Implementing High-Speed Interfaces with MachXO2 Devices

Technical Note

FPGA-TN-02153-1.9

June 2021

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

Acronyms in This Document 8
1. Introduction 9
2. Architecture for High-Speed Interfaces 9
2.1. Gearing Logic Distribution 9
2.2. Different Types of I/O Logic Cells 10
2.3. Clock Domain Transfer at PIO Cells 13
3. External High-Speed Interface Description 17
4. High-Speed Interface Building Blocks 18
4.1. ECLK 18
4.2. ECLKSYNC 18
4.3. SCLK 18
4.4. CLKDIV 18
4.5. PLL 18
4.6. DQSDLL 18
4.7. Input DDR (IDDR) 19
4.8. Output DDR (ODDR) 19
4.9. Delays 19
4.10. DQSBUF 19
4.11. IDDRDQS 19
4.12. ODDRDQS 19
5. Generic High-Speed DDR Interfaces 20
5.1. High-Speed GDDR Interface Types 20
5.2. High-Speed GDDR Interface Details 21
5.3. Receive Interfaces 21
5.3.1. GIREG_RX.SCLK 21
5.3.2. GDDRX1_RX.SCLK.Aligned 22
5.3.3. GDDRX1_RX.SCLK.Centered 22
5.3.4. GDDRX2_RX.ECLK.Aligned 23
5.3.5. GDDRX2_RX.ECLK.Centered 23
5.3.6. GDDRX4_RX.ECLK.Aligned 24
5.3.7. GDDRX4_RX.ECLK.Centered 25
5.3.8. GDDR71_RX.ECLK.7:1 26
5.4. Transmit Interfaces 27
5.4.1. GOREG_TX.SCLK 27
5.4.2. GDDRX1_TX.SCLK.Aligned 27
5.4.3. GDDRX1_TX.SCLK.Centered 28
5.4.4. GDDRX2_TX.ECLK.Aligned 28
5.4.5. GDDRX2_TX.ECLK.Centered 29
5.4.6. GDDRX4_TX.ECLK.Aligned 30
5.4.7. GDDRX4_TX.ECLK.Centered 30
5.4.8. GDDR71_TX.ECLK.7:1 31
6. Using IPexpress to Build Generic High-Speed DDR Interfaces 33
6.1. Building the SDR Interface 34
6.2. Building DDR Generic Interfaces 36
6.3. Building a Generic DDR 7:1 Interface 39
7. Generic High-Speed DDR Design Guidelines 41
7.1. I/O Logic Cells and Gearing Logic 41
7.2. High-Speed ECLK Bridge 41
7.3. Reset Synchronization Requirement 41
7.4. Timing Analysis for High-Speed GDDR Interfaces 42
7.5. Timing Rule Check for Clock Domain Transfers 46
8. DDR/DDR2/LPDDR SDRAM Interfaces Overview 47

9. DDR/DDR2/LPDDR SDRAM Interfaces Implementation 50
9.1. DQS Grouping 50
9.2. DQS Circuitry 50
9.3. I/O Logic Data Path 51
9.4. DDR/DDR2/LPDDR Memory READ Implementation 51
9.5. DDR/DDR2/LPDDR Memory WRITE Implementation 52
10. DDR Memory Interface Generation Using IPexpress 55
11. DDR Memory DQ/DQS Design Rules and Guidelines 58
12. DDR/DDR2/LPDDR Pinout Guidelines 58
13. DDR Software Primitives and Attributes 59
13.1. Input DDR Primitives 59
13.2. IDDRX2E 60
13.3. Output DDR Primitives 62
13.4. DDR Control Logic Primitives 65
Technical Support Assistance 71
Revision History 72

Figures

Figure 2.1. Basic PIO Cell Supports x1 Gearing Ratio 10
Figure 2.2. Memory PIO Cell to Support DDR Memory Applications 11
Figure 2.3. Video PIO Cell for x2/x4 and 7:1 Applications 13
Figure 2.4. 7:1 Deserializer Timing 14
Figure 2.5. x4 Deserializer Timing (Even Phases, SEL=0) 14
Figure 2.6. x4 Deserializer Timing (Odd Phases, SEL=1) 15
Figure 2.7. x2 Deserializer Timing (Even Phases, SEL=0) 15
Figure 2.8. x2 Deserializer Timing (Odd Phases, SEL=1) 15
Figure 2.9. 7:1 Deserializer Timing in response to ALIGNWD 15
Figure 2.10. x4 Deserializer Timing in Response to ALIGNWD (Even-to-Odd Phase) 16
Figure 2.11. x4 Deserializer Timing in Response to ALIGNWD (Odd-to-Even Phase) 16
Figure 3.1. External Interface Definition 17
Figure 5.1. GIREG_RX Interface 21
Figure 5.2. GDDRX1_RX.SCLK.Aligned Interface Using DQSDLL 22
Figure 5.3. GDDRX1_RX.SCLK.Centered 22
Figure 5.4. GDDRX2_RX.ECLK.Aligned Interface 23
Figure 5.5. GDDRX2_RX.ECLK.Centered Interface 24
Figure 5.6. GDDRX4_RX.ECLK.Aligned Interface 24
Figure 5.7. GDDRX4_RX.ECLK.Centered Interface 25
Figure 5.8. GDDR71_RX.ECLK.7:1 Interface 26
Figure 5.9. GOREG_TX.SCLK Interface 27
Figure 5.10. GDDRX1_TX.SCLK.Aligned Interface 27
Figure 5.11. GDDRX1_TX.SCLK.Centered Interface 28
Figure 5.12. GDDRX2_TX.ECLK.Aligned Interface 28
Figure 5.13. GDDRX2_TX.ECLK.Centered Interface 29
Figure 5.14. GDDRX4_TX.ECLK.Aligned Interface 30
Figure 5.15. GDDRX4_TX.ECLK.Centered Interface 31
Figure 5.16. GDDR71_TX.ECLK.7:1 Interface 32
Figure 6.1. SDR Interface Selection at the IPexpress Main Window 34
Figure 6.2. Configuration Tab for the SDR Interfaces 35
Figure 6.3. DDR_Generic Interface Selection at the IPexpress Main Window 36
Figure 6.4. Pre-Configuration Tab of the DDR_Generic Interfaces 37
Figure 6.5. Configuration Tab of the DDR_Generic Modules 38
Figure 6.6. GDDR_71 Interface Selection at the IPexpress Main Window 39
Figure 6.7. GDDR_71 Configuration Tab of GDDR_71 40
Figure 7.1. Reset Synchronization for Receive Interfaces 42
Figure 7.2. Reset Synchronization for Transmit Interfaces 42
Figure 7.3. Receiver RX.CLK.Centered Waveforms 43
Figure 7.4. Receiver RX.CLK.Aligned and MEM DDR Input Waveforms 43
Figure 7.5. Receiver GDDR71_RX. Waveforms 44
Figure 7.6. t_{CO} Minimum and Maximum Timing Analysis 44
Figure 7.7. Transmitter TX.CLK.Centered and MEM DDR Output Waveforms 45
Figure 7.8. Transmitter TX.CLK.Aligned Waveforms 45
Figure 7.9. Transmitter GDDR71_TX. Waveforms 46
Figure 8.1. Typical DDR SDRAM Interface 47
Figure 8.2. Typical DDR2 SDRAM Interface 48
Figure 8.3. Typical LPDDR SDRAM Interface 48
Figure 8.4. DQ-DQS Relationship During READ 48
Figure 8.5. DQ-DQS Relationship During WRITE 49
Figure 9.1. DDR/DDR2/LPDDR READ Implementation 52
Figure 9.2. DDR/DDR2/LPDDR WRITE Implementation 53
Figure 9.3. CLK, Address and Command Control Pin Generation 54

Figure 10.1. IPexpress Main Window 55
Figure 10.2. Configuration Tab for DDR_MEM 56
Figure 10.3. Clock/Address/Command Tab for DDR_MEM 57
Figure 13.1. IDDRXE Symbol 60
Figure 13.2. IDDRX2E Symbol 60
Figure 13.3. IDDRX4B Symbol 61
Figure 13.4. IDDRX71A Symbol 61
Figure 13.5. IDDRXQSX1A Symbol 62
Figure 13.6. ODDRXE Symbol 62
Figure 13.7. ODDRX2E Symbol 63
Figure 13.8. ODDRX4B Symbol 63
Figure 13.9. ODDRX71A Symbol 64
Figure 13.10. ODDRDQSX1A Symbol 64
Figure 13.11. TDDRA Symbol 64
Figure 13.12. DQSBUFH Symbol 65
Figure 13.13. DQSBUFH Block Diagram 66
Figure 13.14. READ Pulse Positioning Optimization 67
Figure 13.15. DQSDLLC Symbol 67
Figure 13.16. DELAYE Symbol 68
Figure 13.17. DELAYD Symbol 69
Figure 13.18. DLLDELC Symbol 70

Tables

Table 2.1. Gearing Logic Distribution for MachXO2-640U, MachXO2-1200/U and Higher Density Devices 9
Table 5.1. Generic High-Speed I/O DDR Interfaces 20
Table 6.1. Signal Names Used by IPexpress Modules 33
Table 6.2. User Interface Options for the SDR Interfaces 35
Table 6.3. User Interface Options for the Pre-Configuration Tab of DDR_Generic Modules 37
Table 6.4. User Interface Options of the Configuration Tab of the DDR_Generic Modules 38
Table 6.5. Gearing Ratio Selection by the Software 39
Table 6.6. User Interface Options of the Configuration Tab for GDDR_71 40
Table 7.1. Gearing Logic Supported by Mixed Mode of I/O Logic Cells 41
Table 8.1. DDR / DDR2 and LPDDR Specification for MachXO2-640U, MachXO2-1200/U and Higher Density Devices 47
Table 10.1. Options of the Configuration Tab of the DDR_MEM Module 56
Table 10.2. Options of the Configuration Tab of the DDR_MEM Module 57
Table 13.1. MachXO2 DDR Software Primitives 59
Table 13.2. TDDRA Attributes 65
Table 13.3. DQSBUFH signals 65
Table 13.4. DQSDLLC Signals 67
Table 13.5. Attribute for DQSDLLC 68
Table 13.6. DELAYE Signals 68
Table 13.7. DELAYE Attributes 68
Table 13.8. DEL_MODE Values Corresponding to the GDDR Interface 69
Table 13.9. DELAYD signals 69
Table 13.10. DLLDELC Signals 70

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
BL	Burst Length
DDR	Double Data Rate
DLL	Delay-Locked Loop
DM	Data Mask
ECLK	Edge Clock
FPGA	Field-Programmable Gate Array
GDDR	Generic Double Data Rate
ODDR	Output Double Data Rate
PCLK	Primary Clock
PIO	Programmable Input/Output
PLL	Phase-Locked Loop
SCLK	System Clock
SDR	Single Data Rate
UI	Unit Intervals

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 DDR clocking technique has largely been used in memory interfaces such as DDR SDRAM. As a result, DDR SDRAM memories achieve twice the bandwidth as SDR SDRAM memories without increasing the signal integrity requirements in the system.

The Lattice MachXO2™ PLD family supports high-speed interfaces for both DDR and SDR applications through built-in Programmable I/O (PIO) logic. The MachXO2 devices also have dedicated circuitry to support DDR, DDR2, and LPDDR SDRAM memory interfaces. This document focuses on the implementation of high-speed generic DDR interfaces, and memory DDR/DDR2 and LPDDR interfaces in the MachXO2 devices. It also provides guidelines for making use of the built-in capabilities of the MachXO2 devices to achieve the best performance for highspeed 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 MachXO2 devices depending on the I/O bank locations and the logic density. The x1 gearing ratio is available in all banks for all the device densities. The x2, x4, and the 7:1 gearing ratio are available in the top and bottom banks of the MachXO2-640U, MachXO2-1200/U and higher density devices. The 7:1 gearing ratio is mainly used for video display applications. The x2/x4 gearing circuit is shared with the 7:1 circuit on both receive and the transmit sides. The right bank of the MachXO2-640U, MachXO2-1200/U and higher density devices support the memory DDR interface. The memory DDR uses x1 gearing logic in the dedicated memory PIO cells. Table 2.1 gives a breakdown of gearing logic support in the different I/O banks. Details of PIO cells can be found in the MachXO2 Family Data Sheet (FPGA-DS-02056).

Table 2.1. Gearing Logic Distribution for MachXO2-640U, MachXO2-1200/U and Higher Density Devices

Gearing Logic	Definition	Gearing Ratio	Left	Right	Bottom	Top
DDR x1*	GDDR	1:2 or 2:1	Yes	Yes	Yes	Yes
Input DDR x2	GDDR	1:4	—	—	Yes	—
Input DDR x4	GDDR	1:8	—	—	Yes	—
Input DDR 7:1	GDDR	1:7	—	—	Yes	—
Output DDR x2	GDDR	4:1	—	—	—	Yes
Output DDR x4	GDDR	8:1	—	—	—	Yes
Output DDR 7:1	GDDR	7:1	—	—	—	Yes
mem DDR x1	Memory DDR	1:2 or 2:1	—	Yes	—	—

***Note:** DDRx1 is available for all MachXO2 device densities.

2.2. Different Types of I/O Logic Cells

In order to support various gearing ratios, the MachXO2 devices support three types of PIO logic cells. These include a basic PIO cell, a memory PIO cell, and a video PIO cell.

The basic PIO cell supports traditional SDR registers and DDR x1 registers. It is available on all sides of all MachXO2 devices. The memory PIO cell supports DDR memory applications and is available on the right side of the MachXO2-640U, MachXO2-1200/U and higher density devices. The video PIO cell supports the x2/x4 and 7:1 gearing applications. They are available on MachXO2-640U, MachXO2-1200/U and larger devices on the bottom side for the receive interfaces, and on the top side for the transmit interfaces. The input and output structures of each type of PIO cell are discussed in detail in the MachXO2 Family Data Sheet (FPGA-DS-02056). The block diagrams of the PIO cells are shown in this document for reference.

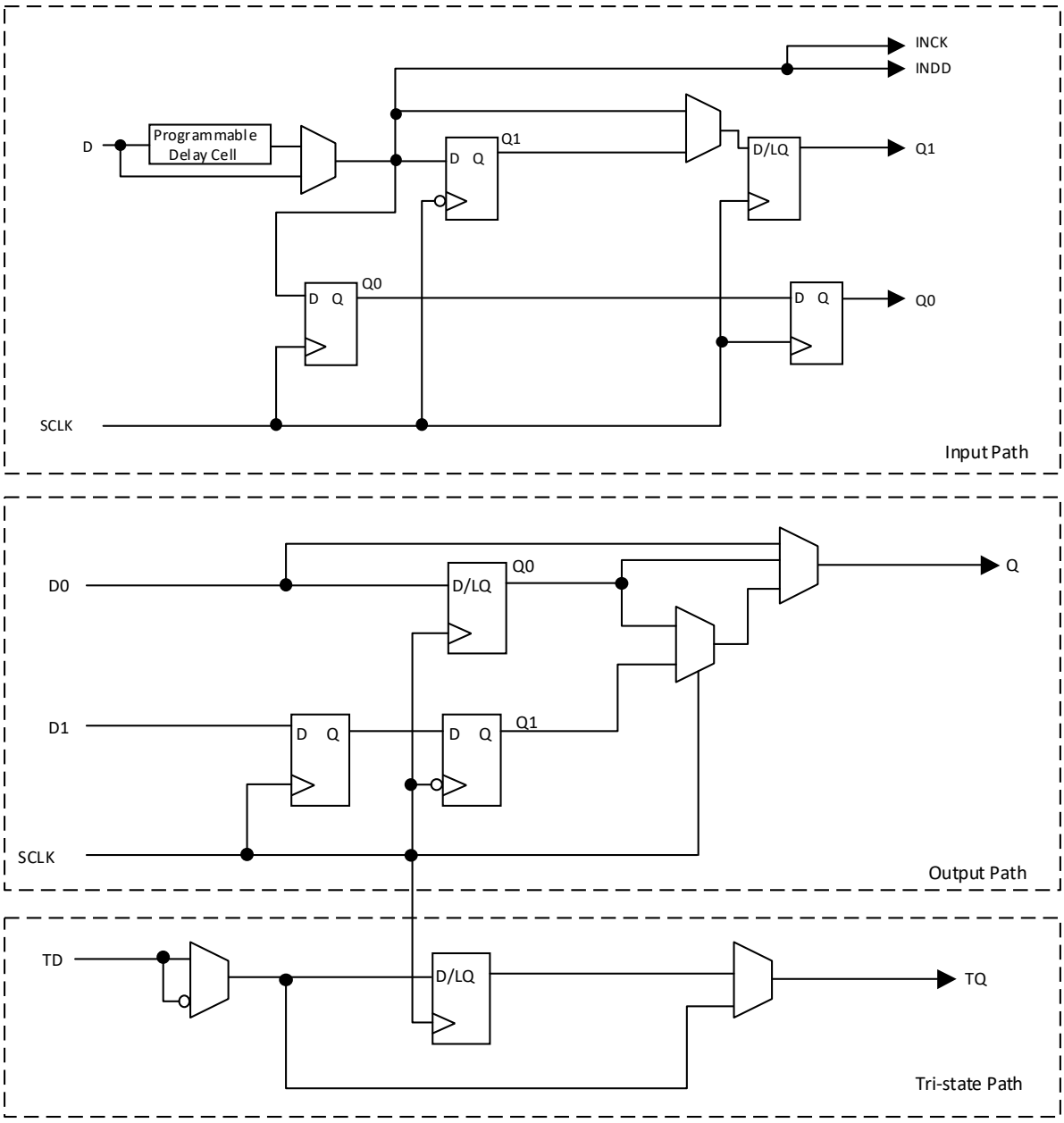


Figure 2.1. Basic PIO Cell Supports x1 Gearing Ratio

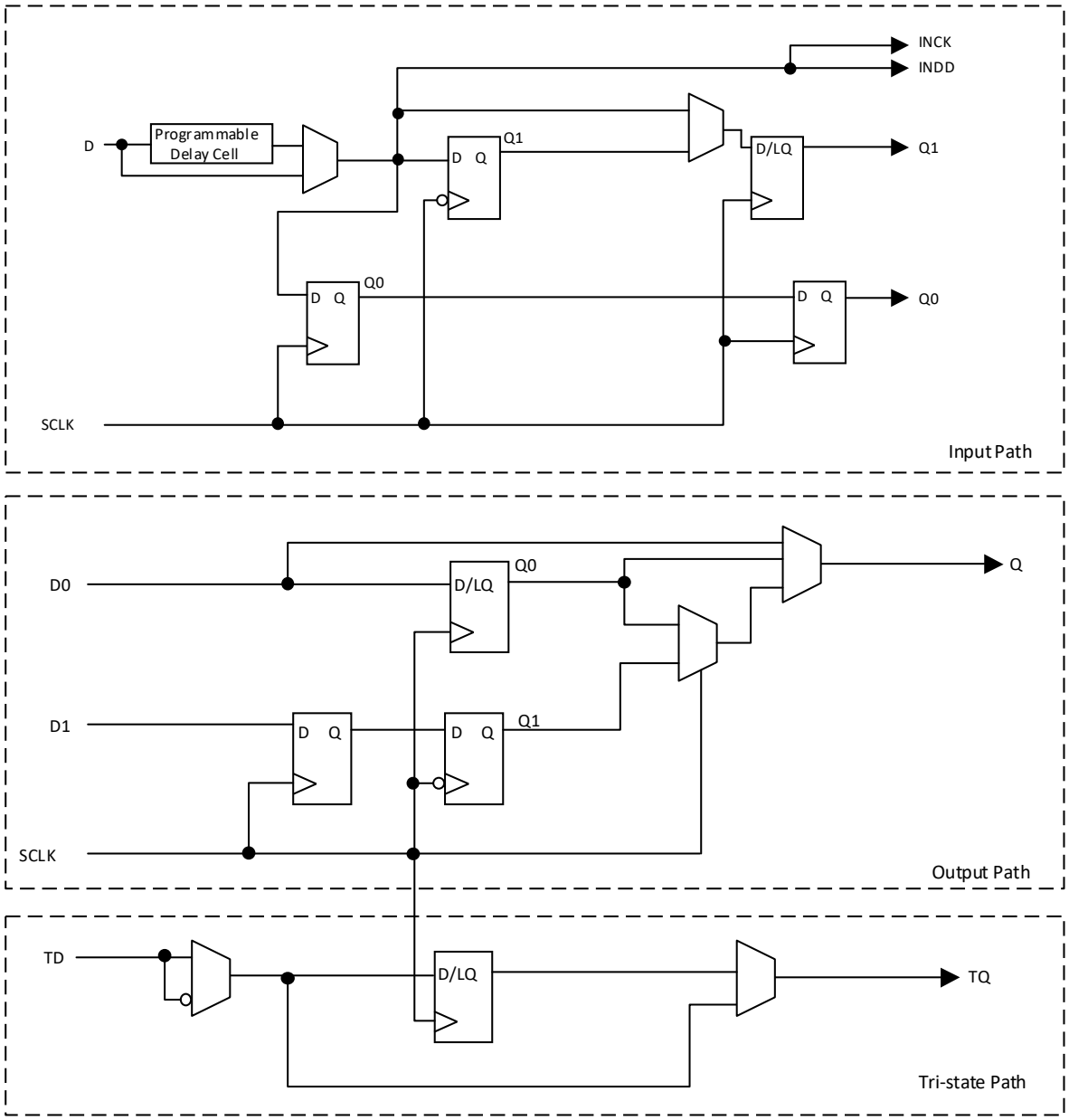
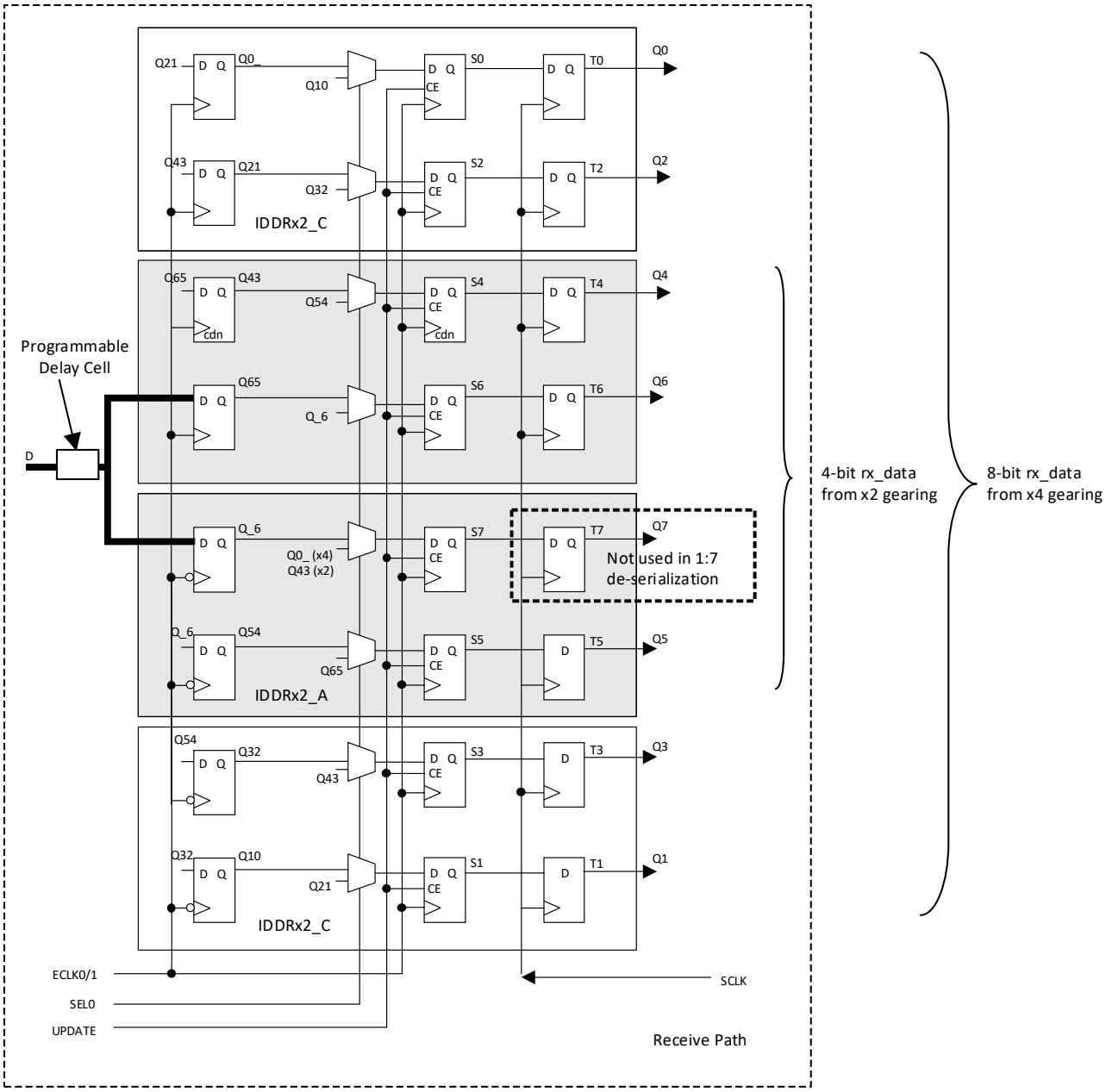


Figure 2.2. Memory PIO Cell to Support DDR Memory Applications



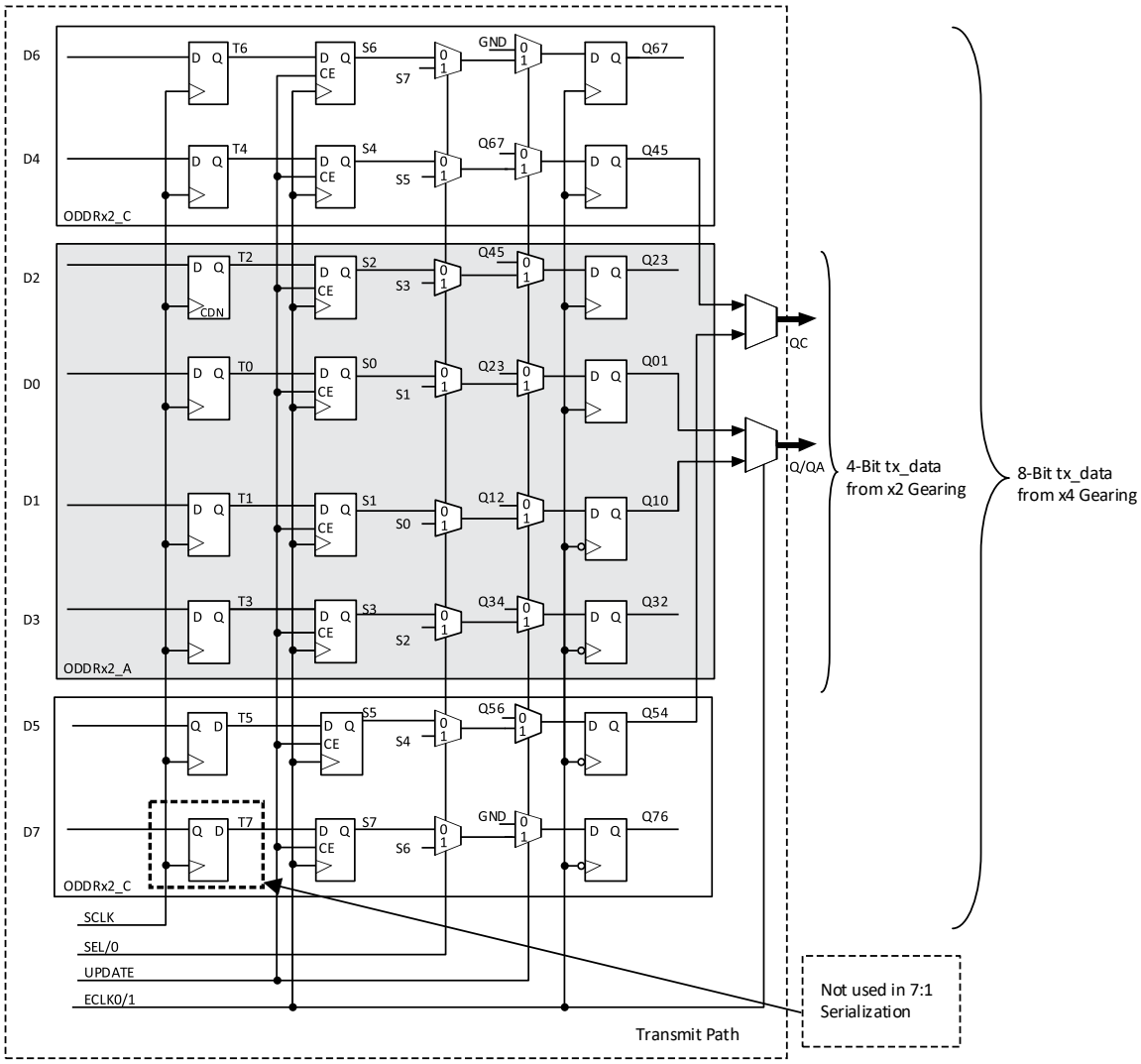


Figure 2.3. Video PIO Cell for x2/x4 and 7:1 Applications

2.3. Clock Domain Transfer at PIO Cells

The MachXO2 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 x2/x4 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 2.3. illustrates the architecture of x2 /x4 input gearing logic.

MachXO2 devices 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 x2/x4 gearings, ALIGNWD must be pulsed eight times to step through maximum eight possible word orders.

The MachXO2 PIO receive de-serializer primitives contains a fundamental logic error for x2 and x4 applications. *Odd* word alignments (SEL = '1') do not present the correct data at the 'Q3' output port (x2), or 'Q7' output port (x4). The timing diagrams below correctly show the actual behavior of the output port, with the actual output data bit highlighted in BOLD. You must be aware of this behavior if instantiating the primitive directly into your design, and take appropriate steps to correct the temporal misalignment. Note the 7:1 gearing primitives are not impacted by this issue.

In Diamond 3.12 and later versions, the temporal misalignment mentioned above is corrected when using the Generic IDDRX4 and IDDRX2 logic interfaces (GDDRX2/X4_RX.ECLK.Centered/Aligned) generated with the Lattice IPexpress tool, and which are described in Generic High-Speed DDR Interfaces. As a result of the fix, the Q bus output is delayed by one SCLK cycle with respect to the primitive output, and with respect to previous versions of the GDDRX2/X4_RX~ interfaces. In most applications, the additional delay is negligible.

Figure 2.4 through Figure 2.8 provide a timing relationship of UPDATE, SEL, ECLK, and SCLK signals under different gearing requirements, Figure 2.9 through Figure 2.11 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. Refer to MachXO2, MachXO3 and ECP5 7:1 LVDS Video Interface (FPGA-RD-02093), for more details on implementing word alignment using the 7:1 gearing function.

The clock domain transfer for the DDR memory interface is accomplished by using the built-in, 90°-shifted clock trees for memory READ and WRITE operations. DLL and DQS detection logic are provided to guarantee the correct receive and transmit of data to and from DDR memories.

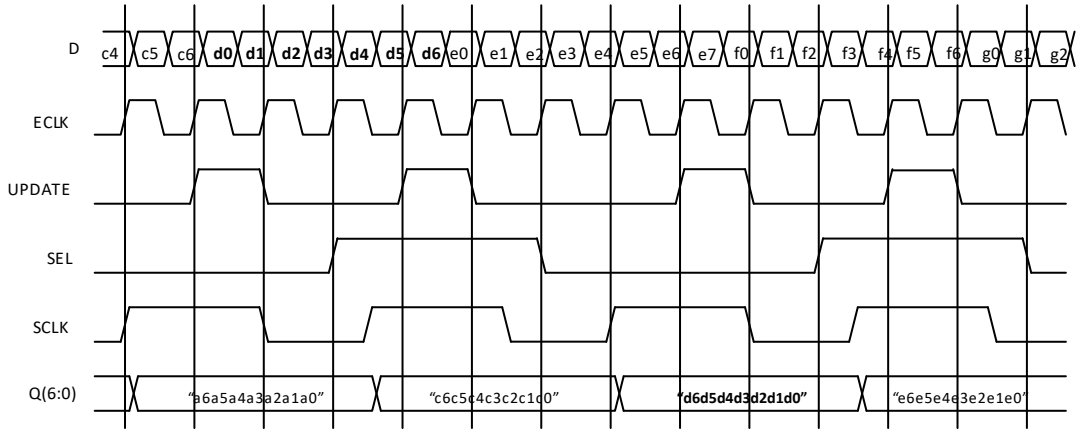


Figure 2.4. 7:1 Deserializer Timing

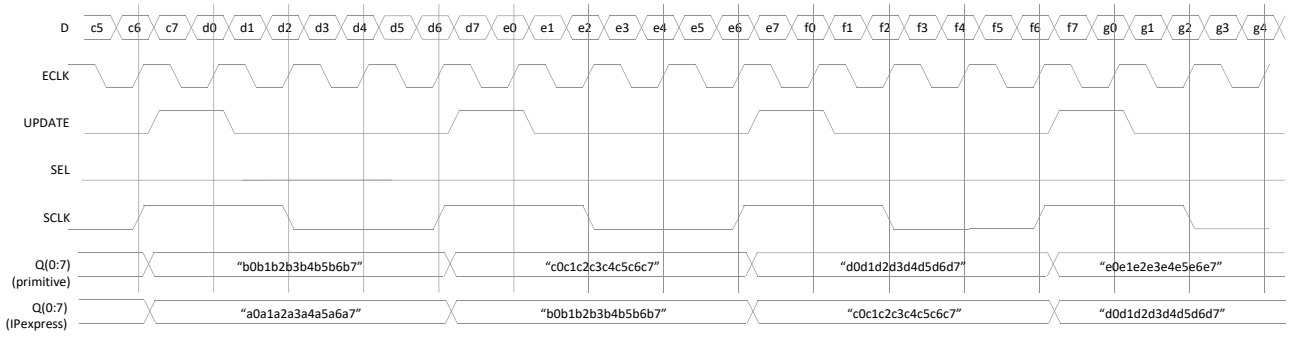


Figure 2.5. x4 Deserializer Timing (Even Phases, SEL=0)

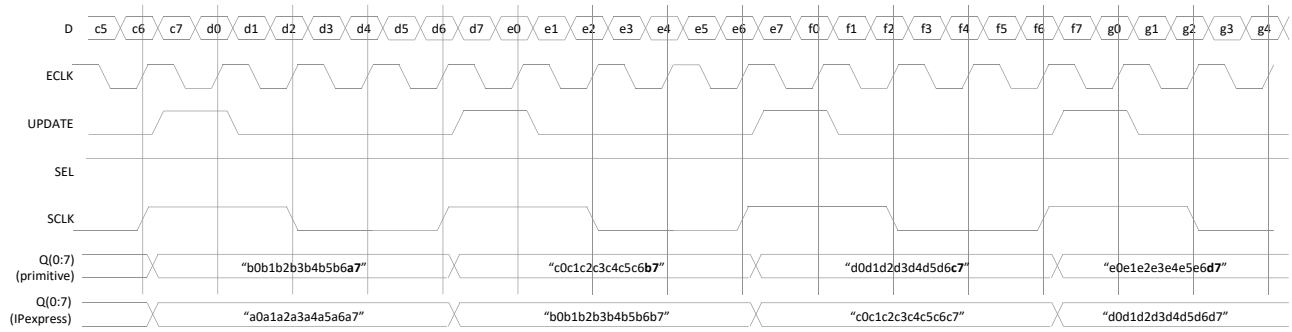


Figure 2.6. x4 Deserializer Timing (Odd Phases, SEL=1)

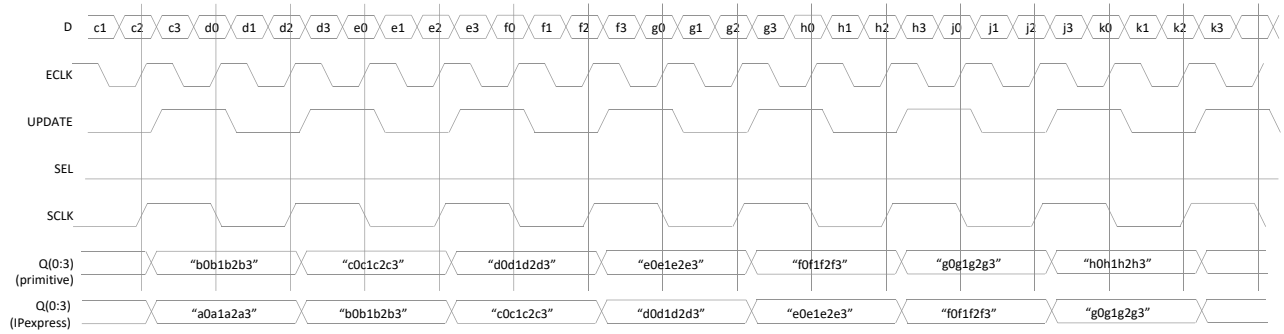


Figure 2.7. x2 Deserializer Timing (Even Phases, SEL=0)

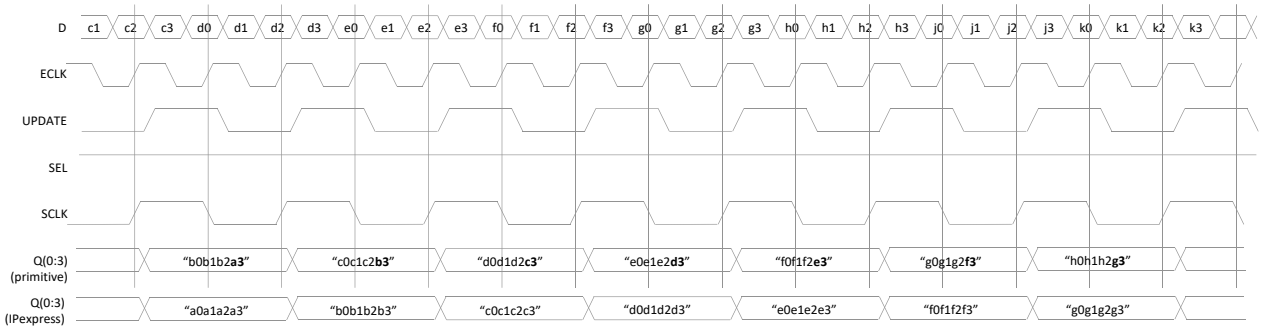


Figure 2.8. x2 Deserializer Timing (Odd Phases, SEL=1)

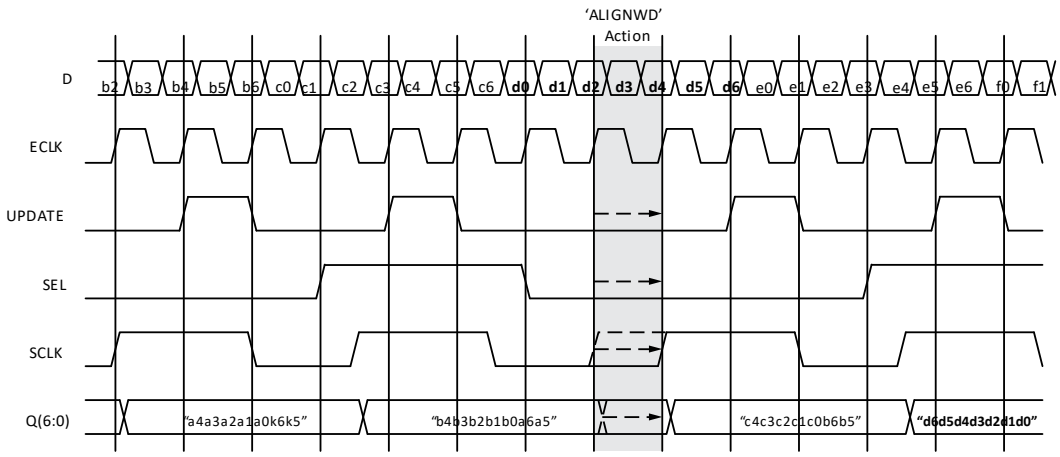


Figure 2.9. 7:1 Deserializer Timing in response to ALIGNWD

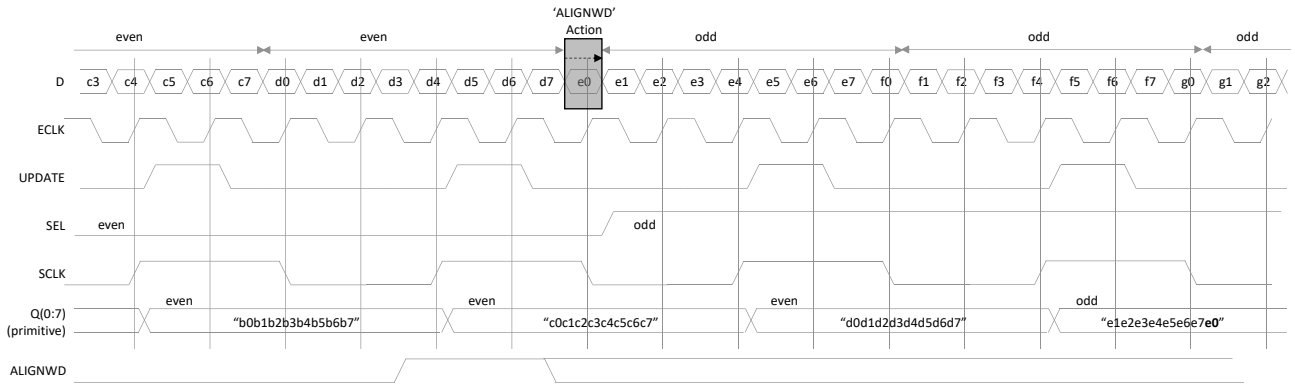


Figure 2.10. x4 Deserializer Timing in Response to ALIGNWD (Even-to-Odd Phase)

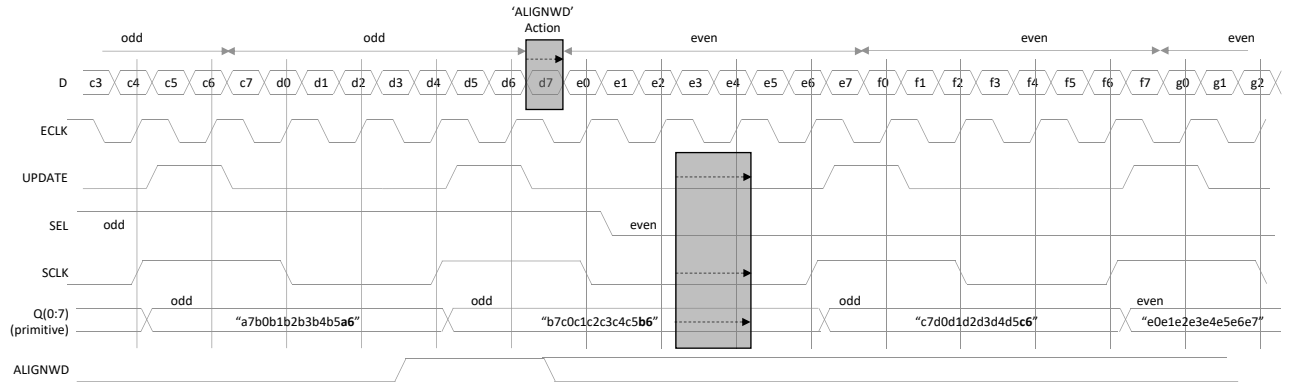


Figure 2.11. x4 Deserializer Timing in Response to ALIGNWD (Odd-to-Even Phase)

3. External High-Speed Interface Description

There are two types of external high-speed interface definitions that can be used with the MachXO2 devices: 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 requires a clock delay adjustment to position the clock at the center of the data opening for transmission.

Note that centered and aligned interfaces might both be used for a given bus. For example, in a DDR SDRAM memory the clock and data relationship during READ is an aligned interface, while the clock and data relationship during WRITE is a centered interface.

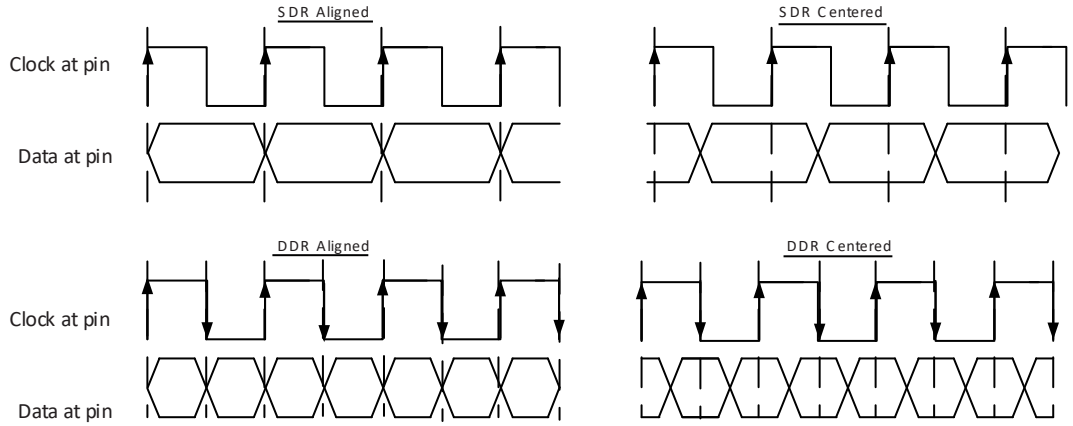


Figure 3.1. External Interface Definition

4. High-Speed Interface Building Blocks

MachXO2 devices provide 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 MachXO2 sysCLOCK PLL Design and Usage Guide (FPGA-TN-02157), 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 on each of the top and bottom sides for MachXO2-640U, MachXO2-1200/U and higher density devices. 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 pins or primary clock nets for high speed interfaces. There are eight primary clock pins (PCLK) available for the MachXO2 device, and eight primary clock nets in the MachXO2 devices.

4.4. CLKDIV

Clock dividers are used to generate low-speed system clocks from a high-speed edge clock. There are two clock dividers per top and bottom sides of the MachXO2-640U, MachXO2-1200/U and higher density devices. 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 MachXO2 devices. The number of PLLs varies with the logic density. The MachXO2-640U, MachXO2-1200/U and MachXO2-2000 have one PLL. MachXO2-2000U, MachXO2-4000, and MachXO2-7000 devices have two PLLs. 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 in the MachXO2-640U, MachXO2-1200/U and higher density devices. The top-right DQSDLL controls the top and right banks. The bottom-left DQSDLL can be used by the bottom and left banks. This component can be used for both DDR memory and generic high-speed interfaces. 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 x1, x2, x4, and 7:1 gearing ratios at the receiving side of the PIO cells. The x1 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 x1 gearing. The x2 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 x4 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 x1, x2, x4, and 7:1 gearing ratios at the transmit side of the PIO cells. The x1 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 x1 gearing. The x2 gearing is supported by ODDRX2. The 4-bit parallel data is clocked by SCLK and is serialized using ECLK. The x4 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 x4 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 the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices 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) at the bottom-side 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 x1 gearing, or match the edge clock injection for x2/x4 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 x1 aligned interface or ECLK for x2/x4 aligned interfaces.

4.10. DQSBUF

This is the dedicated DQS circuitry for DDR memory. It generates a 90° shift on DQS for memory read operations, and a 90° shift on the SCLK for memory write operations. The clock polarity detection, burst detection, and data valid signals are generated from this block. This is available on the right bank of the MachXO2-640U, MachXO2- 1200/U and higher density devices.

4.11. IDDRDQS

DDR memory input buffer supports clock domain transfers from DQS to SCLK. These memory PIO cells are used exclusively for DDR memory interfaces and are used for DDR memory READ operations. They are available on the right bank of the MachXO2-640U, MachXO2-1200/U and higher density devices.

4.12. ODDRDQS

The DDR memory output buffer supports clock domain transfers from SCLK to DQS. These memory PIO cells are used exclusively for DDR memory interfaces and are used for DDR memory WRITE operations. They are available on the right bank of the MachXO2-1200 and higher density devices.

5. Generic High-Speed DDR Interfaces

Generic high-speed interfaces, or Generic DDR (GDDR), are supported in MachXO2 using the dedicated logic blocks. This section discusses 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 MachXO2 device family 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

Mode	Interface Name	Description	Supporting Device & Sides
RX SDR	GIREG_RX.SCLK	SDR input using SCLK	All devices, all sides
RX GDDRx1 Aligned	GDDRX1_RX.SCLK.Aligned	DDRx1 input using SCLK, data is edge-to-edge with incoming clock	640U, 1200/U, and above, all sides
RX GDDRx1 Centered	GDDRX1_RX.SCLK.Centered	DDRx1 input using SCLK, incoming clock is centered at the data opening	All devices, all sides
RX GDDRx2 Aligned	GDDRX2_RX.ECLK.Aligned	DDRx2 input using ECLK, data is edge-to-edge with incoming clock	640U, 1200/U, and above, bottom
RX GDDRx2 Centered	GDDRX2_RX.ECLK.Centered	DDRx2 input using ECLK, incoming clock is centered at the data opening	640U, 1200/U, and above, bottom
RX GDDRx4 Aligned	GDDRX4_RX.ECLK.Aligned	DDRx4 input using ECLK, data is edge-to-edge with incoming clock	640U, 1200/U, and above, bottom
RX GDDRx4 Centered	GDDRX4_RX.ECLK.Centered	DDRx4 input using ECLK, incoming clock is centered at the data opening	640U, 1200/U, and above, bottom
RX GDDR71	GDDR71_RX.ECLK.7:1	GDDR 7:1 input using ECLK	640U, 1200/U, and above, bottom
TX SDR	GOREG_TX.SCLK	SDR output using SCLK	All devices, all sides
TX GDDRx1 Aligned	GDDRX1_TX.SCLK.Aligned	DDRx1 output using SCLK, data is edge-to-edge with outgoing clock	All devices, all sides
TX GDDRx1 Centered	GDDRX1_TX.SCLK.Centered	DDRx1 output using SCLK, outgoing clock is centered at the data opening	640U, 1200/U, and above, all sides
TX GDDRx2 Aligned	GDDRX2_TX.ECLK.Aligned	DDRx2 output using ECLK, data is edge-to-edge with outgoing clock	640U, 1200/U, and above, top
TX GDDRx2 Centered	GDDRX2_TX.ECLK.Centered	DDRx2 output using ECLK, outgoing clock is centered at the data opening	640U, 1200/U, and above, top
TX GDDRx4 Aligned	GDDRX4_TX.ECLK.Aligned	DDRx4 output using ECLK, data is edge-to-edge with outgoing clock	640U, 1200/U, and above, top
TX GDDRx4 Centered	GDDRX4_TX.ECLK.Centered	DDRx4 output using ECLK, outgoing clock is centered at the data opening	640U, 1200/U, and above, top
TX GDDR71	GDDR71_TX.ECLK.7:1	GDDR 7:1 output using ECLK	640U, 1200/U, and above, top

Notes:

- For the “R1” version of the MachXO2 devices GDDRX2, GDDRX4 and GDDR71 modes, ECLKSYNC may have a glitch in the output under certain conditions, leading to possible loss of synchronization. The “R1” versions of the MachXO2 devices have an “R1” suffix at the end of the part number (for example, LCMXO2-1200ZE-1TG144CR1). For more details on the R1 to Standard migration refer to Designing for Migration from MachXO2-1200-R1 to Standard (Non-R1) Devices (FPGA-AN-02012).
- MIPI D-PHY Receive can be built using GDDRX4_RX.ECLK.Centered interface and 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 x1 I/O register
- DDRX2 – DDR x2 I/O register
- DDRX4 – DDR x4 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 MachXO2 clocking structure, refer to MachXO2 sysCLOCK PLL Design and Usage Guide (FPGA-TN-02157). As listed in Table 5.1, each interface is supported in specific bank locations of the MachXO2 devices. 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.

5.3. Receive Interfaces

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

5.3.1. 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.

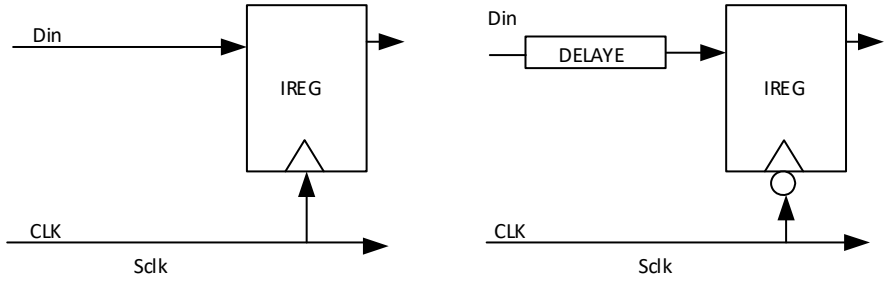


Figure 5.1. GIREG_RX Interface

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 rules:

- Must use a dedicated clock pin PCLK as the clock source

5.3.2. 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 x1 interface.

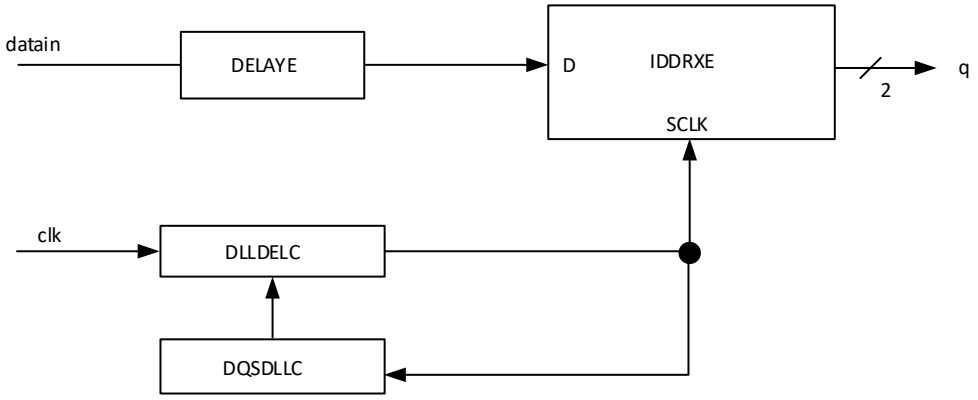


Figure 5.2. GDDRX1_RX.SCLK.Aligned Interface Using DQSDLL

Interface rules:

- Must use a dedicated clock pin PCLK 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.

5.3.3. GDDRX1_RX.SCLK.Centered

This DDR interface uses DELAYE to match the SCLK delay at the IDDRXE. DELAYD is not available for the x1 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.

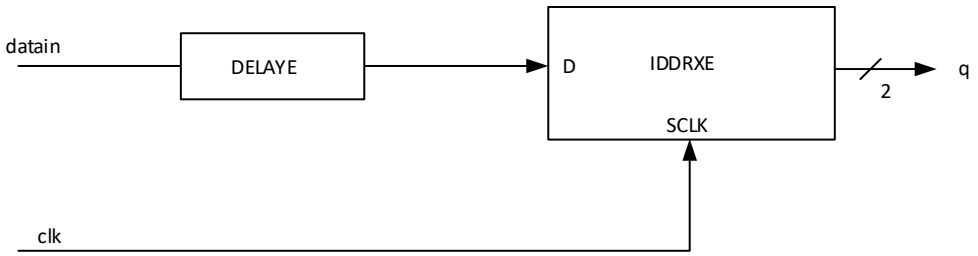


Figure 5.3. GDDRX1_RX.SCLK.Centered

Interface rules:

- Must use a dedicated clock pin PCLK 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

5.3.4. GDDRX2_RX.ECLK.Aligned

This DDR x2 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 x2 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.

The *Temporal Alignment* block (generated in Diamond versions 3.12 and later) introduces a one SCLK period delay and corrects for the temporal misalignment of IDDRX2E 'Q3' data output.

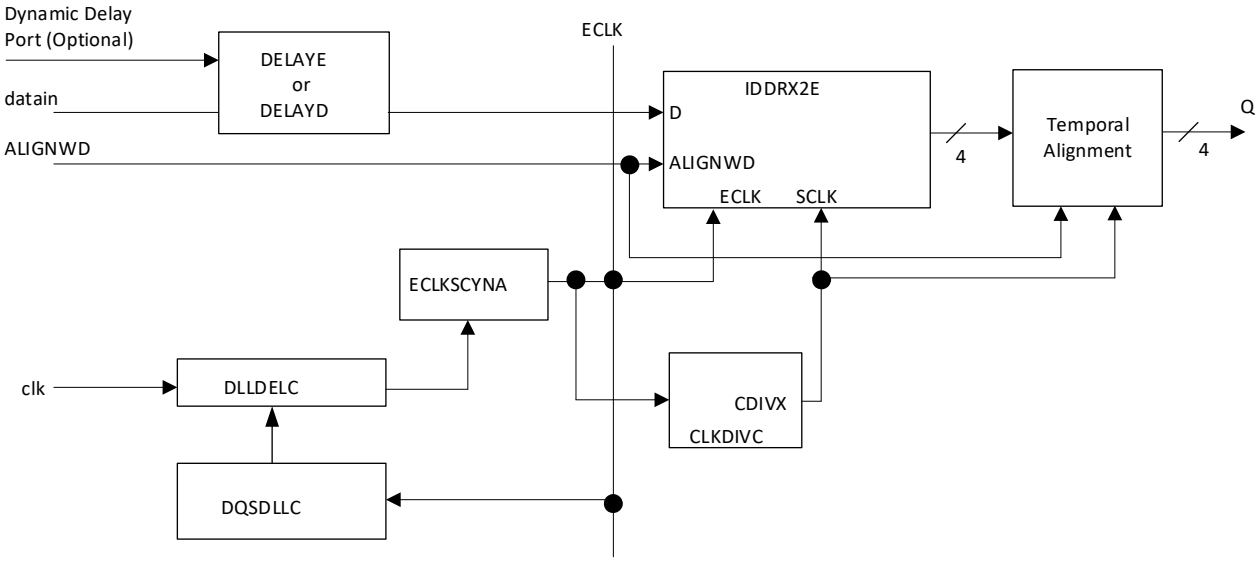


Figure 5.4. GDDRX2_RX.ECLK.Aligned Interface

Interface rules:

- Must use a dedicated clock pin PCLK as the clock source for DLLDELC
- Clock net routed to SCLK must use primary clock net
- There are up to two DQSDLLC 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 at the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices

5.3.5. GDDRX2_RX.ECLK.Centered

This DDR x2 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 x2 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.

The *Temporal Alignment* block (generated in Diamond versions 3.12 and later) introduces a one SCLK period delay and corrects for the temporal misalignment of IDDRX2E 'Q3' data output.

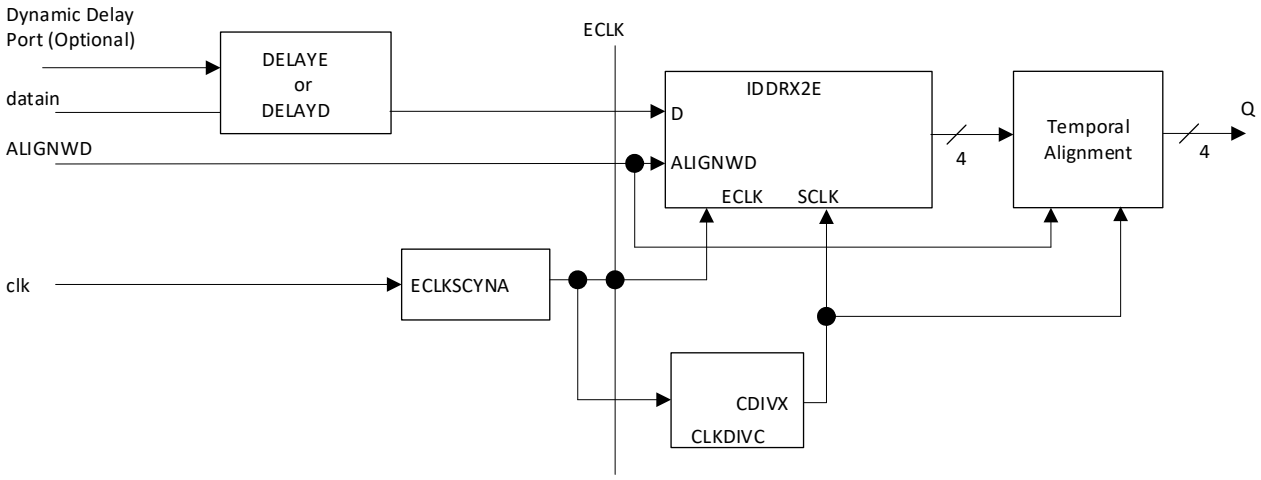


Figure 5.5. GDDRX2_RX.ECLK.Centered Interface

Interface rules:

- Must use a dedicated clock pin PCLK 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 at the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices

5.3.6. GDDRX4_RX.ECLK.Aligned

This DDR x4 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.

The *Temporal Alignment* block (generated in Diamond versions 3.12 and later) introduces a one SCLK period delay and corrects for the temporal misalignment of IDDRX4B 'Q7' data output.

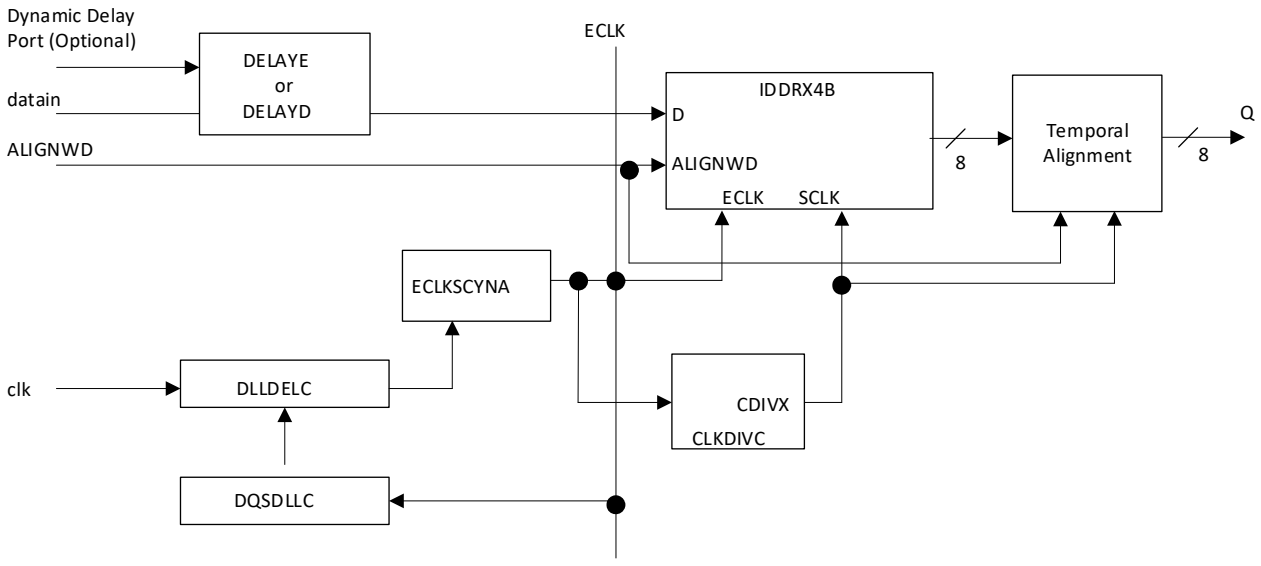


Figure 5.6. GDDRX4_RX.ECLK.Aligned Interface

Interface rules:

- Must use a dedicated clock pin PCLK as the clock source for DLLDELC
- Clock net routed to SCLK must use primary clock net
- There are up to two DQSDLLC 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 at the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices

5.3.7. 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, which is one quarter of the ECLK frequency. The port ALIGNWD can be used for word alignment at the interface.

The *Temporal Alignment* block (generated in Diamond versions 3.12 and later) introduces a one SCLK period delay and corrects for the temporal misalignment of IDDRX4B 'Q7' data output.

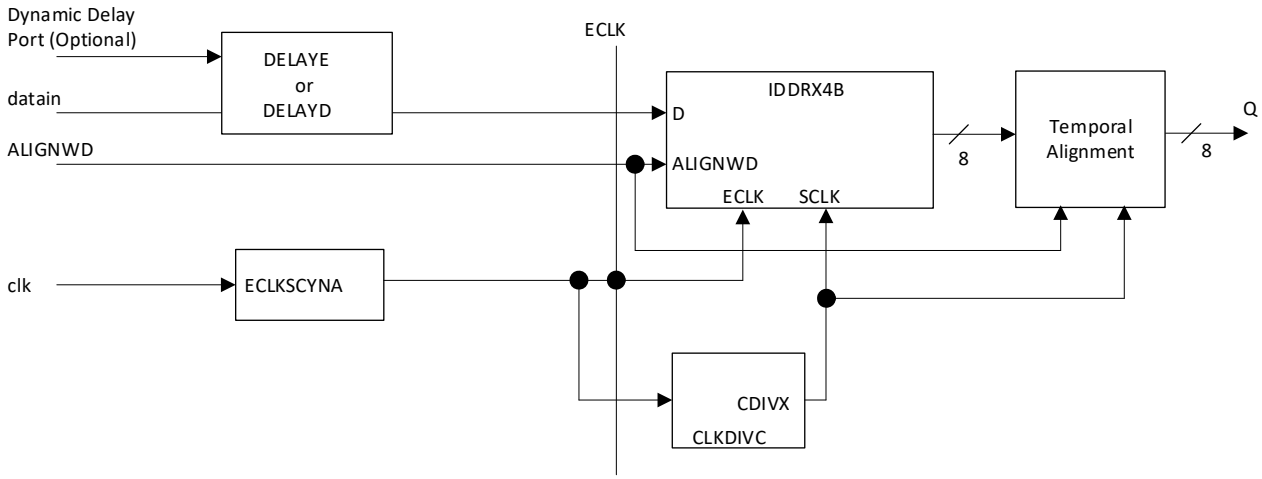


Figure 5.7. GDDRX4_RX.ECLK.Centered Interface

Interface rules:

- Must use a dedicated clock pin PCLK 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 x4 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 at the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices

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.

5.3.8. GDDR71_RX.ECLK.7:1

The GDDR 7:1 receive interface is unique among the supported high-speed DDR interfaces. It uses the PLL tosearch 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.

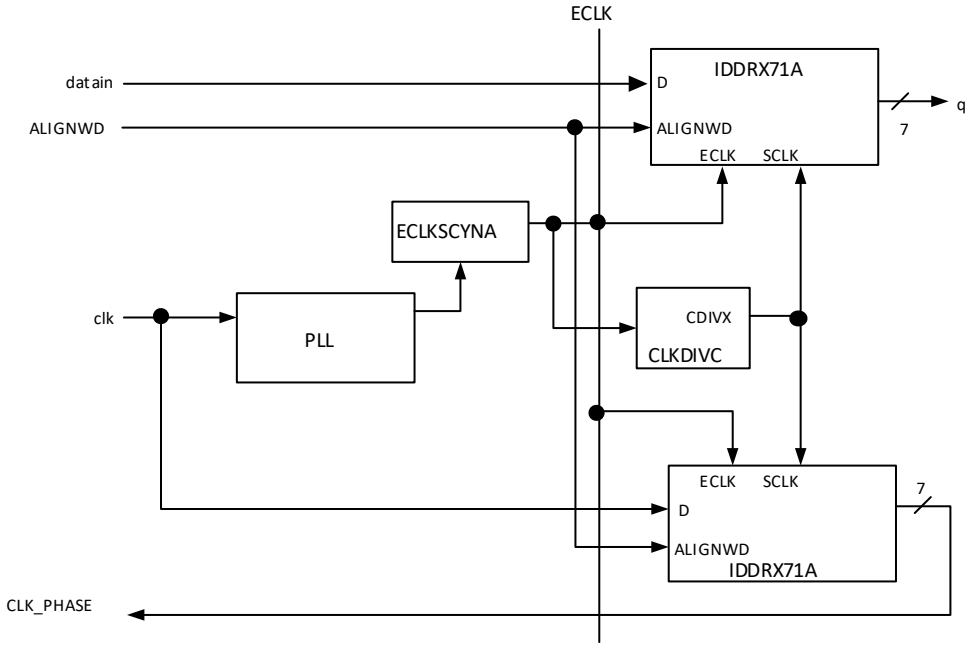


Figure 5.8. GDDR71_RX.ECLK.7:1 Interface

Interface rules:

- Must use a dedicated clock pin PCLK at the bottom side 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 at the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices

5.4. Transmit Interfaces

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

5.4.1. 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.

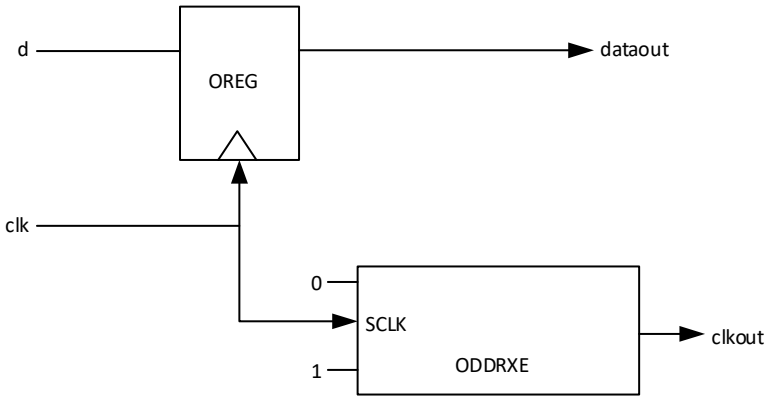


Figure 5.9. GOREG_TX.SCLK Interface

Interface rules:

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

5.4.2. 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.

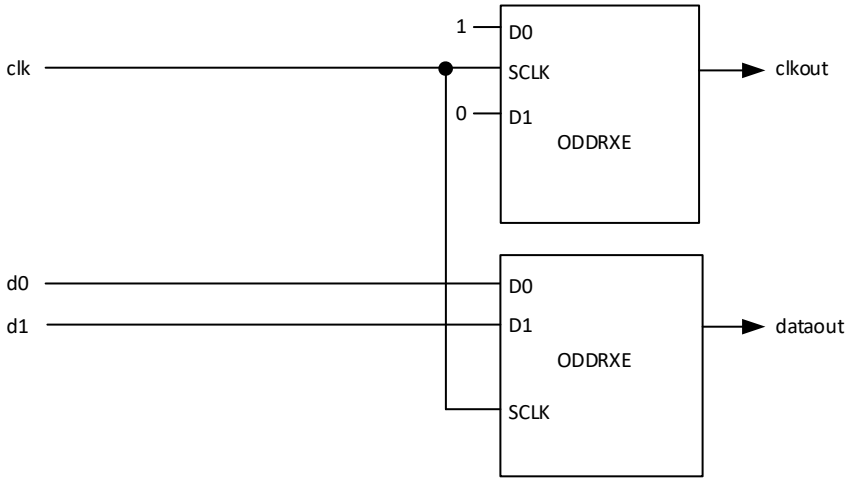


Figure 5.10. GDDRX1_TX.SCLK.Aligned Interface

Interface rules:

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

5.4.3. 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.

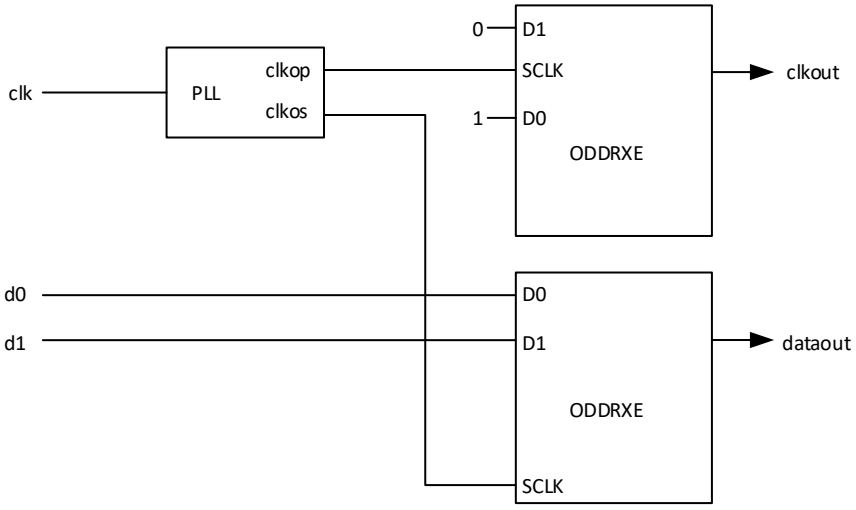


Figure 5.11. GDDRX1_TX.SCLK.Centered Interface

Interface rules:

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

5.4.4. GDDRX2_TX.ECLK.Aligned

This output DDR x2 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.

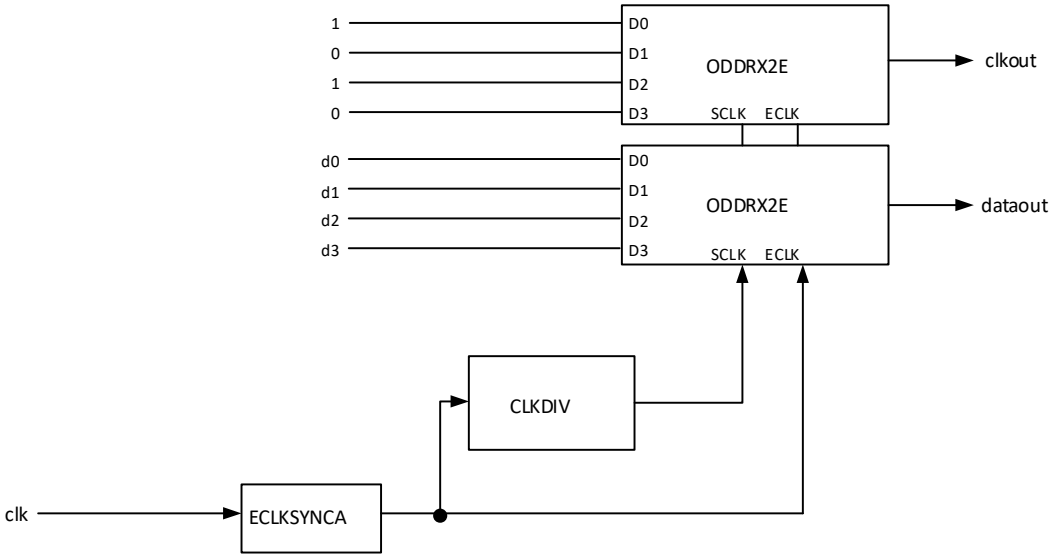


Figure 5.12. GDDRX2_TX.ECLK.Aligned Interface

Interface rules:

- Must use edge clock routing resources for the ECLK
- The routing of SCLK must use primary clock net
- This interface is supported at the top side of the MachXO2-640U, MachXO2-1200/U and higher density devices

5.4.5. GDDRX2_TX.ECLK.Centered

This output DDR x2 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.

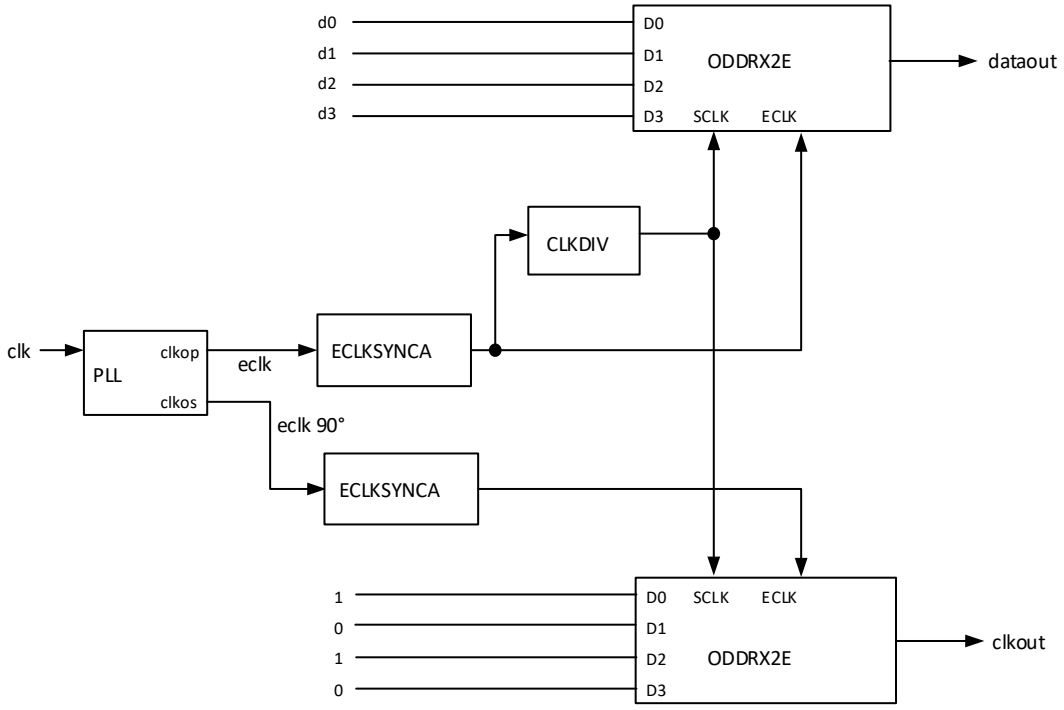


Figure 5.13. GDDRX2_TX.ECLK.Centered Interface

Interface rules:

- Must use edge clock routing resources 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 at the top side of the MachXO2-640U, MachXO2-1200/U and higher density devices

5.4.6. 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.

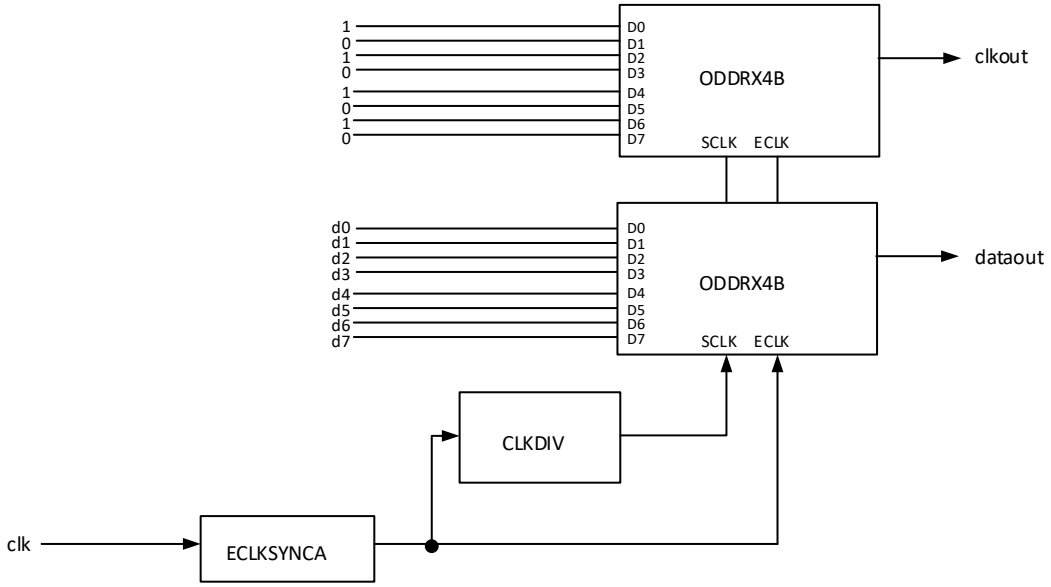


Figure 5.14. GDDRX4_TX.ECLK.Aligned Interface

Interface rules:

- Must use edge clock routing resources 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 x4 gearing
- This interface is supported at the top side of the MachXO2-640U, MachXO2-1200/U and higher density devices

5.4.7. 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.

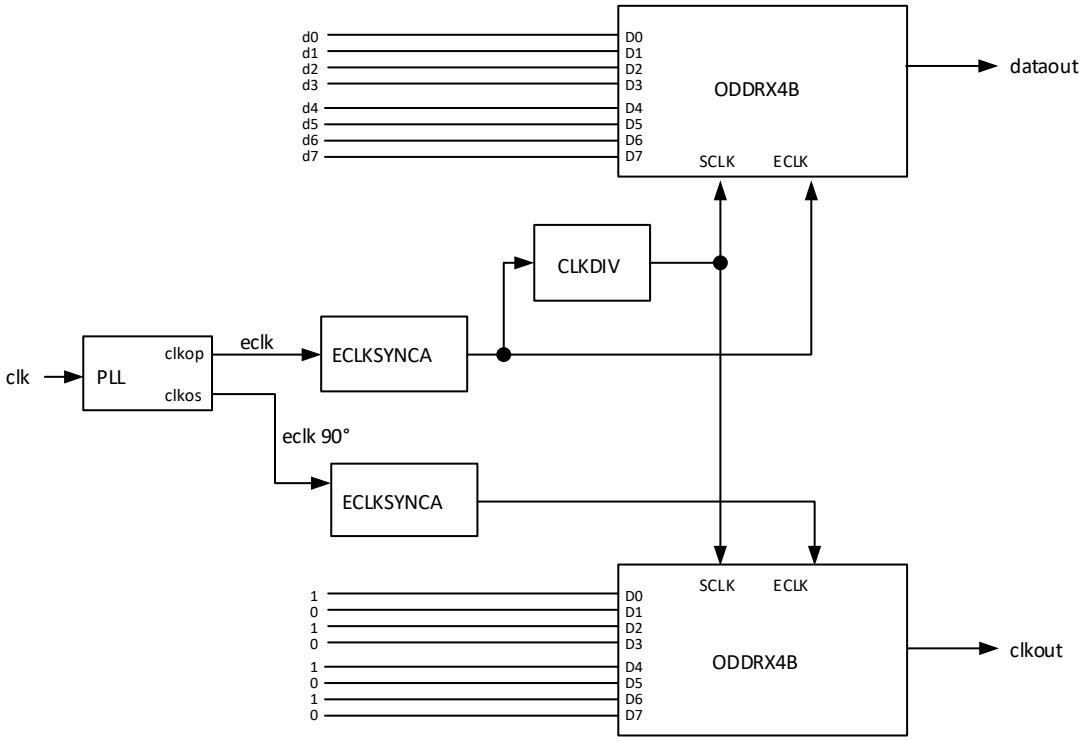


Figure 5.15. GDDRX4_TX.ECLK.Centered Interface

Interface rules:

- Must use edge clock routing resources 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 x4 gearing
- This interface is supported at the top side of the MachXO2-640U, MachXO2-1200/U and higher density devices

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

5.4.8. 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 7 times slower than the transmit data rate. ECLKSYNCA element is associated with the ECLK and must be used to drive the ECLK.

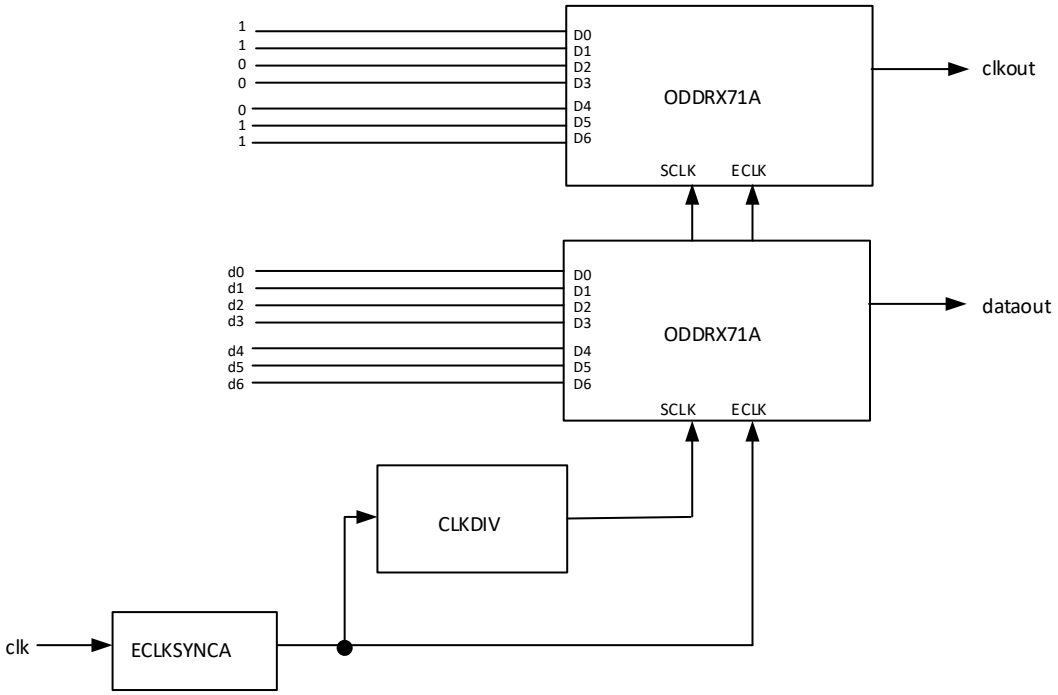


Figure 5.16. GDDR71_TX.ECLK.7:1 Interface

Interface rules:

- Must use edge clock routing resources 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 at the top side of the MachXO2-640U, MachXO2-1200/U and higher density devices

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 generates 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 > I/O. 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 input 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. (The DQSDEL output of DQSDLLC is updated for any PVT variation when UDDCNTLN is held low.)	x1, x2, x4 Aligned
freeze	Input	Freeze or release DLL, active high. (When high, the device freezes the DLL to save power while the delay code is preserved. When low, it releases the DLL to resume operation.)	x1, x2, x4 Aligned
alignwd	Input	Word alignment control signal, active high. (ALIGNWD can be asynchronous to the ECLK domain, but it must be at least two ECLK cycles wide.)	x2, x4, 7:1
dqsdll_reset	Input	Asynchronous DQSDLL reset, active high	x2, x4 Aligned
clk_s[1]	Input	Slow clock for reset synchronization. (This clock must be slower than ECLK.)	x2, x4, 7:1
init	Input	Initialize reset synchronization, active high	x2, x4, 7:1
phase_dir	Input	Dynamic Phase adjustment direction. (When 0, phase shift is delayed by one step. When 1, phase shift is advanced by one step.)	7:1
phase_step	Input	Dynamic Phase Adjustment Step. (Each step generates a 45-degree shift. PHASESTEP is active low.)	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	x2, x4, 7:1
eclk	Output	Edge clock generated from the input clock	x2, x4 Aligned
rx_ready	Output	Indicate completion of reset synchronization	x2, x4, 7:1
clk_phase	Output	7-bit representation of input clock phase. (The clk_phase is used in the FPGA to build the bit alignment logic.)	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[1]	oInput	Slow clock for reset synchronization. (This clock must be slower than eclk.)	x2, x4, 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	x2, x4, 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.

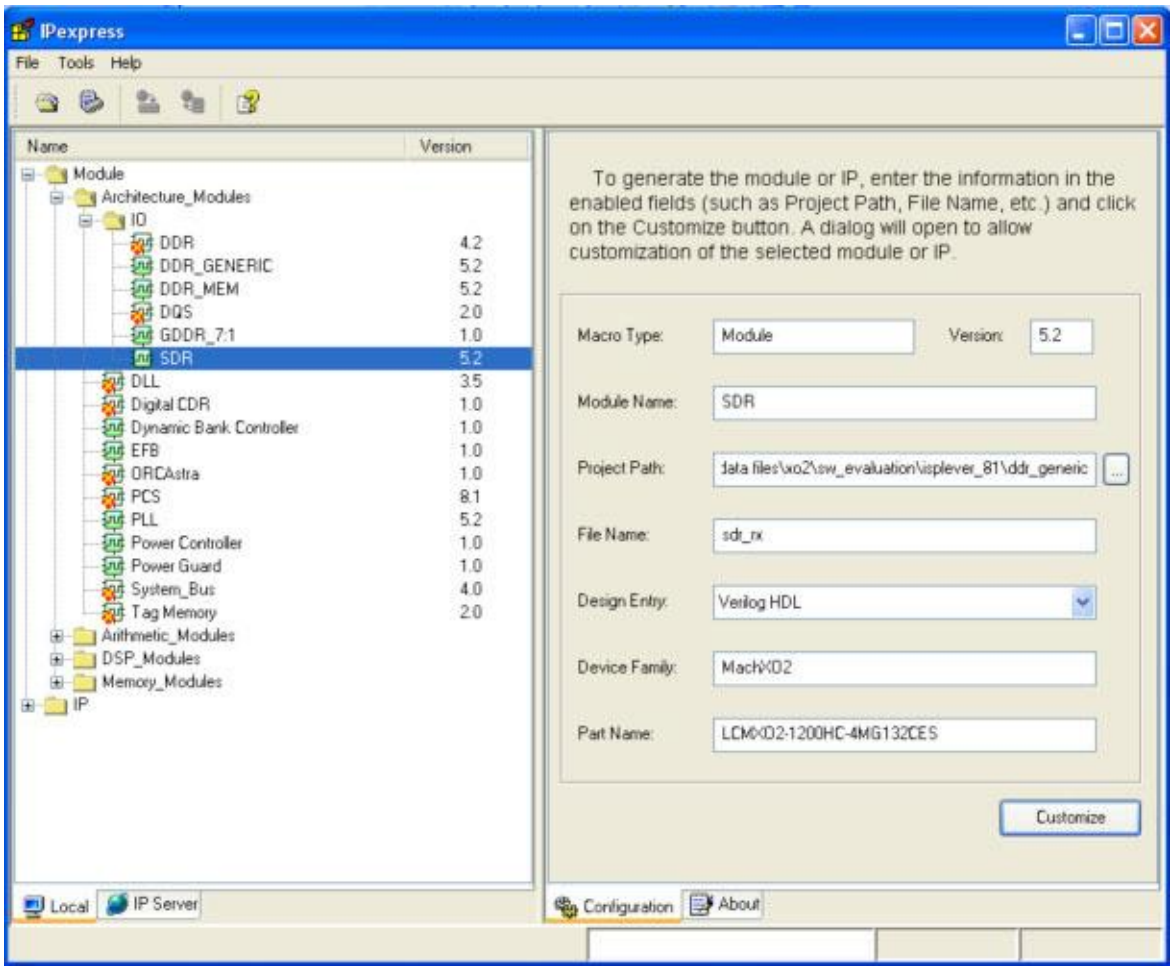


Figure 6.1. SDR Interface Selection at the IPexpress Main Window

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.

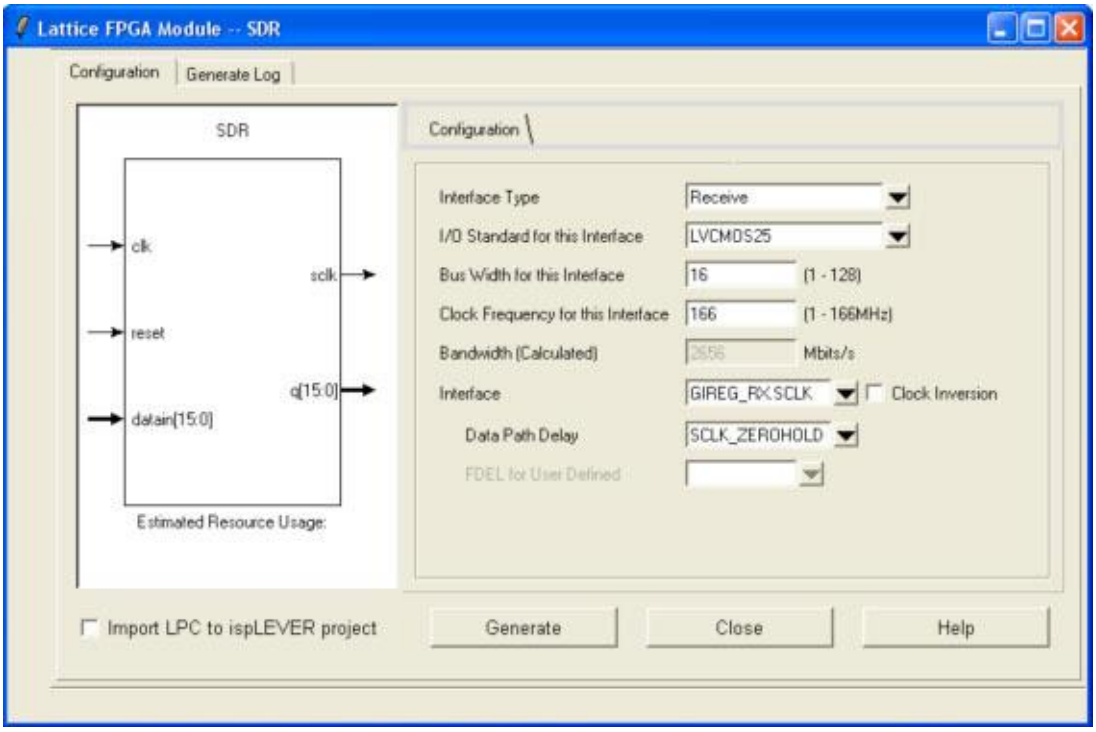


Figure 6.2. Configuration Tab for the SDR Interfaces

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 runs	1-166MHZ	166MHz
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.

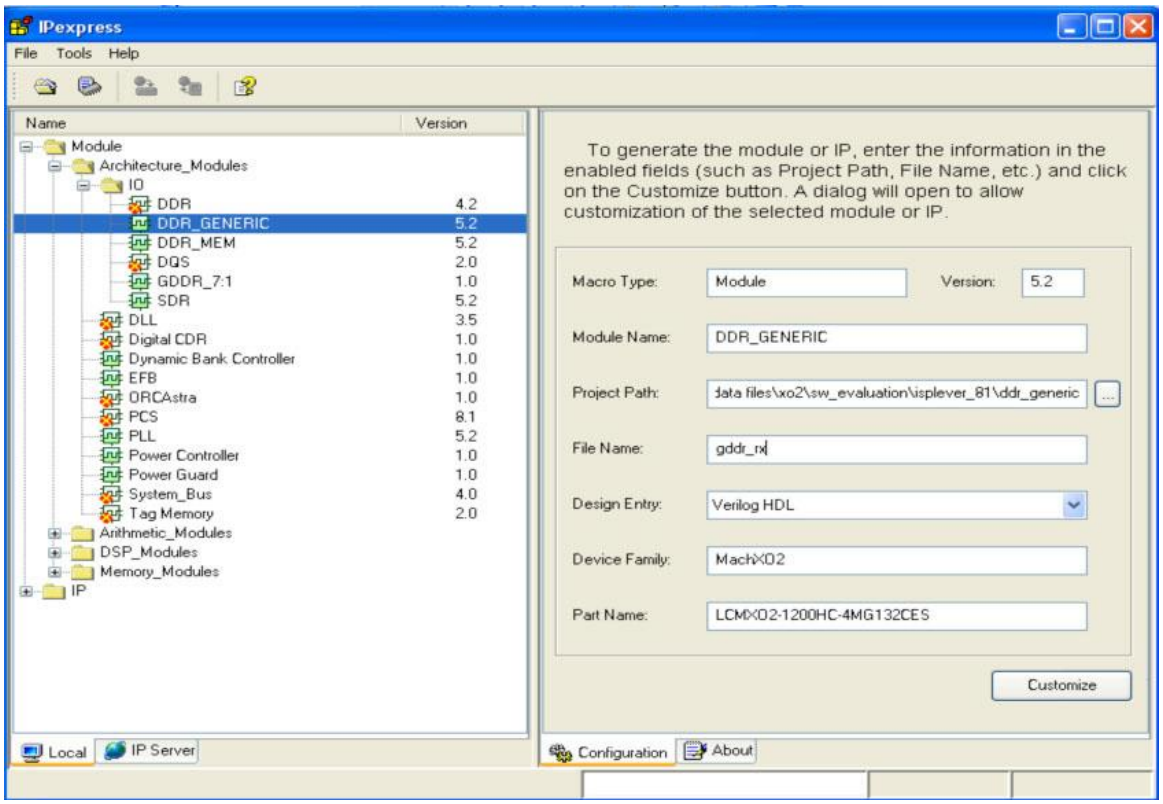


Figure 6.3. DDR_Generic Interface Selection at the IPexpress Main Window

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.

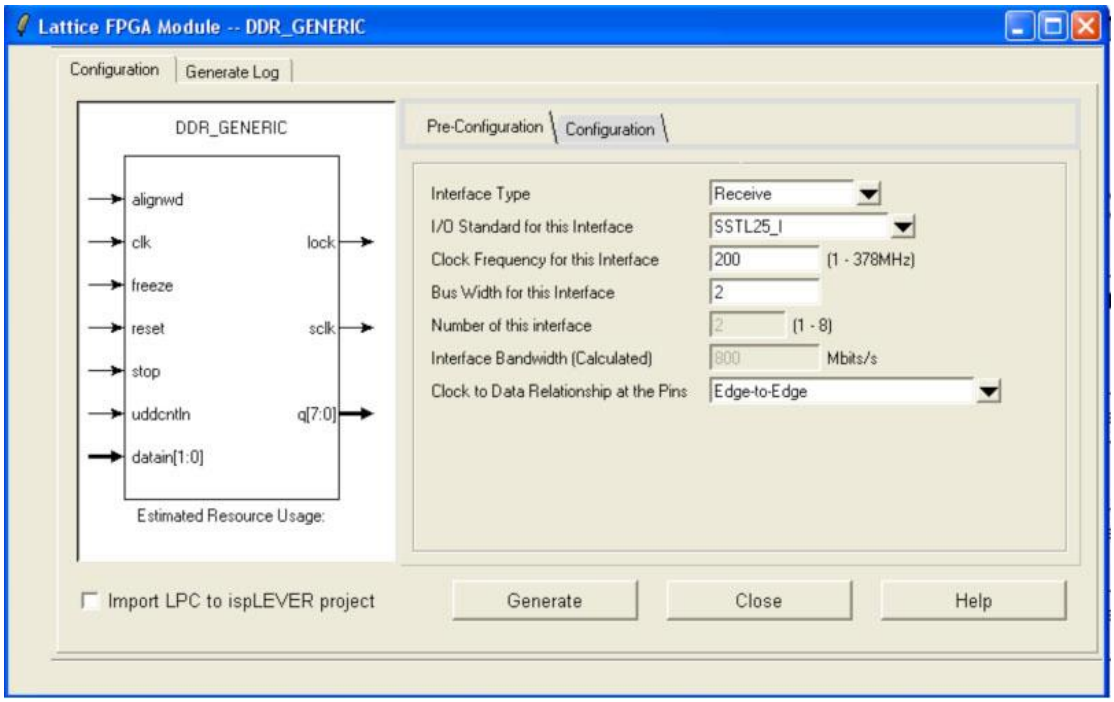


Figure 6.4. Pre-Configuration Tab of the DDR_Generic Interfaces

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

User Interface Option	Description	Range	Default Value
Interface Type	Types of interface	Transmit, Receive	Note 1
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 runs	1-378MHz (for HP) 1-210 MHz (for LP)	Note 1
Bus Width for this Interface	Bus size for the interface	Various depending on the interface selected	Note 1
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 1

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

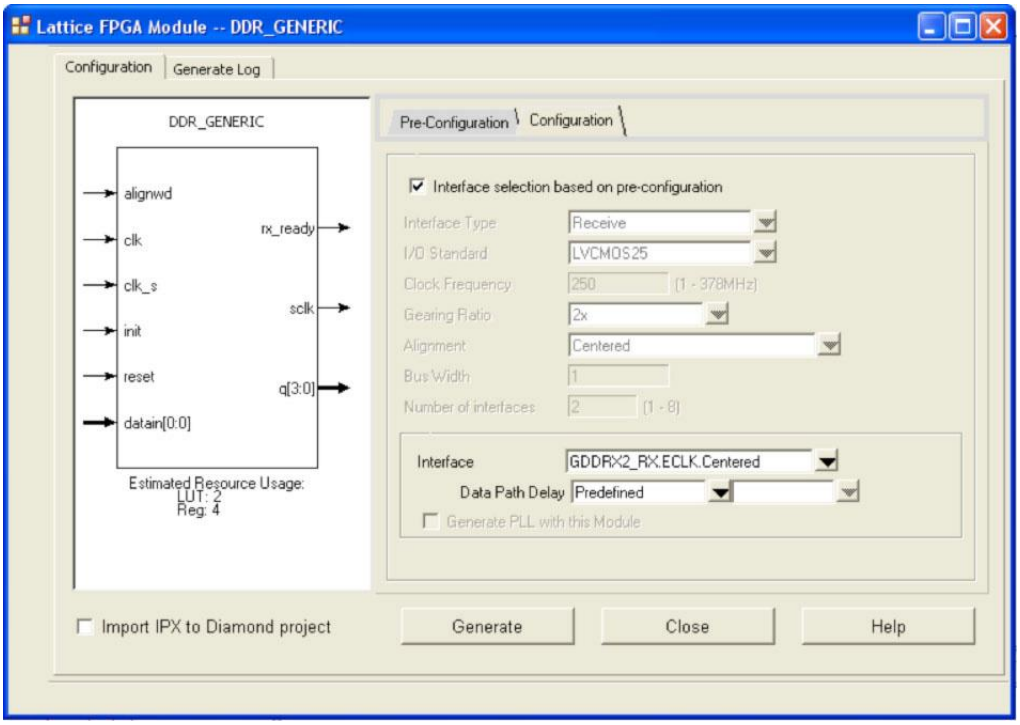


Figure 6.5. Configuration Tab of the DDR_Generic Modules

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 of 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, and so on.	ENABLED, DISABLED	ENABLED
Interface Type	Types of interfaces	Transmit, Receive	Receive
I/O Standard	I/O standard for this interface	All I/O types per selected Interface Type	LVCMOS25
Clock Frequency	Speed at which the interface runs	1-378 MHz (for HP) 1-210 MHz (for LP)	200 MHz 100 MHz
Gearing Ratio	Choose the gearing ratio of the interface	x1, x2, x4	x1
Alignment	Determine the type of external interfaces	Edge-to-Edge, Centered	Edge-to-Edge
Bus Width	Bus size for the interface	1-128	4
Number of Interfaces	Maximum number of buses supported	1-8	calculated
Interface	A list of the supported GDDR interfaces	Dependent of the Gearing ratio and Alignment choice	GDDRX1_RX.SCLK.Aligned
Data Path Delay[1]	Data input can be optionally delayed using the DELAY block.	Bypass, Predefined, User-defined, Dynamic	Predefined
Generate PLL with this Module[2]	Option to generate PLL with this module or not to generate PLL with this module.	Enabled, Disabled	Enabled

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 x2 and x4 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

Device Type	Speed of the Interface	Gearing Ratio
High Performance (HP) devices	=< 166 MHz	x1
	> 166 MHz and =< 266 MHz	x2
	> 266 MHz	x4
Low Power (LP) devices	=< 70 MHz	x1
	>70 MHz and =< 133 MHz	x2
	>133 MHz	x4

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 user interface options are listed in this section. The DDR 7:1 interface is a very specific application so the user interface options are relatively simple. Most of the necessary logic is built into the software for ease of use.

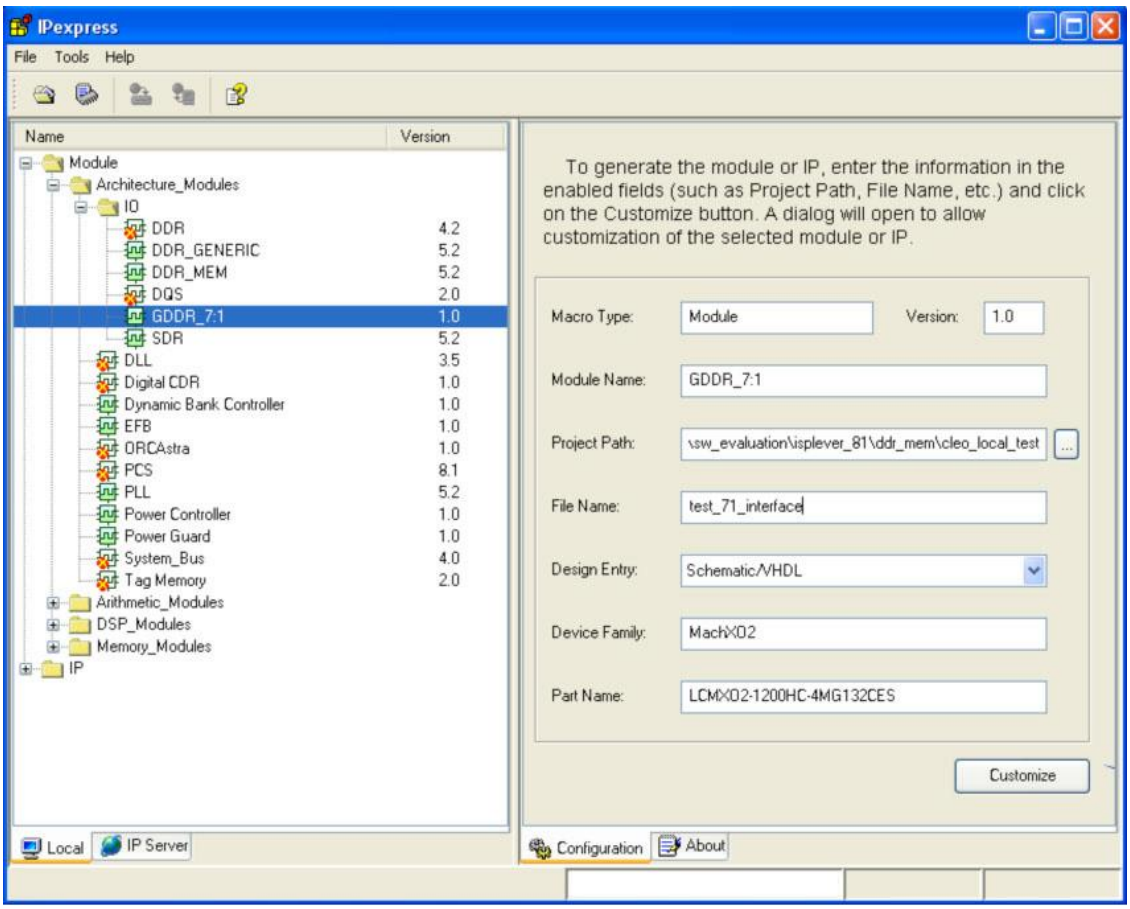


Figure 6.6. GDDR_71 Interface Selection at the IPexpress Main Window

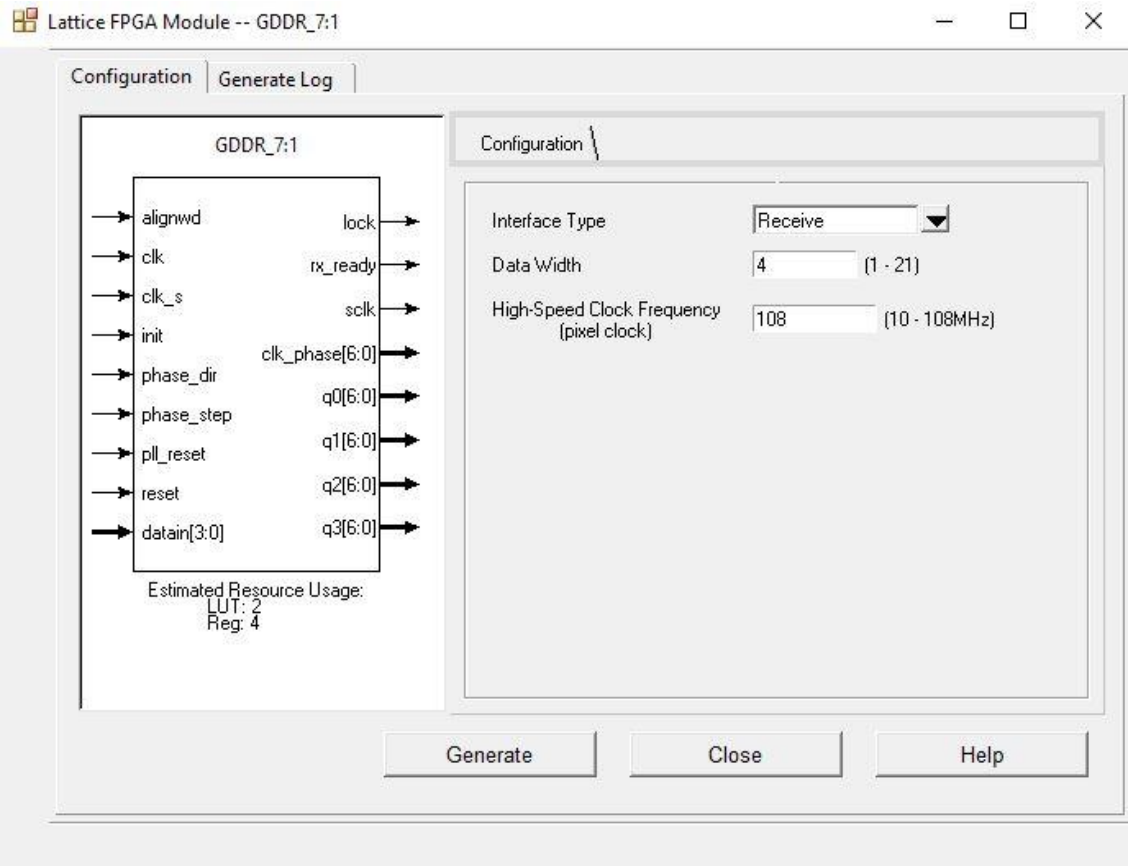


Figure 6.7. GDDR_71 Configuration Tab of GDDR_71

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	1-16	4
High Speed Clock Frequency (Pixel Clock)	Pixel clock frequency for 7:1 LVDS interface	10-108	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 (PIO), which form two pairs of I/O buffers. Each PIO by itself can support a x1 gearing ratio. A pair of PIO, either the A/B pair or C/D pair, can support a x2 gearing ratio. Support of a x4 or 7:1 gearing ratio takes up all four PIO in one PIC block. The x4 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 x2 interfaces available in a specific package is determined by the total number of A/B and C/D pairs. The total number of x4/7:1 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

	I/O Logic A	I/O Logic B	I/O Logic C	I/O Logic D
x2 gearing (A/B pair)	IDDRX2 or ODDRX2	Not available	Basic I/O registers or x1 gearing	Basic I/O registers or x1 gearing
x2 gearing (C/D pair)	Basic I/O registers or x1 gearing	Basic I/O registers or x1 gearing	IDDRX2 or ODDRX2	Not available
x4 gearing	IDDRX4 or ODDRX4	Not available	Basic I/O registers or x1 gearing	Basic I/O registers or x1 gearing
7:1 gearing	IDDRX71 or ODDRX71	Not available	Basic I/O registers or x1 gearing	Basic I/O registers or x1 gearing

7.2. High-Speed ECLK Bridge

The high-speed ECLK bridge is used to enhance communication of ECLKs across the MachXO2 device and is mainly used for high-speed video applications. It is available on MachXO2-640U, MachXO2-1200/U and higher density devices. 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 the top and bottom sides, 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 MachXO2 sysCLOCK PLL Design and Usage Guide (FPGA-TN-02157) for ECLK bridge connectivity details.

The ECLK bridge supports all the generic high-speed interfaces except the high-speed x2 and x4 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 x2, x4, 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 shown below is the STOP port of the ECLKSYNCA component. The RX_RST or the TX_RST is the reset port of the ODDR/IDDR and CLKDIV components. The RX_ECLK or TX_ECLK shown 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.

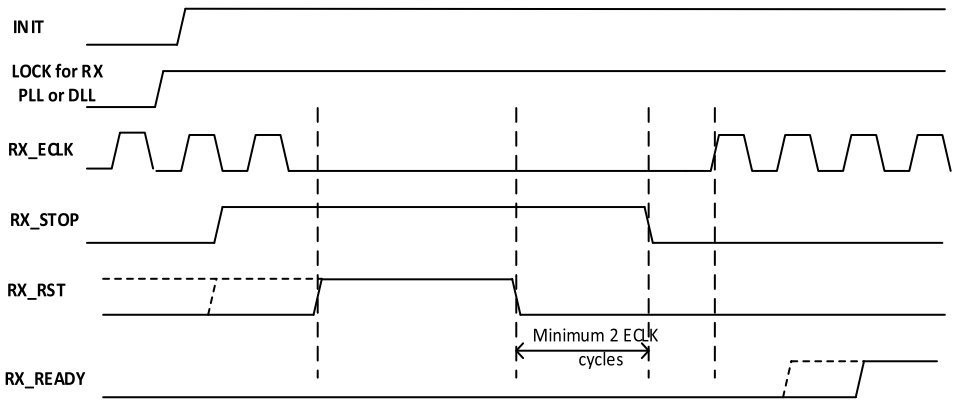


Figure 7.1. Reset Synchronization for Receive Interfaces

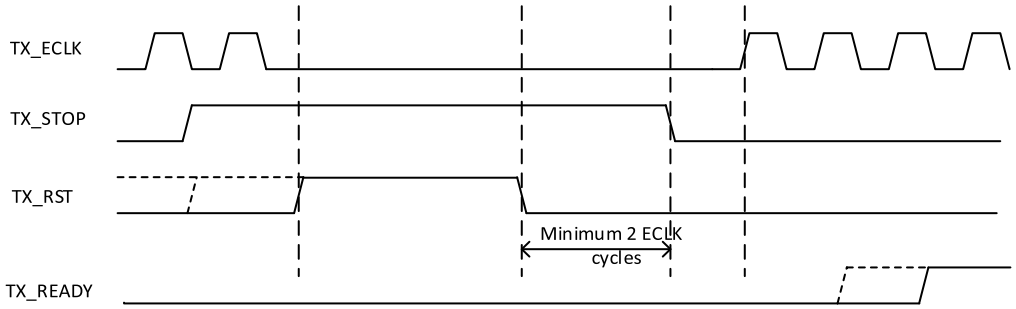


Figure 7.2. Reset Synchronization for Transmit Interfaces

These timing requirements are built into the generic DDR x2/x4/7:1 modules when they are generated by IPexpress. The RX_STOP/TX_STOP, RX_RST/TX_RST, and RX_ECLK/TX_ECLK shown in the figure come from the soft IP that is automatically created by when generating the module in IPExpress. Please note that the names of the ports shown in the figure above may not match the ports in the generated soft IP. These are shown here for logical understanding only. The reset timing requirements must be followed and implemented in RTL code when the generic DDR interfaces are built outside of IPexpress.

7.4. Timing Analysis for High-Speed GDDR 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 use for each type of interface and the expected Trace results. The preferences can either be entered directly in the preference file (.lpf file) or through the Spreadsheet View graphical user interface.

The External Switching Characteristics section of the MachXO2 Family Data Sheet (FPGA-DS-02056) should be used along with this section. The data sheet specifies the actual values for these constraints for each of the interfaces.

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. Refer to the High-Speed GDDR Interface Details section of this document for all the clock pin and clock routing requirements.

Setup and Hold Time Constraints

All of the receive interfaces can be constrained with setup and hold preferences.

Receive Centered Interface – Figure 7.3 shows the data and clock relationship for a Receive Centered Interface. Since the clock is centered to the data, it often provides sufficient setup and hold time at the device interface.

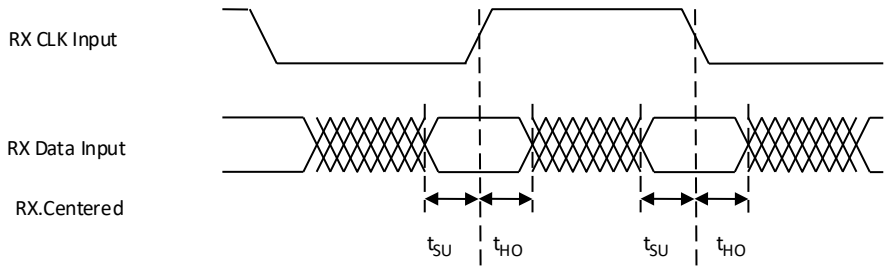


Figure 7.3. Receiver RX.CLK.Centered Waveforms

You must specify in the software preference the amount of setup and hold time available. These parameters are listed in the figure as tSU (setup time) and tHO (hold time). They can be directly provided using the INPUT_SETUP and HOLD preference as shown below:

```
INPUT_SETUP PORT "Data" <tSU> ns HOLD <tHO> ns CLKPORT "CLK";
```

Where: Data = Input Data Port; CLK = Input Clock Port.

The External Switching Characteristics section of the MachXO2 Family Data Sheet (FPGA-DS-02056) specifies the minimum setup and hold times required for each of the high-speed interfaces running at maximum speed. For designs not running at the maximum speed, the Static Timing Analysis tool in the software can be used to calculate the setup and hold time values.

Using a GDDRX2_RX.ECLK.Centered interface running at 250MHz as an example, the preference can be set like this. The software provides the minimum requirement of t_{SU} and t_{HO} for the interface.

```
INPUT_SETUP PORT Data 0.500000 ns HOLD 0.500000 ns CLKPORT "CLK";
```

Receive Aligned Interface – Figure 7.4 shows the data and clock relationship for a Receive Aligned Interface. The clock is aligned edge-to-edge with the data. The DDR memory at the receive side has the same timing behavior.

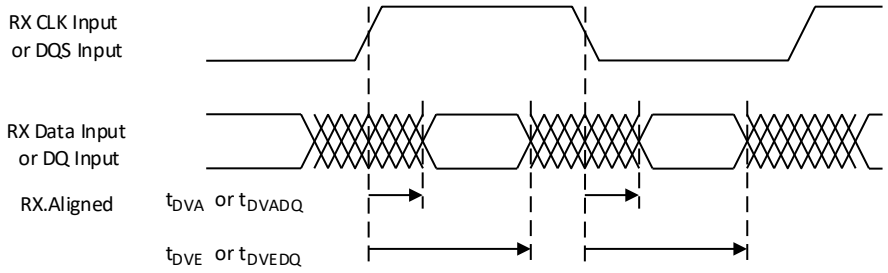


Figure 7.4. Receiver RX.CLK.Aligned and MEM DDR Input Waveforms

The worst case data may occur after the clock edge, and therefore has a negative setup time when entering the device. For this interface, the worst case setup time is specified by tDVA, which is the data valid after the clock edge. The worst case hold time is specified as t_{DVE}, which is the data hold after clock. The setup and hold time for this interface can be specified as below.

```
INPUT_SETUP PORT Data <-tDVA > ns HOLD < tDVE> ns CLKPORT "CLK";
```

Where: Data = Input Data Port; CLK= Input Clock Port

A negative number is used for SETUP time as the data occurs after the clock edge in this case. The External Switching Characteristics section of the MachXO2 Family Data Sheet (FPGA-DS-02056) specifies the maximum t_{DVA} and minimum t_{DVE} values required for each of the high-speed interfaces running at maximum speed. The data sheet numbers for this preference are listed in UI (Unit Intervals). One UI is equal to half of the clock period. Hence, these numbers need to be calculated from the clock period used.

For the GDDRX2_RX.ECLK.Aligned interface running at a speed of 250MHz (UI = 2.0ns)

t_{DVA} = 0.32UI = 0.64ns, t_{DVE} = 0.70UI = 1.4ns

The preference for this case is:

```
INPUT_SETUP PORT Data -0.640000 ns HOLD 1.400000 ns CLKPORT "CLK";
```

Receive 7:1 LVDS Interface – The 7:1 LVDS interface is a unique GDDR interface, which uses one cycle of the pixel clock to align the seven data bits. Figure 7.5 shows the timing of this interface, where t_{RPBi} is the input stroke position for bit i. For this interface, the maximum setup time for bit0 is specified by the t_{RPB0} min, while the minimum hold time is specified as t_{RPB0} max. The t_{RPBi} min and t_{RPBi} max form the boundary of the input strobe position for biti of this interface. The values can be found in the External Switching Characteristics section of the MachXO2 Family Data Sheet (FPGA-DS-02056).

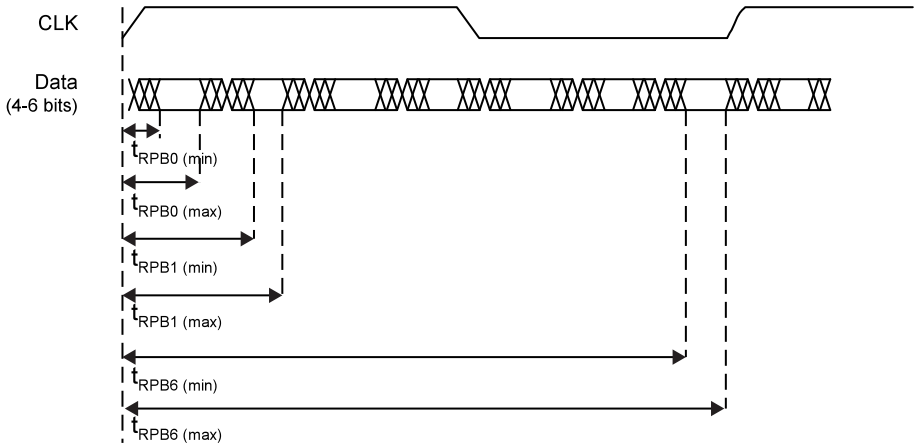


Figure 7.5. Receiver GDDR71_RX. Waveforms

It is recommended to use the MachXO2 Display Interface Reference Design (RD1093) for 7:1 interface implementation.

Receive Dynamic Interfaces – Static Timing Analysis does not show timing for all the dynamic interface cases, either the clock or data delay are dynamically updated at run time.

Clock-to-Out Constraints

All of the transmit (TX) interfaces can be constrained with clock-to-out constraints to detect the relationship between the clock and data when leaving the device.

Figure 7.6 shows how the clock-to-out is constrained in the software. Minimum tCO is the minimum time after the clock edge transition that the data does not transit. Therefore, any data transition must occur between the tCO minimum and maximum values.

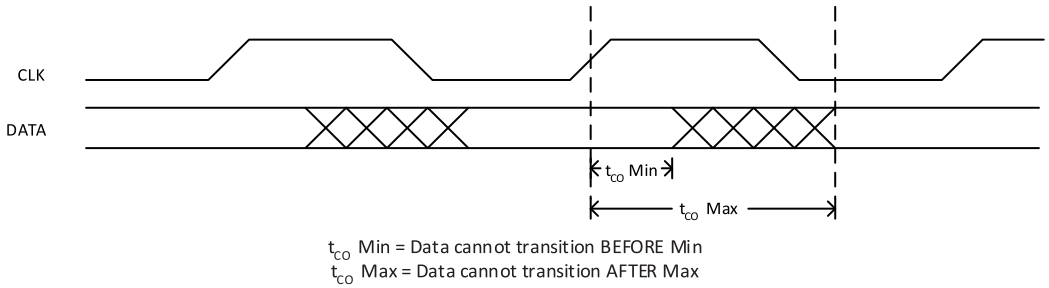


Figure 7.6. t_{CO} Minimum and Maximum Timing Analysis

Transmit Centered Interfaces – The transmit clock is expected to be centered with the data when leaving the device. Figure 7.7 shows the timing for a centered transmit interface. DDR memory transmit side has the same timing behavior as the transmit centered interface.

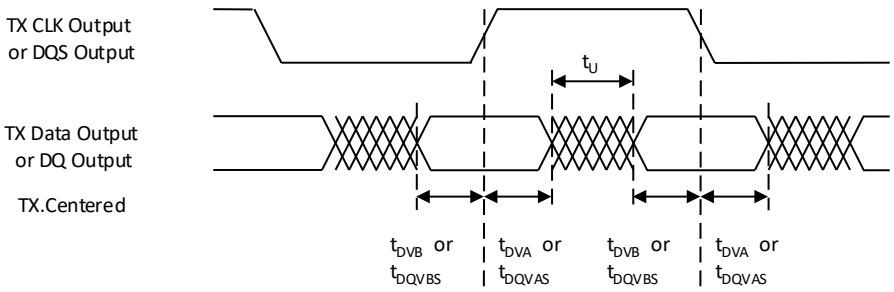


Figure 7.7. Transmitter TX.CLK.Centered and MEM DDR Output Waveforms

Figure 7.7 shows that the maximum value after which the data cannot transit is -t_{DVB}. The t_{DVB} is also the data valid before clock edge value. The minimum value before which the data cannot transition is -(t_U + t_{DVB}), where t_U is the period of time the data is in transition. This is also the data valid after clock value. A negative sign is used because in this particular case where clock is forwarded centered-aligned to the data, these two conditions occur before the clock edge.

The MachXO2 Family Data Sheet (FPGA-DS-02056) specifies the t_{DVB} and t_{DVA} values at maximum speed. But we do not know the tU value, so the minimum tCO can be calculated using the following equations

t_{CO} Min. = -(t_{DVB} + t_U)

½T = t_{DVA} + t_{DVB} + t_U

-(t_{DVB} + t_U) = t_{DVA} - ½T

t_{CO} Min. = t_{DVA} - ½T

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

```
CLOCK_TO_OUT PORT "Data" MAX <-tDVB> MIN <tDVA-1/2 Clock Period> CLKPORT "CLK"
CLKOUT PORT "Clock";
```

Where: Data = Data Output Port; Clock = Forwarded Clock Output Port; CLK = Input Clock Port

The values for tDVB and tDVA can be found in the External Switching Characteristics section of the MachXO2 Family Data Sheet (FPGA-DS-02056) for the maximum speed.

For a GDDRX2_TX.SCLK.Centered interface running at 250MHz, the preference would be:

```
CLOCK_TO_OUT PORT "Data" MAX -0.670000 ns MIN -1.330000 ns CLKPORT "CLK" CLKOUT
PORT "Clock";
```

Transmit Aligned Interfaces – In this case, the clock and data are aligned when leaving the device. Figure 7.8 shows the timing diagram of this interface.

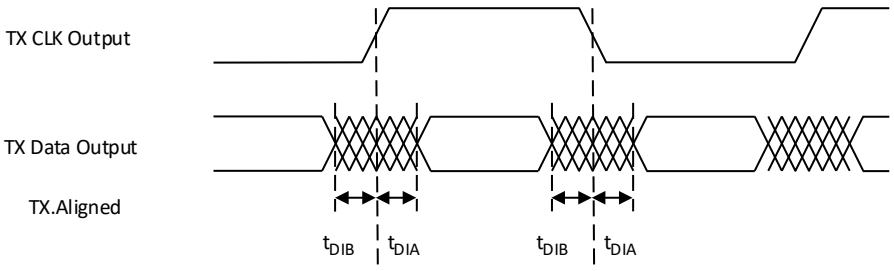


Figure 7.8. Transmitter TX.CLK.Aligned Waveforms

Figure 7.8 shows that maximum value after which the data cannot transition is t_{DIA}. This is the data invalid after the clock value. The minimum value before which the data cannot transition is -t_{DIB}, which is also the data invalid before the clock value. A negative sign is used for the minimum value because the minimum condition occurs before the clock edge.

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

```
CLOCK_TO_OUT PORT “Data” MAX <tDIA> MIN <-tDIB> CLKPORT “CLK” CLKOUT PORT
“Clock”;
```

Where: Data = Data Output Port; Clock = Forwarded Clock Output Port; CLK = Input Clock Port

The t_{DIA} and t_{DIB} values are available in the External Switching Characteristics section of the MachXO2 Family Data Sheet (FPGA-DS-02056) for maximum speed.

For a GDDRX2_TX.Aligned interface running at 250MHz, $t_{DIA} = t_{DIB} = 0.215$ns. The preference would be:

```
CLOCK_TO_OUT PORT "Data" MAX 0.215000 ns MIN -0.215000 ns CLKPORT "CLK" CLKOUT
PORT "Clock”;
```

Transmit 7:1 LVDS Interface – The 7:1 LVDS interface is a unique GDDR interface, which uses one cycle of the pixel clock to transmit the seven data bits. Figure 7.9 shows the timing of this interface. For this interface, the transmit output pulse position for bit0 is bounded by the tTPB0 min and tTPB0 max. The values for tTPBi can be found in the External Switching Characteristics section of the MachXO2 Family Data Sheet (FPGA-DS-02056).

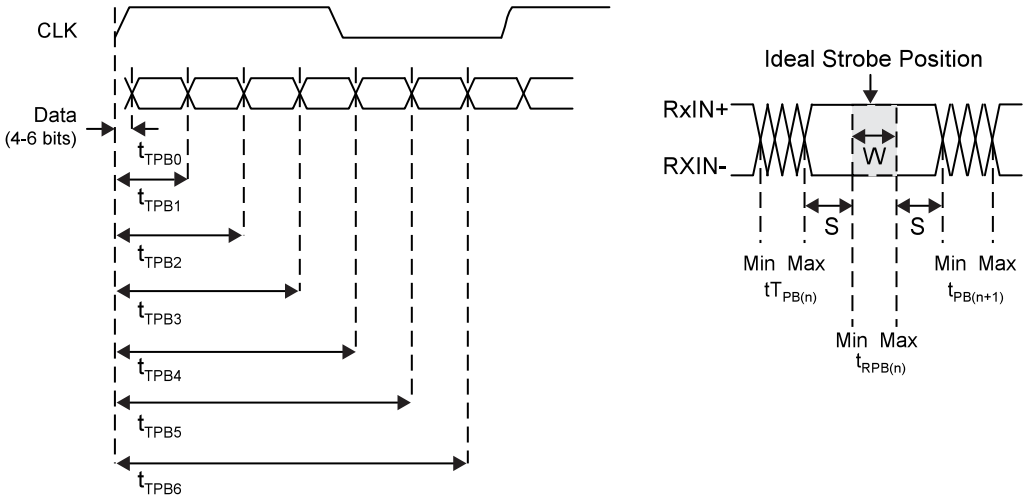


Figure 7.9. Transmitter GDDR71_TX. Waveforms

7.5. Timing Rule Check for Clock Domain Transfers

Clock Domain Transfers within the IDDR and ODDR modules are checked by Trace automatically when these elements are used in a design. Clock domain transfers occur in the GDDR X2, X4, 7:1 modules where there are fastspeed and slow-speed clock inputs.

No special preferences are needed to run this clock domain transfer check in the software. The clock domain transfer checks are automatically done by the software and reported in the Trace report under the section called *Timing Rule Check*. The report lists the timing for both input and output GDDR blocks where a clock domain transfer occurs.

8. DDR/DDR2/LPDDR SDRAM Interfaces Overview

A DDR SDRAM interface transfers data at both the rising and falling edges of the clock. DDR2 is the second generation of the DDR SRDRAM memory, whereas LPDDR is a low-power interface aimed at battery powered applications.

The DDR, DDR2 and LPDDR SDRAM interfaces rely on the use of a data strobe signal, DQS, for high-speed operation. The DDR and LPDDR SDRAM interfaces use a single-ended DQS strobe signal and the DDR2 interface has the option to use a differential DQS strobe. The figures below show typical DDR, DDR2, and LPDDR SDRAM interface signals. SDRAM interfaces are typically implemented with eight DQ data bits per DQS. An x16 interface uses two DQS signals and each DQS is associated with eight DQ bits. Both DQ and DQS are bi-directional ports and are used to read and write to the DDR memory devices.

When reading data from the external memory, data coming into the device is edge-aligned relative 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/DDR2/LPDDR SDRAM, 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 a differential clock (CLKP and CLKN) 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. Note that the DLL that is typically used on standard DDR devices does not exist on LPDDR devices in order to save power. 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 SDRAM also requires 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. An 8-bit interface has one strobe signal.

DDR SDRAM interfaces use the SSTL25 Class I/II I/O standards, DDR2 SDRAM interface uses the SSTL18 Class I/II, and LPDDR SDRAM interface uses the LVCMOS18 standards. DDR2 has an option to use either single-ended or differential DQS.

The following table and figures give an overview of the DDR memory specifications, typical interfaces, and pin-level DQ and DQS relationships.

Table 8.1. DDR / DDR2 and LPDDR Specification for MachXO2-640U, MachXO2-1200/U and Higher Density Devices

	DDR	DDR2	LPDDR
Data Rate	190 to 300 Mbps	266 to 300 Mbps	0 to 300 Mbps
DQS	Single Ended	Single Ended/Differential	Single Ended
Interface	SSTL25	SSTL18	LVCMOS18
Termination	External	On-die	None

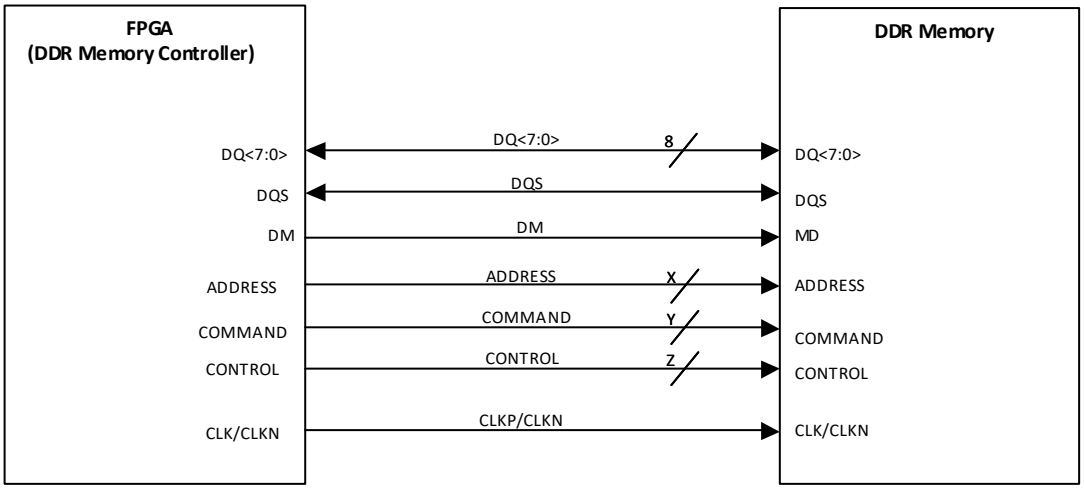


Figure 8.1. Typical DDR SDRAM Interface

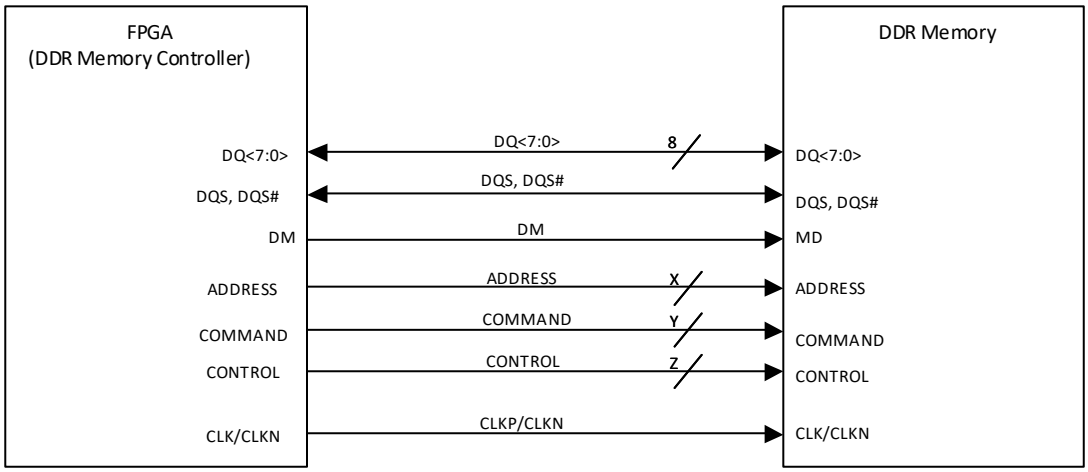


Figure 8.2. Typical DDR2 SDRAM Interface

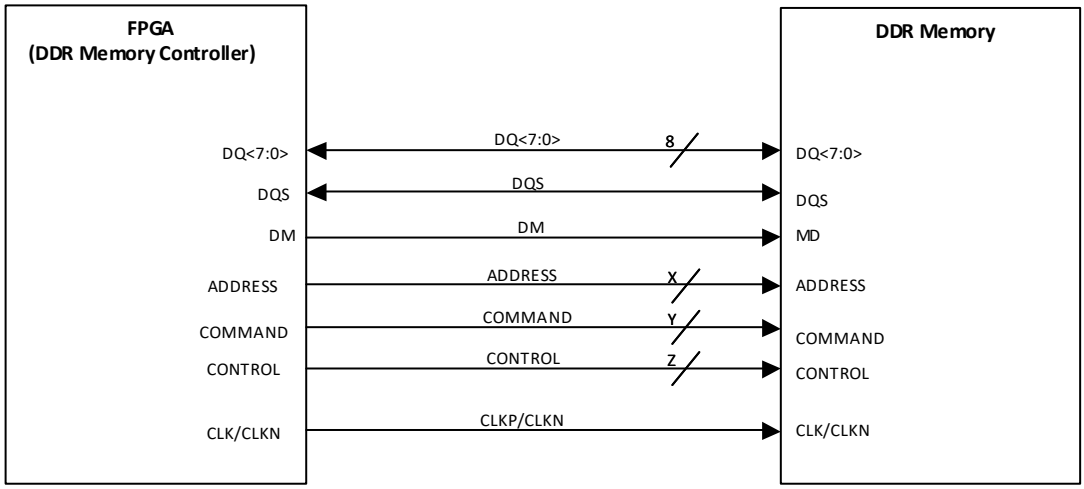


Figure 8.3. Typical LPDDR SDRAM Interface

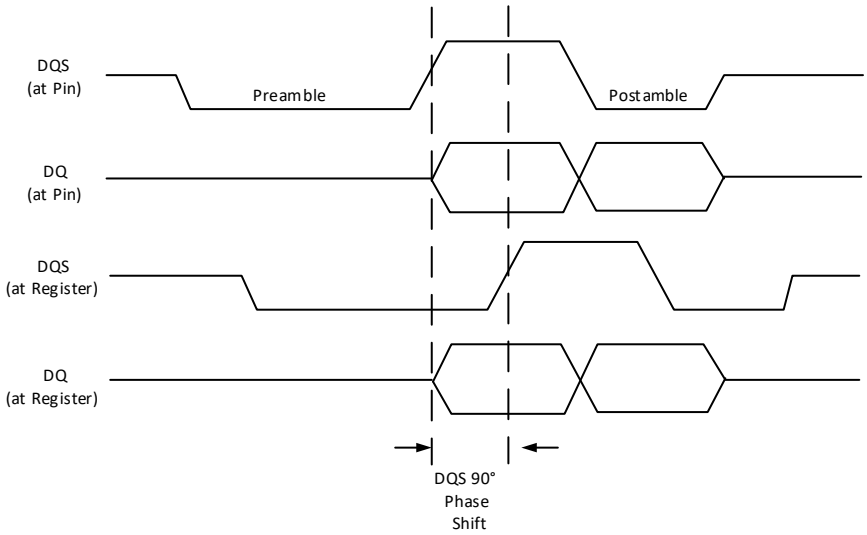


Figure 8.4. DQ-DQS Relationship During READ

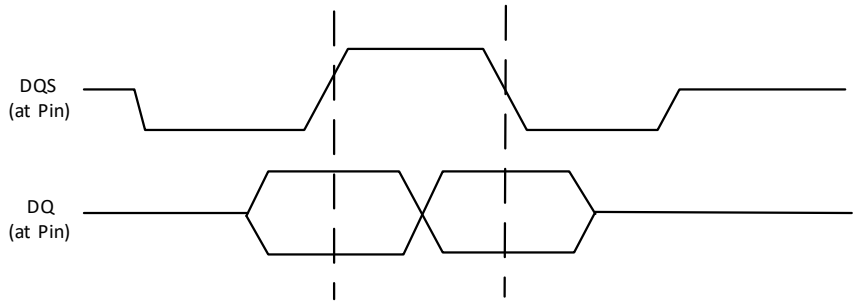


Figure 8.5. DQ-DQS Relationship During WRITE

9. DDR/DDR2/LPDDR SDRAM Interfaces Implementation

As described in the DDR memory overview section, all the DDR SDRAM interfaces rely on the use of a data strobe signal, DQS, for high-speed operation. When reading data from the external memory device, data coming into the MachXO2 device is edge-aligned with respect to the DQS signal. The MachXO2 shifts DQS by 90° before using it to sample the read data. When writing to a DDR SDRAM from the memory controller, the MachXO2 device must generate a DQS signal that is center-aligned with the DQ, the data signals. This is accomplished by ensuring a DQS strobe is shifted 90° relative to DQ data.

MachXO2-640U, MachXO2-1200/U and higher density devices have dedicated DQS support circuitry for generating appropriate phase shifting for DQS. This dedicated circuitry is independent of the generic DDR circuitry and is only available on the right side of the device. The circuitry uses x1 gearing for memory interfaces which results in the memory interface running at 2X frequency relatively to the core performance.

For both DDR2 and LPDDR, XO2 is targeted for embedded chip-to-chip implementation, which requires the fanout of DQ/DQS to be one, whereas fanout of CLKP/CLKN and address/command can potentially be two.

The DQS phase shift circuit uses a frequency referenced DLL to generate delay control signals associated with each of the dedicated DQS pins, and is designed to compensate for process, voltage and temperature (PVT) variations. The frequency reference is provided through one of the PCLK global clock pins. The dedicated DDR support circuit is also designed to provide comfortable and consistent margins for the data sampling window. This section describes how to implement the read and write sections of a DDR memory interface.

9.1. DQS Grouping

A DQS group generally consists of eight DQ pins, one DM pin, one DQS pin or two DQS/DQSN pairs, and one Vref pin. This requires at least 11 I/O logic cells to implement a complete DDR, DDR2, or LPDDR SDRAM interface. MachXO2-640U, MachXO2-1200/U and higher density devices support DQS-14 group to allow DQS signals to span across 12 I/O cells with two reserved for DQS and DQSN. The memory circuitry supported at the right side of the device allows up to 16-bit wide DDR/DDR2/LPDDR bus implementations. This requires two DQS group to be used together.

Each DQS signal spans across 14 I/O. If a single-ended DQS is used, up to 13 I/O spanned by the DQS can be used to implement an 8-bit DDR memory interface. You can assign any eight of the I/O pads within the DQS-14 group to be the DQ data pins. If DQSN is not used, it can be used as a DQ or DM for the DQS group. The MachXO2 device family allows any I/O pin to be used as the Vref pin. This, together with the DQS-14 group structure, increases the flexibility of pin locking for DDR memory interfaces. Refer to the MachXO2 Family Data Sheet (FPGA-DS-02056) for the details of the DQS and DQS-14 group locations.

9.2. DQS Circuitry

The DQS circuitry (DQSBUF) is designed to simplify the implementation of DDR memory interfaces. It integrates several functions, including:

- DQS preamble and postamble management
- DLL-compensated DQS delay
- Data valid and data burst detection

DQS Preamble and Postamble Management

DDR/DDR2/LPDDR SDRAM memory interfaces require preamble and postamble states prior or proceeding the read or write bursts. The pre- and postamble management circuitry is built into the MachXO2 devices. The circuitry ensures clean DQS pulse inside the device, and turns on the DQS for internal logic during preamble, and turn off DQS after the last falling edge of the signal. The clean DQS inside the device helps to guarantee the downstream logic for clock polarity detection (DDRCLKPOL). This signal is used to control the polarity of the clock to the synchronizing registers.

DLL-Compensated Delay Logic

The master DLL (DQSDLL) works in conjunction with delay logic in the DQSBUF to provide the necessary 90° phase shift. The DQSBUF receives the 7-bit delay control code from the DQSDLL at the right side of the chip. The 7-bit delay control is used by the two clock slave delay lines in the DQSBUF at the right side of the device for DDR memory implementation. One is used for the 90° clock shift for the READ operation, and the other is to shift the data by 90° during the WRITE operation.

The DQS circuitry (DQSBUF) generates the DQSR90 by shifting the ECLK by 90°. The DQSR90 signal is then sent to all the DDR memory I/O logic cells at the right side of the chip for clocking the input data during memory READ operations.

DQSW90 is also generated by the DQSBUF and is used for memory WRITE operations. It is the 90° shift of the SCLK signal and is used to send out the DDR data. This meets the requirement of placing the DQS edge at the middle of the data opening of DQ during WRITE.

Data Valid and Data Burst Detection

DQSBUF is also responsible for generating the data valid signal which indicates to the FPGA that valid data is transmitted out the input DDR registers to the FPGA core. The data valid signal is level sensitive and matches data at the FPGA core boundary.

Data burst detection in the DQSBUF is used to position the read pulse in the optimized location for a DDR READ operation. It is critical that a sufficient burst length (BL) is used in the training process. The MachXO2 DDR memory implementation requires at least two consecutive BL2 (BL=2), or one BL4 (BL=4), or a multiple of these burst lengths to be used during training session to allow the device to detect the correct read pulse position.

9.3. I/O Logic Data Path

The DDR memory I/O logic, or the memory PIO cell, is shown in Figure 2.2 at the beginning of the document. At the DDR memory input path, the first set of DDR registers is used to de-mux the DDR data at the positive and negative edges of the phase-shifted DQS signal. The second set is used to transfer the demuxed data from the DQS domain (DQSR90) to the SCLK domain. The final set of registers is used to clock the input data one more time with the SCLK based on DDRCLKPOL. At the output side, the DQS and DQ share the same logic. The parallel data is muxed out by the write clock, DQSW90. The DQS output pulse is controlled by SCLK.

More details of the building blocks of the DDR memory can be found in the DDR Software Primitives and Attributes section.

9.4. DDR/DDR2/LPDDR Memory READ Implementation

MachXO2-640U, MachXO2-1200/U and higher density devices support the DDR/DDR2/LPDDR memory interface functions using the DDR memory module generated through the IPexpress tool. Using IPexpress, you can generate the different modules required to read the data coming from the DDR/DDR2 /LPDDR memory.

The DDR/DDR2/LPDDR read side is implemented using the following three software elements: DQSDLL represents the DLL used for calibration; IDDRDQS implements the input DDR registers; DQSBUF represents the DQS delay block, the clock polarity control logic and the data valid module. The DQSR90 is distributed to each IDDRDQS cell through a dedicated clock tree. The routing of the SCLK must use one of the primary clock nets.

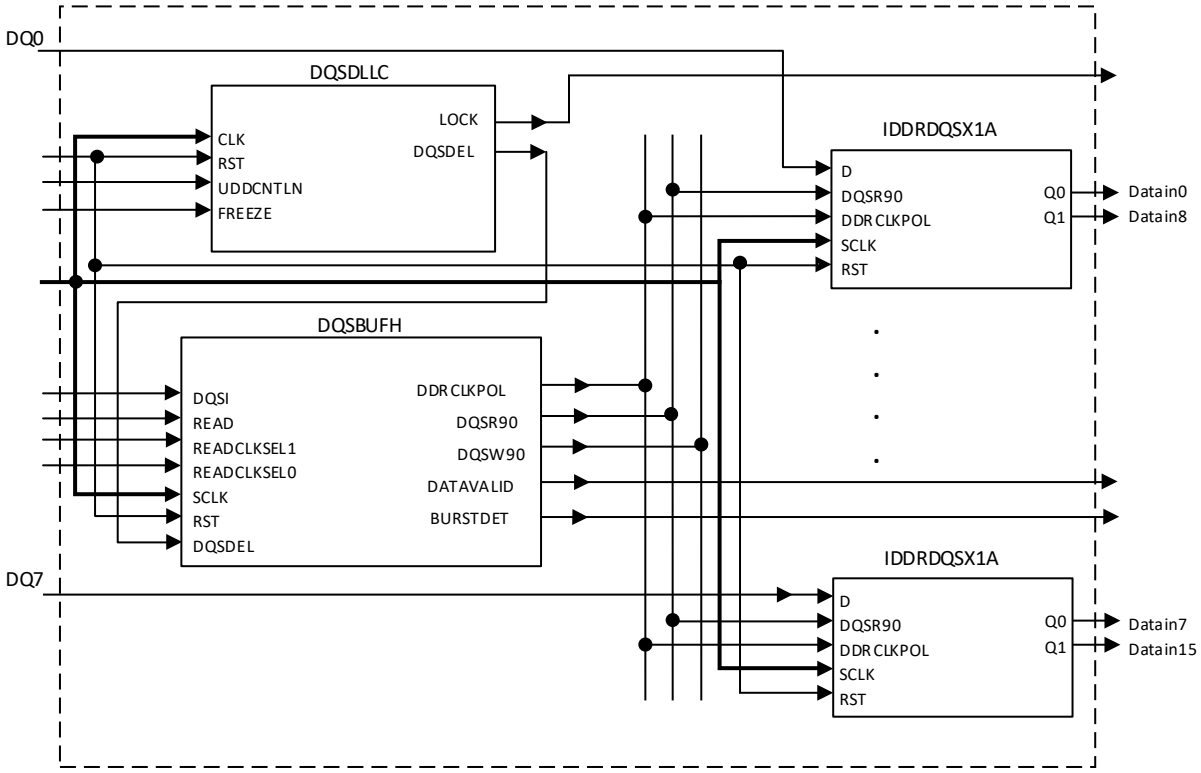


Figure 9.1. DDR/DDR2/LPDDR READ Implementation

9.5. DDR/DDR2/LPDDR Memory WRITE Implementation

To implement the write portion of a DDR memory interface, two streams of single data rate data must be multiplexed together with data transitioning on both edges of the clock. In addition, DQS must arrive at the memory pins center aligned with data (DQ), and edge-aligned with the differential output clock (CLKP/CLKN). Along with these signals, Address/Command and Data Mask (DM) signals also need to be generated. IPexpress should be used to generate this interface.

The DQSW90 is distributed to each ODDRDQS cell through a dedicated clock tree. The routing of the SCLK must use one of the primary clock nets.

DDR/DDR2/LPDDR Write Data (DQ) and Strobe (DQS) Generation

Figure 9.2 shows the DQ/DQS generation for a WRITE operation. DQS90 is a 90°-shifted version of SCLK. As discussed above, DQSW90 is used to shift the DQ instead of the DQS by 90°. Generation of the DQs and DM uses DQSW90, while generation of DQS uses SCLK. This ensures that the edge of DQS is in the middle of the DQ, and DM. DQS is single-ended for DDR and LPDDR, but differential for the DDR2.

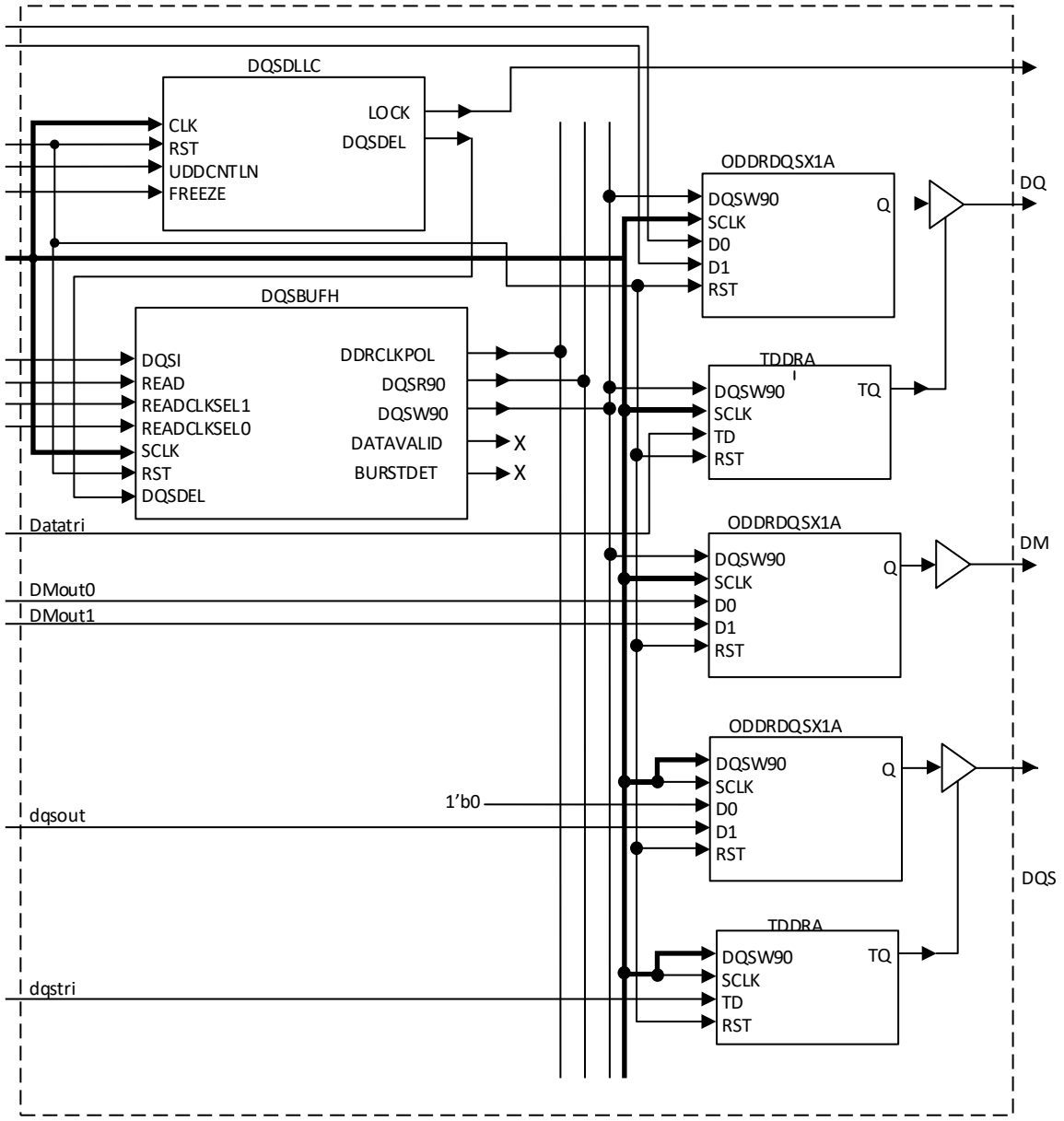


Figure 9.2. DDR/DDR2/LPDDR WRITE Implementation

DDR/DDR2/LPDDR Write Clock, Address and Command Generation

The memory WRITE operation requires the controller to generate a differential clock, address lines, and commands for the SDRAM memory. The clock is generated using the ODDRX (x1) primitive. The inputs of the ODDRX primitive are tied to constants to generate an output clock. When interfacing to the DDR SDRAM memories, CLKP should be connected to the SSTL25D I/O standard. When interfacing to DDR2 memories it should be connected to SSTL18D I/O standard to generate the differential clock outputs. When interfacing to LPDDR SDRAM memories, CLKP should be connected to the LVCMOS18 I/O standard. Generating the CLKN in this manner prevents skew between the two signals.

The address and command signals for DDR/DDR2/LPDDR SDRAM interfaces are generated using the regular output register OREG. The same SCLK is used for the DQ/DQS, as well as clock, address, and command generation. The routing of SCLK must use one of the primary clock nets.

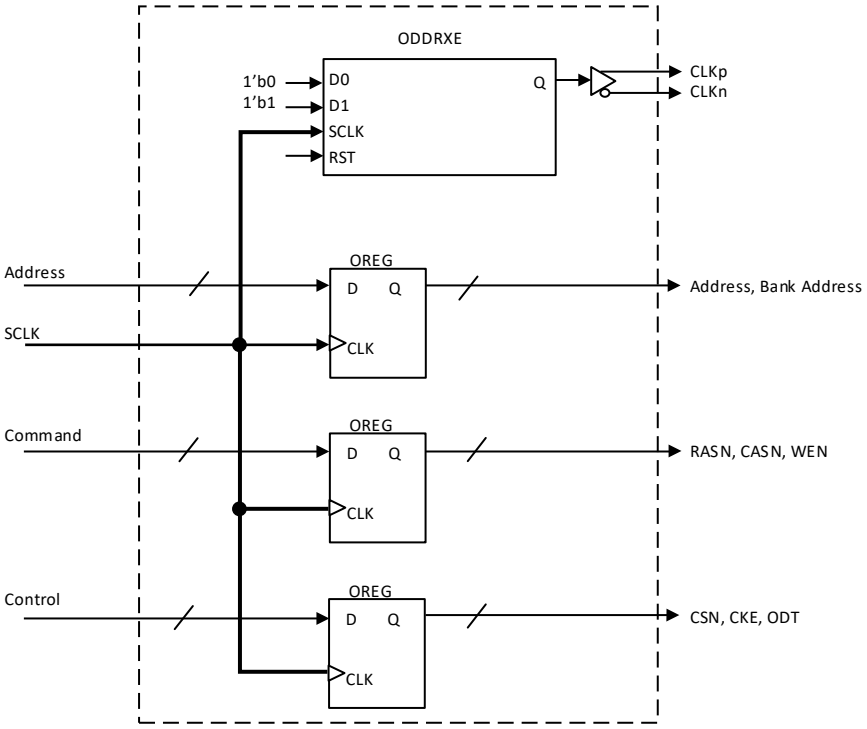


Figure 9.3. CLK, Address and Command Control Pin Generation

10. DDR Memory Interface Generation Using IPexpress

The IPexpress tool should be used to configure and generate all the DDR/DDR2/LPDDR interfaces described above. In the IPexpress user interface, all the DDR memory modules are located under Architecture Modules > I/O. DDR_MEM is used to generate DDR memory interfaces as shown in Figure 10.1.

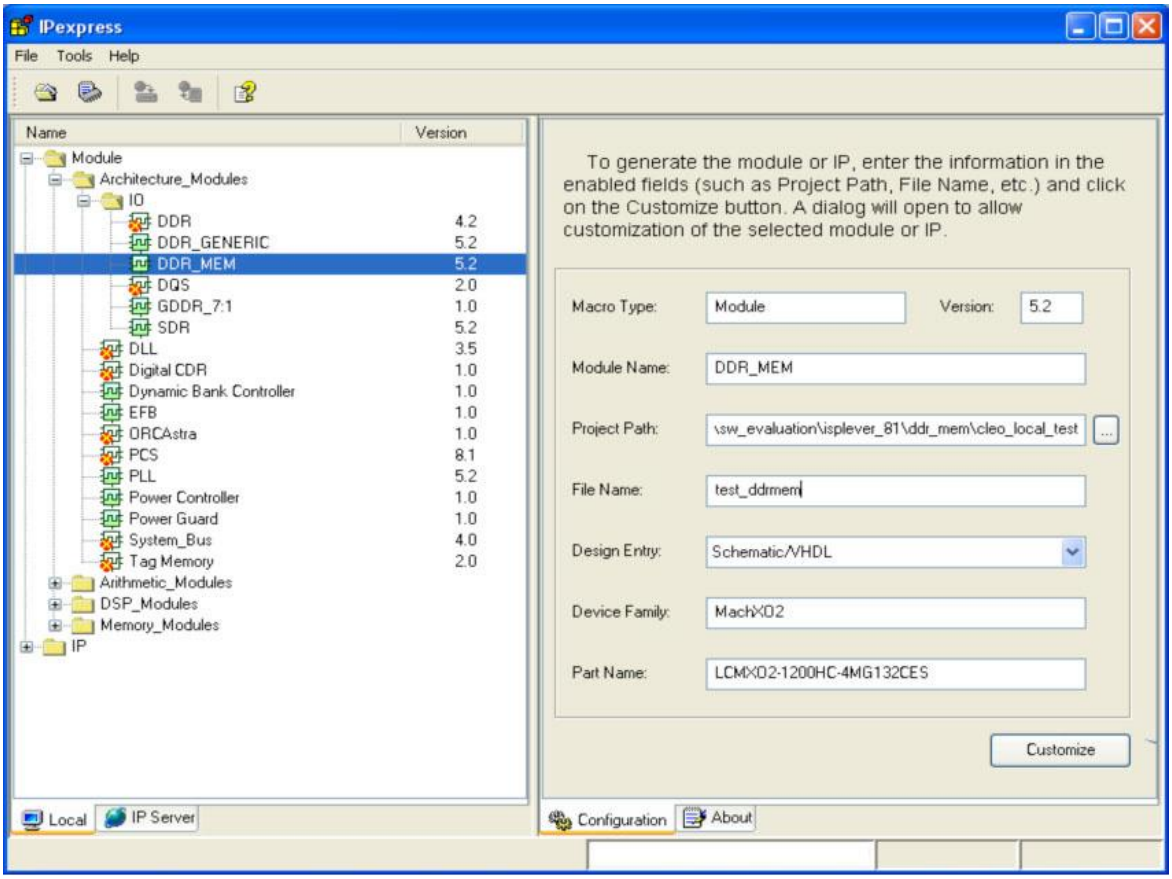


Figure 10.1. IPexpress Main Window

There are two tabs in the DDR_MEM module. The Configuration Tab shown in Figure 10.2 is used to choose among DDR, DDR2, and LPDDR and their corresponding parameters.

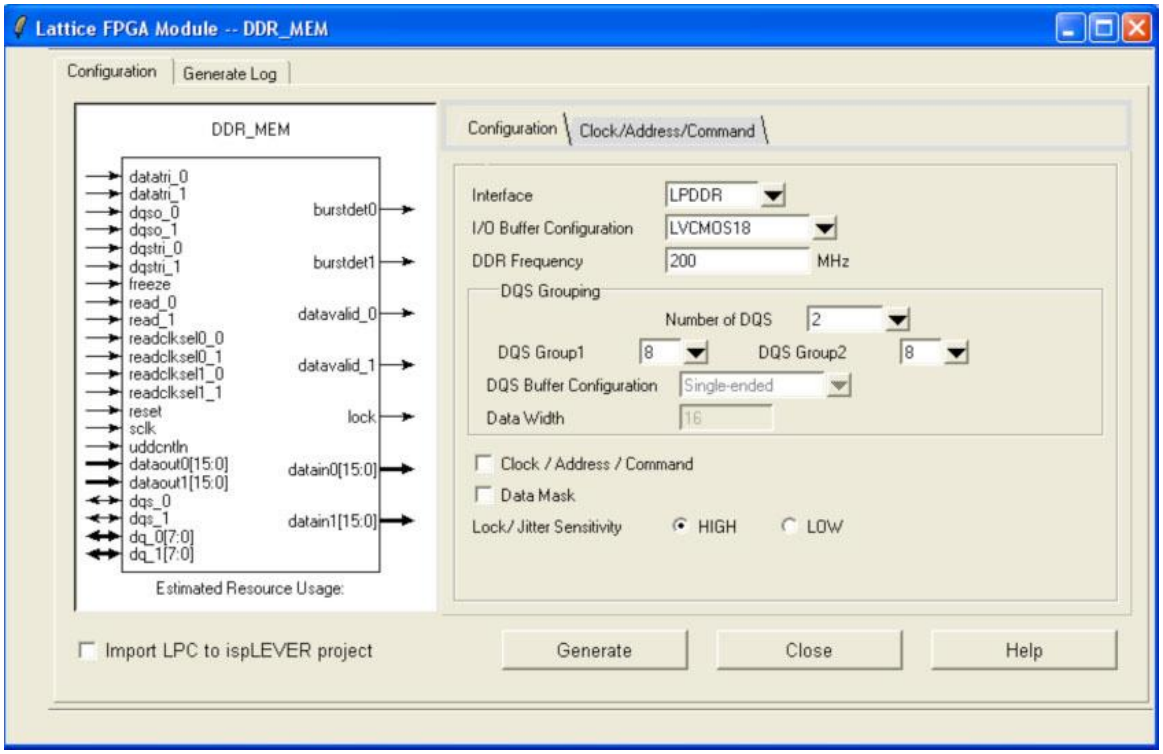


Figure 10.2. Configuration Tab for DDR_MEM

Table 10.1 details the parameters/options for each of the user interface options on the DDR_MEM Configuration Tab.

Table 10.1. Options of the Configuration Tab of the DDR_MEM Module

User Interface Option	Description	Range	Default Value
Interface Type	Types of Interface	DDR, DDR2, LPDDR	DDR
I/O Buffer Configuration	I/O standard associated with each interface type. This is automatically set based on the interface selected	SSTL25_I, SSTL18_I, LVCMOS18	SSTL25_I
DDR Frequency	The speed the DDR memory interface is running.	DDR = 83–100MHz DDR2 = 125–150MHz LPDDR = 0-133MHz	DDR = 100 MHz DDR2 = 150 MHz LPDDR = 133 MHz
Number of DQS	Number of DQS groups available	1, 2	2
Number of DQ for DQS Group1 and DQS Group 2	Data width for the DQS group	1–8	8
DQS Buffer Configuration for DDR2	DQS buffer type	Single-ended, Differential	Single-ended
Data Width (calculated)	Data width for the interface	1–16	(calculated)
Clock/Address/Command	Clock/address/command interface is generated when this option is checked.	Enabled, Disabled	Disabled
Data Mask	Data mask signal is generated when this option is checked	Enabled, Disabled	Disabled
Lock/Jitter Sensitivity	Lock Sensitivity attribute for DQSDLL. Low means less sensitive to jitter.	High, Low	Low

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

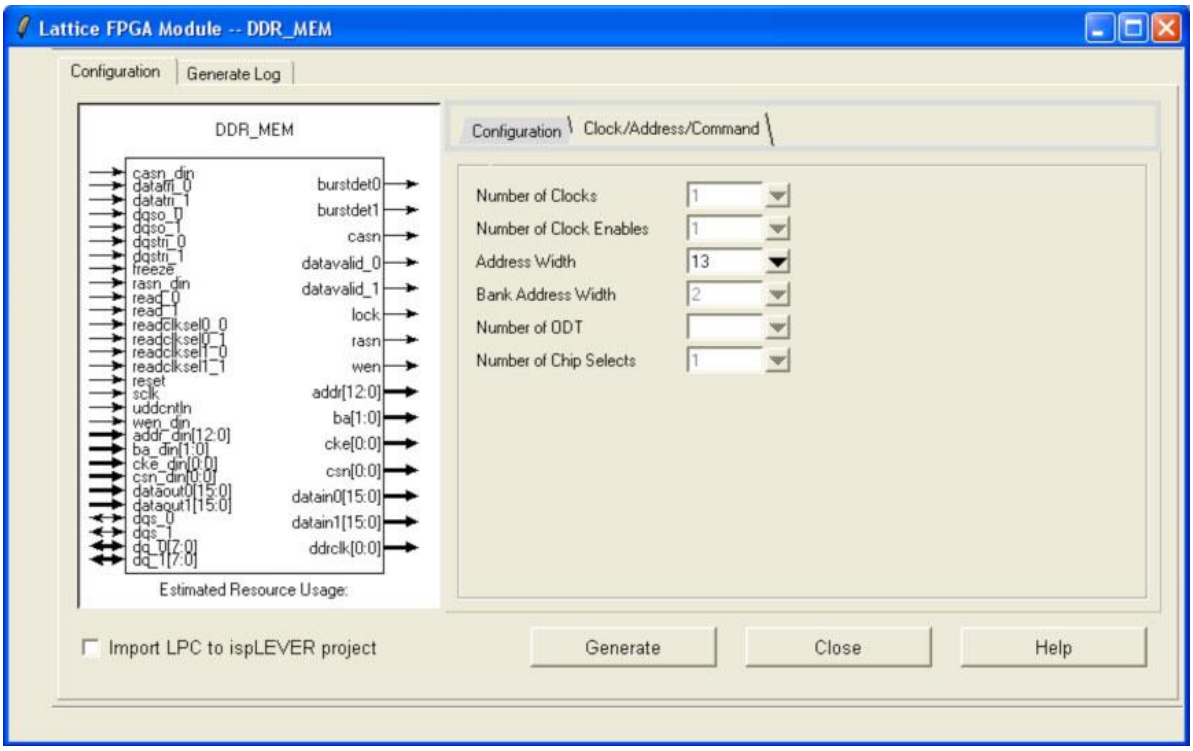


Figure 10.3. Clock/Address/Command Tab for DDR_MEM

Table 10.2. Options of the Configuration Tab of the DDR_MEM Module

User Interface Option	Range	Default Value
Number of Clocks	DDR/DDR2: 1, 2LPDDR: 1	1
Number of Clock Enables	DDR/DDR2: 1, 2 LPDDR: 1	1
Address Width	DDR: 12–14 DDR2: 13–16 LPDDR: 12–14	DDR: 13DDR2: 14LPDDR: 13
Bank Address Width	DDR: 2 DDR2: 2, 3 LPDDR: 2	DDR: 2 DDR2: 2 LPDDR: 2
Number of ODT	DDR: blank DDR2: 1, 2 LPDDR: blank	DDR: Blank DDR2: 1 LPDDR: Blank
Number of Chip Selects	DDR / DDR2: 1, 2 LPDDR: 1	DDR/DDR2: 1 LPDDR: 1

11. DDR Memory DQ/DQS Design Rules and Guidelines

Listed below are some rules and guidelines for implementing DDR memory interfaces in MachXO2-640U, MachXO2-1200/U and higher density devices.

- DDR memory I/O logic and DQS circuitry are available for the bank on the right side of the device. Correspondingly, DDR memory interface can only be implemented at the right side bank of the device.
- There are two DQSDLLs in the device, one for the left and bottom banks, and one for the right and top banks. Only the one for right and top banks can be used for DDR memory interface implementations.
- The delay control code generated by the DQSDLL is applied to all the DDR memory I/O logic on the right side bank.
- When implementing a DDR1 SDRAM interface, all interface signals should be connected to the SSTL25 I/O standard.
- For DDR2 SDRAM interfaces, signals should be connected to the SSTL18 I/O standard.
- For LPDDR SDRAM interfaces, signals should be connected to the LVCMOS18 standard.
- All DDR memory interfaces require a differential clock signal. The clock signal should be connected to corresponding differential I/O standard specified by the memory interface.
- The use of differential DQS is optional for DDR2. If differential DQS is used it should be connected to SSTLD18 I/O standard.

12. DDR/DDR2/LPDDR Pinout Guidelines

- The DQS-DQ association rule must be followed.
  - All associated DQs (8 or 4) to a DQS must be in the same DQS-14 group.
- The data mask (DM) must be part of the corresponding DQS-14 group.
  - Example: DM[0] must be in the DQS-14 group that has DQ[7:0], DQS[0].
- DQS must be allocated to a dedicated DQS pad.
  - DQSN pad is used when differential DQS is selected.
- Do not assign any signal to the DQSN pad if SSTL18D is applied to the DQS pad.
  - The software automatically places DQS# when SSTL18D is applied.
- The clock to the PLL used to generate the outputs must be locked to the correct dedicated PLL pin input.
- In addition to the DQS-14 group, a DDR memory interface typically needs an additional 20-24 pins such as CLK/CLKN, address and command pins depending on memory density. Pins in banks other than the right side of the device can be used for these signals.
- The MachXO2 device family allows any I/O to be used as Vref pin. The Vref for DDR/DDR2 must be part of the DQS-14 group. Refer to MachXO2 sysI/O Usage Guide (FPGA-TN 02158), for Vref assignment rules.

13. DDR Software Primitives and Attributes

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

Table 13.1. MachXO2 DDR Software Primitives

Type	Primitive	Usage
Data Input	IDDRXE	Generic DDR x1
	IDDRX2E	Generic DDR x2
	IDDRX4B	Generic DDR x4
	IDDR71A	Generic DDR 7:1
	IDDRDQSX1A	DDR memory
Data Output	ODDRXE	Generic DDR x1
	ODDRX2E	Generic DDR x2
	ODDRX4B	Generic DDR x4
	ODDR71A	Generic DDR 7:1
	ODDRDQSX1A	DDR memory
DQS Tristate	TDDRA	DDR memory
DQSBUF Logic	DQSBUFH	DDR memory
DLL	DQSDLLC	Master DLL for Generic x2, x4, and DDR Memory
Input Delay	DELAYD	Delay block with dynamic control for Generic x2, x4
	DELAYE	Delay block with fixed delays for Generic DDR x1, x2, x4
	DLLDELC	Clock slave delay cell for Generic DDR x2, x4

13.1. Input DDR Primitives

The input DDR primitives represent the modules used to capture both the GDDR data and the DDR data coming from a memory interface. There are several modes for the DDR input registers to implement different gearings for GDDR interfaces. The memory DDR uses a different primitive than the GDDR primitives. For all the data ports of the input primitives, Q0 of the parallel data is the first bit received.

IDDRXE

The primitive implements the input register block in x1 gearing mode. This is used only for the generic DDR x1 interface (gearing ratio 1:2) available on all sides of the MachXO2 devices. It uses a single clock source, SCLK, for the entire primitive so it does not involve a clock domain transfer.

Figure 13.1. 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.

13.2. IDDRX2E

The primitive implements the input register block in x2 gearing mode. This is used only for the generic DDR x2 interface (gearing ratio 1:4) on the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices. 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 at the bottom side of the device.

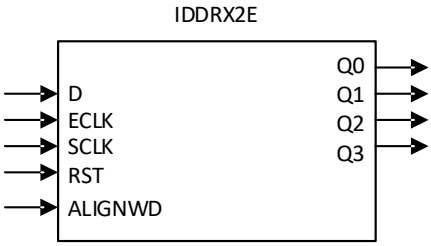


Figure 13.2. IDDRX2E Symbol

The internal register structure for this primitive is based on the video PIO cell, as shown in receive path of Figure 2.3. 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.

The IDDRX2E primitive contains a fundamental logic error. *Odd* word alignments (SEL = '1') do not present the correct data at the *Q3* output port. You must be aware of this behavior if instantiating the primitive directly into your design, and take appropriate steps to correct the temporal misalignment.

In Diamond 3.12 and later versions, the temporal misalignment is corrected when using the Generic IDDRX2 logic interfaces (GDDRX2_RX.ECLK.Centered/Aligned) generated with the Lattice IPexpress tool, and which are described in Generic High-Speed DDR Interfaces.

IDDRX4B

The primitive implements the input register block in x4 gearing mode. This is used only for the generic DDR x4 interface (gearing ratio 1:8) on the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices. 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 at the bottom side of the device.

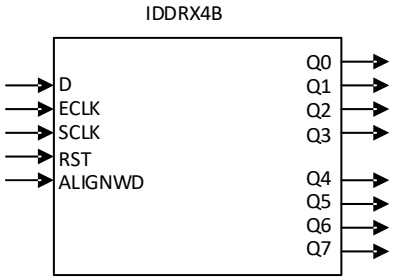


Figure 13.3. IDDRX4B Symbol

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 1c. 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.

The IDDRX4B primitive contains a fundamental logic error. *Odd* word alignments (SEL = '1') do not present the correct data at the *Q7* output port. You must be aware of this behavior if instantiating the primitive directly into your design, and take appropriate steps to correct the temporal misalignment.

In Diamond 3.12 and later versions, the temporal misalignment is corrected when using the Generic IDDRX4 logic interfaces (GDDRX4_RX.ECLK.Centered/Aligned) generated with the Lattice IPexpress tool, and which are described in Generic High-Speed DDR Interfaces.

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) on the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices. 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 at the bottom side of the device.

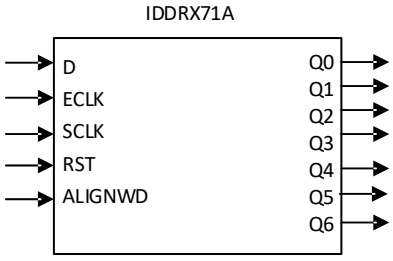


Figure 13.4. IDDRX71A Symbol

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 2.3. 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

IDDRDQSX1A

This primitive is the DDR memory input buffer. It can only be used on the right side of the MachXO2-640U, MachXO2-1200/U and higher density devices for DQ or DQS input pins. This primitive comprises three stages of registers. The first stage of DDR register captures the incoming data at the rising and falling edges of the DQSR90 signal, which is the 90° shifted DQS strobe, generated by the DQS circuitry (DQSBUF). The captured data is then transferred to the second stage of registers with the rising edge SCLK or falling edge SCLK depending on the polarity of the DDRPOLCLK signal, which also is generated from DQSBUF to guarantee the clock domain crossing. The final stage of registers is re-clocked by the rising edge of SCLK to provide the full clock cycle transition to the core.

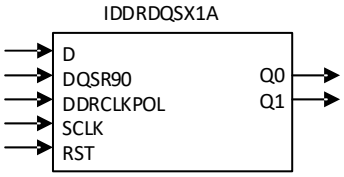


Figure 13.5. IDDRXQSX1A Symbol

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

13.3. 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 or to the DDR memory device. There are several modes for the DDR output registers to implement different gearings for GDDR interfaces. The memory DDR output uses a different primitive than the GDDR interfaces. For all the data ports of the output primitives, D0 of the parallel data is the first bit transmitted.

ODDRXE

This primitive is implemented the output register block in x1 gearing mode. It can be used in all sides of the MachXO2 devices. A single primary clock source, SCLK from the FPGA core, is used for this primitive.

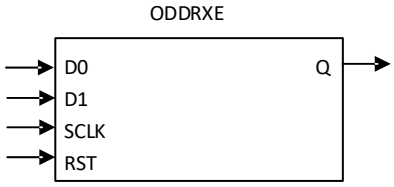


Figure 13.6. ODDRXE Symbol

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.

ODDRX2E

The primitive implements the output register block in x2 gearing mode. This is used only for the generic DDR x2 interface (gearing ratio 4:1) on the top side of the MachXO2-640U, MachXO2-1200/U and higher density devices. 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 at the top side of the device.

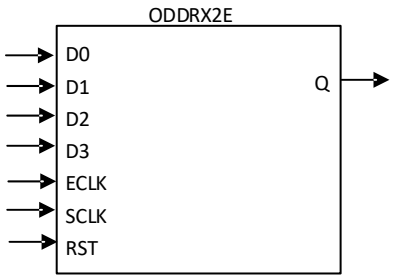


Figure 13.7. ODDRX2E Symbol

The internal register structure for this primitive is based on the video PIO cell, as shown in transmit path of Figure 2.3. 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.

ODDRX4B

The primitive implements the output register block in x4 gearing mode. This is used only for the generic DDR x4 interface (gearing ratio 8:1) on the top side of the MachXO2-640U, MachXO2-1200/U and higher density devices. 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 at the top side of the device.

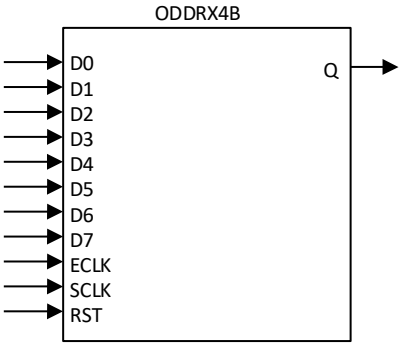


Figure 13.8. ODDRX4B Symbol

The internal register structure for this primitive is based on the video PIO cell, as shown in the transmit path of Figure 2.3. 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.

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) on the top side of the MachXO2-640U, MachXO2-1200/U and higher density devices. 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 at the top side of the device.

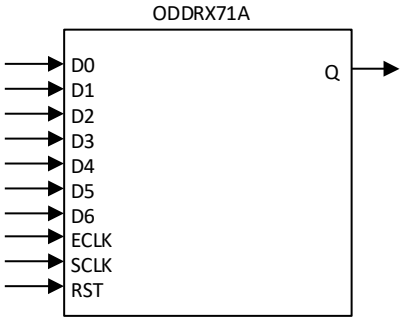


Figure 13.9. ODDRX71A Symbol

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 2.3. 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.

ODDRDQSX1A

This primitive is the DDR memory output buffer. It can only be used on the right side of the MachXO2-640U, MachXO2-1200/U and higher density devices for DQ or DQS output pins. For DQ output data, the parallel data is clocked into this primitive using the system clock, SCLK. The captured data is multiplexed out to the pin by the DQSW90 signal, which is the 90° shift of SCLK, which is generated by the DQS circuitry (DQSBUF). For the DQS output, it uses the same output buffer architecture but the clock is controlled by the SCLK to ensure the DQS is not shifted 90° like the DQ signals. This ensures the DQS to be center aligned with the DQ for transmission.

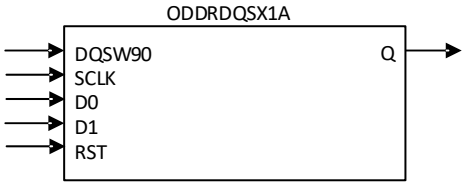


Figure 13.10. ODDRDQSX1A Symbol

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

TDDRA

This is a special tri-state primitive used at the right side of the MachXO2-640U, MachXO2-1200/U and higher density devices for DQ or DQS pins. Its register structure is shown in the Tristate Register block of memory PIO cell, Figure 2.2.

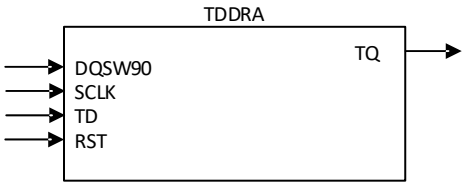


Figure 13.11. TDDRA Symbol

When this component is used for DQ pins, the DQSW90 port should be driven by the DQSW90 signal from the DQSBUF component. When this component is used for DQS, the DQSW90 port should be driven by SCLK. This primitive has an attribute associated with it as listed in the table. This attribute is to ensure the DQS pin is centered at the data, DQ, during DDR memory write.

Table 13.2. TDDRA Attributes

Attribute	Description	Values	Software Default
DQSW90_INVERT	Select clock polarity for the TDDRA for DQS pin. Only used for DQS during DDR Write	ENABLED, DISABLED	DISABLED

13.4. DDR Control Logic Primitives

The DDR primitives discussed below include the DLL, the DQS circuitry, and the delay elements. The DLL is shared between GDDR and DDR memory. The DQS circuitry is only used for DDR memory. The delay elements are for data paths or the clock slave delay paths.

DQSBUFH

This primitive represents the DQS circuitry used in DDR memory interface. It generates a shifted version of DQS for DDR memory READ and DDR WRITE operations. It provides preamble and postamble detection, data valid detection, and the many functions necessary for DDR memory interface applications.

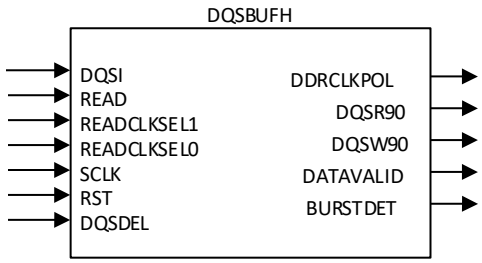


Figure 13.12. DQSBUFH Symbol

Table 13.3. DQSBUFH signals

Signal	I/O	Description
DQSI	I	DQS signal from pin
READ	I	Signal for DDR read mode, from FPGA logic
READCLKSEL1, READCLKSEL0	I	Select read clock source and polarity control for READ pulse position control in T/4 precision. The 4 positions are the rising/falling edges of SCLK or DQSW90. The signals come from FPGA logic.
SCLK	I	System clock
RST	I	RESET for this block
DQSDEL	I	DQS slave delay control from DQSDLLC
DDRCLKPOL	O	SCLK polarity control
DQSR90	O	DQS phase shifted by 90° output DQSW90 O SCLK phase shifted by 90° output
DQSW90	O	SCLK phase shifted by 90° output
DATAVALID	O	Data valid signal for READ mode
BURSTDET	O	Burst detection signal

Figure 13.13 gives an overview of the functional blocks within this DQSBUFH primitive.

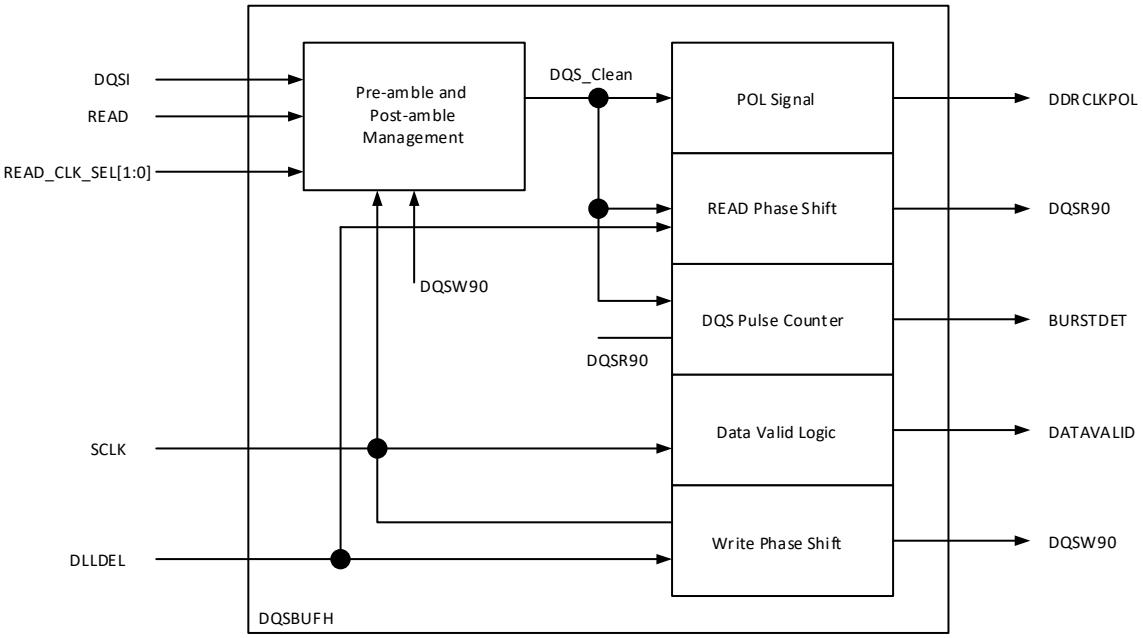


Figure 13.13. DQSBUFH Block Diagram

The DQS_Clean is a digitally-generated DQS signal which is a glitch-less version of the DQSI for downstream logic. The READCLKSEL signal is driven by the user logic and is part of the DDR memory controller IP. It selects one of the four phase options between the SCLK and the signal DQSW90 to start the READ process.

The DDR Clock Polarity (DDRCLKPOL) signal is generated based on the phase of SCLK at the first DQS transition. If the DQS transition happens when the SCLK is high, then DDRCLKPOL is high and the rising edge of SCLK is used to clock the data into the FPGA core. Otherwise, DDRCLKPOL is low and the falling edge of SCLK is used to clock the data into the FPGA core.

The DQSR90 is the 90° phase shift of the DQS_Clean signal based on the DQSDEL delay control code from the DQSDLL. It is distributed to the DQSR90 tree and is used by all the DDR memory incoming data, DQ, to clock the incoming data during the READ operation. The DQSW90 is the 90° phase shift of the SCLK. It is used as a select signal to serialize the outgoing data during the WRITE operation. The DQS buffer during WRITE operation has a 90° phase difference from the data, DQ.

BURSTDET is used during DDR memory READ operations for read pulse positioning optimization at the start of a READ process. A minimum burst length (BL) of 4 must be used in the training process at start up. This can be done with two consecutive BL2 (BL=2), even number of BL2, or any multiplication of BL4 (BL=4) or longer burst lengths. The BURSTDET signal is also used to perform periodic read position calibration during the READ operation. The DDR memory controller IP monitors the BURSTDET signal with necessary steps to ensure the optimized read pulse position is found.

The READ signal to the DQSBUF block is generated by the DDR memory controller IP. The READ signal goes high after the READ command to control the DDR-SDRAM. This should normally precede the DQS preamble by one cycle, but may overlap the trailing bits of a prior read cycle. The READ signal is used to activate an internal signal, DQS_ena within the shaded region of Figure 13.14. If the DQS_ena is activated within this region, BURSTDET goes high and the DQS_Clean signal is turned on. The READ signal has to last as long as the burst length.

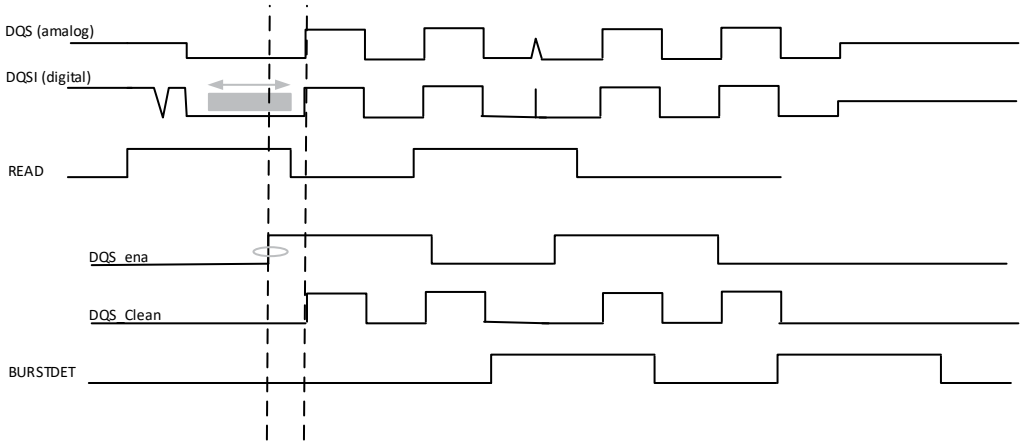


Figure 13.14. READ Pulse Positioning Optimization

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 DQS delay block when this update control (UDDCNTLN) input is asserted. Otherwise, the update is in the hold condition. The active low signal on UDDCNTLN updates the DQS phase alignment.

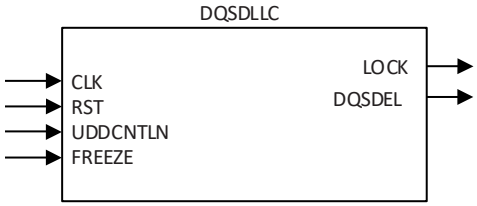


Figure 13.15. DQSDLLC Symbol

Table 13.4. 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 cells, connect to DQSDEL of the DQSBUFH element

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 DDR memory 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 freezes the DLL to save power while the delay code is preserved. When FREEZE goes low, it releases the DLL to resume operation. FREEZE must be applied to the DQSDLLC before the clock stops.

By default, this DLL generates 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. Refer to the MachXO2 Family Data Sheet (FPGA-DS-02056) for the range of frequencies when the FORCE_MAX_DELAY becomes effective.

Table 13.5. 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

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 times. The amount of the delay is determined by the software based on the type of interface implemented using the attribute DEL_MODE. Users 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 ~105ps 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 is incorrect and the performance of the DDR interface is not optimal. The DELAYE block is applicable to the receive mode of the DDR interfaces. It is available for all input register paths at all sides of a MachXO2 device.

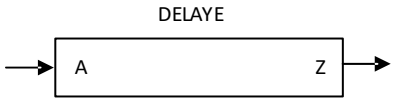


Figure 13.16. DELAYE Symbol

Table 13.6. DELAYE Signals

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

Table 13.7. 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 13.8. DEL_MODE Values Corresponding to the GDDR Interface

Interfaces 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

DELAYD

At the bottom side of the MachXO2-640U, MachXO2-1200/U and higher density devices, 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 ~105ps of delay.

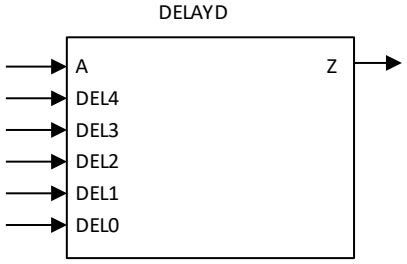


Figure 13.17. DELAYD Symbol

Table 13.9. DELAYD signals

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

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.

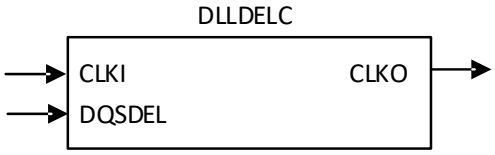


Figure 13.18. DLLDELC Symbol

Table 13.10. 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 1.9, June 2021

Section	Change Summary
All	Minor formatting.
Acronyms in This Document	Added this section.
Architecture for High-Speed Interfaces	• Updated Figure 2.3 to Figure 2.11. • Updated section content in Clock Domain Transfer at PIO Cells.
Generic High-Speed DDR Interfaces	• Updated Figure 5.4 to Figure 5.7. • Added content for Temporal Alignment in GDDRX2_RX.ECLK.Aligned, GDDRX2_RX.ECLK.Centered, GDDRX4_RX.ECLK.Aligned, and GDDRX4_RX.ECLK.Centered.

Revision 1.8, April 2020

Section	Change Summary
All	• Changed document number from TN1203 to FPGA-TN-02153. • Updated document template.
Disclaimers	Added this section.
Generic High-Speed DDR Interfaces	• Added note to Table 5.1. Generic High-Speed I/O DDR Interfaces. • Added note to the GDDRX4_RX.ECLK.Centered and the GDDRX4_TX.ECLK.Centered interface rules.
Using IPexpress to Build Generic High-Speed DDR Interfaces	• In Table 6.1. Signal Names Used by IPexpress Modules, removed 7:1 from eclk signal supported interfaces. • Updated Figure 6.7. GDDR_71 Configuration Tab of GDDR_71.
—	Updated links to referenced documents.

Revision 1.7, January 2014

Section	Change Summary
Architecture for High-Speed Interfaces	Updated Table 2.3., Signal Names Used by IPexpress Modules. Added information to signal name descriptions.
Generic High-Speed DDR Design Guidelines	Updated the Reset Synchronization Requirement section.

Revision 1.6, August 2013

Section	Change Summary
Using IPexpress to build Generic High-Speed DDR Interfaces	Updated the following tables: • User Interface Options for the Pre-Configuration Tab of DDR_Generic Modules • User Interface Options of the Configuration Tab of the DDR_Generic Modules
Technical Support Assistance	Updated Technical Support Assistance information.

Revision 1.5, April 2013

Section	Change Summary
DDR Software Primitives and Attributes	• Updated the Attribute for DQSDLLC table. • Updated delay information for DELAYE and DELAYD blocks

Revision 1.4, February 2012

Section	Change Summary
All	• Updated document with new corporate logo. • Document status changed from Preliminary to Final.

Revision 1.3, July 2011

Section	Change Summary
DDR Memory DQ/DQS Design Rules and Guidelines	Updated Generic High-Speed I/O DDR Interfaces table with footnote about migration from MachXO2-1200-R1 to Standard (non-R1) devices

Revision 1.2, April 2011

Section	Change Summary
Using IPexpress to build Generic High-Speed DDR Interfaces	Added signal names of IPexpress modules and timing analysis information.

Revision 1.1, January 2011

Section	Change Summary
All	Updated for ultra-high I/O (“U”) devices.

Revision 1.0, November 2010

Section	Change Summary
All	Initial release.

www.latticesemi.com