Implementing High-Speed Interfaces with Mach-NX Usage Guide

Technical Note

FPGA-TN-02234-1.1

March 2024

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. 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. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
DDR	Double Data Rate
ECLK	Edge Clock
GDDR	Generic Double Data Rate
PCLK	Primary Clock
PIO	Programmable I/O
PLD	Programmable Logic Device
SCLK	System Clock
SDR	Single Data Rate

1. Introduction

In response to the increasing need for higher data bandwidth, the industry has migrated from the traditional Single Data Rate (SDR) to the Double Data Rate (DDR) architecture. SDR uses either the rising edge or the falling edge of the clock signal to transfer data, while DDR uses both edges of the clock signal for data transfer. This essentially doubles the data transmission rate using the same clock frequency because the data is transferred twice per clock cycle. The Mach™-NX PLD family supports high-speed interfaces for both DDR and SDR applications through built-in Programmable I/O (PIO) logic. This document focuses on the implementation of high-speed generic DDR interfaces in the Mach-NX device. It also provides guidelines for making use of the built-in capabilities of the Mach-NX device to achieve the best performance for high-speed interfaces.

2. Architecture for High-Speed Interfaces

2.1. Gearing Logic Distribution

The high-speed generic DDR (GDDR) interfaces are supported through the built-in gearing logic in the Programmable I/O (PIO) cells. This gearing is necessary to support high-speed I/O while reducing the performance requirement on the FPGA fabric.

There are four gearing ratio settings available in the Mach-NX device depending on the sysI/O bank locations and the logic density. The ×1 gearing ratio is available in all banks for all the device densities. The ×2, ×4, and the 7:1 gearing ratio are available in Bank 0 and Bank 2 of the Mach-NX device. The 7:1 gearing ratio is mainly used for video display applications. The ×2/×4 gearing circuit is shared with the 7:1 circuit on both receive and the transmit sides. Table 2.1 gives a breakdown of gearing logic support in the different sysI/O banks. Details of PIO cells can be found in Mach-NX Family Data Sheet (FPGA-DS-02084).

Table 2.1. Gearing Logic Distribution for Mach-NX Device

Gearing Logic	Definition	Gearing Ratio	Bank 0	Bank 1	Bank 2
DDR ×1*	GDDR	1:2 or 2:1	Yes	Yes	Yes
Input DDR ×2	GDDR	1:4	—	—	Yes
Input DDR ×4	GDDR	1:8	—	—	Yes
Input DDR 7:1	GDDR	1:7	—	—	Yes
Output DDR ×2	GDDR	4:1	Yes	—	—
Output DDR ×4	GDDR	8:1	Yes	—	—
Output DDR 7:1	GDDR	7:1	Yes	—	—

***Note**: DDR ×1 is available for all Mach-NX device densities.

2.2. Different Types of I/O Logic Cells

In order to support various gearing ratios, the Mach-NX device support two types of PIO logic cells. These include a basic PIO cell and a video PIO cell. The basic PIO cell supports traditional SDR registers and DDR ×1 registers. It is available on all sides of all Mach-NX device. The video PIO cell supports the ×2, ×4 (in Mach-NX device) and 7:1 gearing applications. They are available in Bank 2 for the receive interfaces, and in Bank 0 for the transmit interfaces. The input and output structures of each type of PIO cell are discussed in detail in Mach-NX Family Data Sheet (FPGA-DS-02084). The block diagrams of the PIO cells are shown here again in this document for reference.

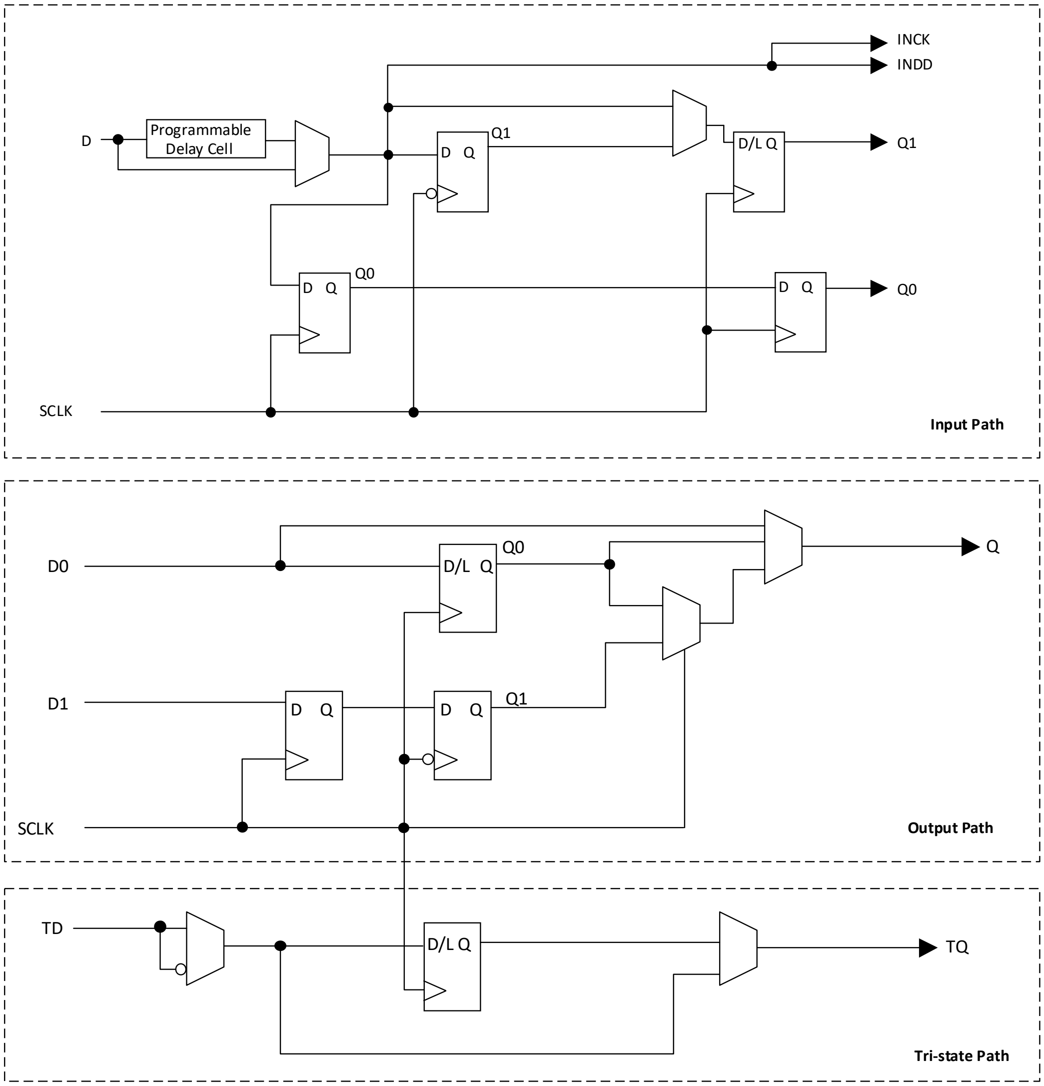
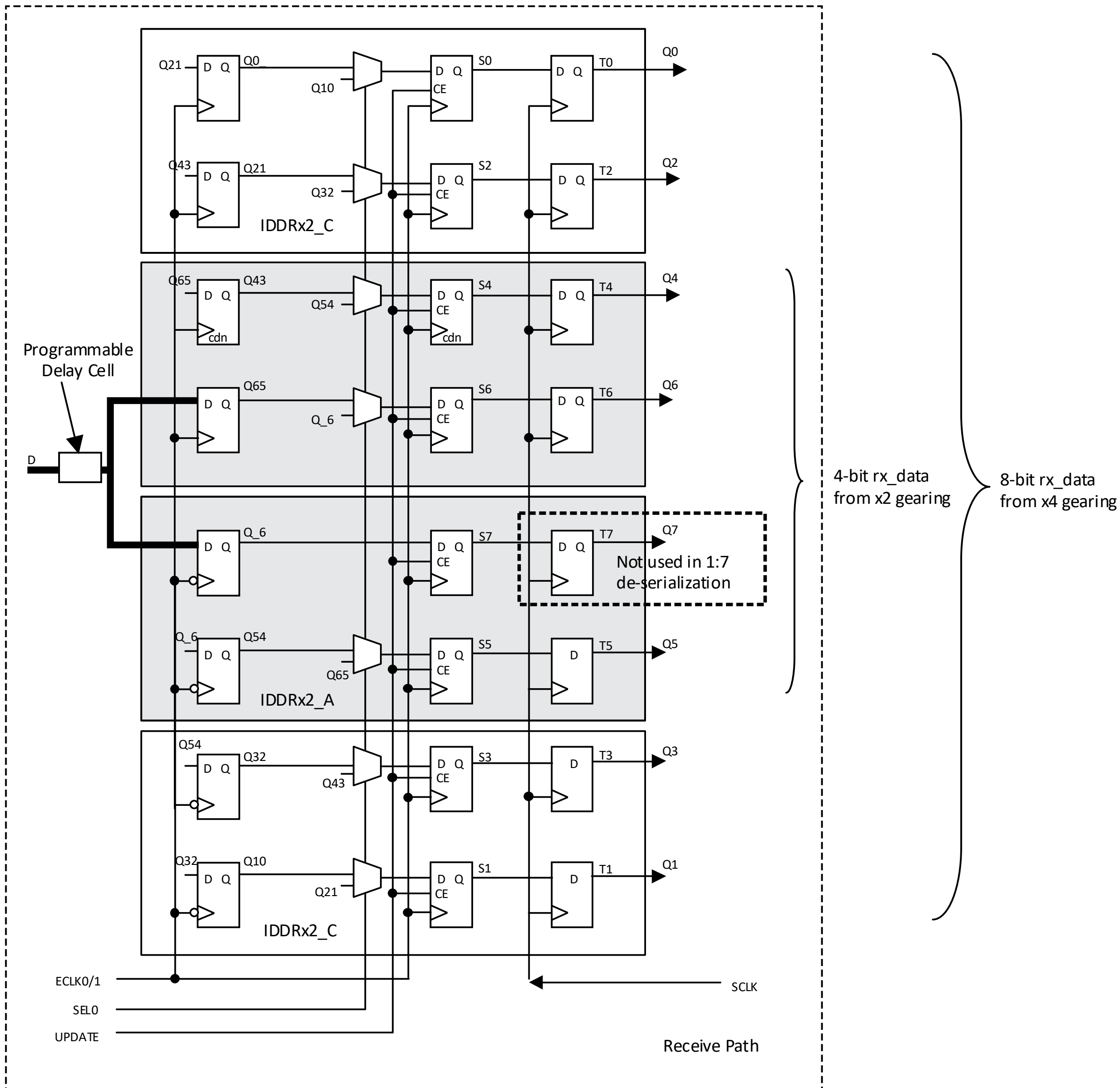


Figure 2.1. Basic PIO Cell Support ×1 Gearing Ratio


Programmable
Delay Cell
D
IDDRx2_C
IDDRx2_A
IDDRx2_C
Not used in 1:7
de-serialization
4-bit rx_data
from x2 gearing
8-bit rx_data
from x4 gearing
ECLK0/1
SEL0
UPDATE
SCLK
Receive Path
Q0
Q2
Q4
Q6
Q7
Q5
Q3
Q1

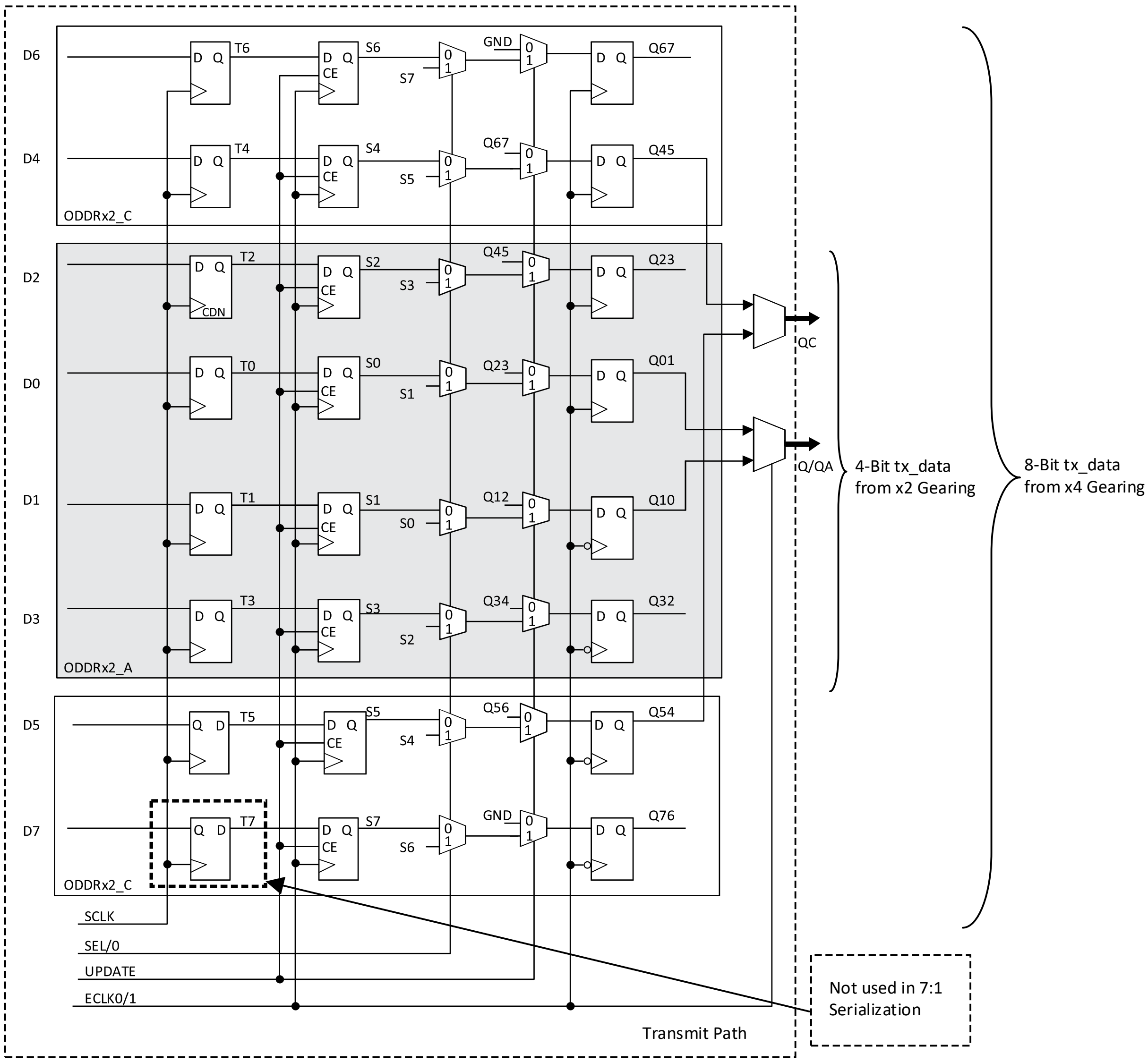


Figure 2.2. Video PIO Cell for ×2/×4 and 7:1 Applications in Mach-NX Device

2.3. Clock Domain Transfer at PIO Cells

The Mach-NX gearing logic performs serializing and de-serializing of high-speed data in the PIO cells. The clock domain transfer for the data from the high-speed edge clock (ECLK) to the low-speed system clock (SCLK) is guaranteed by design through two internal signals, UPDATE and SEL. The SEL signal toggles between '0' and '1' to sample three bits or four bits of data at a time for the 7:1 gearing. It remains static during the ×2/×4 gearings. The UPDATE signal behaves the same for all the gearings to update the register with the correct byte of data. This data is then clocked by the SCLK for downstream processing. Figure 2.2 illustrates the architecture of ×2/×4 input gearing logic.

Mach-NX device provide logic to support word alignment with minimal FPGA resources. The word alignment results in a shift to the UPDATE, SEL and the SCLK signals. It can be activated by providing an alignment request signal to the ALIGNWD port of the high-speed interface components. ALIGNWD can be asynchronous to the ECLK domain, but it must be at least two ECLK cycles wide. For the 7:1 gearing, ALIGNWD must be pulsed seven times to loop through a maximum of seven combinations of word orders. For the ×2/×4 gearings, ALIGNWD must be pulsed eight times to step through maximum eight possible word orders.

Figure 2.3 and Figure 2.4 provide a timing relationship of UPDATE, SEL, ECLK, and SCLK signals under different gearing requirements. Figure 2.5 and Figure 2.6 show the word alignment procedure for various gearing ratios. The discussion of gearing logic is applicable to both receive and transmit sides of the high-speed interfaces.

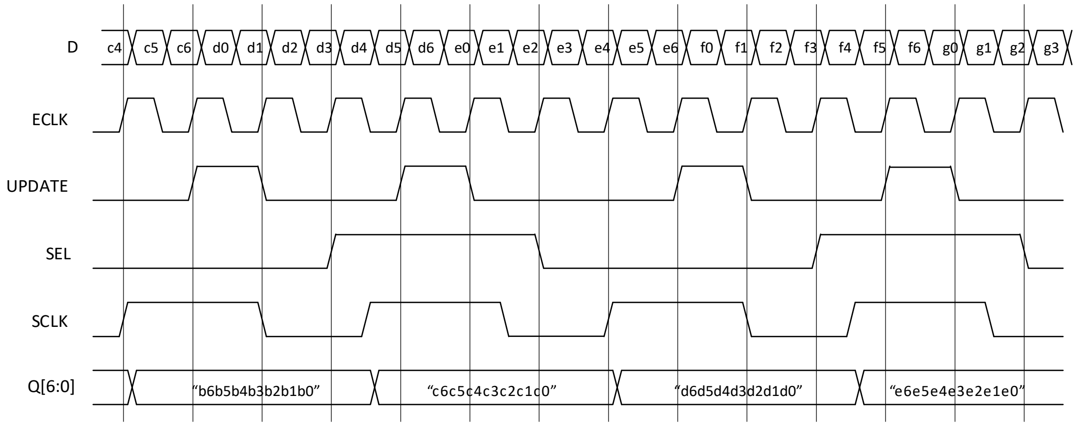


Figure 2.3. 7:1 Deserializer Timing in Mach-NX Device

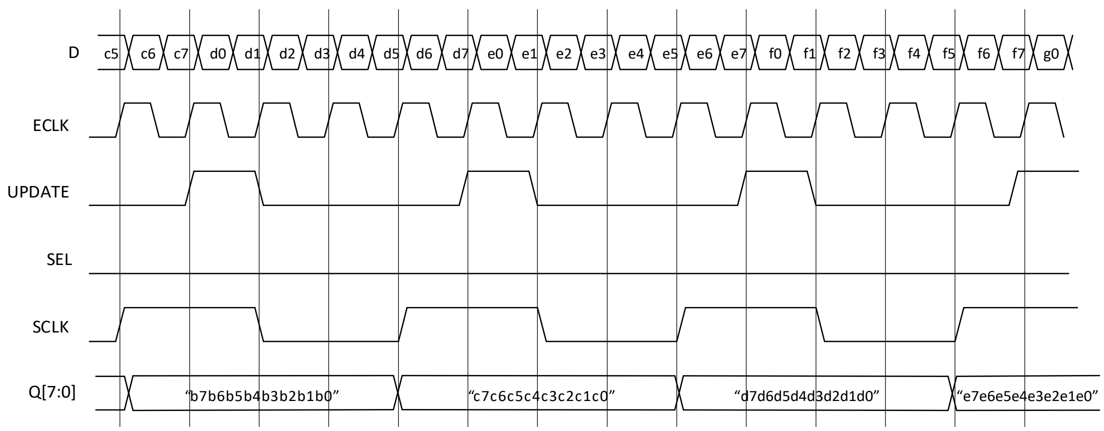


Figure 2.4. ×2/×4 Deserializer Timing in Mach-NX Device

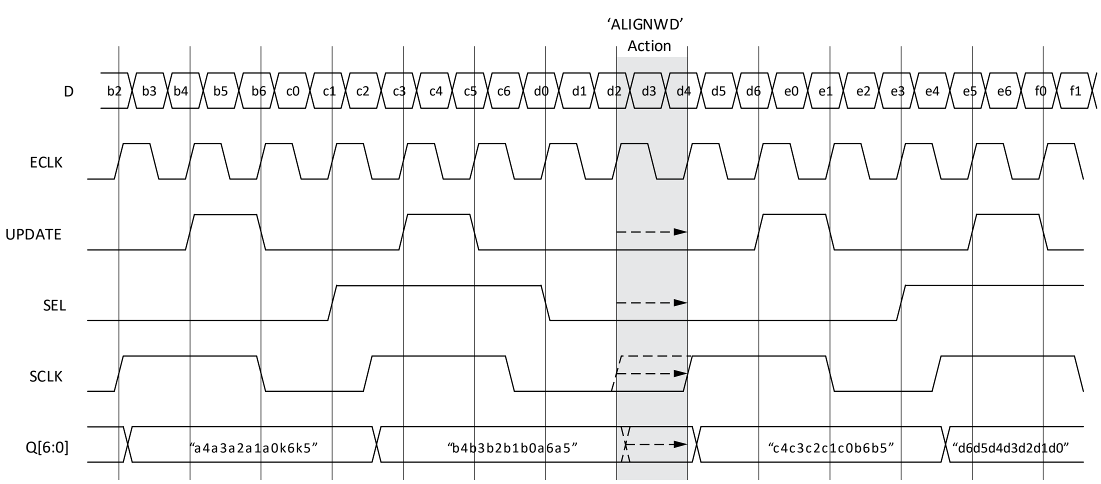


Figure 2.5. 7:1 Deserializer Timing in Response to ALIGNWD in Mach-NX Device

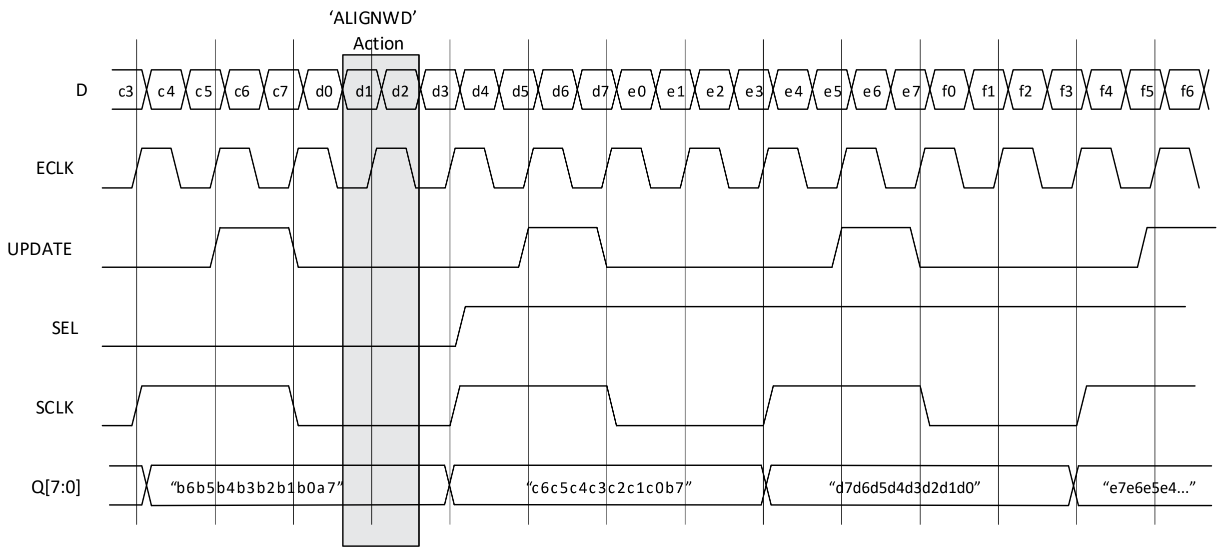


Figure 2.6. ×2/×4 Deserializer Timing in Response to ALIGNWD in Mach-NX Device

3. External High-Speed Interface Description

There are two types of external high-speed interface definitions that can be used with the Mach-NX device: centered and aligned. In a centered external interface, at the device pins, the clock is centered in the data opening. In an aligned external interface, the clock and data transition are aligned at the device pins. This is sometimes called *edge-on-edge*. Figure 3.1 shows external interface waveforms for SDR and DDR. At the receive side, an aligned interface requires clock delay adjustment to position the clock edge at the middle of the data opening to ensure that the capture flip-flop setup and hold times are not violated. Similarly a centered interface at the transmit side will require a clock delay adjustment to position the clock at the center of the data opening for transmission.

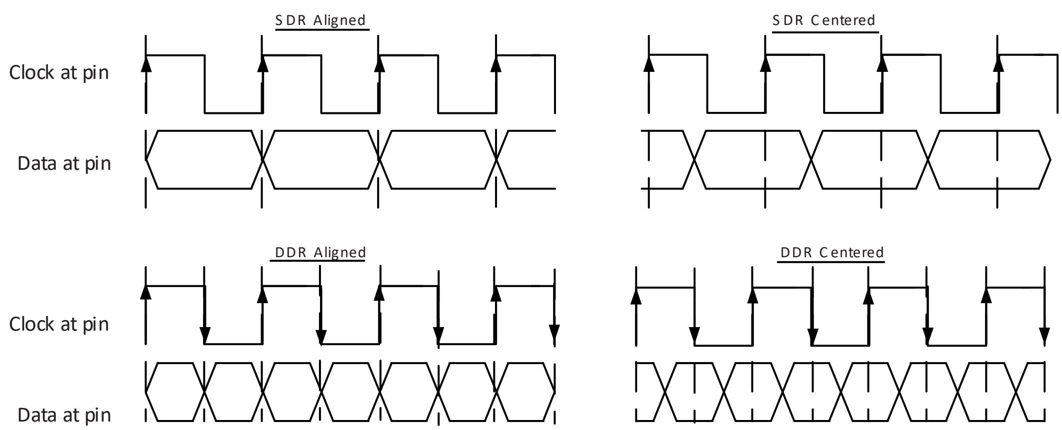


Figure 3.1. External Interface Definition

4. High-Speed Interface Building Blocks

Mach-NX device provides dedicated logic blocks for building high-speed interfaces, with each block performing a unique function. Combining various blocks gives ultimate performance of a specific interface. The hardware components in the device are described in this section. The DDR Software Primitives and Attributes section describes the library elements for these components. Refer to Mach-NX sysCLOCK PLL Usage Guide (FPGA-TN-02215) for an in-depth discussion of clocking and PLL architectures.

4.1. ECLK

Edge clocks are high-speed, low-skew I/O dedicated clocks. Two edge clocks (ECLK) are available along each Bank 0 and Bank 2 edge. The primary clock nets (PCLK) have direct connectivity to ECLKs. The bottom PCLK pins also have minimal routing to PLLs for video applications.

4.2. ECLKSYNC

This is the ECLK synchronization block. Each ECLK has its own ECLKCYNC component to synchronize the clock domain transfer of data. This block can also be used to dynamically disable an edge clock to save power during operation.

4.3. SCLK

SCLK refers to the system clock of the design. SCLK must use primary clock (PCLK) pins or primary clock (PCLK) nets for high speed interfaces. There are eight PCLK pins available for the Mach-NX device, and eight PCLK nets in the Mach-NX device.

4.4. CLKDIV

Clock dividers are used to generate low-speed system clocks from a high-speed edge clock. The ECLK frequency can be divided down by 2, by 3.5, or by 4 through the CLKDIV component.

4.5. PLL

A maximum of two PLLs are available in the Mach-NX device. There are pre-assigned dual-purpose I/O pins that drive to PLLs as reference clock inputs.

4.6. DQSDLL

A maximum of two DQSDLLs are available. The top-right DQSDLL is used by Bank 0 and Bank 1. The bottom-left DQSDLL is used by Bank 2. The DQSDLL, together with the clock slave delay cell (DLLDEL), is used to create a 90° clock shift/delay for aligned receiver interfaces.

4.7. Input DDR (IDDR)

Generic input DDR components support ×1, ×2, ×4, and 7:1 gearing ratios at the receiving side of the PIO cells.

The ×1 gearing is supported by IDDRX, or the basic PIO cell. It receives 1-bit DDR data and outputs 2-bit wide parallel data synchronized to the SCLK. There is no clock domain transfer involved in the ×1 gearing. The ×2 gearing is supported by IDDRX2. It receives 1-bit DDR data synchronized to the ECLK and outputs four bits of parallel data synchronized to the SCLK. The same function applies to the IDDRX4, which receives a single bit of DDR data synchronized to the ECLK and outputs eight bits of parallel data synchronized to the SCLK. The 7:1 gearing shares the same structure as the ×4 gearing. The 7:1 gearing outputs seven bits of parallel data instead of eight. The generic high-speed interface gearings are supported by the video PIO cells.

4.8. Output DDR (ODDR)

Generic output DDR components support ×1, ×2, ×4, and 7:1 gearing ratios at the transmit side of the PIO cells.

The ×1 gearing is supported by ODDRX, or the basic PIO cell. It serializes the 2-bit data based on SCLK. There is no clock domain transfer involved in ×1 gearing. The ×2 gearing is supported by ODDRX2. The 4-bit parallel data is clocked by SCLK and is serialized using ECLK. The ×4 gearing is supported by ODDRX4. The 8-bit parallel data is clocked by SCLK and is serialized using ECLK. The 7:1 gearing shares the same structure as the ×4 gearing. The 7-bit parallel data is serialized by the ECLK. The generic high speed interface gearings are supported by the video PIO cells.

4.9. Delays

There are two types of delay available for high-speed interfaces. The first type is the I/O logic delay that can be applied on the input data paths, as shown in the block diagram of PIO architectures at the beginning of the document.

Although the 32-tap I/O logic delay can be static or dynamic, only I/O in Bank 2 of the Mach-NX device supports dynamic data path delay. The static I/O logic delay (DELAYE) is used by default when configuring the interface in the Lattice design software. Software applies fixed delay values based on the interface used. Dynamic delay (DELAYD) in Bank 2 input data path provides dynamic or user-defined delay. Dynamic delay requires extra ports available on the module to be connected to user logic for delay control. The I/O logic delay is used to achieve the SDR zero hold timing, or match primary clock injection for ×1 gearing, or match the edge clock injection for ×2/×4 gearings.

The second type of delay is the clock slave delay cell (DLLDEL), which delays the incoming clock by 90° to place the clock in the middle of the data opening. This block is digitally controlled by the DQSDLL through 7-bit control code. There is one clock slave delay cell per primary clock pin. Its input comes from the primary clock pins and its output can drive the primary clock net for the ×1 aligned interface or ECLK for ×2/×4 aligned interfaces.

5. Generic High-Speed DDR Interfaces

Generic high-speed interfaces, or Generic DDR (GDDR), are supported in Mach-NX device using the dedicated logic blocks. This section will discuss the GDDR types, the interface logic, and the software to support the GDDR capability in the silicon.

5.1. High-Speed GDDR Interface Types

The GDDR interfaces supported by the Mach-NX device are pre-defined in the software and characterized in the silicon. Table 5.1 lists all the supported interfaces and gives a brief description of each interface.

Table 5.1. Generic High-Speed I/O DDR Interfaces[1]

Mode	Interface Name	Description	Supporting Banks
RX SDR	GIREG_RX.SCLK	SDR input using SCLK	Bank 0/1/2
RX GDDRx1 Aligned	GDDRX1_RX.SCLK.Aligned	DDRx1 input using SCLK, data is edge-to-edge with incoming clock	Bank 0/1/2
RX GDDRx1 Centered	GDDRX1_RX.SCLK.Centered	DDRx1 input using SCLK, incoming clock is centered at the data opening	Bank 0/1/2
RX GDDRx2 Aligned	GDDRX2_RX.ECLK.Aligned	DDRx2 input using ECLK, data is edge-to-edge with incoming clock	Bank 2
RX GDDRx2 Centered	GDDRX2_RX.ECLK.Centered	DDRx2 input using ECLK, incoming clock is centered at the data opening	Bank 2
RX GDDRx4 Aligned	GDDRX4_RX.ECLK.Aligned	DDRx4 input using ECLK, data is edge-to-edge with incoming clock	Bank 2
RX GDDRx4 Centered	GDDRX4_RX.ECLK.Centered	DDRx4 input using ECLK, incoming clock is centered at the data opening	Bank 2
RX GDDR71	GDDR71_RX.ECLK.7:1	GDDR 7:1 input using ECLK	Bank 2
TX SDR	GOREG_TX.SCLK	SDR output using SCLK	Bank 0/1/2
TX GDDRx1 Aligned	GDDRX1_TX.SCLK.Aligned	DDRx1 output using SCLK, data is edge-to-edge with outgoing clock	Bank 0/1/2
TX GDDRx1 Centered	GDDRX1_TX.SCLK.Centered	DDRx1 output using SCLK, outgoing clock is centered at the data opening	Bank 0/1/2
TX GDDRx2 Aligned	GDDRX2_TX.ECLK.Aligned	DDRx2 output using ECLK, data is edge-to-edge with outgoing clock	Bank 0
TX GDDRx2 Centered	GDDRX2_TX.ECLK.Centered	DDRx2 output using ECLK, outgoing clock is centered at the data opening	Bank 0
TX GDDRx4 Aligned	GDDRX4_TX.ECLK.Aligned	DDRx4 output using ECLK, data is edge-to-edge with outgoing clock	Bank 0
TX GDDRx4 Centered	GDDRX4_TX.ECLK.Centered	DDRx4 output using ECLK, outgoing clock is centered at the data opening	Bank 0
TX GDDR71	GDDR71_TX.ECLK.7:1	GDDR 7:1 output using ECLK	Bank 0

Note: MIPI D-PHY Receive can be built using GDDRX4_RX.ECLK.Centered interface & MIPI D-PHY Transmit can be built using GDDRX4_TX.ECLK.Centered interface. Refer to MIPI D-PHY Interface IP (FPGA-RD-02040) for details.

The following describes the naming conventions used for each of the interfaces listed in Table 5.1.

- G – Generic
- IREG – SDR input I/O register
- OREG – SDR output I/O register
- DDRX1 – DDR ×1 I/O register
- DDRX2 – DDR ×2 I/O register
- DDRX4 – DDR ×4 I/O register
- DDR71 – DDR 7:1 I/I register

- _RX – Receive interface
- _TX – Transmit interface
- ECLK – Uses ECLK (edge clock) clocking resource at the GDDR interface
- SCLK – Uses SCLK (primary clock) clocking resource at the GDDR interface
- Centered – Clock is centered to the data when coming into the device
- Aligned – Clock is aligned edge-on-edge to the data when coming into the device

5.2. High-Speed GDDR 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 Mach-NX clocking structure, refer to Mach-NX sysCLOCK PLL Usage Guide (FPGA-TN-02215). As listed in Table 5.1, each interface is supported in specific bank locations of the Mach-NX device. It is important to follow the architecture and various interface rules and preferences listed under each interface in order to build these interfaces successfully. The discussion of each component can be found in the DDR Software Primitives and Attributes section of this document.

5.2.1. Receive Interfaces

There are eight receive interfaces pre-defined and supported through Lattice IPexpress™ software.

GIREG_RX.SCLK

This is a generic interface for single data rate (SDR) data. The standard I/O register in the basic PIO cell (Figure 2.1) is used for the implementation. An optional inverter can be used to center the clock for aligned inputs. PLLs or DLLs can be used to remove the clock injection delay or adjust the setup and hold times. There are a limited number of DLLs in the architecture and these should be saved for high-speed interfaces when necessary. This interface can either be built using IPexpress, instantiating an I/O register element, or inferred during synthesis.

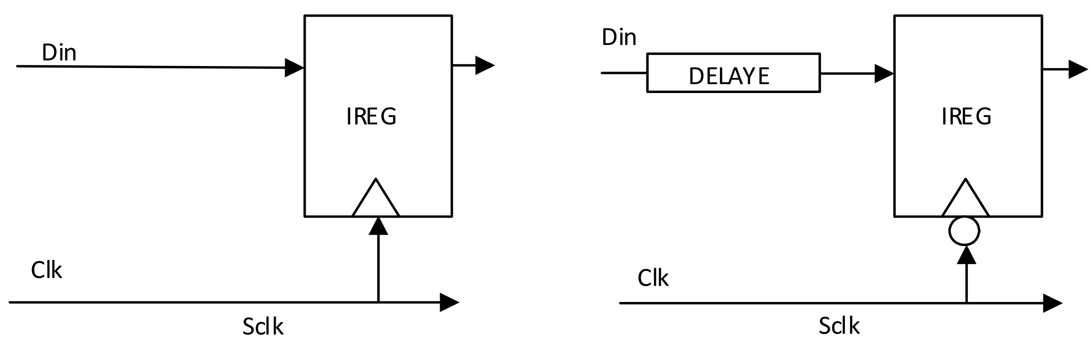


Figure 5.1. GIREG_RX Interface

The input data path delay cells can be used on the Din path of the interface. A DELAYE element provides a fixed delay to match the SCLK injection time. The dynamic input delay, DELAYD, is not available for this interface. Figure 5.1 shows possible implementations of this interface.

Interface rule:

A dedicated clock pin PCLK must be used as the clock source.

GDDRX1_RX.SCLK.Aligned

This DDR interface uses the SCLK and the DQSDLL to provide a 90° clock shift to center the clock at the IDDRXE. A DELAYE element is used to adjust data delay for the SCLK clock injection time. The DELAYD is not available for the ×1 interface.

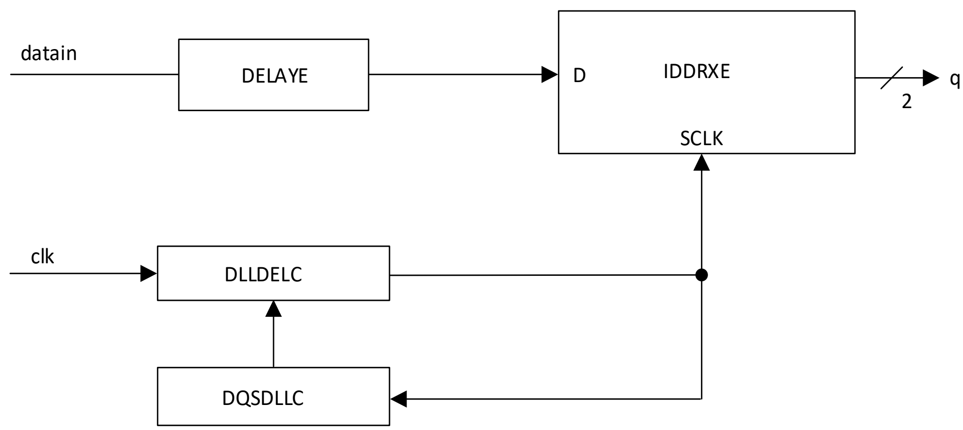


Figure 5.2. GDDRX1_RX.SCLK.Aligned Interface Using DQSDLL

Interface rules:

- A dedicated clock pin PCLK must be used as the clock source for DLLDELC.
- A primary clock net must be used to connect DLL outputs to the SCLK port.
- The DELAYE value should be set to SCLK_ALIGNED for the best timing.
- There are up to two DQSDLLCs per device. This limits the interface to a maximum of two clock frequencies per device.

GDDRX1_RX.SCLK.Centered

This DDR interface uses DELAYE to match the SCLK delay at the IDDRXE. DELAYD is not available for the ×1 interface. Since it is a centered interface, the clock edge is already in the middle of the data opening. There is no logic required to shift the clock.

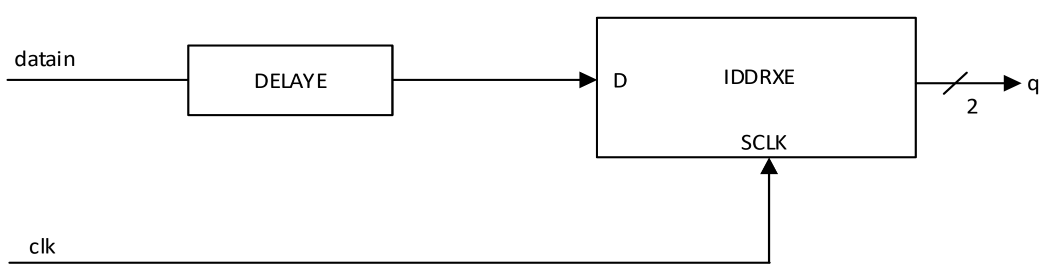


Figure 5.3. GDDRX1_RX.SCLK.Centered

Interface rules:

- A dedicated clock pin PCLK must be used as the clock source.
- DELAYE value should be set to SCLK_CENTERED for the best timing.
- The clock connected to SCLK should be on a primary clock net.

GDDRX2_RX.ECLK.Aligned

This DDR ×2 interface uses the DQSDLL to provide a 90° clock shift to center the clock at the IDDRX2E buffer. DELAYE is used to delay data to match the ECLK injection delay. DELAYD can also be used to control the delay dynamically. This interface uses ×2 gearing with the IDDRX2E element. This requires the use of a CLKDIVC to provide the SCLK which is half the frequency of the ECLK. The ECLKSYNCA element is associated with the ECLK and must be used to drive the ECLK. The port ALIGNWD can be used for word alignment at the interface.

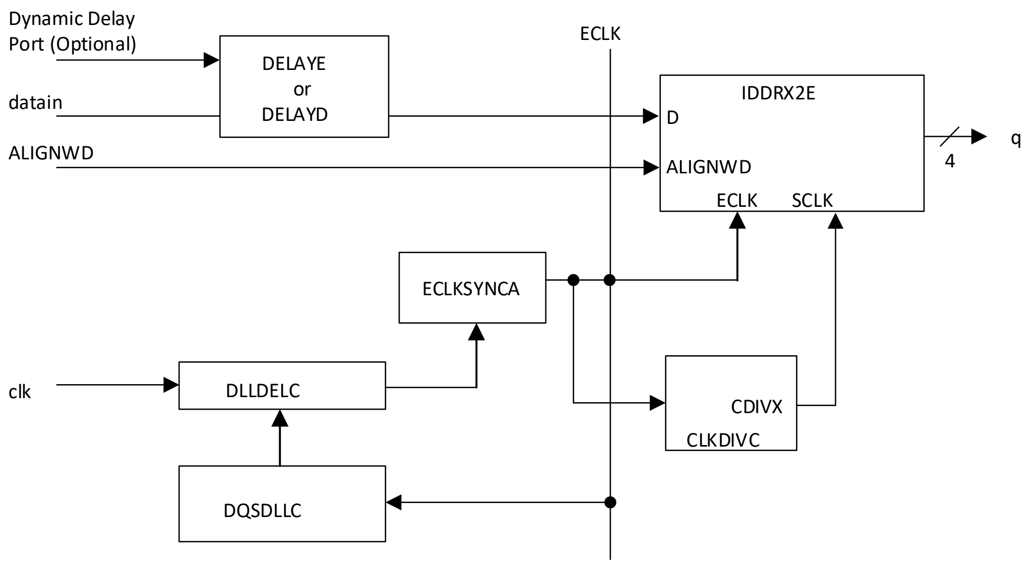


Figure 5.4. GDDRX2_RX.ECLK.Aligned Interface

Interface rules:

- A dedicated clock pin PCLK must be used as the clock source for DLLDELC.
- Clock net routed to SCLK must use primary clock net.
- There are up to two DQSDLLCs per device. It limits this interface to a maximum of two clock frequencies per device.
- DELAYE should be set to ECLK_ALIGNED.
- When DELAYD is used, only one dynamic delay port is needed for the entire bus.
- This interface is supported in Bank 2.

GDDRX2_RX.ECLK.Centered

This DDR ×2 interface uses DELAYE or DELAYD to match edge clock delay at the IDDRX2E. Since this interface uses the ECLK, it can be extended to support large data bus sizes for the entire side of the device. This interface uses ×2 gearing with the IDDRX2D element. This requires the use of a CLKDIVC to provide the SCLK which is half the frequency of the ECLK. The port ALIGNWD can be used for word alignment at the interface.

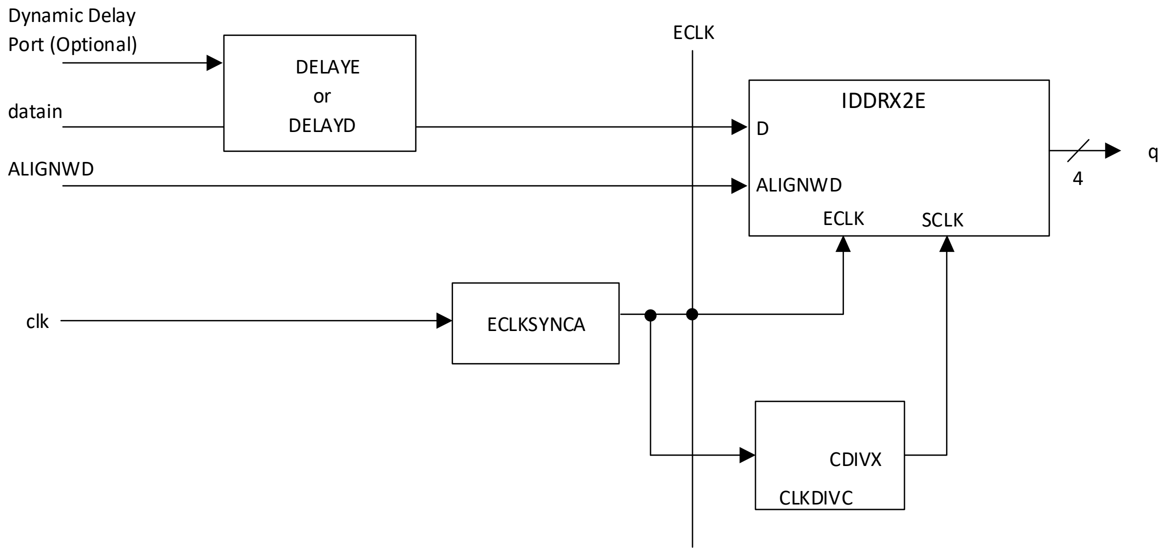


Figure 5.5. GDDRX2_RX.ECLK.Centered Interface

Interface rules:

- A dedicated clock pin PCLK must be used as the clock source for ECLKSYNCA.
- Clock net routed to SCLK must use primary clock net.
- DELAYE should be set to ECLK_CENTERED.
- When DELAYD is used, only one dynamic delay port is needed for the entire bus.
- This interface is supported in Bank 2.

GDDRX4_RX.ECLK.Aligned

This DDR ×4 interface uses the DQSDLL to provide a 90° clock shift to center the edge clock at the IDDRX4B buffer. DELAYE is used to delay data to match the ECLK injection delay. DELAYD can also be used to control the delay dynamically. Since this interface uses the ECLK, it can be extended to support large data bus sizes for the entire side of the device. This interface uses x4 gearing with the IDDRX4B element. This requires the use of a CLKDIVC to provide the SCLK which is one quarter of the ECLK frequency. ECLKSYNCA element is associated with the ECLK and must be used to drive the ECLK. The port ALIGNWD can be used for word alignment at the interface.

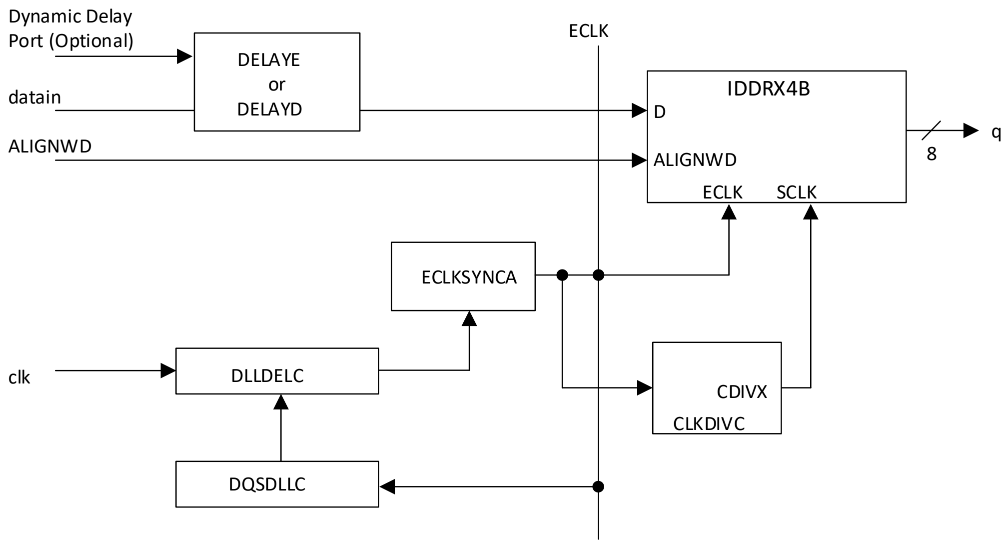


Figure 5.6. GDDRX4_RX.ECLK.Aligned Interface

Interface rules:

- A dedicated clock pin PCLK must be used as the clock source for DLLDELC.
- Clock net routed to SCLK must use primary clock net
- There are up to two DQSDLLCs per device. It limits this interface to have maximum of two clock frequencies per device.
- Data input must use A/B pair of the I/O logic cells for x4 gearing.
- DELAYE should be set to ECLK_ALIGNED
- When DELAYD is used, only one dynamic delay port is needed for the entire bus.
- This interface is supported in Bank 2.

Note: The GDDRX4_RX.ECLK.Centered interface is used to build MIPI D-PHY Receive Interface. Refer to MIPI D-PHY Interface IP (FPGA-RD-02040) for details.

GDDRX4_RX.ECLK.Centered

This DDR x4 interface uses DELAYE or DELAYD to match edge clock delay at the IDDRX4B. Since this interface uses the ECLK it can be extended to support large data bus sizes for the entire side of the device. This interface uses x4 gearing with the IDDRX4B element. This requires the use of a CLKDIVC to provide the SCLK that is one quarter of the ECLK frequency. The port ALIGNWD can be used for word alignment at the interface.

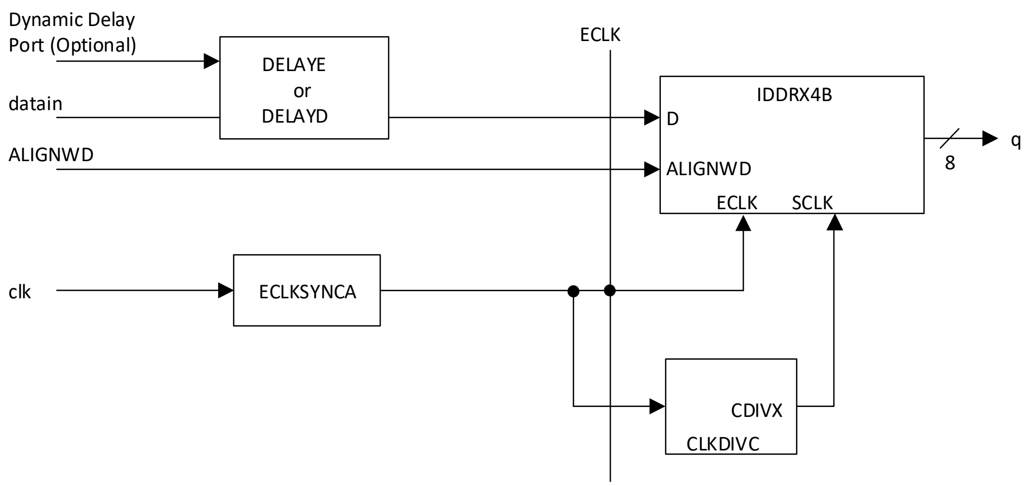


Figure 5.7. GDDRX4_RX.ECLK.Centered Interface

Interface rules:

- A dedicated clock pin PCLK must be used as the clock source for ECLKSYNCA.
- Clock net routed to SCLK must use primary clock net.
- Data input must use A/B pair of the I/O logic for ×4 gearing.
- DELAYE should be set to ECLK_CENTERED.
- When DELAYD is used, one dynamic delay port is needed for the entire bus.
- This interface is supported in Bank 2.

GDDR71_RX.ECLK.7:1

The GDDR 7:1 receive interface is unique among the supported high-speed DDR interfaces. It uses the PLL to search the best clock edge to the data opening position during bit alignment process. The PLL steps through the 16 phases to give eight sampling points per data. The data path delay is not used in this interface. CLKDIVC is used to divide down the ECLK by 3.5 due to the nature of the 1:7 deserializing requirement. This means the SCLK is running seven times slower than the incoming data rate. ECLKSYNCA element is associated with the ECLK and must be used to drive the ECLK. The complete 7:1 LVDS video display application requires bit alignment and word alignment blocks to be built in the FPGA resources in addition to the built-in I/O gearing logic and alignment logic. The CLK_PHASE signal is sent to the FPGA side to build the bit alignment logic.

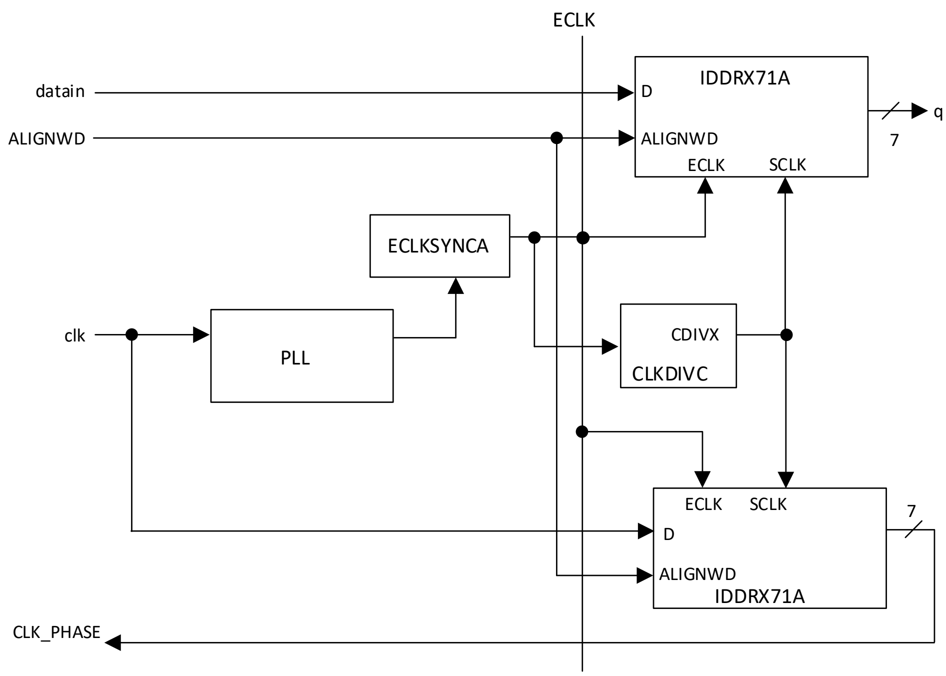


Figure 5.8. GDDR71_RX.ECLK.7:1 Interface

Interface rules:

- A dedicated clock pin PCLK in Bank 2 must be used as the clock source for PLL.
- Clock net routed to SCLK must use primary clock net.
- There are up to two PLLs per device. It limits this interface to have maximum of two clock frequencies per device.
- The data input must use A/B pair of the I/O logic cell for 7:1 gearing.
- This interface is supported in Bank 2.

5.2.2. Transmit Interfaces

There are eight transmit interfaces pre-defined and supported through Lattice IPexpress software.

GOREG_TX.SCLK

This is a generic interface for SDR data and a forwarded clock. The standard register in the basic PIO cell is used to implement this interface. The ODDRXE used for the output clock balances the clock path to match the data path. A PLL can also be used to clock the ODDRXE to phase shift the clock to provide a precise clock to data output. There are a limited number of PLLs in the architecture and these should be saved for high-speed interfaces when necessary. This interface can either be built using IPexpress, instantiating an I/O register element, or inferred during synthesis.

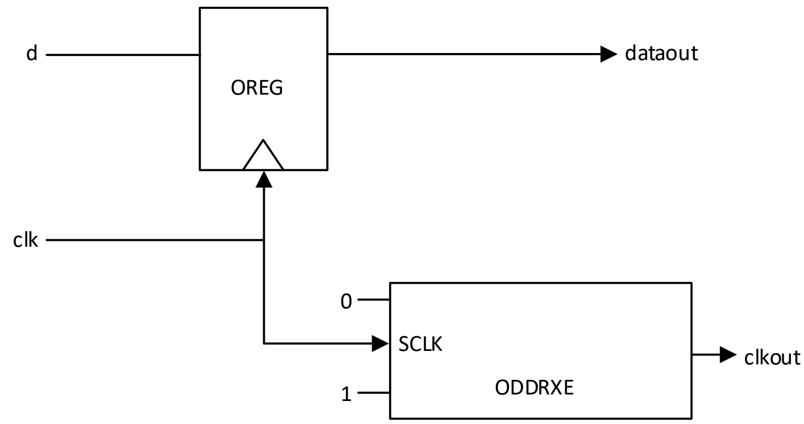


Figure 5.9. GOREG_TX.SCLK Interface

Interface rule:

The clock source for SCLK must be routed on a primary clock net.

GDDRX1_TX.SCLK.Aligned

This output DDR interface provides clock and data that are aligned using a single SCLK. The ODDRXE used for the output clock balances the clock path to match the data path.

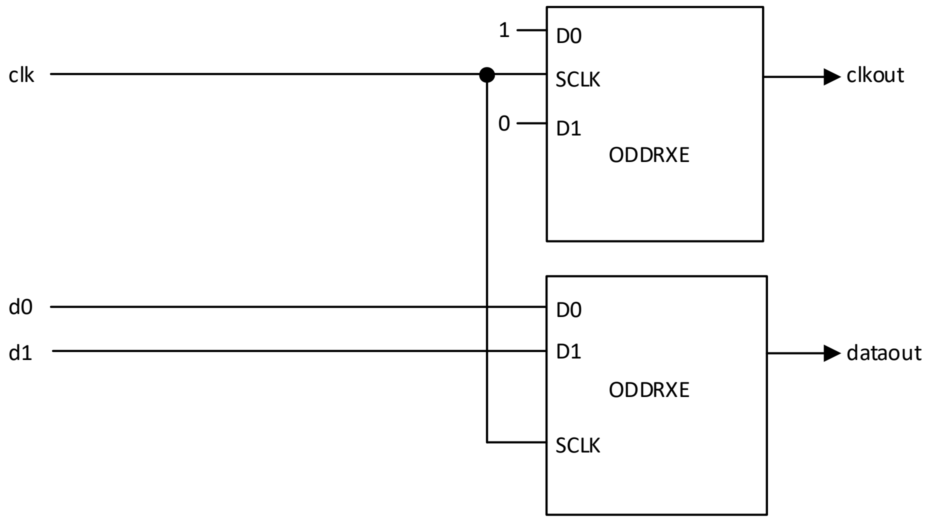


Figure 5.10. GDDRX1_TX.SCLK.Aligned Interface

Interface rule:

The clock source for SCLK must be routed on a primary clock net.

GDDRX1_TX.SCLK.Centered

This output DDR interface provides clock and data that are pre-centered. PLL uses clkop and clkos ports to provide the 90° phase difference between the data and the clock. It requires two SCLK resources to drive the output data I/O cell and the output clock I/O cell.

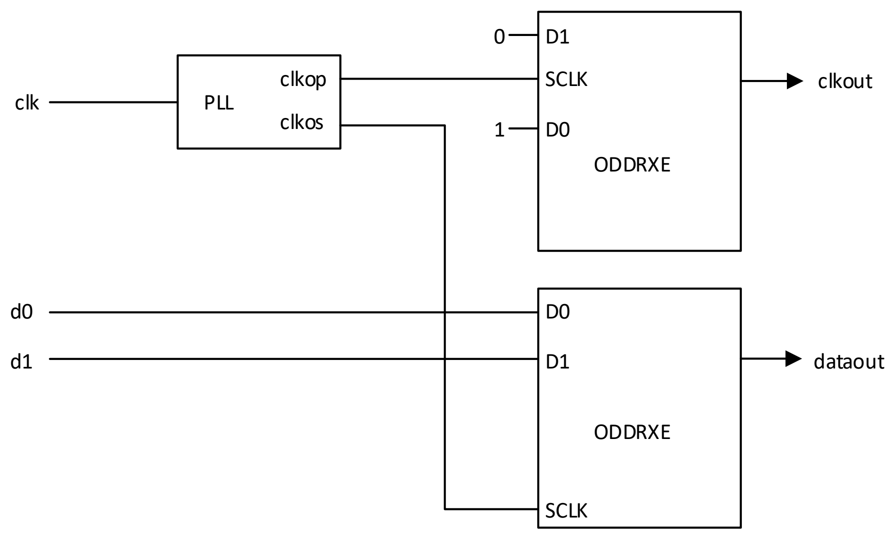


Figure 5.11. GDDRX1_TX.SCLK.Centered Interface

Interface rule:

SCLK and 90°-shifted SCLK must be routed on primary clock nets.

GDDRX2_TX.ECLK.Aligned

This output DDR ×2 interface provides clock and data that are aligned. A CLKDIV is used to generate the SCLK which is half of the ECLK frequency. The ECLKSYNC element is used on the ECLK path for data synchronization.

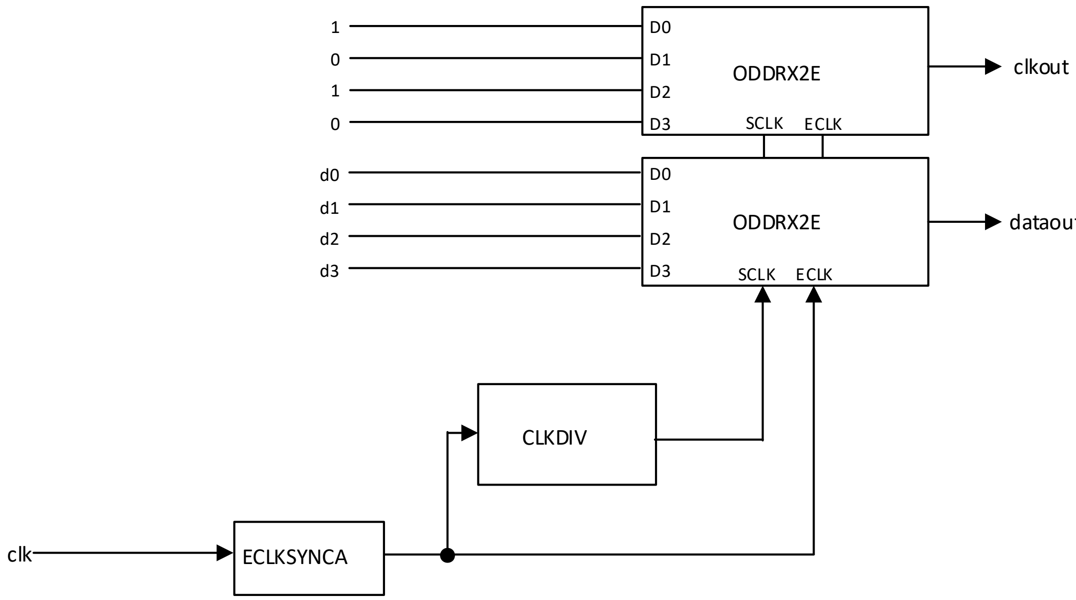


Figure 5.12. GDDRX2_TX.ECLK.Aligned Interface

Interface rules:

- Edge clock routing resources must be used for the ECLK.
- The routing of SCLK must use primary clock net.
- This interface is supported in Bank 0.

GDDRX2_TX.ECLK.Centered

This output DDR ×2 interface provides a clock that is centered at the data opening. The PLL uses clkop and clkos ports to provide the 90° phase difference between the data and the clock. Two ECLK routing resources are used in this interface to drive the output data and the output clock. A CLKDIV is used to generate the SCLK which is half of the ECLK frequency.

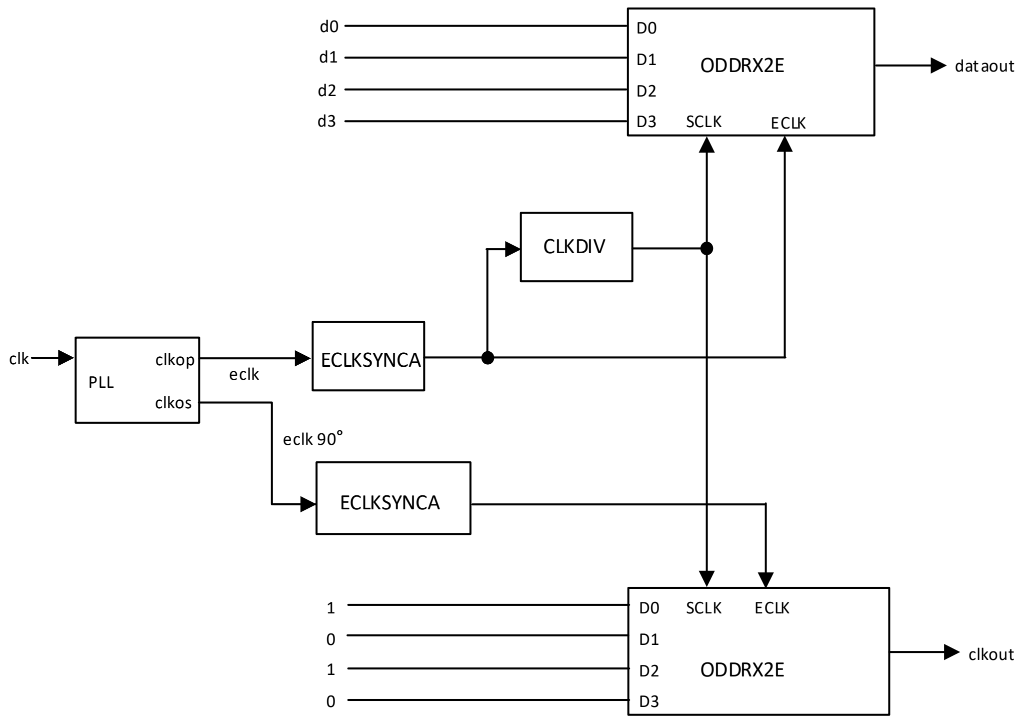


Figure 5.13. GDDRX2_TX.ECLK.Centered Interface

Interface rules:

- Edge clock routing resources must be used for the ECLK.
- Since two ECLKs are used on this interface, maximum one bus of this interface can be implemented at a time.
- The routing of the SCLK must use primary clock net.
- This interface is supported in Bank 0.

GDDRX4_TX.ECLK.Aligned

This output DDR x4 interface provides clock and data that are aligned. A CLKDIV is used to generate the SCLK which is a quarter of the ECLK frequency. The ECLKSYNC element is used on the ECLK path for data synchronization.

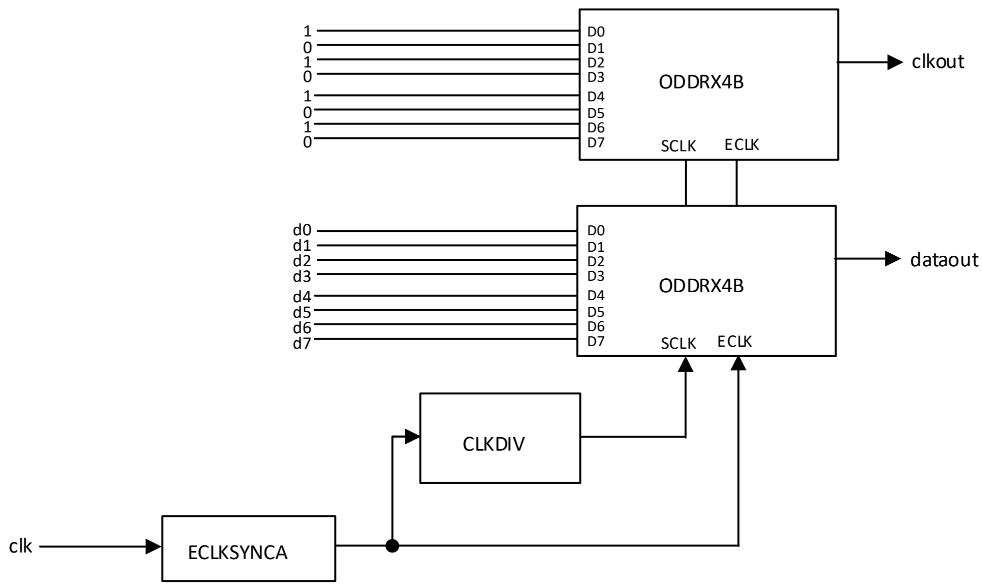


Figure 5.14. GDDRX4_TX.ECLK.Aligned Interface

Interface rules:

- Edge clock routing resources must be used for the ECLK.
- The routing of SCLK must use primary clock net.
- Data output must use A/B pair of the I/O logic for ×4 gearing.
- This interface is supported in Bank 0.

GDDRX4_TX.ECLK.Centered

This output DDR x4 interface provides a clock that is centered at the data opening. The PLL uses clkop and clkos ports to provide the 90° phase difference between the data and the clock. Two ECLK routing resources are used in this interface to drive the output data and the output clock. A CLKDIV is used to generate the SCLK which is one-quarter of the ECLK frequency.

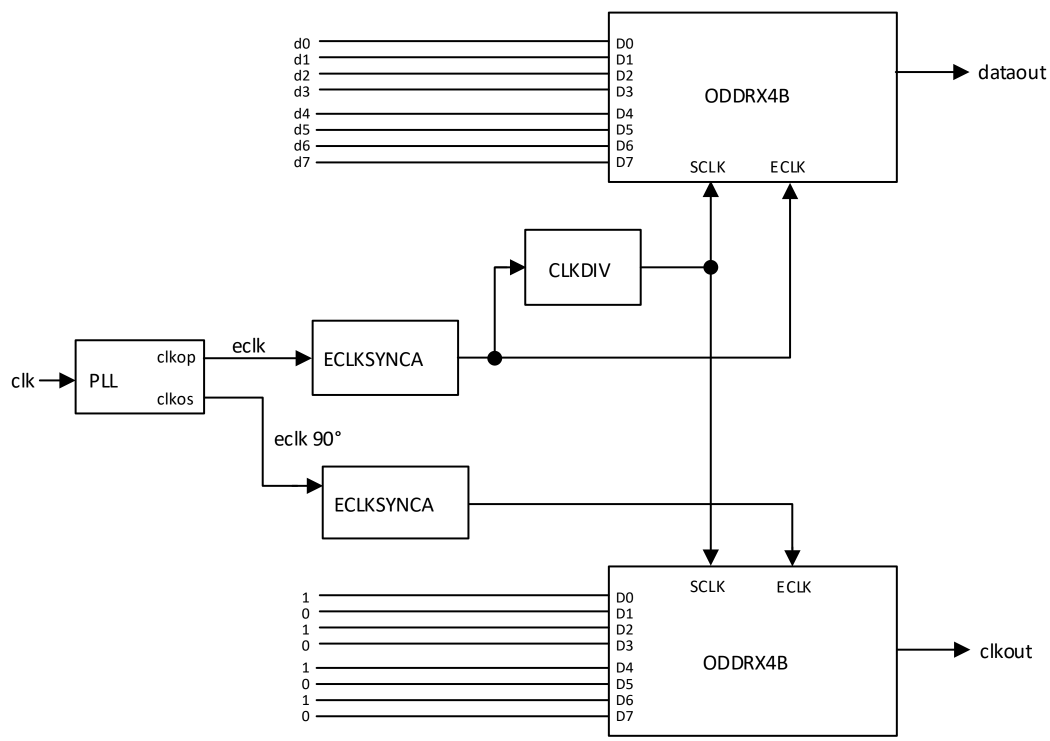


Figure 5.15. GDDRX4_TX.ECLK.Centered Interface

Interface rules:

- Edge clock routing resources must be used for the ECLK.
- Since two ECLKs are used on this interface, a maximum of one bus of this interface can be implemented at a time.
- The routing of the SCLK must use primary clock net.
- Data output must use A/B pair of the I/O logic for ×4 gearing.
- This interface is supported in Bank 0.

Note: The GDDRX4_TX.ECLK.Centered interface is used to build MIPI D-PHY Transmit Interface. Refer to MIPI D-PHY Interface IP (FPGA-RD-02040) for details.

GDDR71_TX.ECLK.7:1

The GDDR 7:1 transmit interface is unique among the supported high-speed DDR interfaces. It uses a specific pattern to generate the output clock, known as pixel clock. The CLKDIVC is used to divide down the ECLK by 3.5 due to the nature of the 7:1 serializing requirement. This means the SCLK is running seven times slower than the transmit data rate. ECLKSYNCA element is associated with the ECLK and must be used to drive the ECLK.

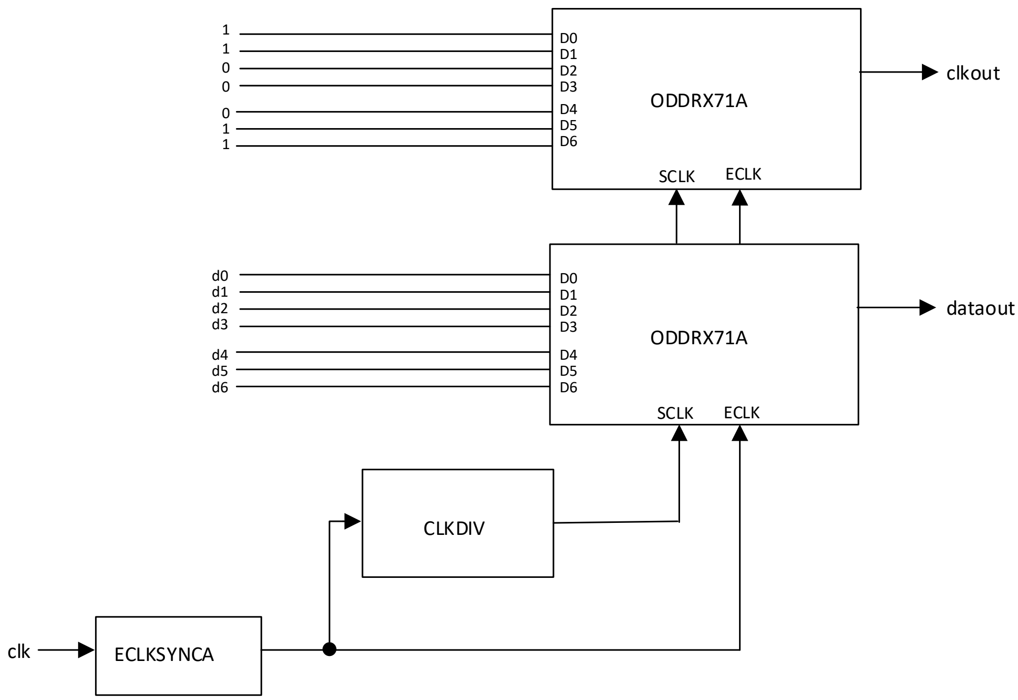


Figure 5.16. GDDR71_TX.ECLK.7:1 Interface

Interface rules:

- Edge clock routing resources must be used for the ECLK.
- The routing of SCLK must use primary clock net.
- Data output must use A/B pair of the I/O logic for 7:1 gearing.
- This interface is supported in Bank 0.

6. Using IPexpress to Build Generic High-Speed DDR Interfaces

The IPexpress tool of the Lattice development software should be used to configure and generate all the generic high-speed interfaces described above. IPexpress will generate a complete HDL module including clocking requirements for each of the interfaces described above. In the IPexpress user interface, all the DDR modules are located under Architecture Modules > IO. This section covers the SDR, DDR_GENERIC, and GDDR_71 interfaces in IPexpress.

Table 6.1 shows the signal names used in the IPexpress modules. Each signal can be used in all or some specified interfaces. The signals are listed separately for the GDDR receive interfaces and the transmit interfaces.

Table 6.1. Signal Names Used by IPexpress Modules

Signal Name	Direction	Description	Supported Interfaces
Receive Interface			
clk	Input	Source synchronous clock	All
reset	Input	Asynchronous reset to the interface, active high	All
datain	Input	Serial data input at Rx interfaces	All
uddcntln	Input	Hold/update control of delay code, active low	×1, ×2, ×4 Aligned
freeze	Input	Freeze or release DLL, active high	×1, ×2, ×4 Aligned
alignwd	Input	Word alignment control signal, active high	×2, ×4, 7:1
dqsdll_reset	Input	Asynchronous DQSDLL reset, active high	×2, ×4 Aligned
clk_s*	Input	Slow clock for reset synchronization	×2, ×4, 7:1
init	Input	Initialize reset synchronization, active high	×2, ×4, 7:1
phase_dir	Input	PLL phase direction	7:1
phase_step	Input	PLL phase step	7:1
sclk	Output	System clock for the FPGA fabric	All
q	Output	Parallel data output of the Rx interfaces	All
lock	Output	DLL or PLL lock	×2, ×4, 7:1
eclk	Output	Edge clock generated from the input clock	×2, ×4 Aligned, 7:1
rx_ready	Output	Indicate completion of reset synchronization	×2, ×4, 7:1
clk_phase	Output	7-bit representation of input clock phase	7:1
Transmit Interface			
clk	Input	Main input clock for Tx interfaces	All
reset	Input	Asynchronous reset to the interface, active high	All
dataout	Input	Parallel input data of the Tx interfaces	All
clk_s*	Input	Slow clock for reset synchronization	×2, ×4, 7:1
sclk	Output	System clock for the FPGA fabric	All
dout	Output	Serial data output for the Tx interfaces	All
clkout	Output	Source synchronous clock	All
Tx_ready	Output	Indicate completion of reset synchronization	×2, ×4, 7:1

***Note**: clk_s can be any slow clock to be used for reset synchronization process. This clock must be slower than eclk.

6.1. Building the SDR Interface

As shown in Figure 6.1, you can choose interface type SDR, enter the module name and click Customize to open the Configuration tab. Note that ×4 gearing is only on the Mach-NX device.

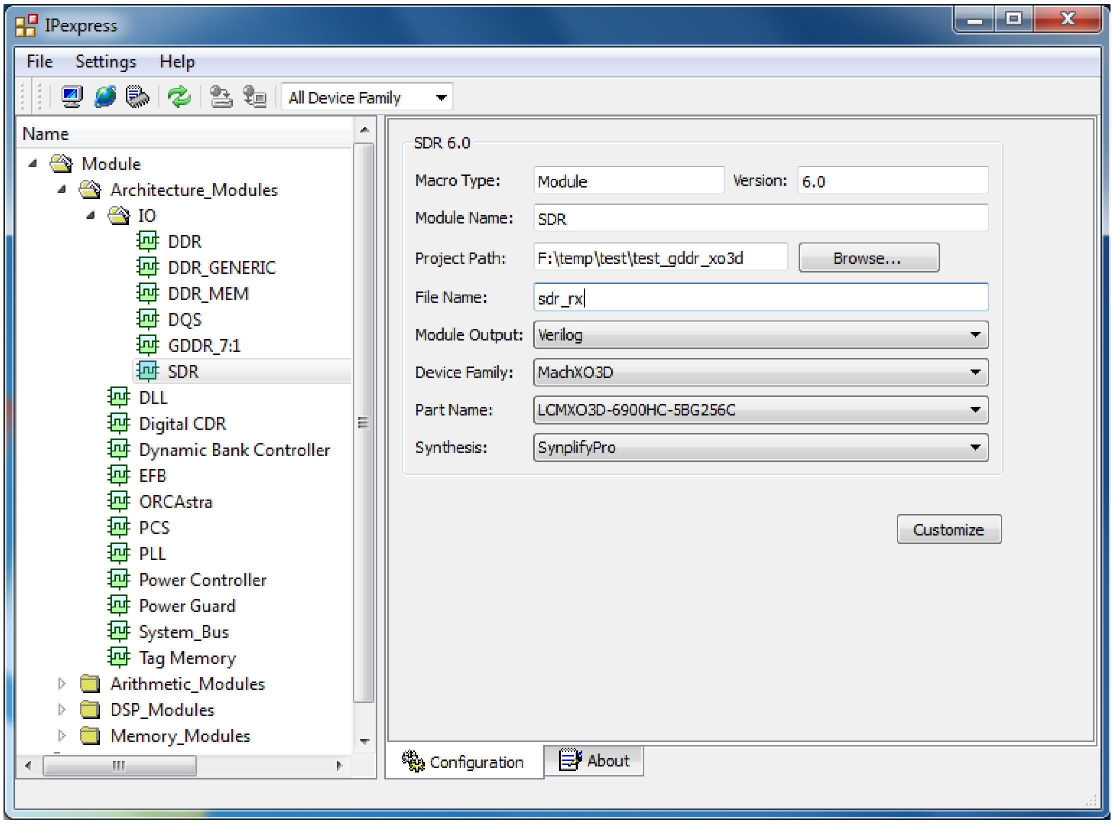


Figure 6.1. SDR Interface Selection at the IPexpress Main Window

Figure 6.2 shows the Configuration tab for the SDR module in IPexpress. Table 6.2 lists the various configurations options available for SDR modules.

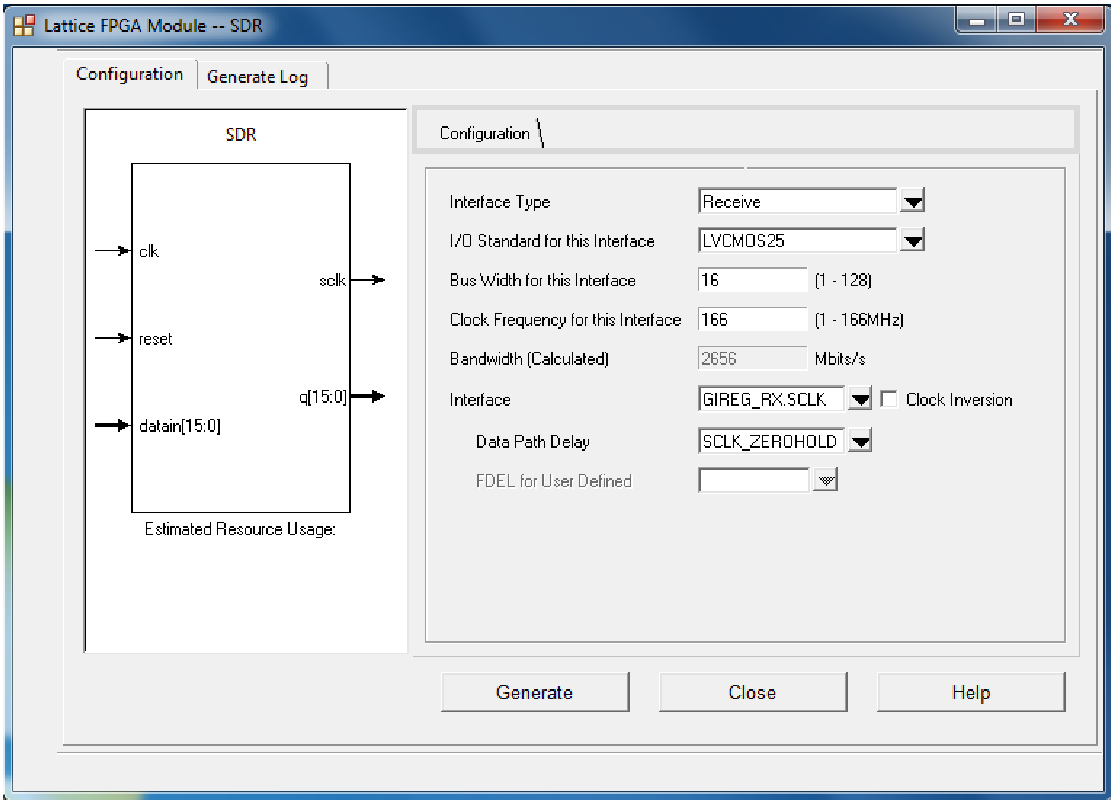


Figure 6.2. Configuration Tab for the SDR Interface

Table 6.2. User Interface Options for the SDR Interfaces

User Interface Option	Description	Range	Default Value
Interface Type	Types of interfaces	Transmit, Receive	Receive
I/O Standard for this interface	I/O standard to be used for the interface.	Supports all I/O types per selected Interface Type	LVCMOS25
Bus Width for this Interface	Bus size for the interface.	1-128	16
Clock Frequency for this Interface	Speed at which the interface will run	1-166 MHz	166 MHz
Interface Bandwidth (calculated)	Calculated from the clock frequency entered.	(calculated)	(calculated)
Interface	Interface selected based on previous entries.	Transmit: GOREG_TX.SCLK Receive: GIREG_RX.SCLK	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.	Bypass, SCLK_ZEROHOLD, User Defined	Bypass
FDEL for User Defined	If Delay type selected above is “User Defined”, delay values can be entered with this parameter.	Delay0 to Delay31	Delay0

6.2. Building DDR Generic Interfaces

As shown in Figure 6.3, you can choose interface type DDR_Generic, enter module name and click Customize to open the Configuration tab.

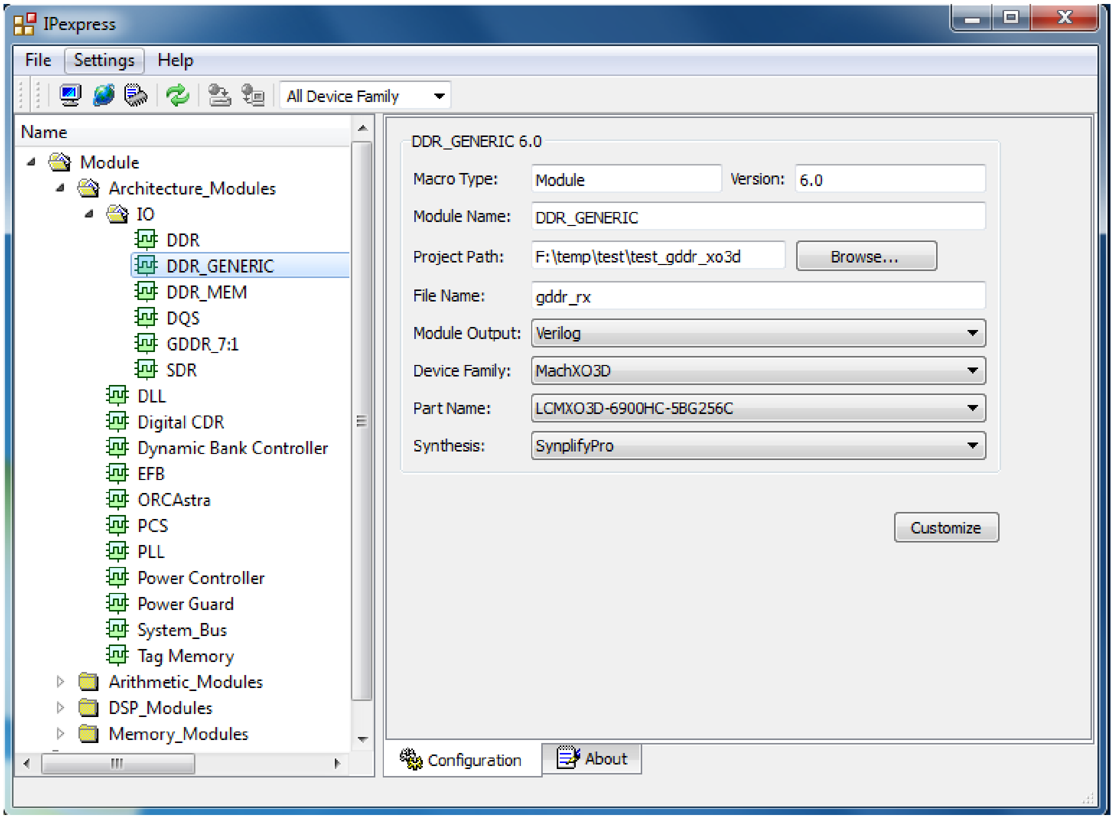


Figure 6.3. DDR_Generic Interface Selection at the IPexpress Main Window

DDR_Generic interfaces 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 necessary, override the selection made for the interface in the Configuration tab and customize the interface based on design requirements.

The following figures show the two tabs of the DDR_Generic modules in IPexpress. Table 6.3 and Table 6.4 list the various configuration options available for DDR_Generic modules.

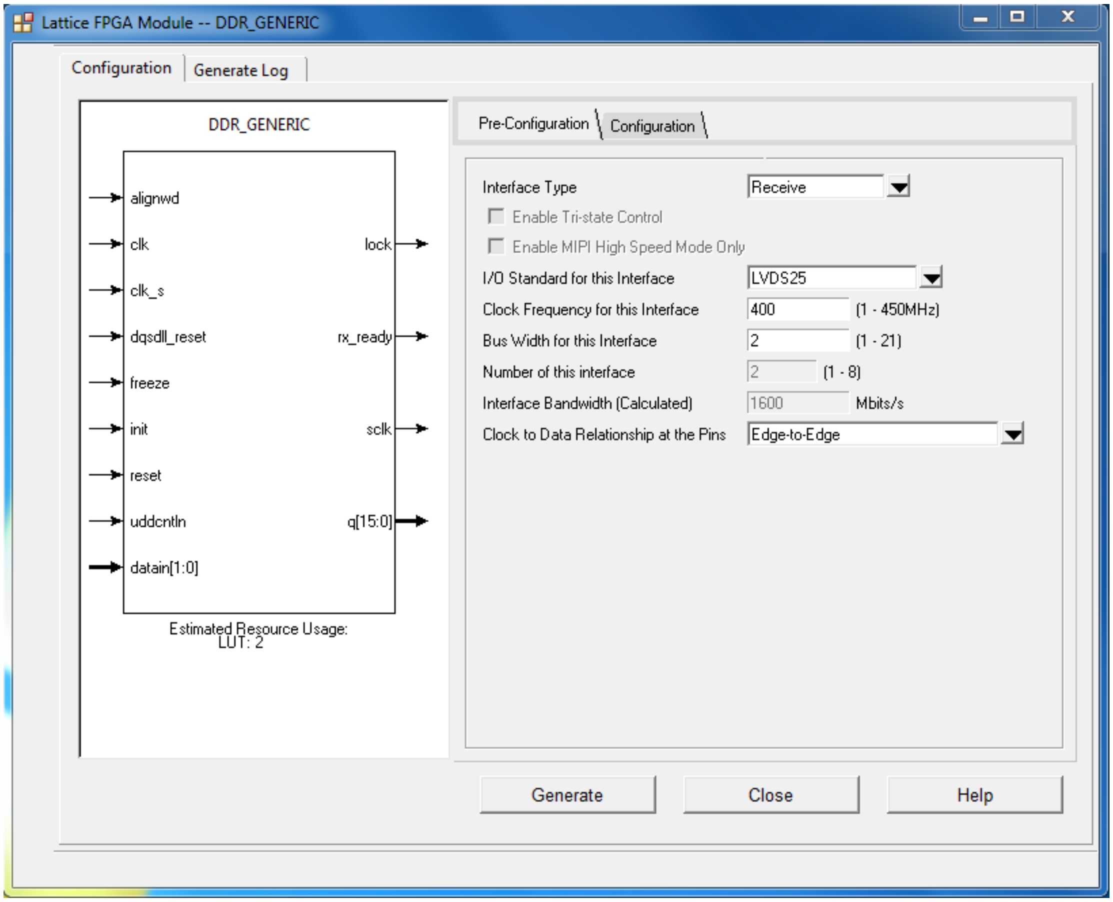


Figure 6.4. Pre-Configuration Tab of the DDR_Generic Interfaces

Table 6.3. User Interface Option for the Pre-Configuration Tab of DDR_Generic Modules

User Interface Option	Description	Range	Default Value
Interface Type	Types of interfaces	Transmit, Receive, Transmit_MIPI, Receive_MIPI	*Note
Enable MIPI High Speed Mode Only (only for MIPI)	Supports only HS mode, not LP mode	DISABLED, ENABLED	DISABLED
I/O Standard for this interface	I/O Standard for this interface	Supports all I/O types per selected Interface Type	LVCMOS25
Clock Frequency for this Interface	Speed at which the interface will run	1-450 MHz	*Note
Bus Width for this Interface	Bus size for the interface	Various depending on the interface selected	*Note
Number of this Interface	Maximum number of buses supported	(Calculated)	(Calculated)
Interface Bandwidth (calculated)	Calculated from the clock frequency and bus width	(Calculated)	(Calculated)
Clock to Data Relationship at the Pins	Select the type of external interfaces	Edge-to-Edge, Centered	*Note

***Note**: All fields of the Pre-Configuration tab are blank as default.

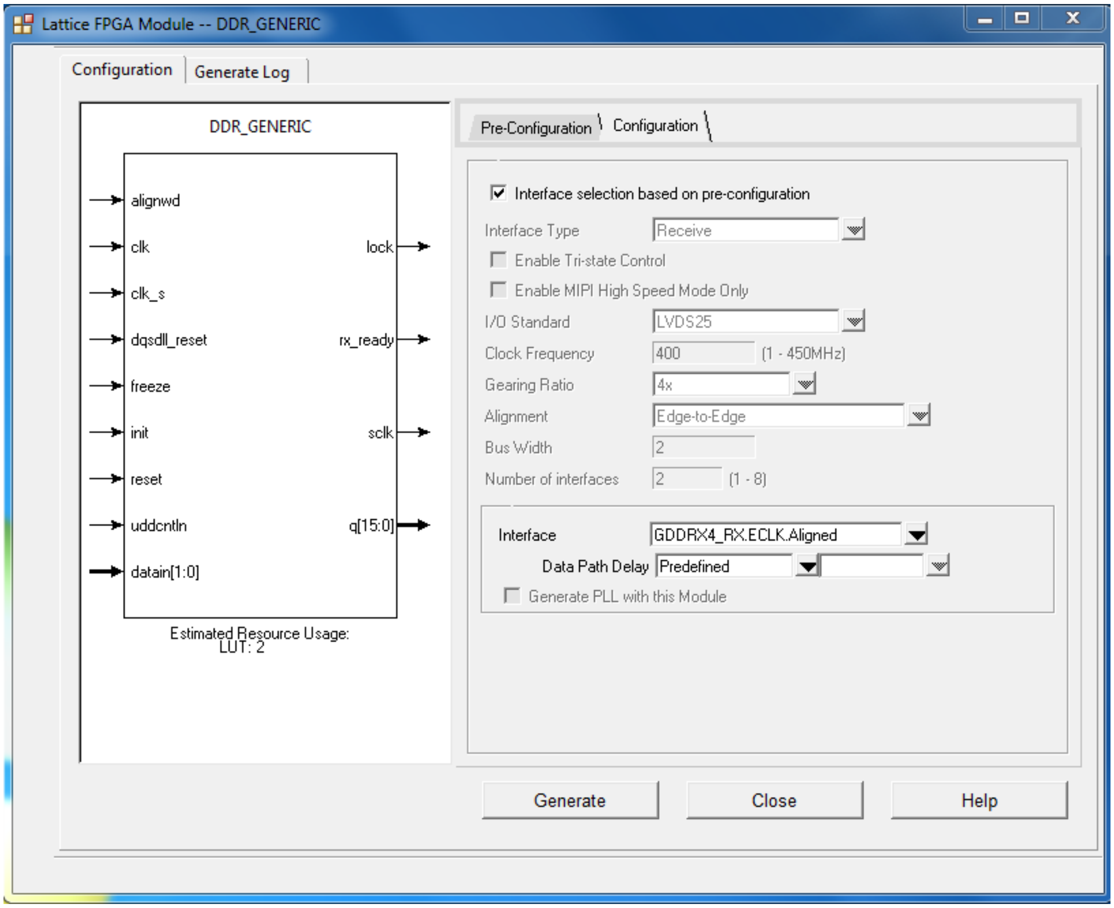


Figure 6.5. Configuration Tab of the DDR_Generic Modules

Based on the selections made in the Pre-Configuration tab, the Configuration tab is populated with the selections as shown in Figure 6.6. The checkbox at 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. Note that IPexpress chooses the most suitable interface based on selections made in the Pre-Configuration tab.

Table 6.4. User Interface Options in the Configuration Tab of the DDR_Generic Modules

User Interface Option	Description	Range	Default Value
Interface Selection based on pre-configuration	Indicates interface is selected based on selection made in the Pre-configuration tab. Disabling this checkbox allows you to select gearing ratio, delay types etc.	Enabled, Disabled	Enabled
Interface Type	Types of interfaces	Transmit, Receive, Transmit_MIPI, Receive_MIPI	—
I/O Standard	I/O standard for this interface	All I/O types per selected Interface Type	—
Clock Frequency	Speed at which the interface will run	1-378 MHz (for HP) 1-210 MHz (for LP)	—
Gearing Ratio	Choose the gearing ratio of the interface	×1, ×2, ×4	—
Alignment	Determine the type of external interfaces	Edge-to-Edge, Centered	—
Bus Width	Bus size for the interface	1-128	—

User Interface Option	Description	Range	Default Value
Number of Interfaces	Maximum number of buses supported	1-8	—
Interface	A list of the supported GDDR interfaces	Dependent of the Gearing ratio and Alignment choice	—
Data Path Delay[1]	Data input can be optionally delayed using the DELAY block.	Bypass, Predefined, User defined, Dynamic	—
Generate PLL with this Module[2]	Option to generate PLL with this module or not to generate PLL with this module.	Enabled, Disabled	—

Notes:

1. When "User Defined" is selected, the delay value field is enabled to allow you to select the delay values 0 to 31. When "Dynamic" is selected, a 5-bit delay port is added to the module. "Dynamic" can only be used for ×2 and ×4 receive interfaces.
2. This option is only available for interfaces that are using a PLL. This includes, GDDRX1_RX.SCLK.Aligned, GDDRX1_TX.SCLK.Centered, GDDRX2_TX.ECLK.Centered, and GDDRX4_TX.ECLK.Centered interfaces.

If the Pre-Configuration tab is used, the gearing ratio of the interface is determined by the speed of the interface. Table 6.5 shows how the gearing ratio is selected.

Table 6.5. Gearing Ratio Selection by the Software for Mach-NX Device

Device Type	Speed of the Interface	Gearing Ratio
High Performance (HP) device	=< 166 MHz	×1
	> 166 MHz and =< 266 MHz	×2
	> 266 MHz	×4
Low Power (LP) device	=< 70 MHz	×1
	>70 MHz and =< 133 MHz	×2
	>133 MHz	×4

6.3. Building a Generic DDR 7:1 Interface

As shown in Figure 6.6, you can choose interface type GDDR_71, enter module name and click Customize to open the Configuration tab. The Configuration tab options are listed in this section. The DDR 7:1 interface is a very specific application, so the options are relatively simple. Most of the necessary logic is built into the software for ease of use.

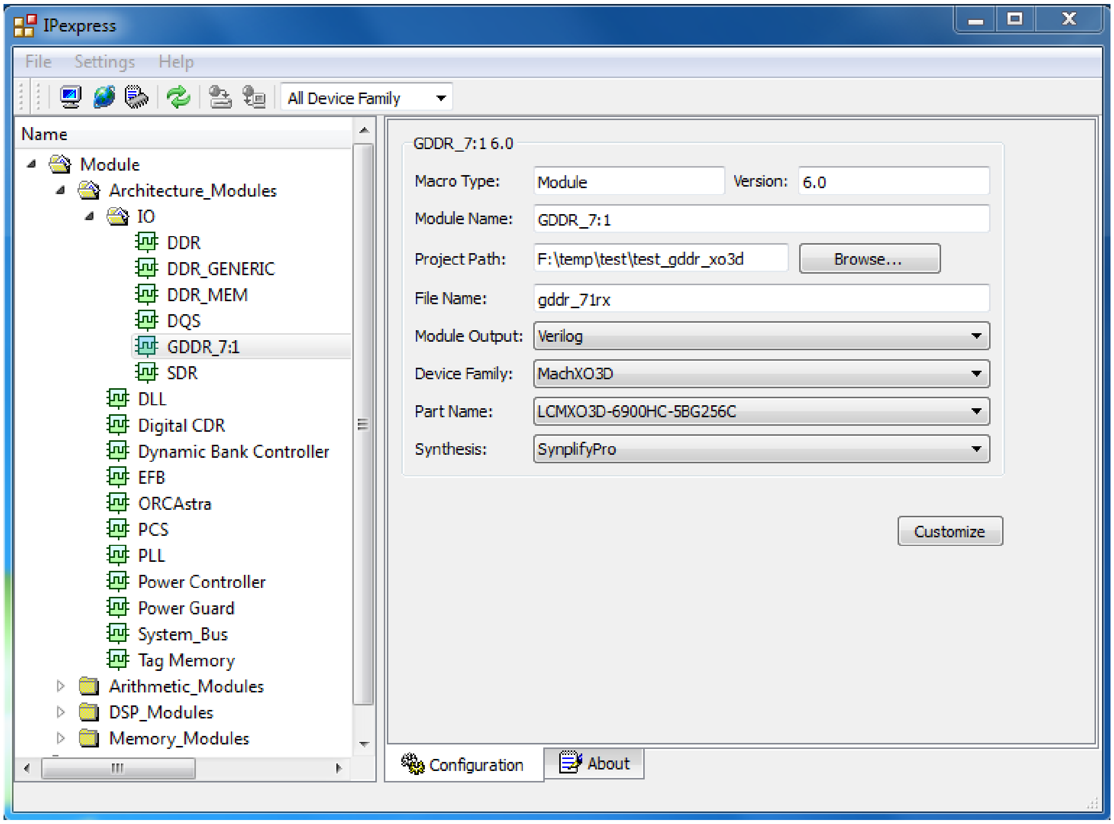


Figure 6.6. GDDR_71 Interface Selection at the IPexpress Main Window

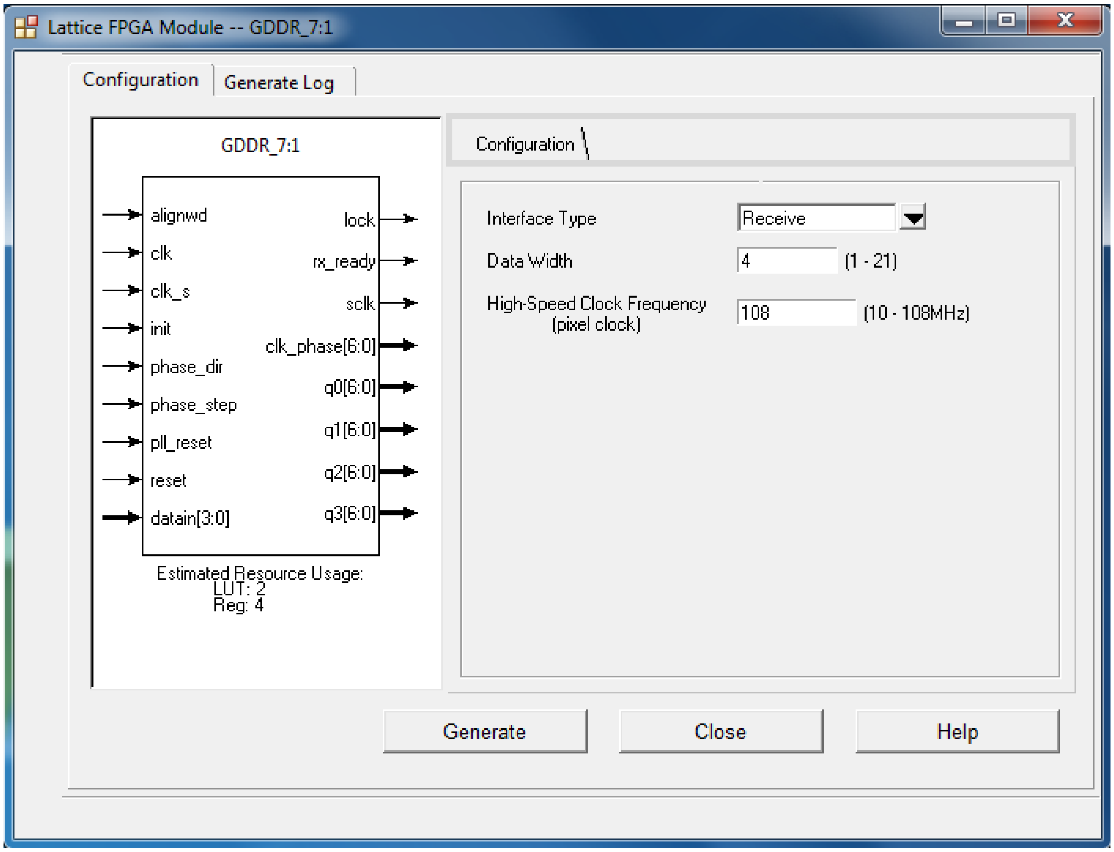


Figure 6.7. Configuration Tab of GDDR_71

Table 6.6. User Interface Options of the Configuration Tab for GDDR_71

User Interface Option	Description	Range	Default Value
Interface Type	Types of interfaces	Transmit, Receive	Receive
Data Width	The number of incoming channels	1-21	4
High speed Clock Frequency (Pixel Clock)	Pixel clock frequency for 7:1 LVDS interface	10-108 MHz	108

7. Generic High-Speed DDR Design Guidelines

7.1. I/O Logic Cells and Gearing Logic

Each Programmable I/O Cell (PIC) has four programmable I/O (PIOs), which form two pairs of I/O buffers. Each PIO by itself can support a ×1 gearing ratio. A pair of PIOs, either the A/B pair or C/D pair, can support a ×2 gearing ratio. Support of a ×4 or 7:1 gearing ratio will take up all four PIOs in one PIC block. The ×4 or 7:1 gearing ratio can only be supported when the A/B pair pins are available in the package, and are independent of the availability of the C/D pins. The total number of ×2 interfaces available in a specific package is determined by the total number of A/B and C/D pairs. The total number of ×4/7:1 interfaces available in a specific package is determined by the total number of A/B pairs.

Table 7.1. Gearing Logic Supported by Mixed Mode of I/O Logic Cells for Mach-NX Device

Gearing Type	I/O Logic A	I/O Logic B	I/O Logic C	I/O Logic D
×2 gearing (A/B pair)	IDDRX2 or ODDRX2	Not available	Basic I/O registers or ×1 gearing	Basic I/O registers or ×1 gearing
×2 gearing (C/D pair)	Basic I/O registers or ×1 gearing	Basic I/O registers or x1 gearing	IDDRX2 or ODDRX2	Not available
×4 gearing	IDDRX4 or ODDRX4	Not available	Basic I/O registers or ×1 gearing	Basic I/O registers or ×1 gearing
7:1 gearing	IDDRX71 or ODDRX71	Not available	Basic I/O registers or ×1 gearing	Basic I/O registers or ×1 gearing

7.2. High-Speed ECLK Bridge

The high-speed ECLK bridge is used to enhance communication of ECLKs across the Mach-NX device and is mainly used for high-speed video applications. The bridge allows a clock source to drive the edge clocks on the top and bottom edges of the device with minimal skew. The inputs to the bridge include primary clock pins from Bank 0 and Bank 2, PLL outputs from both sides, and clock tree routings.

Two bridge muxes are available in the ECLK bridge: one for each ECLK on the same side of the device. These muxes allow dynamic switching between two edge clocks. Refer to Mach-NX sysCLOCK PLL Usage Guide (FPGA-TN-02215) for ECLK bridge connectivity details.

The ECLK bridge supports all the generic high-speed interfaces except the high-speed ×2 and ×4 receive interfaces. The ECLK bridge component must be instantiated in the design in order to use the bridge function or to use it for routing purpose.

7.3. Reset Synchronization Requirement

The generic DDR interfaces are built with multiple dedicated circuits that are optimized for high-speed applications. It is therefore necessary to make sure all the components, such as CLKDIV and IDDR/ODDR, start with the same high-speed edge clock cycle to maintain the clock domain crossing margin between ECLK and SCLK, and to avoid bus bit-order scrambling due to the various delay of the reset pulse.

The ECLKSYNCA component and a particular reset sequence are required to guarantee a successful system implementation for interfaces using ×2, ×4, and 7:1 gearings. Figure 7.1 and Figure 7.2 show the timing requirements for receive interfaces and transmit interfaces. The RX_STOP or TX_STOP signal must be connected to the STOP port of the ECLKSYNCA component. The RX_RST or the TX_RST should be connected to the reset port of the ODDR/IDDR and CLKDIV components. The RX_ECLK or TX_ECLK are the outputs of the ECLKSYNCA components. It is necessary to have a minimum of two ECLK cycles between the RST and STOP signals as shown in the figures below. The receive interface reset process should not start until the transmit interface reset process is complete in a loopback implementation. The clock signal used to generate the minimum two ECLK delay can be any clock that is slower than the ECLK frequency.

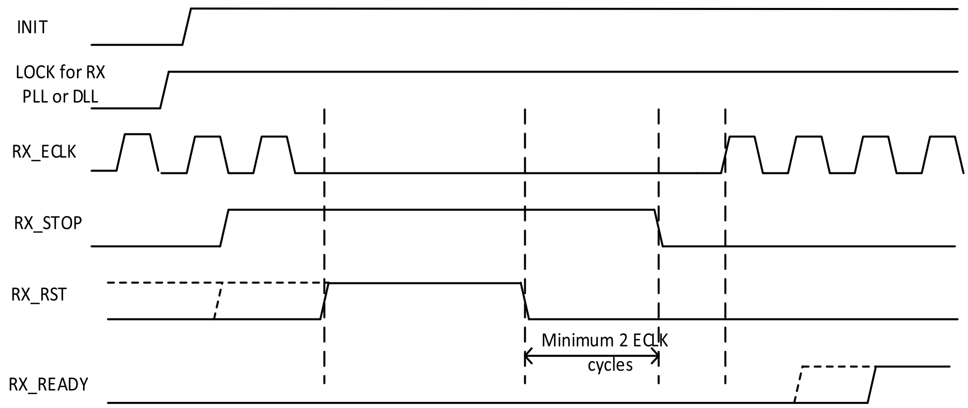


Figure 7.1. Reset Synchronization for Receive Interfaces

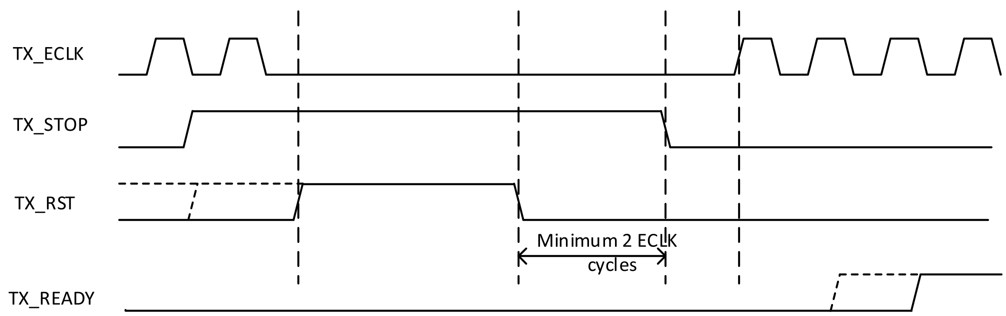


Figure 7.2. Reset Synchronization for Transmit Interfaces

These timing requirements are built into the generic DDR ×2/×4/7:1 modules when they are generated by IPexpress. The RX_STOP/TX_STOP, RX_RST/TX_RST, and RX_ECLK/TX_ECLK are internal signals when the modules are generated by IPexpress. The reset timing requirements must be followed and implemented in RTL code when the generic DDR interfaces are built outside of IPexpress.

8. DDR Software Primitives and Attributes

Software primitives used for all the generic DDR interfaces implementation are discussed in this section. The primitives are divided according to their usage. Some of them are used for generic DDR interfaces and others are control functions. The DDR input primitives are discussed first, followed by the DDR output primitives, then the DDR control logic primitives.

Table 8.1. Mach-NX DDR Software Primitives

Type	Primitive	Usage
Data Input	IDDRXE	Generic DDR ×1
	IDDRX2E	Generic DDR ×2
	IDDRX4B	Generic DDR ×4
	IDDRX71A	Generic DDR 7:1
Data Output	ODDRXE	Generic DDR ×1
	ODDRX2E	Generic DDR ×2
	ODDRX4B	Generic DDR ×4
	ODDRX71A	Generic DDR 7:1
DLL	DQSDLLC	Master DLL for Generic ×2 and ×4
Input Delay	DELAYD	Delay block with dynamic control for Generic ×2, ×4
	DELAYE	Delay block with fixed delays for Generic DDR ×1, ×2, ×4
	DLLDELC	Clock slave delay cell for Generic DDR ×2, ×4

8.1. Input DDR Primitives

The input DDR primitives represent the modules used to capture both the GDDR data. There are several modes for the DDR input registers to implement different gearings for GDDR interfaces. For all the data ports of the input primitives, Q0 of the parallel data is the first bit received.

8.1.1. IDDRXE

The primitive implements the input register block in ×1 gearing mode. This is used only for the generic DDR ×1 interface (gearing ratio 1:2) available for Bank 0, Bank 1, and Bank 2 of the Mach-NX device. It uses a single clock source, SCLK, for the entire primitive so it does not involve a clock domain transfer.

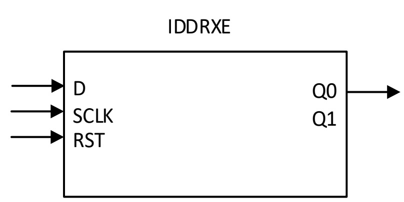


Figure 8.1. IDDRXE Symbol

The internal register structure for this primitive is based on the basic PIO cell, as shown in Figure 2.1. The first set is the DDR register to capture the data at both edges of the SCLK. The second set is the synchronization registers to transfer the captured data to the FPGA core.

8.1.2. IDDRX2E

The primitive implements the input register block in ×2 gearing mode. This is used only for the generic DDR ×2 interface (gearing ratio 1:4) in Bank 2. Its registers are designed to use edge clock routing on the GDDR interface and the system clock for FPGA core. The edge clock is connected to the ECLK port, while the system clock is connected to the SCLK port. This primitive can be used on both A/B or C/D pairs of I/O cells in Bank 2.

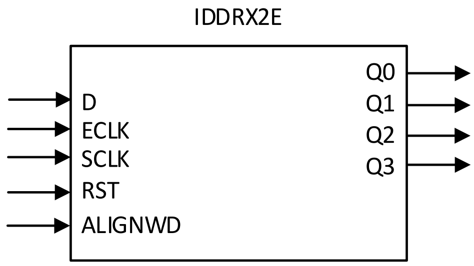


Figure 8.2. IDDRX2E Symbol

The internal register structure for this primitive is based on the video PIO cell, as shown in receive path of Figure 2.2.

The first set of the registers is the DDR register to capture the data at both edges of the ECLK. The second set is the synchronization registers to hold the data ready for clock domain transfer. The third set of registers performs the clock domain transfer from ECLK to SCLK.

8.1.3. IDDRX4B

The primitive implements the input register block in ×4 gearing mode. This is used only for the generic DDR ×4 interface (gearing ratio 1:8) in Bank 2. Its registers are designed to use edge clock routing on the GDDR interface and the system clock on the FPGA side. The edge clock is connected to the ECLK port, while the system clock is connected to the SCLK port. This primitive can only be used on the A/B pair of the I/O cells in Bank 2.

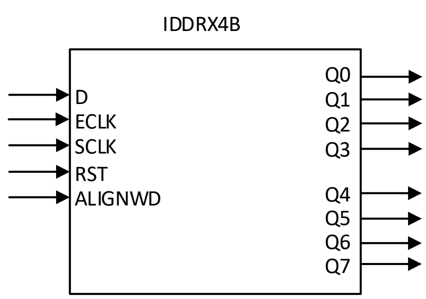


Figure 8.3. IDDRX4B Symbol

The 1:8 gearing of IDDRX4B uses two of the 1:4 gearing and shares the basic architecture with IDDRX2E. The internal register structure for this primitive is based on the video PIO cell, as shown in receive path of Figure 2.2. The first set of registers is the DDR register to capture the data at both edges of the ECLK. The second set of registers is synchronization registers to hold the data ready for clock domain transfer. The third set of registers performs the clock domain transfer from ECLK to SCLK.

8.1.4. IDDRX71A

The primitive implements the input register block in 7:1 gearing mode. This is used only for the generic DDR 71 interface (gearing ratio 1:7) in Bank 2. Its registers are designed to use edge clock routing on the GDDR interface and the system clock on the FPGA side. The edge clock is connected to the ECLK port, while the system clock is connected to the SCLK port. This primitive, like the input x4 gearing primitive, can only be used on the A/B pair of the PIO cell in Bank 2.

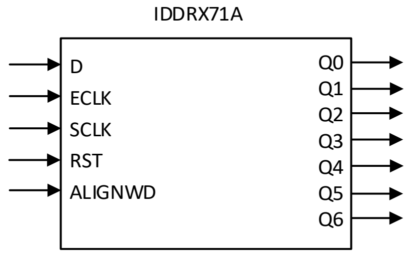


Figure 8.4. IDDRX71A Symbol

The 1:7 gearing of IDDRX71A shares the same architecture as the 1:8 gearing of the IDDRX4B primitive. It depends on an internal control signal to select three bits or four bits data at a time. The internal register structure for this primitive is based on the video PIO cell, as shown in the receive path of Figure 2.2. The first set of the registers is the DDR register to capture the data at both edges of ECLK. The second set is the synchronization registers to hold the data ready for clock domain transfer. The third set of registers performs the clock domain transfer from ECLK to SCLK.

8.2. Output DDR Primitives

The output DDR primitives represent the output DDR module used to multiplex two data streams before sending them out to the GDDR interface.

8.2.1. ODDRXE

This primitive will implement the output register block in ×1 gearing mode. It can be used in Bank 0, Bank 1, and Bank 2 of the Mach-NX device. A single primary clock source, SCLK from the FPGA core, is used for this primitive.

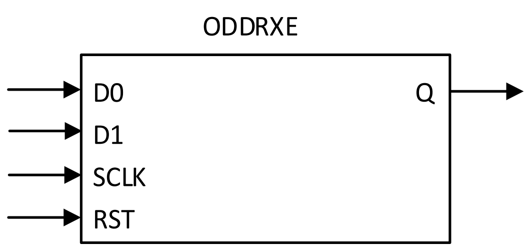


Figure 8.5. ODDRXE Symbol

The internal register structure for this primitive is based on the basic PIO cell, as shown in Figure 2.1. The SCLK is used to multiplex between the 2-bit parallel data to generate a serial data stream.

8.2.2. ODDRX2E

The primitive implements the output register block in ×2 gearing mode. This is used only for the generic DDR ×2 interface (gearing ratio 4:1) in Bank 0. Its registers are designed to use the system clock on the FPGA side and the edge clock at the DDR interface. The edge clock is connected to ECLK port, while the system clock is connected to SCLK port. This primitive can be used on both the A/B or C/D pairs of PIO cells in Bank 0.

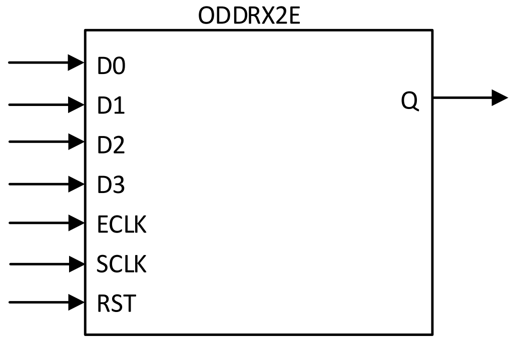


Figure 8.6. ODDRX2E Symbol

The internal register structure for this primitive is based on the video PIO cell, as shown in transmit path of Figure 2.2. The parallel data is registered by SCLK at the first set of registers. At each update, the parallel data is clocked in by SCLK and is held by the second set of registers. The third set of registers has the data ready to be multiplexed out as serial data by the ECLK.

8.2.3. ODDRX4B

The primitive implements the output register block in x4 gearing mode. This is used only for the generic DDR ×4 interface (gearing ratio 8:1) in Bank 0. Its registers are designed to use the system clock on the FPGA side and the edge clock at the GDDR interface. The edge clock is connected to the ECLK port, while the system clock is connected to the SCLK port. This primitive can only be used on the A/B pair of I/O cells in Bank 0.

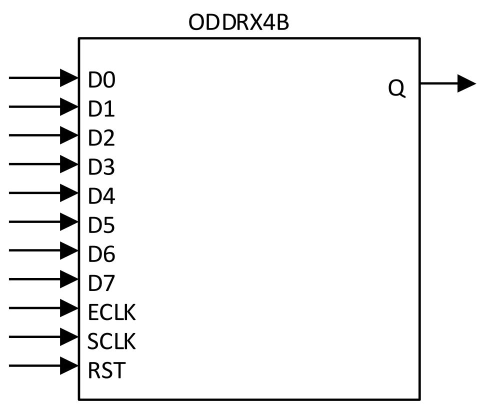


Figure 8.7. ODDRX4B Symbol

The internal register structure for this primitive is based on the video PIO cell, as shown in the transmit path of Figure 2.2. The parallel data is registered by SCLK at the first set of registers. At each update, the parallel data is clocked in by SCLK and is held by the second set of registers. The third set of registers has the data ready to be multiplexed out as serial data by ECLK.

8.2.4. ODDRX71A

The primitive implements the output register block in 7:1 gearing mode. This is used only for the generic DDR 71 interface (gearing ratio 7:1) in Bank 0. Its registers are designed to use the system clock on the FPGA side and the edge clock routing on the GDDR interface. The edge clock is connected to the ECLK port, while the system clock is connected to the SCLK port. This primitive can only be used on the A/B pair of I/O cells in Bank 0.

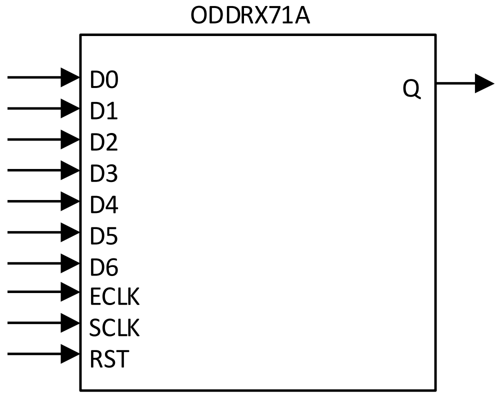


Figure 8.8. ODDRX71A Symbol

The 7:1 gearing of ODDRX71A shares the same architecture as the 8:1 gearing of the ODDRX4B primitive. It depends on the internal control signal to select the three or four bits of data at a time for transmission. The internal register structure for this primitive is based on the video PIO cell, as shown in the transmit path of Figure 2.2. The parallel data is registered by SCLK at the first set of registers. At each update, the parallel data is clocked in by SCLK and is held by the second set of registers. The third set of registers has the data ready to be multiplexed out as serial data by ECLK.

8.3. DDR Control Logic Primitives

The DDR primitives discussed below include the DLL and the delay elements. The delay elements are for data paths or the clock slave delay paths.

8.3.1. DQSDLLC

The DQSDLLC is the on-chip DLL, which generates the 90° phase shift required for the DQS signal. Only one DQSDLLC can be used for the DDR implementations on one-half of the device. The clock input to this DLL should be at the same frequency as the DDR interface. The DQSDLLC generates the delay based on this clock frequency and the update control input to this block. The DQSDLLC updates the dynamic delay control code (DQSDEL) to the DLLDELC block when this update control (UDDCNTLN) input is asserted. Otherwise, the update will be in the hold condition. The active low signal on UDDCNTLN updates the clock phase using DQSDEL delay.

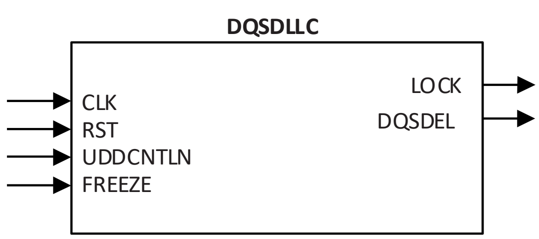


Figure 8.9. DQSDLLC Symbol

Table 8.2. DQSDLLC Signals

Signal	I/O	Description
CLK	I	Input clock to the DLL, same frequency as DDR interface
RST	I	DLL reset control
UDDCNTLN	I	Update/hold control to delay code before adjustment. Active low signal updates the delay code.
FREEZE	I	Use to freeze or release DLL input CLK
LOCK	O	DLL lock signal
DQSDEL	O	DLL delay control code to slave delay

The DQS delay can be updated for PVT variation using the UDDCNTLN input. The DQSDEL is updated when the UDDCNTLN is held low. The DQSDEL can be updated when variations are expected. It can be updated anytime except during a READ or WRITE operation.

The FREEZE input port of this component is used to freeze or release the DLL. When FREEZE goes high, the device will freeze the DLL to save power while the delay code is preserved. When FREEZE goes low, it will release the DLL to resume operation. FREEZE must be applied to the DQSDLLC before the clock stops.

By default, this DLL will generate a 90° phase shift for the DQS strobe based on the frequency of the input reference clock to the DLL. You can control the sensitivity to jitter by using the LOCK_SENSITIVITY attribute. This configuration bit can be programmed to be either HIGH or LOW. Lock_sensitivity HIGH means more sensitive to jitter. It is recommended that the bit be programmed LOW.

The DQSDLLC supports a wide range of frequencies up to 400 MHz. The FIN attribute associated with this primitive allows you to set the DLL frequency. It is possible to bypass the DLL locking process when the frequency becomes very low. The attribute FORCE_MAX_DELAY can be used for this purpose. When FORCE_MAX_DELAY is set in the software, the DLL does not go through the locking process. Instead, DLL is locked to maximum delay steps. The effect of the FORCE_MAX_DELAY attribute is not reflected in the simulation model. The simulation model always models the 90° phase shift for DLL.

Table 8.3. Attribute for DQSDLLC

Attribute	Description	Values	Software Default
LOCK_SENSITIVITY	Jitter sensitivity	High, Low	Low
FIN	Input clock frequency of DLL	Range supported by DLL	100 MHz
FORCE_MAX_DELAY	Bypass DLL locking procedure at low frequency and sets the maximum delay setting	Yes, No	No

8.3.2. DELAYE

Input data going to the DDR registers can optionally be delayed using the delay block, DELAYE. The 32-tap DELAYE block is used to compensate for clock injection delay time. The amount of the delay is determined by the software based on the type of interface implemented using the attribute DEL_MODE. You are allowed to set the delay by choosing the USER_DEFINED mode for the block. When in USER_DEFINED mode, you must manually set the number of delay steps to be used. Each delay stay would generate ~105 ps of delay. It is recommended to use the PREDEFINED mode for all generic DDR interfaces. If an incorrect attribute value is used for a given interface, the DELAYE setting will be incorrect and the performance of the DDR interface will not be optimal. The DELAYE block is applicable to the receive mode of the DDR interfaces. It is available for all input register paths at Bank 0, Bank 1, and Bank 2 of a Mach-NX device.

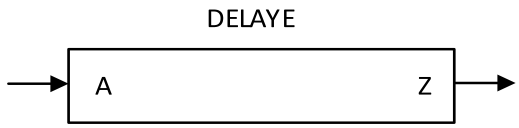


Figure 8.10. DELAYE Symbol

Table 8.4. DELAYE Signals

Signal	I/O	Description
A	I	DDR input from sysI/O buffer
Z	O	Output with delay

Table 8.5. DELAYE attributes

Attribute	Description	Values	Software Default
DEL_MODE	Fixed delay value depending on interface and user-defined delay values	SCLK_ZEROHOLD ECLK_ALIGNED ECLK_CENTERED SCLK_ALIGNED SCLK_CENTERED USER_DEFINED	USER_DEFINED
DEL_VALUE	User-defined value	DELAY0…DELAY31	DELAY0

Table 8.6. DEL_MODE Values Corresponding to the GDDR Interface

Interface Name	DEL_MODE Values
GIREG_RX.SCLK	SCLK_ZERHOLD
GDDRX1_RX.SCLK.Aligned	SCLK_ALIGNED
GDDRX1_RX.SCLK.Centered	SCLK_CENTERED
GDDRX2_RX.ECLK.Aligned	ECLK_ALIGNED
GDDRX2_RX.ECLK.Centered	ECLK_CENTERED
GDDRX4_RX.ECLK.Aligned	ECLK_ALIGNED
GDDRX4_RX.ECLK.Centered	ECLK_CENTERED
GDDRX71_RX.ECLK.71	Bypass
GOREG_TX.SCLK	N/A
GDDRX1_TX.SCLK.Centered	N/A
GDDRX1_TX.SCLK.Aligned	N/A
GDDRX2_TX.ECLK.Aligned	N/A
GDDRX2_TX.ECLK.Centered	N/A
GDDRX4_TX.ECLK.ALIGNED	N/A
GDDRX4_TX.ECLK.CENTERED	N/A
GDDRX_TX.ECLK.7:1	N/A

8.3.3. DELAYD

In Bank 2, input data going to the DDR registers can also be delayed by the DELAYD block. Unlike the DELAYE block where the delay is determined during the operation of the device, the DELAYD block allows you to control the amount of data delay while the device is in operation. This block receives 5-bit (32 taps) delay control. The 5-bit delay is dynamically controlled by the user logic through the delay port. Each delay step would generate ~105 ps of delay.

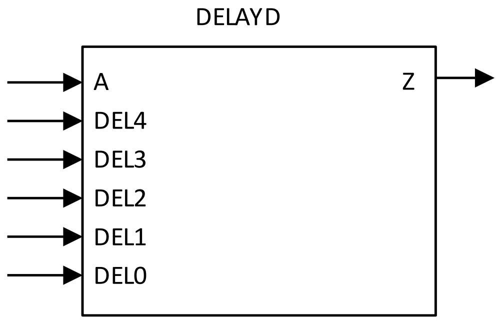


Figure 8.11. DELAYD Symbol

Table 8.7. DELAYD Signals

Signal	I/O	Description
A	I	DDR input from I/O buffer
DEL4, DEL3, DEL2, DEL1, DEL0	I	Dynamic delay input port from FPGA logic
Z	O	Output with delay

8.3.4. DLLDELC

This is the clock slave delay cell, which is used to generate a 90° delay in all receive aligned interfaces. The 90° delay is calculated based on the input clock to the DQSDLLC element. The amount of delay required is based on the delay control code, DQSDEL, generated from the DQSDLLC.

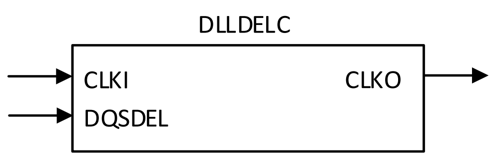


Figure 8.12. DLLDELC Symbol

Table 8.8. DLLDELC Signals

Signal	I/O	Description
CLKI	I	Data Input from I/O buffer
DQSDEL	I	Dynamic delay inputs from DQSDLLC
CLKO	O	Output with delay

Technical Support Assistance

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

Revision History

Revision 1.1, March 2024

Section	Change Summary
Disclaimers	Updated disclaimers.
Generic High-Speed DDR Design Guidelines	Removed Timing Analysis for High-Speed GDDR Interfaces section.

Revision 1.0, December 2020

Section	Change Summary
All	Changed document status from preliminary to final.

Revision 0.81, October 2020

Section	Change Summary
Architecture for High-Speed Interfaces	• Indicated sysI/O banks. • Updated Table 2.1. Gearing Logic Distribution for Mach-NX Device.
High-Speed Interface Building Blocks	Updated DQSDLL section.
Generic High-Speed DDR Interfaces	Updated Table 5.1. Generic High-Speed I/O DDR Interfaces.
DDR Software Primitives and Attributes	Indicated Bank 0, Bank 1, and Bank 2 in the IDDRXE, ODDRXE and DELAYE sections.

Revision 0.80, September 2020

Section	Change Summary
All	First preliminary release.

www.latticesemi.com