, Q3	O	Data at negative edge of input ECLK

8.8.3. IDDR71B

This primitive is used for 7:1 LVDS input side implementation.

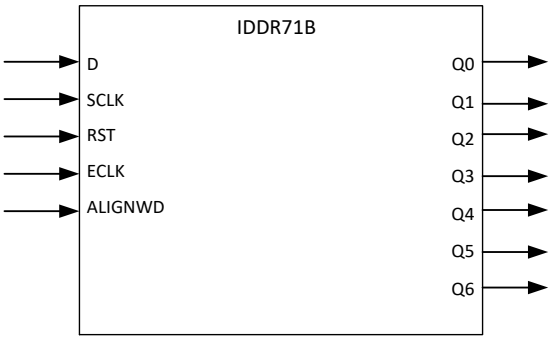

Figure 8.7. IDDR71B

Table 8.11. IDDRX2F Port List

Port	I/O	Description
D	I	DDR data input
ECLK	I	Edge Clock
SCLK	I	Primary Clock (divide-by-3.5 of ECLK)
RST	I	Reset to DDR registers
ALIGNWD	I	This signal is used for word alignment. It shifts the word by one bit.
Q0 to Q6	O	7 bits of output data.

8.9. Output DDR Primitives

The following are the primitives used to implement various Generic DDR output configurations.

8.9.1. ODDRX1F

This primitive is used to transmit Generic DDR with 1x gearing.

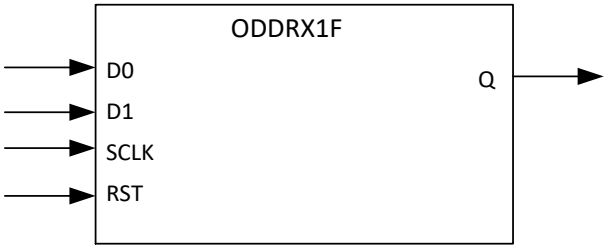

Figure 8.8. ODDRX1F

Table 8.12. ODDRX1F Port List

Port	I/O	Description
D0, D1	I	Parallel data input to ODDR (D0 is sent out first then D1)
SCLK	I	SCLK input
RST	I	Reset input
Q	O	DDR data output on both edges of SCLK

8.9.2. ODDRX2F

This primitive is used to receive Generic DDR with 2x gearing.

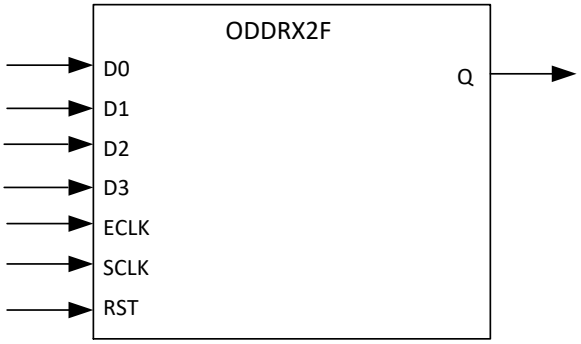

Figure 8.9. ODDRX2F

Table 8.13. ODDRX2F Port List

Port	I/O	Description
D0, D1, D2, D3	I	Parallel Data input to the ODDR (D0 is sent out first and D3 last)
ECLK	I	ECLK input (2x speed of SCLK)
SCLK	I	SCLK input
RST	I	Reset input
Q	O	DDR data output on both edges of ECLK

8.9.3. ODDR71B

This primitive is used for 7:1 LVDS ODDR implementation.

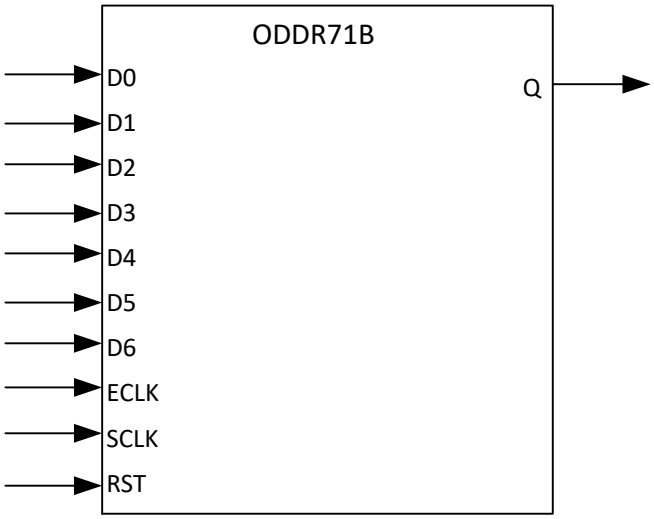

Figure 8.10. ODDR71B Primitive

Table 8.14. ODDR71B Port List

Port	I/O	Description
D0, D1, D2, D3, D4, D5, D6	I	Parallel data input to the ODDR (D0 is sent out first and D6 last)
ECLK	I	ECLK input (3.5x speed of SCLK)
SCLK	I	SCLK input
RST	I	Reset input
Q	O	DDR data output on both edges of ECLK

8.10. Memory DDR Primitives

This section describes the primitives used to build DDR2, DDR3, DDR3L, LPDDR2, and LPDDR3 memory interfaces.

8.10.1. DQSBUF (DQS Strobe Control Block)

DQSBUF block is used to delay the incoming DQS signal by 90°. DQSBUF receives a delay code from DDRDLL and shifts the signal accordingly. There is one DQSBUF block for every 16 I/O. The DQSBUF should be used when the DQS clock tree is used for clocking the IDDR module.

The following describes the functions of the DQSBUF module:

- Receives the delay code from the DDRDLL and generates the 90 delayed DQS signal that is used as a Write clock in the IDDR module
- You can choose to move the delay up or down using the dynamic margin control signals (loadn, move, and direction).
- When this margin control circuit is used and LOADN goes high, any further delay code changes from the DDRDLL are not reflected in the delay DQS signal. A soft IP is required to detect the code changes from the DDRDLL and update the MOVE pulse input to the DQSBUF so that the DDRDLL code changes can be tracked.
- If margin control is not used then LOADN should be low to continuously get code from DDRDLL.
- Pause should be asserted prior to changing readclksel, DYNDEL<> or DLL code update.
- Receives READ clock select signals that are used to correctly position the READ signal with respect to the DQS preamble

- Generates a BURSTDET output that can be used to validate the READ pulse positioning.
- Generates the Read and Write pointers required in the IDDR to correctly transfer data between the DQS and ECLK clock domains
- Generates DQS write clocks to be used in ODDR modules to generate DQ and DQS
- Generates the Write Leveling delay required for DDR3 or LPDDR3 interfaces

8.10.2. DQSBUFM

DQSBUFM element is used for all the DDR Memory interfaces.

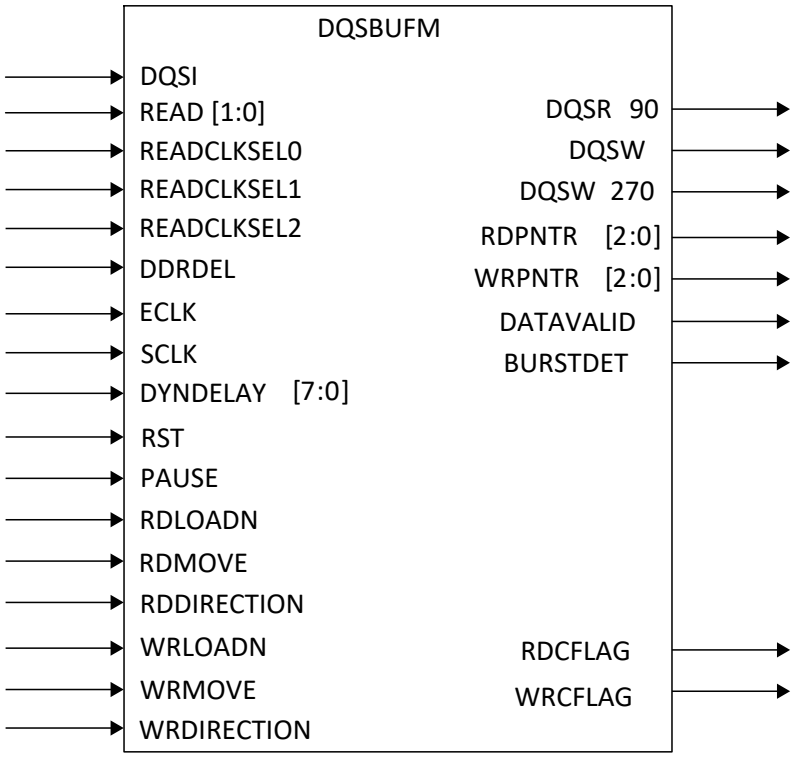

Figure 8.11. DQSBUFM Primitive

Table 8.15. DQSBUF Port List

Port	I/O	Description
DQSI	I	DQS input from the DQS pin
DDRDEL	I	Delay code from DDRDLL
ECLK	I	Edge Clock
SCLK	I	System Clock
RST	I	Reset input
READ[1:0]	I	Read input width for DQSBUFM
READCLKSEL0, READCLKSEL1, READCLKSEL2	I	Read clock pulse selection
DYNDELAY[7:0]	I	Dynamic Write leveling delay (only for DDR3)
PAUSE	I	Pause input to stop the DQSW/DQSW270 during write leveling or DDRDLL delay code change.
RDLOADN	I	Used to reset back to 90° delay for read side DQS
RDMOVE	I	Pulse is required to change delay settings. The value on Direction is sampled at *falling edge* of MOVE. Used to change delay on the read side DQS.

Port	I/O	Description
RDDIRECTION	I	Indicates delay direction. '1' decreases the delay count, '0' increases the delay count. Used to change delay on the read side DQS.
RDCFLAG	O	Indicates the delay counter has reached max value for the read side DQS delay.
WRLOADN	I	Used to reset back to 90° delay for write side DQSW270
WRMOVE	I	Pulse is required to change delay settings. The value on Direction is sampled at the falling edge of MOVE. Used to change delay on the write side DQSW270.
WRDIRECTION	I	Indicates delay direction. '1' decrease delay count, '0' increases the delay count. Used to change delay on the write side DQSW270.
WRCFLAG	O	Indicates the delay counter has reached the maximum value for the write side DQSW270 delay.
DQSR90	O	90° delay DQS used for read
DQSW270	O	90° delay clock used for DQ write
DQSW	O	Clock used for DQS write
RDPNTR[2:0]	O	Read pointer for IFIFO module
WRPNTR[2:0]	O	Write pointer for IFIFO module
DATAVALID	O	Signal indicating start of valid data
BURSTDET	O	Burst Detect indicator

Table 8.16. DQSBUFM Attributes

Attribute	Description	Values	Default	DELAY
DQS_LI_DEL_ADJ	Sign bit for READ delay adjustment, DDR input	PLUS, MINUS	PLUS	All
DQS_LI_DEL_VA	Value of delay for input DDR.	0 to 255 (PLUS) 1 to 256 (MINUS)	Note*	All
DQS_LO_DEL_ADJ	Sign bit for WRITE delay adjustment, DDR output	PLUS, MINUS	PLUS	All
DQS_LO_DEL_VAL	Value of delay for output DDR	0 to 255 (PLUS) 1 to 256 (MINUS)	Note*	All

***Note:** Default value is set based on device characterization to achieve the 90 degree phase shift.

8.11.Input and Output Memory DDR Primitives

The ECP5 and ECP5-5G device IDDR/ODDR modules support 4:1 (2x) gearing mode that are used to implement the memory functions.

Table 8.17 shows a summary of all the DDR memory primitives. See the sections below for detailed descriptions.

Table 8.17. Summary of all DDR Memory Primitives

DDR Memory	DQ Input	DQ Output	DQ Tristate	DQS Output	DQS Tristate	Addr/Cmd	Clock
DDR2 DDR3 DDR3L	IDDRX1DQA	ODDRX2DQA	TSHX2DQA	ODDRX2DQSB	TSHX2DQSA	ODDRX1F CS_N: OSHX2A	ODDRX2F
LPDDR2	IDDRX2DQA	ODDRX2DQA	TSHX2DQA	ODDRX2DQSB	TSHX2DQSA	ODDRX2DQA CS_N & CKE: ODDRX2DQA*	ODDRX2DQSB
LPDDR3	IDDRX2DQA	ODDRX2DQA	TSHX2DQA	ODDRX2DQSB	TSHX2DQSA	ODDRX2DQA CS_N, CKE & ODT: ODDRX2DQA1	ODDRX2DQSB

***Note:** The D0 and D1 inputs are tied together. The D2 and D3 inputs are also tied together.

8.12.Memory Input DDR Primitives

The following are the primitives used to implement various memory DDR input configurations.

8.12.1. IDDRX2DQA

This primitive is used to implement the DDR2 memory input interface at higher speeds and DDR3 memory interface.

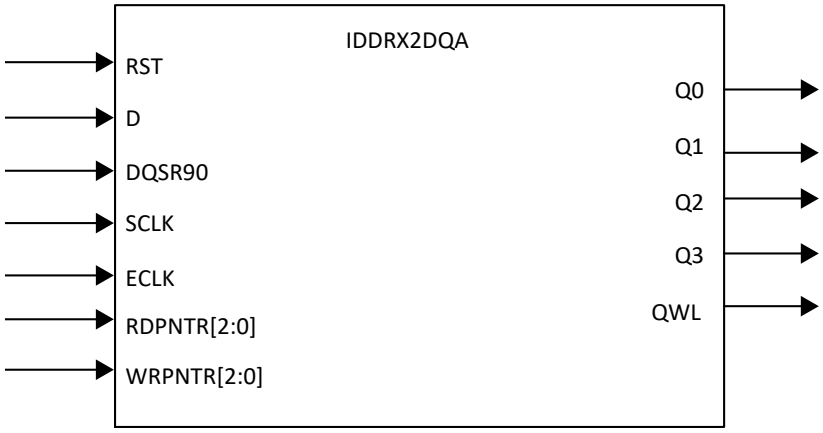

Figure 8.12. IDDRX2DQA Primitive

Table 8.18. DQSBUF Port List

Port	I/O	Description
D	I	DDR data input
RST	I	Reset to DDR registers
DQSR90	I	DQS clock Input
ECLK	I	Fast Edge Clock
SCLK	I	Primary Clock input (divide-by-2 of ECLK)
RDPNTR[2:0]	I	Read pointer from the DQSBUF module used to transfer data to ECLK
WRPNTR[2:0]	I	Write pointer from the DQSBUF module used to transfer data to ECLK
Q0, Q2	O	Data at positive edge of DQS
Q1, Q3	O	Data at negative edge of DQS
QWL	O	Data output used for write leveling

Table 8.19. Memory Primitive Attributes

Attribute	Description	Values	Default	Valid Primitives
REGSET	Set the Tristate register to either SET or RESET. By Default value is SET so that all output buffers are tristated by default	SET, RESET	SET	TSHX2DQA

8.13. Memory Output DDR Primitives for DQ Outputs

The following are the primitives used to implement various memory DDR output configurations to generate the DQ outputs.

8.13.1. ODDRX2DQA

This primitive is used to generate DQ data output for DDR2 with x2 gearing and for DDR3 memory interface.

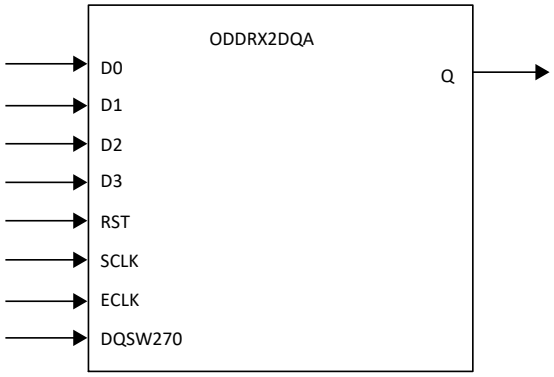

Figure 8.13. ODDRX2DQA

Table 8.20. ODDRX2DQA Port List

Port	I/O	Description
D0, D1, D2, D3	I	Data input to the ODDR (D0 is output first, D3 last)
ECLK	I	Fast Edge Clock input
DQSW270	I	Clock that is 90° ahead of clock used to generate the DQS output
SCLK	I	SCLK input
RST	I	Reset input
Q	O	DDR data output on both edges of DQSW270

8.14.Memory Output DDR Primitives for DQS Output

Following are the primitives used to implement the DQS outputs to the DDR memory.

8.14.1. ODDRX2DQSB

This primitive is used to generate DQS clock output for DDR2 and DDR3 memory.

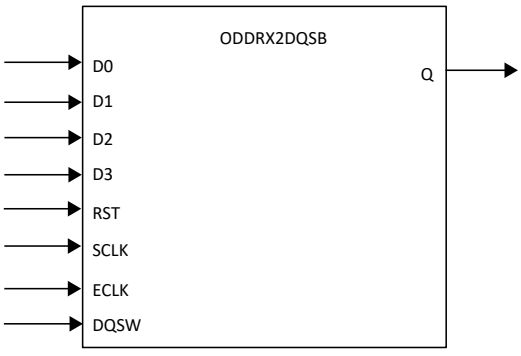

Figure 8.14. ODDRX2DQSB Primitive

Table 8.21. ODDRX2DQA Port List

Port	I/O	Description
D0, D1, D2, D3	I	Data input to the ODDR (D0 is output first, D3 last)
ECLK	I	ECLK input
DQSW270	I	SCLK input
SCLK	I	DQSW includes write leveling phase shift from ECLK
RST	I	Reset input
Q	O	DDR data output on both edges of DQSW

8.15.Memory Output DDR Primitives for Tristate Output Control

The following are the primitives used to implement tristate control for the outputs to the DDR memory.

8.15.1. TSHX2DQA

This primitive is used to generate the tristate control for DQ data output.

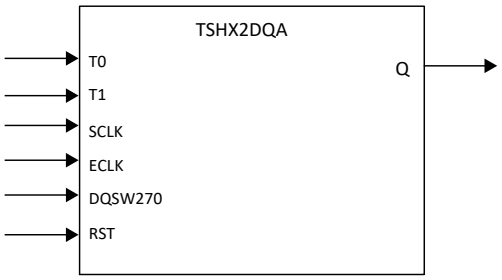

Figure 8.15. TSHX2DQA Primitive

Table 8.22. TSHX2DQA Port List

Port	I/O	Description
T0, T1	I	Tristate input (T0 is output first, followed by T1)
ECLK	I	ECLK input (2x speed of SCLK)
DQSW270	I	Clock that is 90° ahead of the clock used to generate the DQS output
SCLK	I	SCLK input
RST	I	Reset input
Q	O	Tristate output

8.15.2. TSHX2DQSA

This primitive is used to generate the tristate control for DQS output.

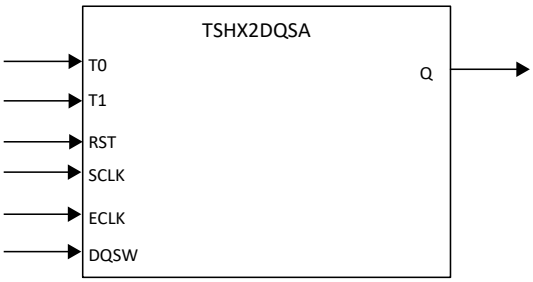

Figure 8.16. TSHX2DQSA Primitive

Table 8.23. TSHX2DQSA Port List

Port	I/O	Description
T0, T1	I	Tristate input (T0 is output first, followed by T1)
ECLK	I	ECLK input (2x speed of SCLK)
DQSW270	I	Clock that is 90° ahead of the clock used to generate the DQS output
SCLK	I	SCLK input
RST	I	Reset input
Q	O	Tristate output

8.16. Memory Output DDR Primitives for Address and Command

The following are the primitives used to implement the address and command outputs to the DDR memory.

8.16.1. OSHX2A

This primitive is used to generate the address and command for DDR3 memory with x2 gearing and write leveling.

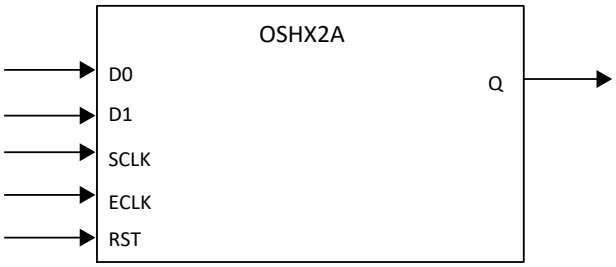

Figure 8.17. OSHX2A Primitive

Table 8.24. OSHX2A Port List

Port	I/O	Description
D0, D1	I	Data input (D0 is output first then D1)
ECLK	I	ECLK input (2x speed of SCLK)
SCLK	I	SCLK input
RST	I	Reset input
Q	O	Address and command output

9. Soft IP Modules

The following soft IP Modules are available for use with the Generic DDR interfaces described above. All of the soft IP Modules can be generated using Clarity Designer. Table 9.1 below summarizes the list of soft IPs available and the ones that are optional versus the ones that are automatically generated with the interface in Clarity Designer.

Table 9.1. List of Soft IPs supported

Soft IP Name	Function	Required
RX_SYNC	Used to break up the DDRDLL to DLLDEL clock loop for Aligned Interfaces	Yes
GDDR_SYNC	Needed to tolerate large skew between stop and reset input	Yes
MEM_SYNC	Needed to avoid issues on DDR memory bus and update code in operation without interrupting interface operation.	Yes
7:1 LVDS Bit and Word Alignment (BW_ALIGN)	The soft IP is used to perform bit and word alignment using PLL's dynamic phase shift interface and aligned input of IDDR71C.	Optional
MIPI_FILTER	Implements low pass filter on low speed MIPI data	Optional

Table 9.2 below summarized the soft IPs used in each interface.

Table 9.2. Soft IP Used in Each Interface

Interface	Soft IP
GDDRX1_RX.SCLK.Centered	None
GDDRX1_RX.SCLK.Aligned	GDDRX71_TX.ECLK
GDDRX2_RX.ECLK.Centered	GDDR_SYNC
GDDRX2_RX.ECLK.Aligned	RX_SYNC
GDDRX2_RX.MIPI	GDDR_SYNC, MIPI_FILTER
GDDRX71_RX.ECLK	GDDR_SYNC, BW_ALIGN
GDDRX1_TX.SCLK.Centered	None
GDDRX1_TX.SCLK.Aligned	None
GDDRX2_TX.ECLK.Centered	GDDR_SYNC
GDDRX2_TX.ECLK.Aligned	GDDR_SYNC
GDDRX71_TX.ECLK	GDDR_S

9.1. Detailed Description of Each Soft IP

9.1.1. GDDR_SYNC

This module is needed to startup al RX Centered and all TX interfaces with 2x gearing.

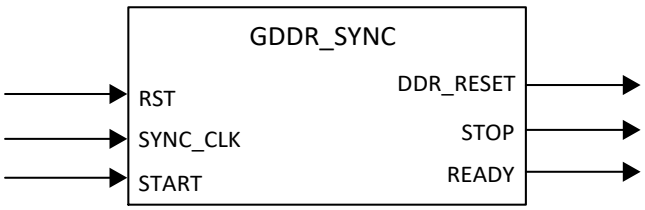

Figure 9.1. GDDR_SYNC Ports

Table 9.3. GDDR_SYNC Port List description

Port	I/O	Description
SYNC_CLK	I	Startup clock. This cannot be the RX_CLK or divided version. It can be other low speed continuously running clock. For example, oscillator clock
RST	I	Active high reset to this sync circuit. When RST=1, STOP=0, DDR_RESET=1, READY=0
START	I	Start sync process. This is used to wait for PLL lock, then start sync process in 7:1 LVDS interface
STOP	O	Connects to ECLKSYNC.STOP
DDR_RESET	O	Reset to all IDDRX or ODDRX components and CLKDIV
READY	O	Indicate that startup is finished and RX circuit is ready to operate

9.1.2. RX_SYNC

This module is needed to startup RX Aligned interfaces with 2x gearing.

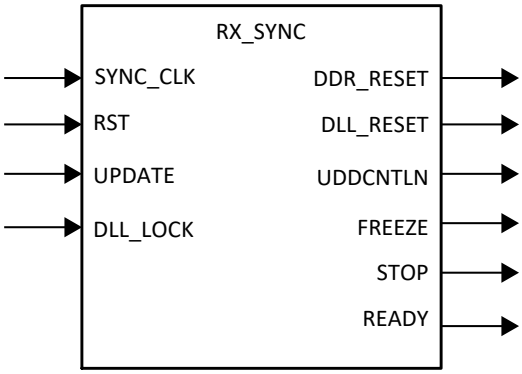

Figure 9.2. RX_SYNC Ports

Table 9.4. GDDR_SYNC Port List description

Port	I/O	Description
SYNC_CLK	I	Startup clock. This cannot be the RX_CLK or divided version. It can be other low speed continuously running clock. For example, oscillator clock.
RST	I	Active high reset to this sync circuit. When RST=1, STOP=0, FREEZE=0, UDDCNTLN=1, DLL_RESET=1, DDR_RESET=1, READY=0.
DLL_LOCK	I	LOCK output from DDRDLL
UPDATE	I	UPDATE can be used to re-start sync process. READY goes low and waits for the sync process to be completed before going high again. This can only be performed when no traffic is present.
STOP	O	Connect to ECLKSYNC.STOP
FREEZE	O	Connect to DDRDLL.FREEZE
UDDCNTLN	O	Connect to DDRDLL.UDDCNTLN
DLL_RESET	O	Reset to DDRDLL
DDR_RESET	O	Reset to all IDDRX components and CLKDIV
READY	O	Indicate that startup is finished and RX circuit is ready to operate.

9.1.3. MEM_SYNC

This module is needed to startup external memory controller interfaces with 2x gearing.

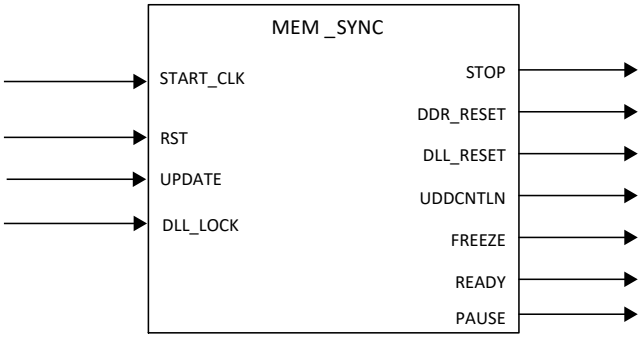

Figure 9.3. MEM_SYNC Ports

Table 9.5. MEM_SYNC Port Description

Port	I/O	Description
SYNC_CLK	I	Startup clock. This cannot be the RX_CLK or divided version. It can be other low speed continuously running clock. For example, oscillator clock.
RST	I	Active high reset to this sync circuit. When RST=1, STOP=0, FREEZE=0, UDDCNTLN=1, DLL_REEST=1, DDR_RESET=1, READY=0, PAUSE =0.
DLL_LOCK	I	LOCK output from DDRDLL
UPDATE	I	After ready goes high, you can use UPDATE to update code in DQSBUF, perform training (change read_clk_sel) or write leveling (change dyndelay<>).
PAUSE	O	Connect to DQSBUF.PAUSE
STOP	O	Connect to ECLKSYNC.STOP
FREEZE	O	Connect to DDRDLL.FREEZE
UDDCNTLN	O	Connect to DDRDLL.UDDCNTLN
DLL_RESET	O	Reset to DDRDLL
DDR_RESET	O	Reset to all IDDRX, ODDRX, OSHX components, DQSBUF, and CLKDIV
READY	O	Indicate that startup is finished and RX circuit is ready to operate.

9.1.4. BW_ALIGN

This module is used to perform 7:1 video RX bit and word alignment. This module is optional and can be enabled in Clarity Designer.

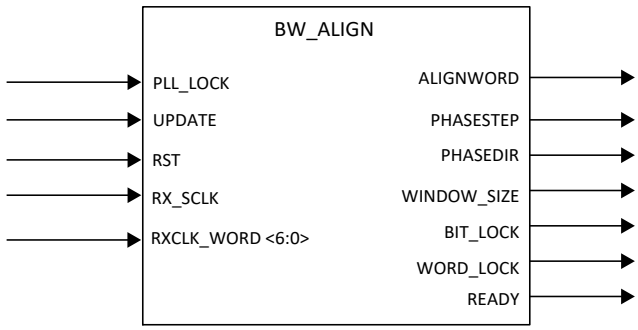

Figure 9.4. BW_ALIGN Ports

Table 9.6. BW_ALIGN Port Description

Port	I/O	Description
RX_SCLK	I	Divided RX clock from the 7:1 RX interface, produced by CLKDIV.
RST	I	Active high reset to this circuit. When RST=1, All outputs=0.
PLL_LOCK	I	Connect to PLL's LOCK output. Start the alignment procedures after PLL lock goes high.
UPDATE	I	Start the procedure, or re-start if need to optimize again.
RXCLK_WORD<6:0>	I	Parallel data output from the 2nd IDDRX71 attached to RX CLK Input.
PHASESTEP	O	Rotate phase for PLL.
PHASEDIR	O	Phase rotation direction for PLL, fixed to forward (0) for this design.
ALIGNWORD	O	Connect to IDDRX71.ALIGNWORD, for word rotation.
WINDOW_SIZE	O	Final valid window size.
BIT_LOCK	O	Status output, bit lock has been achieved.
WORD_LOCK	O	Status output, word lock has been achieved.
READY	O	Indicate that alignment procedure is finished and RX circuit is ready to operate.

With Bit Alignment, the goal is to place Edge Clock (under PLL dynamic phase shift control) to the center of valid window for the clock word and data words. The PLL phase rotation goes through all 16 phases. The PLL's high speed output is used to sample RX input clock. Transitions are detected on 2nd IDDR71 output which inputs the RX Clock and phases close to transition are identified. The IP chooses the phase most away from transition as the final phase to use.

The low speed clock has two transitions per 7-bit word. It is not the worst case in terms of inter-symbol interference. On the other hand, we do have 8 possible sample points per bit period. Minimum eye-opening of 3/8 UI is needed to achieve lock. Jitter tolerance is around 0.25 UI, about 300 ps at 756 Mb/sec.

After bit alignment is achieved, word alignment is needed so video data (in 7-bit words) can be processed in core. The IP uses the ALIGNWD function of the IDDRX71 primitive for word alignment. Each pulse on ALIGNWD rotates the 7-bit bus by 2 bits. In maximum 7 ALIGNWD operations, the word loops through all seven possibilities. The goal is to get 7'b1100011 (7'h63) in the clock word. The clock word is the clock (4 bit 1 and 3'b 0) converted to parallel data, exactly as the video data traffic. For the 7:1 video, the RX input Clock serves as:

- Frequency reference to generate high speed.
- Phase reference as source synchronized RX, since the clock is edge aligned with the data bits.
- Word alignment reference.

9.1.5. MIPI_FILTER

This module is needed to filter low speed signal for MIPI RX. It filters out narrow pulses. It allows pulse width above 40 ns to pass.

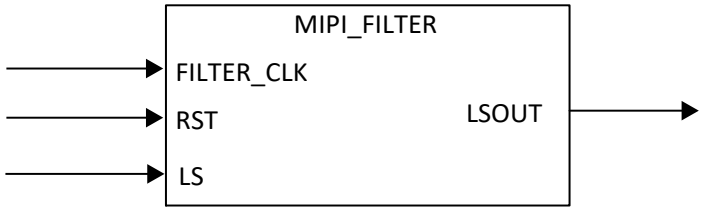

Figure 9.5. MIPI_FILTER Ports

Table 9.7. MIPI_FILTER Port Description

Port	I/O	Description
FILTER_CLK	I	Clock used to drive digital filter. Min freq=100 MHz. Recommendation is to use internal oscillator at 133 MHz.
LS	I	Low speed signal from MIPI PHY
RST	I	Active high reset. When RST=1, LSOUT=0.
LS_OUT	O	Filtered output signal
cutoff	Parameter	Parameterize the circuit, default=4, pass signal above 4 cycles. cutoff=round (20 ns/period (filter_clk)). Filter_clk=100 MHz, cutoff=4. Filter_clk=133 MHz, cutoff=6. Filter_clk=175 MHz, cutoff=7. Filter_clk=200 MHz. cutoff=8.

Technical Support Assistance

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

Revision History

Revision 1.2, May 2019

Section	Change Summary
All	• Changed document number from TN1265 to FPGA-TN02035. • Updated document template. • Added Disclaimers section.
Using Clarity Designer to Build and Plan High Speed DDR Interfaces	In the Configuring SDR Modules section, deleted footnote 2 from Table 7.1. SDR Configuration Parameters.
High-Speed DDR Interface Details	• Changed speed to 250 MHz for GDDRX1_RX.SCLK.Centered. • Changed speed to 400 MHz for GDDRX2_RX.ECLK.Centered. • Changed speed to 400 MHz for GDDRX2_RX.ECLK.Aligned. • Changed speed to 400 MHz for GDDRX2_RX.MIPI. • Corrected references to the Timing Analysis for High Speed DDR Interfaces section.
Memory DDR Primitives	Changed LPDDR2 and LPDDR3 Clock values in Table 8.17. Summary of all DDR Memory Primitives.
All	Minor editorial changes.

Revision 1.1, November 2015

Section	Change Summary
All	• Added support for ECP5-5G. • Changed document title to ECP5 and ECP5-5G High-Speed I/O Interface.
General	Updated the following figures: • Figure 4, GDDRX1_RX.SCLK.Centered Interface (Static Delay) • Figure 5, GDDRX1_RX.SCLK.Centered Interface (Dynamic Data delay) • Figure 6, GDDRX1_RX.SCLK.Aligned Interface (Static Delay) • Figure 7, GDDRX1_RX.SCLK.Aligned Interface (Dynamic Data/Clock Delay) • Figure 8, GDDRX2_RX.ECLK.Centered Interface (Static Delay) • Figure 9, GDDRX2_RX.ECLK.Centered Interface (Dynamic Data delay) • Figure 10, GDDRX2_RX.ECLK.Aligned Interface (Static Delay) • Figure 11, GDDRX2_RX.ECLK.Aligned Interface (Dynamic Data/Clock Delay) • Figure 12, GDDRX2_RX.MIPI • Figure 13, GDDRX71_RX.ECLK Interface • Figure 15, GDDRX1_TX.SCLK.Centered Interface • Figure 16, GDDRX2_TX.ECLK.Aligned Interface • Figure 17, GDDRX2_TX.ECLK.Centered Interface • Figure 18, GDDRX71_TX.ECLK.Centered Interface • Figure 31, DDR2, DDR3/DDR3L, LPDDR2, and LPDDR3 Read side Implementation • Figure 39, SDR Option Selected in the Catalog tab of Clarity Designer • Figure 41, DDR_Generic Option Selected in the Catalog tab of Clarity Designer • Figure 44, Option Selected in the Catalog tab of Clarity Designer • Figure 46, DDR_MEM Option Selected in the Catalog tab of Clarity Designer
Technical Support Assistance	Updated Technical Support information.

Revision 1.0, March 2014

Section	Change Summary
All	Initial release.

www.latticesemi.com