Lattice Nexus 2 High-Speed I/O and External Memory Interface User Guide

Preliminary Technical Note

FPGA-TN-02372-0.81

December 2024

Disclaimers

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

Inclusive Language

This document was created consistent with Lattice Semiconductor's inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice's inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

Contents

Figures

Tables

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviations	Definition
CA	Command and Address
CLKDIV	Edge Clock Dividers
DDR	Double Data Rate
DDRPHY	Double Data Rate Physical Layer
DLL	Delay-Locked Loop
DM	Data Mask
DQ	Data
DQS	Data Strobe
ECLK	Edge Clock
FPGA	Field Programmable Gate Array
HP	High Performance
IDDR	Input DDR
LPDDR	Low Power Double Data Rate
LVDS	Low Voltage Differential Signaling
MIPI	Mobile Industry Processor Interface
ODDR	Output DDR
ODT	On-Die Termination
PCB	Printed Circuit Board
PCLK	Primary Clock
PIC	Programmable I/O Cell
PIO	Programmable I/O
RX	Receiver
SCLK	System Clock
SDR	Single Data Rate
SDRAM	Synchronous Dynamic Random Access Memory
SGMII	Serial Gigabit Media-Independent Interface
SSN	Simultaneous Switching Noise
TX	Transmitter
VREF	Voltage Reference
WR	Wide Range
xSPI	Expanded Serial Peripheral Interface

1. Introduction

Lattice Nexus™ 2 FPGAs support high-speed I/O interfaces, including Single Data Rate (SDR) and Double Data Rate (DDR) interfaces, using the logic built into the Programmable I/O (PIO). In SDR applications, data is captured on one edge of the clock (either rising or falling), whereas in DDR applications, data is captured on both the rising and falling edges of the clock, thus doubling the performance. The Lattice Nexus 2 device also contains dedicated circuitry that is used along with the DDR I/O to support DDR3L, DDR4, and LPDDR4 external memory interfaces. This document discusses how to utilize the capabilities of the Lattice Nexus 2 devices to implement high-speed I/O interfaces.

1.1. Device Features and Support

The different types of high-speed I/O interfaces supported in Lattice Nexus 2 devices include the following:

- Single Data Rate (SDR) – the SDR Input/Output Module is designed to be used in a range of applications requiring fast data transmission up to 250 Mbps.
- Generic DDR (GDDR) – the GDDR Input/Output Module is designed to be used in a range of applications requiring high-speed data transfers up to 1,800 Mbps.
- GDDR 7:1 – the GDDR 7:1 Input/Output Module is designed for flat-panel display interfaces up to 1,050 Mbps.
- MIPI D-PHY – the MIPI D-PHY Interface Module is a physical serial data communication layer that supports protocols like CSI-2 (Camera Serial Interface-2) and DSI (Display Serial Interface). It physically connects a camera sensor or display interface to an application processor and supports up to 1,500 Mbps per lane.
- DDRPHY – the DDR PHY Interface Module is an implementation of the DFI 4.0 specification that defines the interface between a DDR memory controller and the physical interface (PHY). It supports interfacing with DDR3L, DDR4, and LPDDR4 external memories.
- Memory Controller – the Memory Controller IP provides a complete solution consisting of a controller, DDR PHY, and the associated clocking and training logic to interface with DDR3L, DDR4, and LPDDR4 external memories.

The following table summarizes the high-speed I/O interface support for Lattice Certus™-N2.

Table 1.1. Features Overview

High-Speed I/O Interface	Certus-N2
SDR	Yes
GDDR	Yes
GDDR 7:1	Yes
MIPI D-PHY	Yes
DDR3L (DDRPHY and Memory Controller)	Yes
DDR4 (DDRPHY and Memory Controller)	Yes
LPDDR4 (DDRPHY and Memory Controller)	Yes

***Note:** *Yes,* implies that the specified high-speed I/O interface is currently supported or will be in the future.

1.2. Interface Naming Conventions

This technical note references two types of external interfaces: centered and aligned.

- A centered external interface means that the clock is centered on the data window at the device pins.
- An aligned (also referred to as edge-on-edge) external interface means that the clock and data transitions are edge-aligned at the device pins.

The following figure shows a waveform representation for centered and aligned SDR/DDR interfaces.

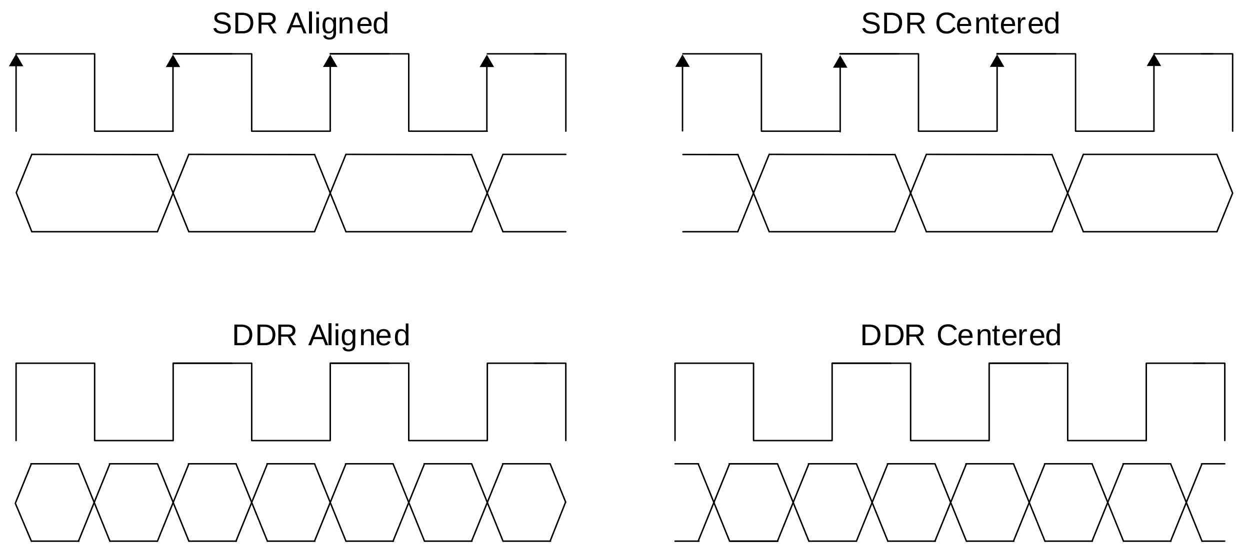


Figure 1.1. SDR and DDR Interface Definitions

2. High-Speed I/O Building Blocks

The Lattice Nexus 2 device contains dedicated functions for building high-speed interfaces, where each Lattice Nexus 2 device contains arrays of logic blocks that are arranged into Clock Regions (CKR). Each CKR is associated with an I/O bank, where the I/O bank is classified as either a Wide-Range I/O (WRIO) bank or a High-Performance I/O (HPIO) bank. The Lattice Nexus 2 device also contains a hardened PHY (DDRPHY), which provides a physical interface between the FPGA soft memory controller and external SDRAM to support DDR3L, DDR4, and LPDDR4 memory standards. Each Lattice DDRPHY consists of up to 3 HPIO banks. The following figure represents the Lattice Nexus 2 device floorplan in relation to building high-speed I/O interfaces.

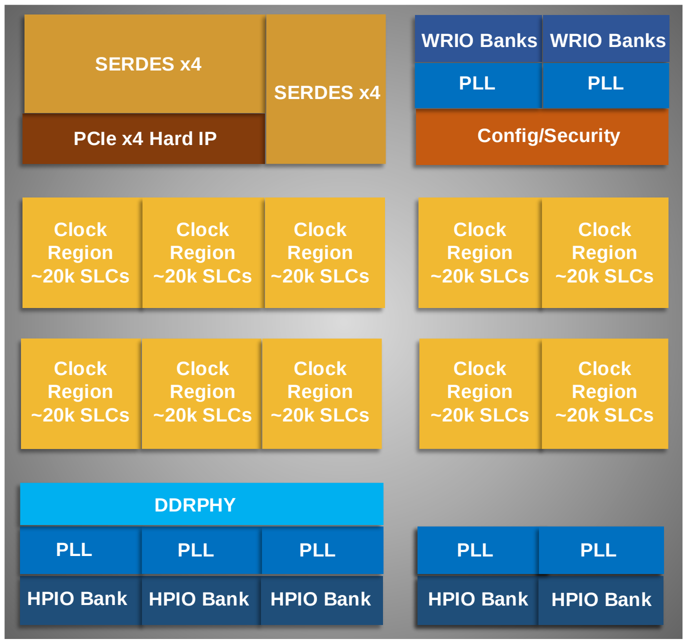


Figure 2.1. High-level Device Floorplan (LN2-CT20 Device)

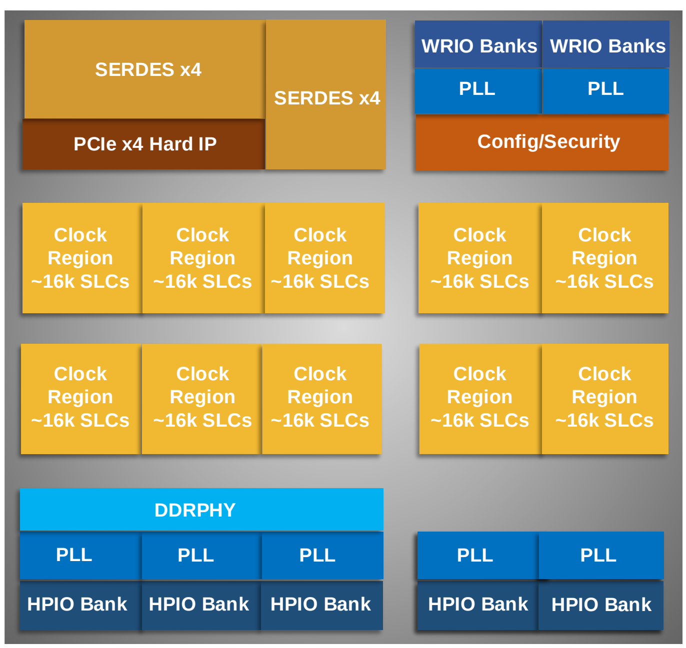


Figure 2.2. High-Level Device Floorplan (LN2-CT16 Device)

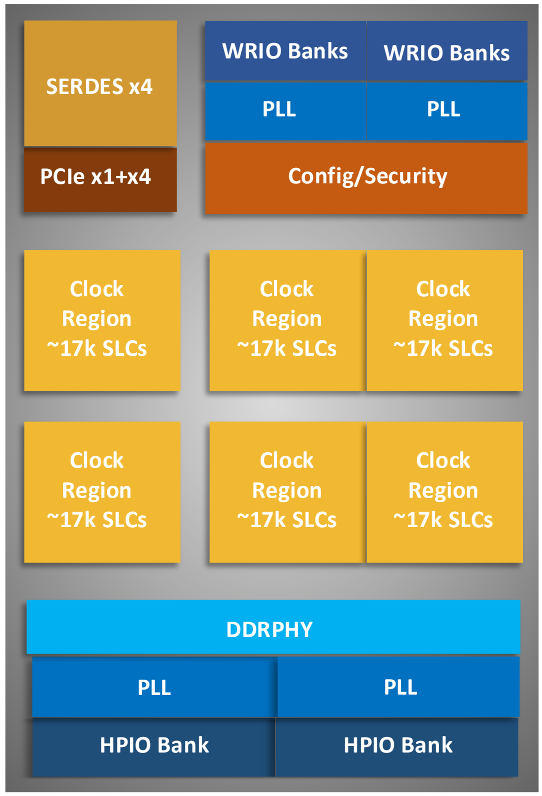


Figure 2.3. High-level Device Floorplan (LN2-CT10 Device)

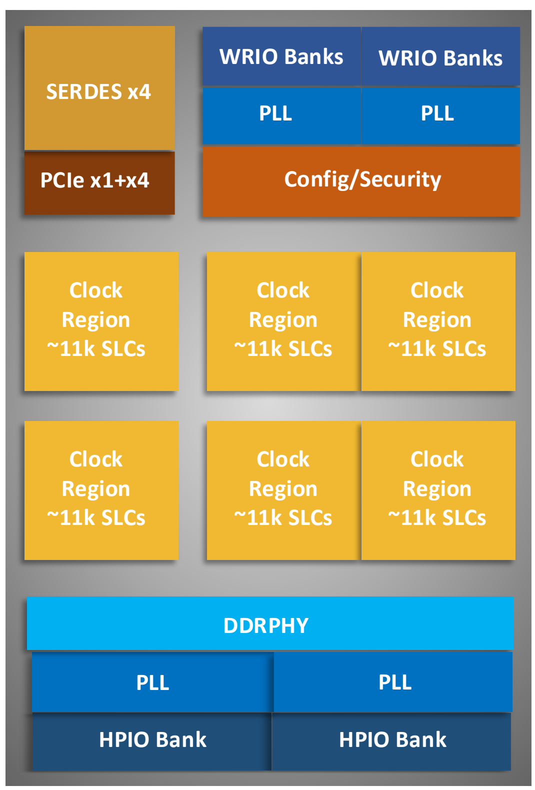


Figure 2.4. High-level Device Floorplan (LN2-CT06 Device)

2.1. I/O Banks

The Lattice Nexus 2 device supports two kinds of I/O banks: Wide-Range I/O (WRIO) and High-Performance I/O (HPIO).

2.1.1. Wide-Range I/O (WRIO)

Wide-Range I/O banks are located at the top of The Lattice Nexus 2 devices and contain the following capabilities:

- Supports I/O bank voltage ranging from 1.2 V to 3.3 V.
- Contains one PLL, one DDR Delay-Locked Loop (DDRDLL), and one DLL Delay (DLLDEL) primitive per WRIO bank.
- Supports one Expanded Serial Peripheral Interface (xSPI) clock tree for xSPI interfaces per WRIO bank.

For more information regarding WRIO banks and xSPI, refer to the Lattice Nexus 2 Platform Datasheet Overview (FPGA-TN-02122) and the Lattice Nexus 2 System Configuration User Guide (FPGA-TN-02370).

2.1.2. High-Performance I/O (HPIO)

PVT Calibration

All supported HPIO standards have calibration capability. High speed IO must be calibrated to meet performance requirements (DDR, LVDS, SLVS, and the like). Calibration is recommended, but optional for low-speed standards (LVCMOS).

High-Performance I/O banks are located at the bottom of The Lattice Nexus 2 devices and contain the following capabilities:

- Supports I/O bank voltage ranging from 0.9 V to 1.8 V.
- Supports LVDS, MIPI, and DDR3L/DDR4/LPDDR4 high-speed interfaces.
- Contains one PLL, one DDRDLL, and one DLLDEL primitive per HPIO bank.
- Supports Edge Clock (ECLK) and PHY Clock (PHYCLK) trees for DDRPHY and Memory Controller IP interfaces.

2.2. Clock Tree

The Lattice Nexus 2 Clock Network consists of four different clock structures:

- Global Clock Network (GCLK) – clock source that drives the Regional Clock Network (RCLK) of all clock regions.
- Regional Clock Network (RCLK) – clock source that drives all blocks (PFU, EBR, and DSP) within a clock region.
- Edge Clock Network (ECLK) – clock source for high-speed DDR I/O interfaces.
- PHY Clock Network (PHYCLK) – a high frequency clock to support DDRPHY interfaces within HPIO banks.

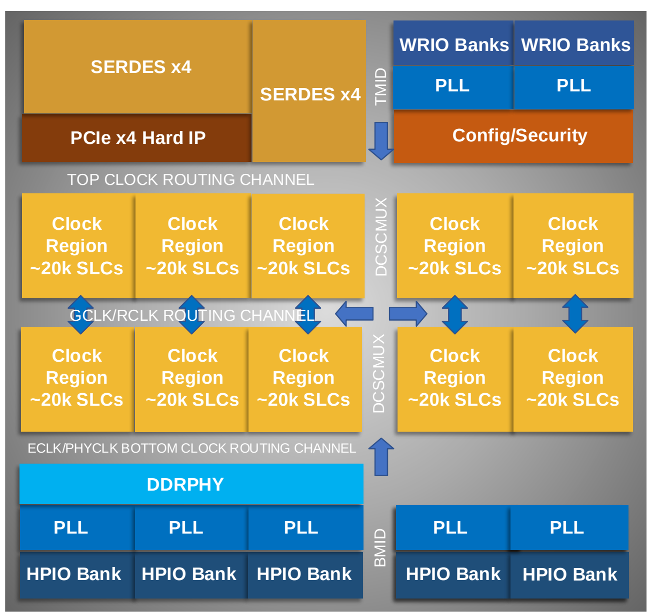


Figure 2.5. High-Level View of Clock Networks (Lattice Nexus 2)

Refer to the Lattice Nexus 2 System Clock PLL Design User Guide (FPGA-02364) for a complete description of the Lattice Nexus 2 device family clocking resources. The following sections provide a brief description of each of the major clock elements for building high-speed interfaces.

2.2.1. Primary Clocks

Primary Clocks (PCLK) are the system clocks of a design and originate from Dedicated Clock Pins (PCLKT pins) on the Lattice Nexus 2 device. The PCLK network is used to drive the System Clock (SCLK) of various I/O primitives.

2.2.2. Edge Clocks

Edge Clocks (ECLK) are high-speed, low-skew I/O dedicated clocks that provide a divided-down clock for DDR interfaces. A HPIO bank contains four ECLKs, where each ECLK network is made up of the following blocks:

- Edge Clock Input Multiplexer (ECLKINMUX)
- Edge Clock Synchronizer and Divider (ECLKSYNCDIV)
- Edge Clock Bridge Multiplexer (ECLKBRGMUX)

Refer to the Lattice Nexus 2 System Clock PLL Design User Guide (FPGA-02364) for additional details on the ECLK network.

2.2.2.1. Edge Clock Input Multiplexer (ECLKINMUX)

The ECLKINMUX determines which source is used to drive the high-speed I/O interface of each HPIO bank. The possible clock inputs to the ECLKINMUX can originate from the following elements:

- Output of a HPIO PLL
- Fabric clock (from GCLK)
- Output of the DLLDEL primitive
- Dedicated clock pins (PCLKT pins)

2.2.2.2. Edge Clock Synchronizer and Divider (ECLKSYNC and ECLKDIV)

The ECLKSYNC provides synchronization capabilities for each ECLK. It allows designers to dynamically enable or disable a clock and its associated logic. There are five ECLKSYNC modules available in each HPIO bank (four for ECLK and one for PHYCLK). The ECLKDIV generates a divided down clock with a x1,x2,x3.5,x4, or x5 dividing ratio that is used for DDR I/O domain crossing.

2.2.2.3. Edge Clock Bridge Multiplexer (ECLKBRGMUX)

The ECLKBRGMUX enables clocks in the current region to drive adjacent clock regions to form a wider clock. The four ECLK structures of each HPIO bank can bridge to adjacent HPIO banks (left and right) to form a wider ECLK network.

2.2.3. PHY Clocks (PHYCLK)

PHY Clocks (PHYCLK) are a special output of the HPIO PLL that is designed to support SGMII TX and DDR memory interfaces up to 2.4 GHz. There is only one PHYCLK per HPIO bank, where each PHYCLK network is made up of the following blocks:

- Edge Clock Synchronizer and Divider (ECLKSYNC and ECLKDIV)
- PHY Clock Bridge Multiplexer (PHYCLKBRGMUX)

Refer to the Lattice Nexus 2 System Clock PLL Design User Guide (FPGA-02364) for additional details on the PHYCLK network.

2.2.3.1. PHY Clock Bridge Multiplexer (PHYCLKBRGMUX)

The PHYCLKBRGMUX enables clocks from one clock region to drive adjacent (left and right) clock regions to support wider interface buses.

2.3. Phase-Locked Loop (PLL)

The PLL provides frequency synthesis with additional static and dynamic phase adjustments. There is one PLL in each HPIO bank, and there are seven output ports provided: CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5, and CLKOPHY. All seven outputs have the same set of dividers. Refer to the Lattice Nexus 2 System Clock PLL Design User Guide (FPGA-TN-02364) for additional information on PLLs.

2.4. DDR Delay-Locked Loop (DDRDLL)

The DDRDLL is a dedicated DLL for creating a 90-degree clock delay that is provided to each of the DLL Delay (DLLDEL) primitives within each HPIO bank. This enables data synchronization between the output data and the system clock. There is one DDRDLL in each HPIO bank.

2.5. DLL Delay (DLLDEL)

The DLLDEL provides a phase shift on the receive side clocks for each SCLK or ECLK. This shifts the clock input before it drives the clock tree by the delay set in the DDRDLL delay code. The DLLDEL primitive allows further adjustment of the delay set by the DDRDLL code for margin test purposes. The adjusted phase can be dynamic or static controlled.

- In dynamic control mode, the delay code comes directly from the associated DDRDLL.
- In static control mode, the delay code is set by software.

There are four DLLDEL primitives in each HPIO bank, where each is associated with an ECLK. There is only one DLLDEL delay code that is provided to all the DLLDEL modules within an HPIO bank, where the delay element inside the DLLDEL can be bypassed if it is not used. Each DLLDEL primitives consist of:

- A LOAD control signal that asynchronously resets the delay setting to the original DLLDEL primitive.
- A MOVE control pulse that changes the delay setting by ±1 tap each time according to the DIRECTION value. Note that MOVE needs to meet a 6 ns minimum pulse width requirement.
- A DIRECTION signal that:
  - When set to 0, increases the setting by 1 tap.
  - When set to 1, it decreases the setting by 1 tap.
- A COUT flag is asserted when the delay setting reaches the minimum value of 0 or the maximum value of 511 (9-bit CONTROL) to indicate underflow or overflow. In this scenario, the delay setting does not roll over, even if MOVE is pulsed.

Refer to the DLLDELA User Primitive section of this technical note for more information on the DLLDEL primitive.

2.6. Input DDR (IDDR)

The input DDR function can be used in the ×1 (2:1), ×2 (4:1), ×4 (8:1), x71 (7:1), and x5 (10:1) gearing modes.

- In ×1 gearing mode, 1-bit of DDR data is received, and the output is 2-bits of parallel data synchronized to SCLK. There is no clock domain transfer involved in x1 gearing.
- In x2 gearing mode, 1-bit of DDR data is received and synchronized to ECLK, and the output is 4-bit of parallel data synchronized to SCLK.
- In x4 gearing mode, the same function applies as in x2 gearing, except the output is 8-bit of parallel data.
- In x71 gearing mode, the same function applies as in x2 and x4 gearing, except the output is 7-bit of parallel data.
- x5 gearing mode is used for SGMII and Clock Data Recovery (CDR) applications. Refer to the SGMII and Gb Ethernet PCS IP User Guide for more information.

2.7. Output DDR (ODDR)

The output DDR function can also be supported in ×1 (2:1), ×2 (4:1), ×4 (8:1), x71 (7:1), or x5 (10:1) gearing modes.

- In ×1 gearing mode, the ODDR element receives 2-bit wide data from the FPGA fabric and generates a 1-bit DDR data or clock output.
- In x2 gearing mode, the ODDR element receives 4-bit wide data from the FPGA fabric and generates a 1-bit DDR data or clock output. ECLK is used to clock the output data for generic high-speed interfaces, and the Data Strobe (DQS) is used to clock the output data for DDR3L, DDR4, and LPDDR4 memory interfaces.
- In x4 gearing mode, the same function applies as in x2 gearing, except the ODDR element receives 8-bit wide data.
- In x71 gearing mode, the same function applies as in x2 and x4 gearing, except the ODDR element receives 7-bit wide data and that ECLK is used to clock the output data.
- x5 gearing mode is used for SGMII and Clock Data Recovery (CDR) applications. Refer to the SGMII and Gb Ethernet PCS IP User Guide for more information.

2.8. Input/Output DELAY

In addition to the DLLDEL primitive, there are three different delay cells within the PIC that can be utilized to fine-tune the data path to compensate for clock injection delay:

- DELAYD – can be used as either an input delay or an output delay for the SCLK interface with a dynamic control delay.
- DELAYE – can be used as either an input delay or an output delay for the SCLK interface with a static delay.
- DELAYF – can be used as either an input delay or an output delay for the ECLK interface with a dynamic control delay.

These delay primitives allow 512 steps of delay, where each delay step generates 12.5 ps of delay. You can overwrite the delay setting dynamically using the MOVE and DIRECTION control inputs, where the MOVE input on the DELAYD and DELAYF primitives needs to meet a 6 ns minimum pulse width requirement. The LOADN input resets the delay back to the default value. Refer to the Input/Output Delay User Primitives section of this technical note for more information on the DELAYD, DELAYE, and DELAYF primitives.

2.9. Hardened DDR Physical Layer (DDRPHY)

The DDRPHY Module is an implementation of the DDR PHY Interface (DFI) v4.0 Specification, which is an interface protocol between memory controller logic and PHY interfaces. A Lattice Nexus 2 device can contain up to 3 DDRPHY interfaces, where each DDRPHY is composed of up to 3 HPIO banks. Each DDRPHY can support one of the following external memory interfaces: DDR3L, DDR4, and LPDDR4.

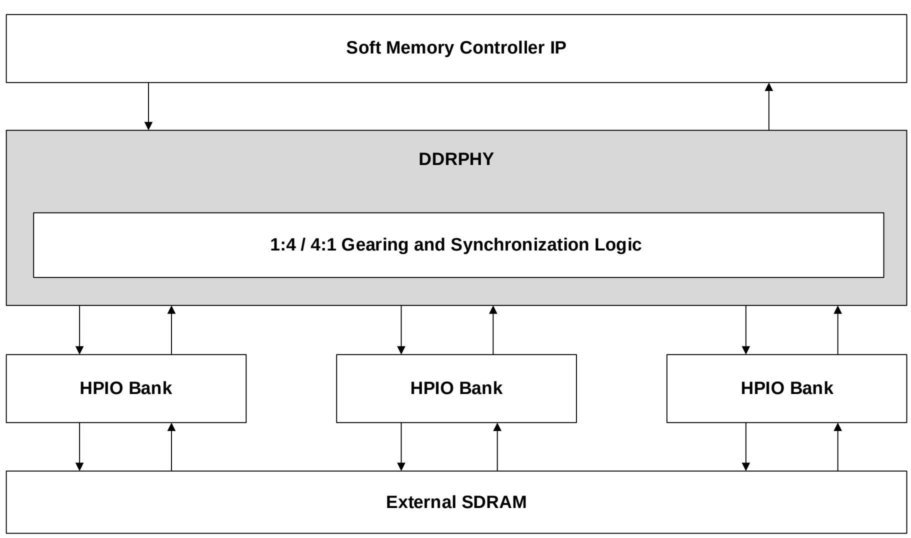


Figure 2.6. Lattice Nexus 2 DDRPHY Block Diagram

The DDRPHY contains built-in 1:4 and 4:1 gearing and synchronization logic to handle clock domain transfers among three different clock domains:

- 2,400 MHz PHYCLK
- 600 MHz DFICLK
- 300 MHz PCLK

2.9.1. Memory Interface Lanes

Each HPIO bank within a DDRPHY consists of 4 lanes, where these lanes are partitioned either as DQ/DQS lanes or CA lanes.

- Each HPIO bank within a DDRPHY has 52 I/O, with each lane containing 12 pins.
- Each DQ/DQS lane provides a fixed clock pair (DQS/DQSN) for the Data Strobe (DQS).
- Each data group for a memory interface occupies 11 pins in total: 8 Data (DQ), 1 Data Mask (DM), and dedicated DQS_P/DQS_N.
- The number of pins occupied for command, address, and control signals varies depending on the memory interface.

Table 2.1. Memory Interface CA Pin Count

Hardened DDRPHY Port	DDR3L	DDR4	LPDDR4
mem_address[15:0]	A15 – A0	{BG[1], ACT_n, A[13:0]}	{Unused[9:0], CA_A[5:0]} / {Unused[9:0], CA_B[5:0]}
mem_bank[2:0]	BA2 – BA0	{BG[0], BA[1:0]}	Unused
mem_ras_n	RAS#	RAS_n / A16	Unused
mem_cas_n	CAS#	CAS_n / A15	Unused
mem_we_n	WE#	WE_n / A14	Unused
mem_rst_n	RESET#	RESET_n	RESET_n
mem_cs_n[#RANKS-1:0]	CS#	CS_n	CS_A / CS_B
mem_odt[#RANKS-1:0]	ODT	ODT	Unused[2]
mem_cke	CKE	CKE	CKE_A / CKE_B
mem_address[16]	Unused	PAR	Unused
alert_n[1]	Unused	ALERT	Unused

Notes:

1. alert_n is supported by GPIO, not through the DDRPHY block.
2. ODT is tied off on the board for both channels for LPDDR4.

For more information regarding the placement of external memory interface signals, refer to the External Memory Interfaces Pin and Resource Planning section of this technical note.

3. Generic High-Speed I/O Interfaces

This section describes each of the generic high-speed interfaces and associated clocking. The following table summarizes the different types of high-speed I/O interfaces supported in the Lattice Nexus 2 devices.

Table 3.1. Types of Generic High-Speed I/O Interfaces

Mode	Interface Name	Description
Receive SDR	GIREG_RX.SCLK	SDR input register using SCLK. Supports bypassed, static, and dynamic data path delays and optional clock inversion.
RX DDRX1 Centered	GDDRX1_RX.SCLK.Centered	DDR x1 input register using SCLK. Supports bypassed, static, and dynamic data path delays and dynamic clock path delays. Clock is centered to data window.
RX DDRX1 Aligned	GDDRX1_RX.SCLK.Aligned	DDR x1 input register using SCLK. Supports bypassed, static, and dynamic data path delays and dynamic clock path delays. Data is edge-to-edge with clock.
RX DDRX2 Centered	GDDRX2_RX.ECLK.Centered	DDR x2 input register using ECLK. Supports bypassed, static, and dynamic data path delays and dynamic clock path delays. Clock is centered to data window.
RX DDRX2 Aligned	GDDRX2_RX.ECLK.Aligned	DDR x2 input register using ECLK. Supports bypassed, static, and dynamic data path delays and dynamic clock path delays. Data is edge-to-edge with clock.
RX DDRX4 Centered	GDDRX4_RX.ECLK.Centered	DDR x4 input register using ECLK. Supports bypassed, static, and dynamic data path delays and dynamic clock path delays. Clock is centered to data window.
RX DDRX4 Aligned	GDDRX4_RX.ECLK.Aligned	DDR x4 input register using ECLK. Supports bypassed, static, and dynamic data path delays and dynamic clock path delays. Data is edge-to-edge with clock.
RX DDRX5 Centered	GDDRX5_RX.ECLK.Centered	DDR x5 input register using ECLK. Supports bypassed, static, and dynamic data path delays and dynamic clock path delays. Clock is centered to data window.
RX DDRX5 Aligned	GDDRX5_RX.ECLK.Aligned	DDR x5 input register using ECLK. Supports bypassed, static, and dynamic data path delays and dynamic clock path delays. Data is edge-to-edge with clock.
RX DDR71	GDDR71_RX.ECLK	DDR 7:1 input register using ECLK. Supports bypassed and dynamic data path delays, and optional bit and word alignment.
RX DDRX4 MIPI	GDDRX4_RX.MIPI	DDR x4 input register using ECLK-to-MIPI interface.
Transmit SDR	GOREG_TX.SCLK	SDR output register using SCLK. Supports bypassed, static, and dynamic data path delays, and optional tri-state control.
TX DDRX1 Centered	GDDRX1_TX.SCLK.Centered	DDR x1 input register using SCLK. Supports bypassed, static, and dynamic data path delays, and optional tri-state control. Clock is centered to data window.
TX DDRX1 Aligned	GDDRX1_TX.SCLK.Aligned	DDR x1 output register using SCLK. Supports bypassed, static, and dynamic data path delays, and optional tri-state control. Data is edge-to-edge with clock.
TX DDRX2 Centered	GDDRX2_TX.ECLK.Centered	DDR x2 input register using ECLK. Supports bypassed, static, and dynamic data path delays, and optional tri-state control. Clock is centered to data window.
TX DDRX2 Aligned	GDDRX2_TX.ECLK.Aligned	DDR x2 output register using ECLK. Supports bypassed, static, and dynamic data path delays, and optional tri-state control. Data is edge-to-edge with clock.
TX DDRX4 Centered	GDDRX4_TX.ECLK.Centered	DDR x4 input register using ECLK. Supports bypassed, static, and dynamic data path delays, and optional tri-state control. Clock is centered to data window.

Mode	Interface Name	Description
TX DDRX4 Aligned	GDDRX4_TX.ECLK.Aligned	DDR x4 output register using ECLK. Supports bypassed, static, and dynamic data path delays, including tri-state control. Data is edge-to-edge with clock.
TX DDRX5 Centered	GDDRX5_TX.ECLK.Centered	DDR x5 input register using ECLK. Supports bypassed, static, and dynamic data path delays, including tri-state control. Clock is centered to data window.
TX DDRX5 Aligned	GDDRX5_TX.ECLK.Aligned	DDR x5 output register using ECLK. Supports bypassed, static, and dynamic data path delays, including tri-state control. Data is edge-to-edge with clock.
TX DDR71	GDDR71_TX.ECLK	DDR 7:1 output register using ECLK. Supports bypassed and dynamic data path delays.
TX DDRX4 MIPI	GDDRX4_TX.MIPI	DDR x4 output register using ECLK-to-MIPI interface.

The following describes the naming conventions used for each of the interfaces listed in Table 3.1.

- G – Generic
- IREG – SDR Input I/O Register
- OREG – SDR Output I/O Register
- DDRX1 – DDR x1 gearing I/O Register
- DDRX2 – DDR x2 gearing I/O Register
- DDRX4 – DDR x4 gearing I/O Register
- DDRX5 – DDR x5 gearing I/O Register
- DDRX71 – DDR 7:1 gearing I/O Register
- _RX – Receiver Interface
- _TX – Transmit Interface
- .ECLK – Uses Edge Clock resource
- .SCLK – Uses Primary Clock resource
- .Centered – Clock is centered to data when coming into device.

3.1. Generic SDR Receive Interfaces

The Lattice Semiconductor Single Data Rate Input/Output (SDR I/O) Module provides receive interfaces up to 250 Mbps and supports up to 256-bit data bus widths with single-ended or differential signaling.

3.1.1. GIREG_RX.SCLK Bypass Delay

This section describes the Generic SDR Receive interface with bypass delay. These interfaces can be used for speeds up to 250 MHz. This SDR interface uses SCLK and is made up of the following modules:

- The IFD1P3DX primitive is a positive edge-triggered register that is used to capture the data.
- The input clock can be optionally inverted (via the INV primitive) and is routed through the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.

Figure 3.1 shows the bypass data delay option for the Generic SDR Receive interface.

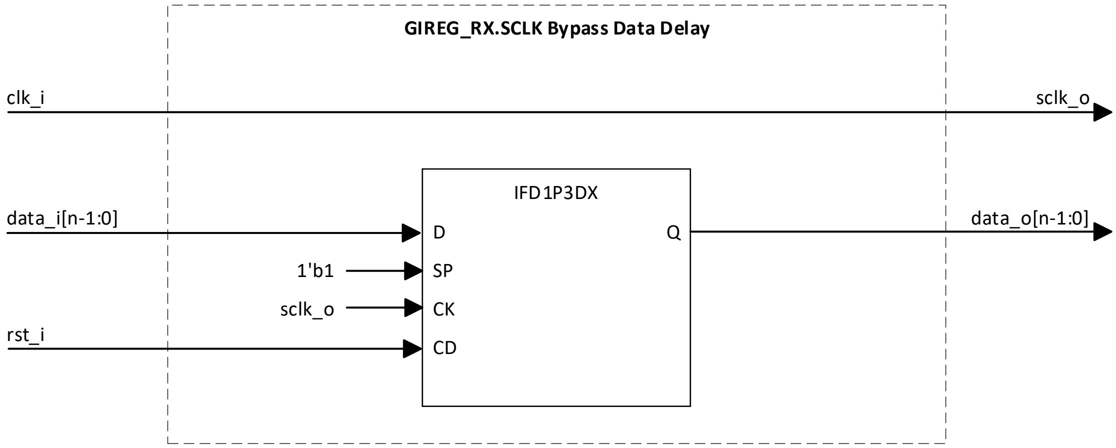


Figure 3.1. GIREG_RX.SCLK (Bypass Data Delay)

3.1.2. GIREG_RX.SCLK Static Default/User-Defined Delay

This section describes the Generic SDR Receive interface with static default/user-defined delays. These interfaces can be used at speeds up to 250 MHz. This SDR interface uses SCLK and is made up of the following modules:

- The IFD1P3DX primitive is a positive edge-triggered register that is used to capture the data.
- The static delay primitive DELAYE is used to delay the incoming data to remove clock injection time. The type of delay required can be selected through the Module/IP Block Wizard.
- The DEL_MODE attribute is used with the DELAYE primitive to indicate the interface type so that the correct delay value can be set in the delay element. SCLK_ZEROHOLD is used for static default delay mode, and USER_DEFINED is used for static user-defined delay mode.
- The input clock can be optionally inverted (via the INV primitive) and is routed through the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.

Figure 3.2 shows the static default data delay option for this interface.

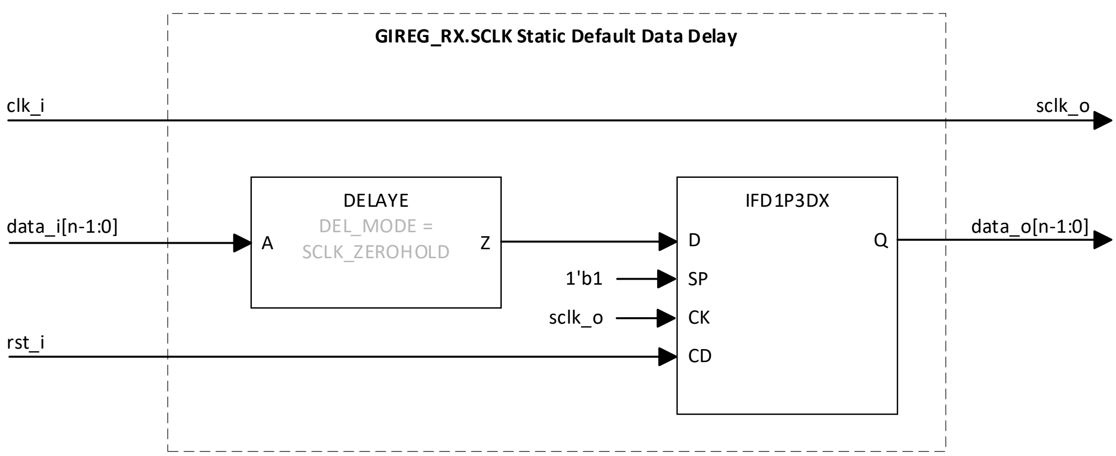


Figure 3.2. GIREG_RX.SCLK (Static Default Data Delay)

3.1.3. GIREG_RX.SCLK Dynamic Default/User-Defined Delay

This section describes the Generic SDR Receive interface with dynamic default/user-defined delays. These interfaces can be used for speeds up to 250 MHz. This SDR interface uses SCLK and is made up of the following modules:

- The IFD1P3DX primitive is a positive edge-triggered register that is used to capture the data.
- The dynamic delay primitive DELAYD is used to control the delay on the data dynamically. DELAYD allows you to override the set input delay through the Module/IP Block Wizard. Note that the MOVE input on the DELAYD primitive needs to meet a 6 ns minimum pulse width requirement.
- The DEL_MODE attribute is used with the DELAYD primitive to indicate the interface type so that the correct delay value can be set in the delay element. SCLK_ZEROHOLD is used for dynamic default delay mode, and USER_DEFINED is used for dynamic user-defined delay mode.
- The input clock can be optionally inverted (via the INV primitive) and is routed through the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.

Figure 3.3 shows the dynamic default data delay option for this interface.

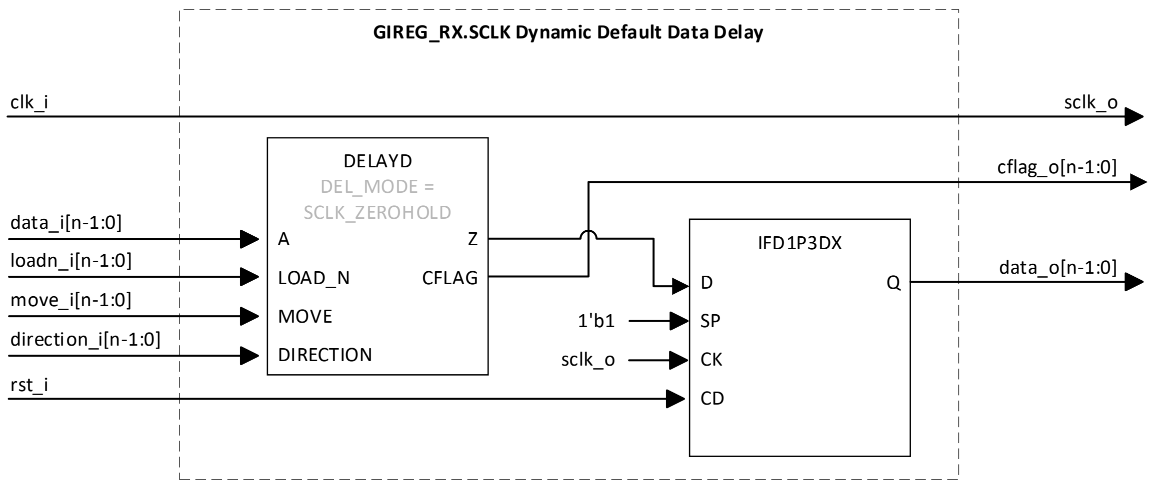


Figure 3.3. GIREG_RX.SCLK (Dynamic Default Data Delay)

3.2. Generic DDR Receive Interfaces

The Lattice Semiconductor Generic Double Data Rate Input/Output (GDDR I/O) Module provides receive interfaces up to 1,800 Mbps and supports up to 256-bit data bus widths with x1, x2, x4, and x5 gearing. The Lattice Semiconductor Generic Double Data Rate 7:1 Input/Output (GDDR 7:1 I/O) Module provides receive interfaces up to 1,050 Mbps and supports up to 16-bit data bus widths.

3.2.1. GDDRX1_RX.SCLK.Centered

This section describes the Generic DDR 1x Centered Receive interface with bypass, static default/user-defined delays, and dynamic default/user-defined delays. These interfaces can be used for speeds up to 300 MHz. This DDR interface uses SCLK and is made up of the following modules:

- The IDDRX1A primitive is a register that is used to capture the data both at the positive (Q0) and negative (Q1) edges of the clock.
- The static delay primitive DELAYE is used in bypass and static default/user-defined delay modes to delay the incoming data to remove clock injection time. The type of delay required can be selected through the Module/IP Block Wizard.
- The dynamic delay primitive DELAYD is used in dynamic default/user-defined delay modes. DELAYD allows you to override the set input delay on the data dynamically through the Module/IP Block Wizard. Note that the MOVE input on the DELAYD primitive needs to meet a 6 ns minimum pulse width requirement.

- The DEL_MODE attribute is used with the DELAYE and DELAYD primitives to indicate the interface type so that the correct delay value can be set in the delay element. SCLK_CENTERED is used for bypass and static default delay modes, and USER_DEFINED is used for static user-defined and dynamic default/user-defined delay modes.
- The input clock must use a PCLK input so it can be routed directly to the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.

Figure 3.4 shows the bypass and static default delay options for this interface. Figure 3.5 shows the dynamic default/user-defined delay options for this interface.

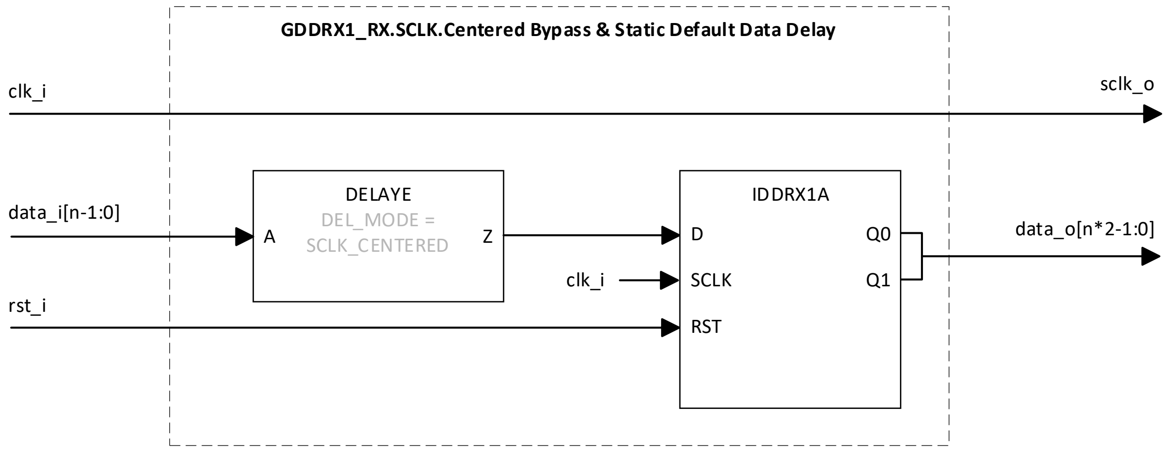


Figure 3.4. GDDRX1_RX.SCLK.Centered (Bypass and Static Default Data Delay)

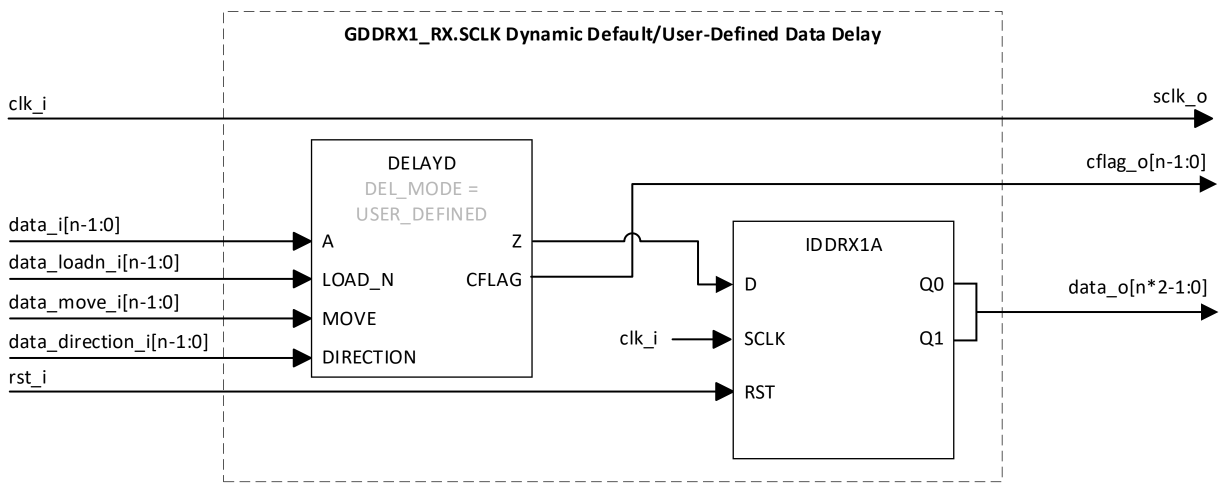


Figure 3.5. GDDRX1_RX.SCLK.Centered (Dynamic Default/User-Defined Data Delay)

3.2.2. GDDRX1_RX.SCLK.Aligned

This section describes the Generic DDR 1x Aligned Receive interface with bypass, static default/user-defined delays, and dynamic default/user-defined delays. These interfaces can be used for speeds up to 300 MHz. This DDR interface uses SCLK and is made up of the following modules:

- The IDDRX1A primitive is a register that is used to capture the data both at the positive (Q0) and negative (Q1) edges of the clock.
- The static delay primitive DELAYE is used in bypass and static default/user-defined delay modes to delay the incoming data to remove clock injection time. The type of delay required can be selected through the Module/IP Block Wizard.

- The dynamic delay primitive DELAYD is used in dynamic default/user-defined delay modes. DELAYD allows you to override the set input delay on the data dynamically through the Module/IP Block Wizard. Note that the MOVE input on the DELAYD primitive needs to meet a 6 ns minimum pulse width requirement.
- The DEL_MODE attribute is used with the DELAYE and DELAYD primitives to indicate the interface type so that the correct delay value can be set in the delay element. SCLK_ALIGNED is used for bypass and static default delay modes, and USER_DEFINED is used for static user-defined and dynamic default/user-defined delay modes.
- The input clock must use a PCLK input so it can be routed directly to the DLLDELA and DLLDELA primitives to be phase-shifted and routed to primary (SCLK) tree. CFLAG output for DLLDELA is left unconnected for bypass and static default/user-defined delay modes with fixed clock delay.
- The DLLDELA input signals are controlled via an internal delay component that triggers DLLDELA initialization and delay counter. This internal delay component is controlled via the DEL_INIT_START output of RX_SYNC and signals completion of the DLLDELA delay counter via the DEL_INIT_DONE input of RX_SYNC. In dynamic default/user-defined delay modes, the DDRDLLA primitive provides the delay code to the DLLDELA inputs to adjust the delay dynamically. Note that the MOVE input on the DLLDELA primitive needs to meet a 6ns minimum pulse width requirement.
- The RX_SYNC input reset is an active-high asynchronous reset.
- The receiver synchronization soft IP (RX_SYNC) is required and automatically generated for aligned interfaces to ensure the DLLDELA primitive does not update until the DDRDLLA primitive is locked. STOP output for RX_SYNC is left unconnected for all delay modes.

Figure 3.6 and Figure 3.7 show the bypass and static default delay modes for this interface. Figure 3.8 shows the dynamic default/user-defined delay options for this interface.

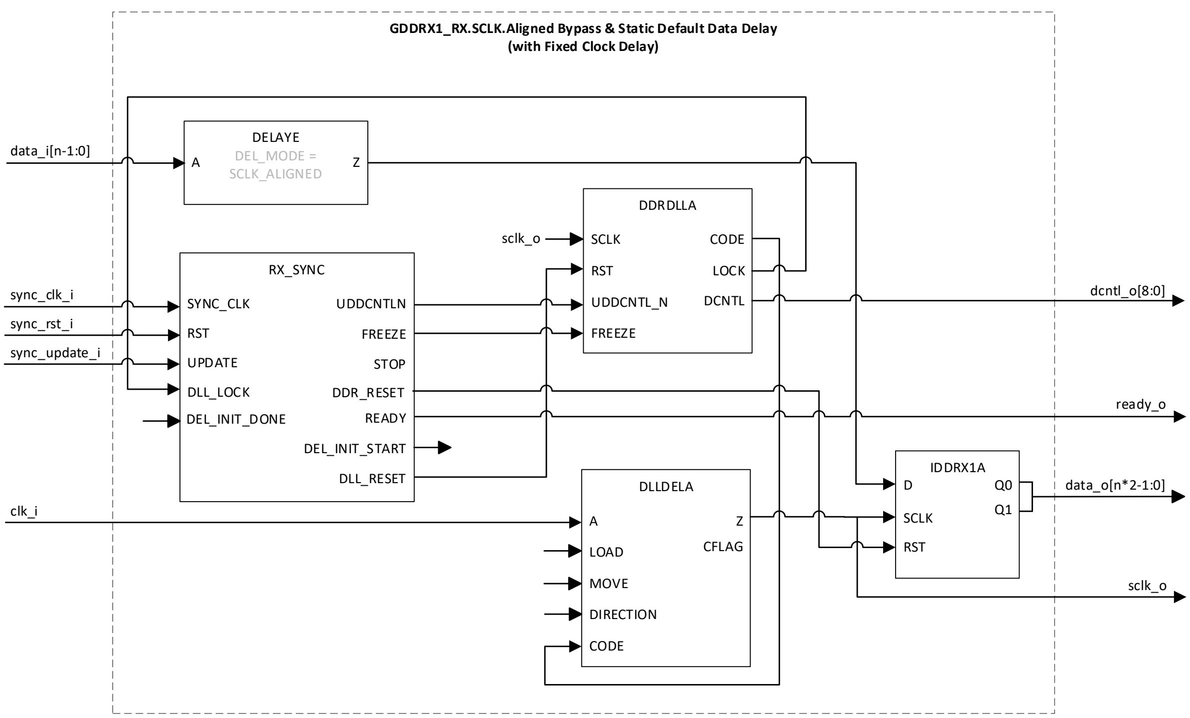


Figure 3.6. GDDRX1_RX.SCLK.Aligned (Bypass and Static Default Data Delay with Fixed Clock Delay)

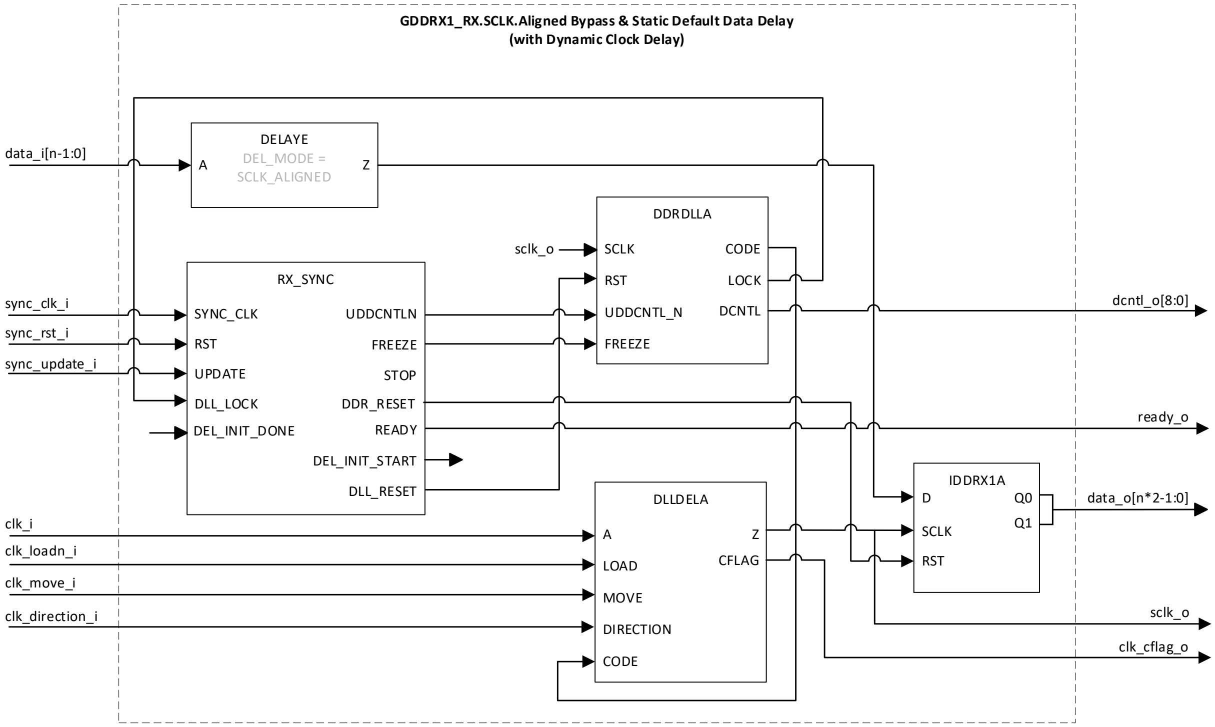


Figure 3.7. GDDRX1_RX.SCLK.Aligned (Bypass and Static Default Delay with Dynamic Clock Delay)

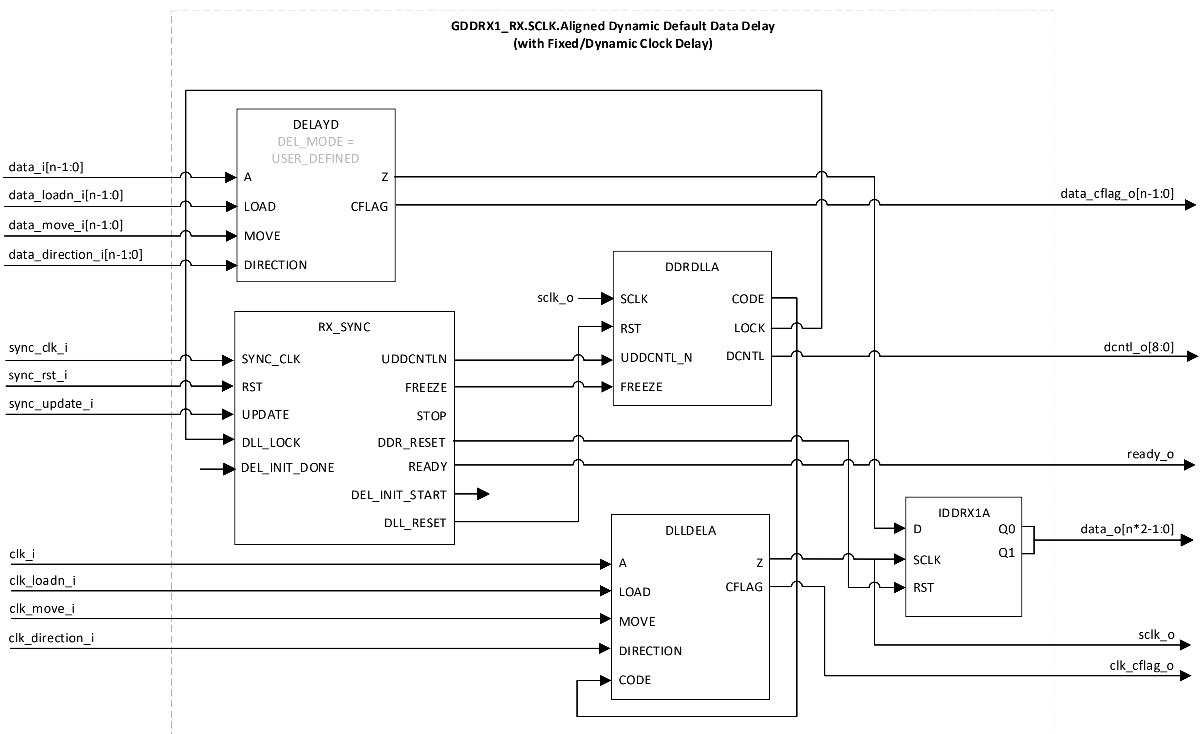


Figure 3.8. GDDRX1_RX.SCLK.Aligned (Dynamic Default Data Delay with Fixed/Dynamic Clock Delay)

3.2.3. GDDRX2_RX.ECLK.Centered/GDDRX4_RX.ECLK.Centered/GDDRX5_RX.ECLK.Centered

This section describes the Generic DDR 2x/4x/5x Centered Receive interface with bypass, static default/user-defined delays, and dynamic default/user-defined delays. The Generic 2x interfaces can be used for speeds up to 600 MHz and the Generic 4x/5x interfaces can be used for speeds up to 900 MHz. These DDR interfaces use ECLK and are made up of the following modules:

- The IDDRX2A/IDDRX4A/IDDRX5A primitives are registers that are used to capture data both at the positive (Q0/Q2/Q4/Q6/Q8) and negative (Q1/Q3/Q5/Q7/Q9) edges of the clock.
- The dynamic delay primitive DELAYF is used for all data delay modes. DELAYF allows you to override the set input delay on the data through the Module/IP Block Wizard. CFLAG output for DELAYF is left unconnected for bypass and static default/user-defined delay modes. Note that the MOVE input on the DELAYF primitive needs to meet a 6 ns minimum pulse width requirement.
- The DEL_MODE attribute is used with the DELAYF primitive to indicate the interface type so that the correct delay value can be set in the delay element. ECLK_CENTERED is used for bypass, static default, and dynamic default delay modes, and USER_DEFINED is used for static user-defined and dynamic user-defined delay modes.
- The input clock must use PCLK input so it can be routed directly through the ECLKSYNCA module to the ECLK tree. The SCLK output of the ECLKDIVA module is routed through the primary (SCLK) clock tree.
- The GDDR_SYNC input reset is an active-high asynchronous reset.
- The synchronization soft IP (GDDR_SYNC) is required and automatically generated for bypass and static modes to account for the skew between the SYNC input to ECLKSYNCA, and the RST input to ECLKDIVA and IDDRX2A/IDDRX4A/IDDRX5A primitives.

Figure 3.9, Figure 3.10, and Figure 3.11 show the bypass and static default delay options for this interface. Figure 3.12, Figure 3.13, and Figure 3.14 show the dynamic default delay options for this interface.

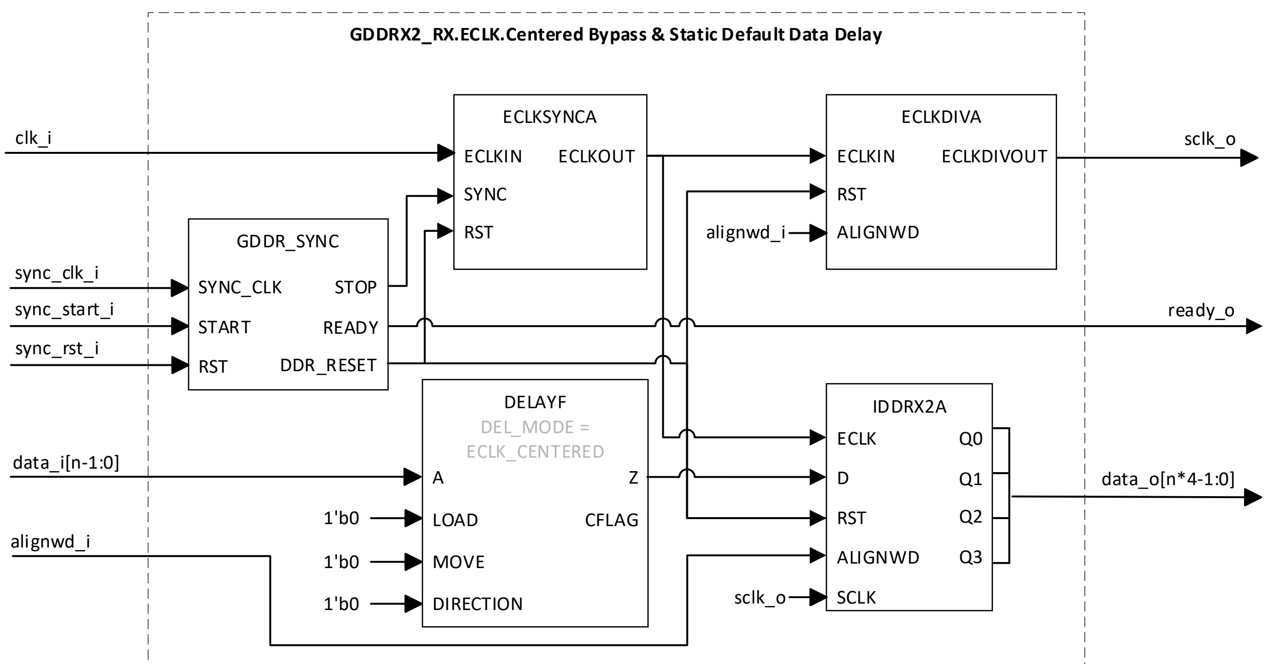


Figure 3.9. GDDRX2_RX.ECLK.Centered (Bypass and Static Default Data Delay)

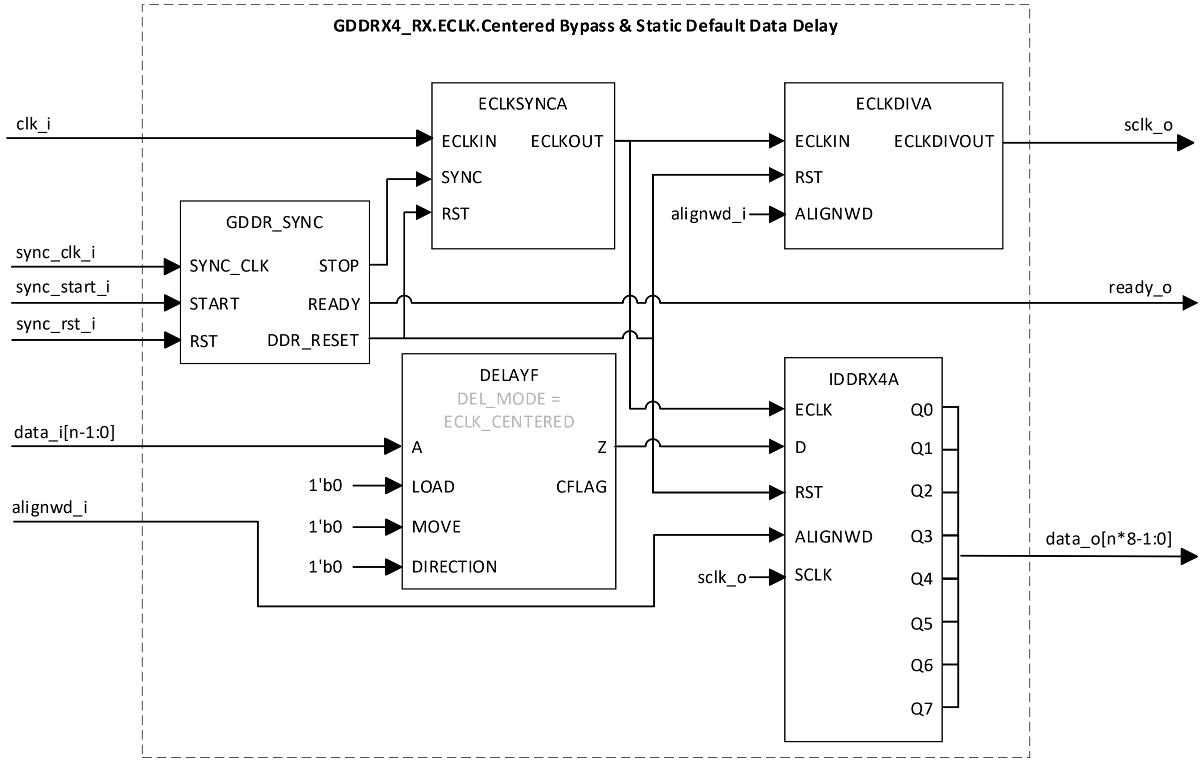


Figure 3.10. GDDRX4_RX.ECLK.Centered (Bypass and Static Default Data Delay)

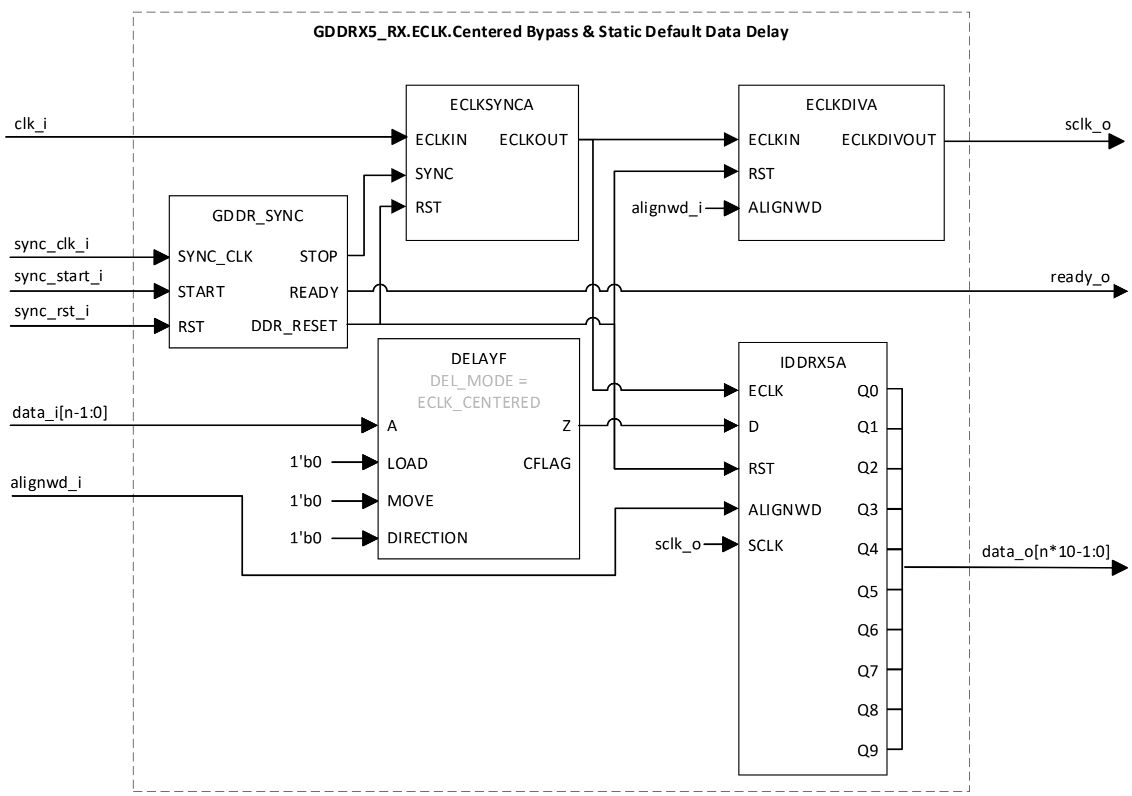


Figure 3.11. GDDRX5_RX.ECLK.Centered (Bypass and Static Default Data Delay)

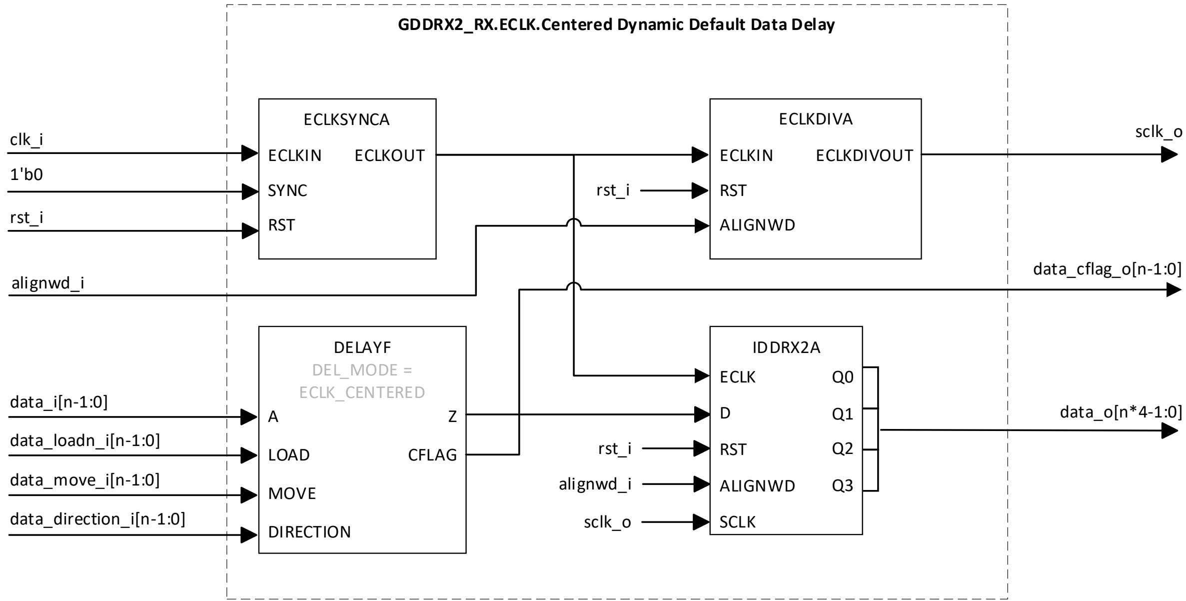


Figure 3.12. GDDRX2_RX.ECLK.Centered (Dynamic Default Data Delay)

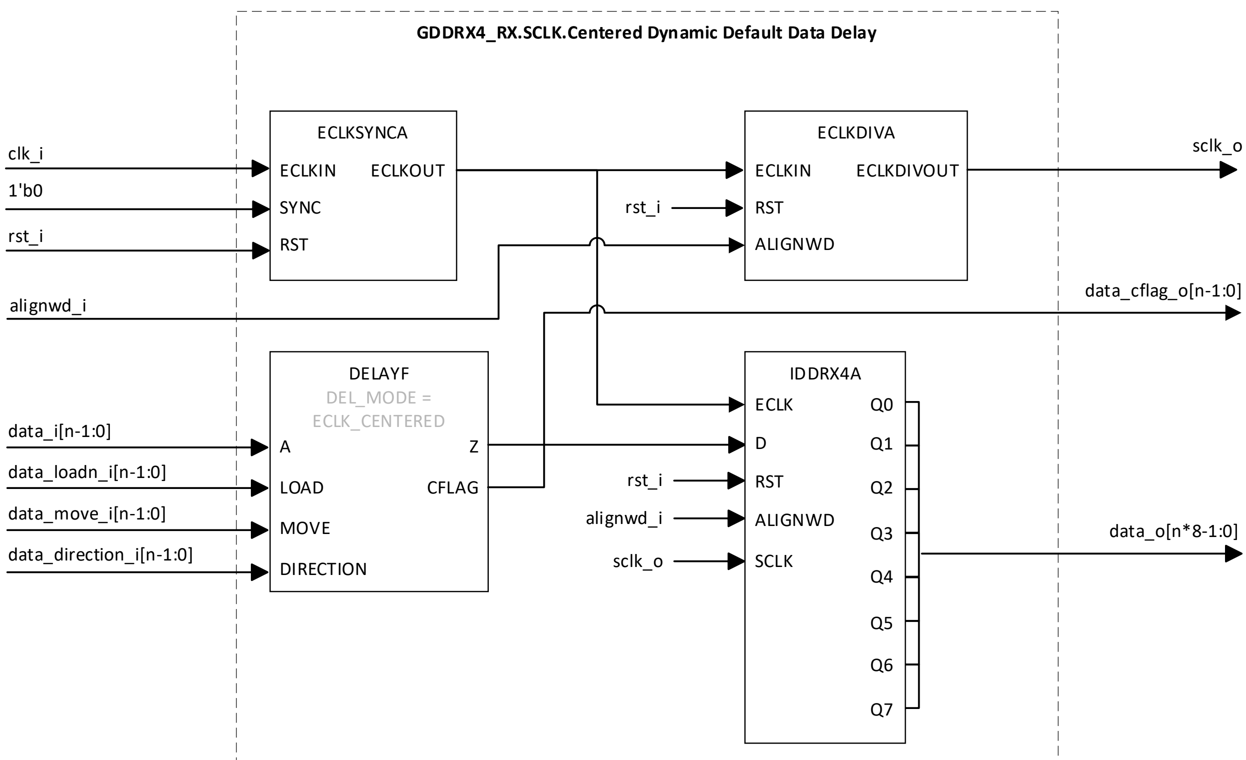


Figure 3.13. GDDRX4_RX.ECLK.Centered (Dynamic Default Data Delay)

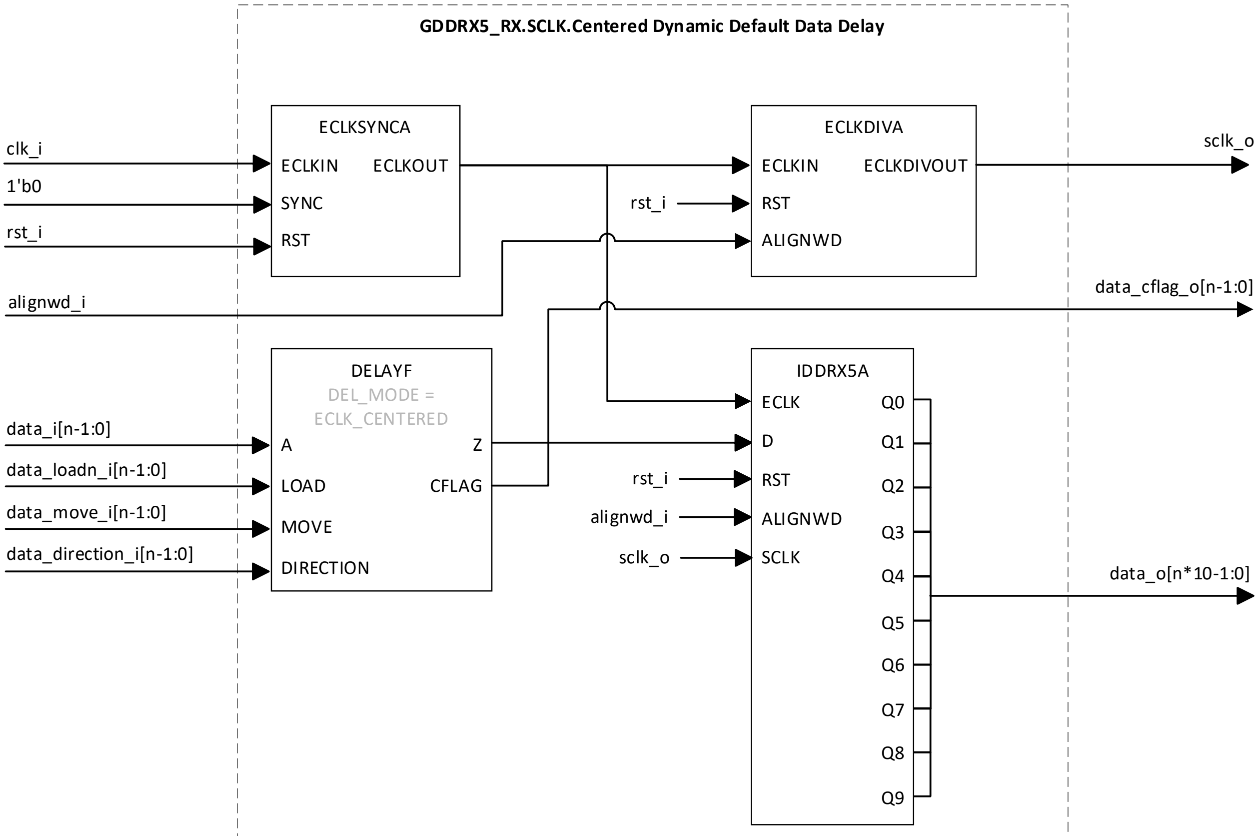


Figure 3.14. GDDRX5_RX.ECLK.Centered (Dynamic Default Data Delay)

3.2.4. GDDRX2_RX.ECLK.Aligned/GDDRX4_RX.ECLK.Aligned/GDDRX5_RX.ECLK.Aligned

This section describes the Generic DDR 2x/4x/5x Aligned Receive interface with bypass, static default/user-defined, and dynamic default/user-defined delays. The Generic 2x interfaces can be used for speeds up to 600 MHz and the Generic 4x/5x interfaces can be used for speeds up to 900 MHz. These DDR interfaces use ECLK and are made up of the following modules:

- The IDDRX2A/IDDRX4A/IDDRX5A primitives are registers that are used to capture data both at the positive (Q0/Q2/Q4/Q6/Q8) and negative (Q1/Q3/Q5/Q7/Q9) edges of the clock.
- The dynamic delay primitive DELAYF is used in bypass and static default/user-defined delay modes. DELAYF allows you to override the set input delay on the data through the Module/IP Block Wizard. CFLAG output for DELAYF is left unconnected for bypass and static default/user-defined delay modes. Note that the MOVE input on the DELAYF primitive needs to meet a 6 ns minimum pulse width requirement.
- The DEL_MODE attribute is used with the DELAYF primitive to indicate the interface type so that the correct delay value can be set in the delay element. ECLK_ALIGNED is used for bypass and static default delay modes, and USER_DEFINED is used for static user-defined and dynamic default/user-defined delay modes.
- The input clock must use a PCLK input so it can be routed directly to the DLLDELA and DLLDELA primitives to be phase-shifted and then routed through the ECLKSYNCA and ECLKDIVA modules to the ECLK tree. The clock is divided by 2,4 or 5. The SCLK output of the ECLKDIVA module is routed through the primary (SCLK) clock tree. CFLAG output for DLLDELA is left unconnected for bypass and static default/user-defined delay modes.

- In bypass and static default/user-defined delay modes, the DLLDELA input signals are controlled via an internal delay component that triggers DLLDELA initialization and a delay counter. This internal delay component is controlled via the DEL_INIT_START output of RX_SYNC and signals completion of the DLLDELA delay counter via the DEL_INIT_DONE input of RX_SYNC. In dynamic default/user-defined delay modes, the DDRDLLA primitive provides the delay code to the DLLDELA inputs to adjust the delay dynamically. Note that the MOVE input on the DLLDELA primitive needs to meet a 6 ns minimum pulse width requirement.
- The RX_SYNC input reset is an active-high asynchronous reset.
- The receiver synchronization soft IP (RX_SYNC) is required and automatically generated for aligned interfaces to ensure the DLLDELA primitive does not update until the DDRDLLA primitive is locked.

Figure 3.15, Figure 3.16, and Figure 3.17 show the bypass and static default delay options for this interface. Figure 3.18, Figure 3.19, and Figure 3.20 show the dynamic default delay options for this interface.

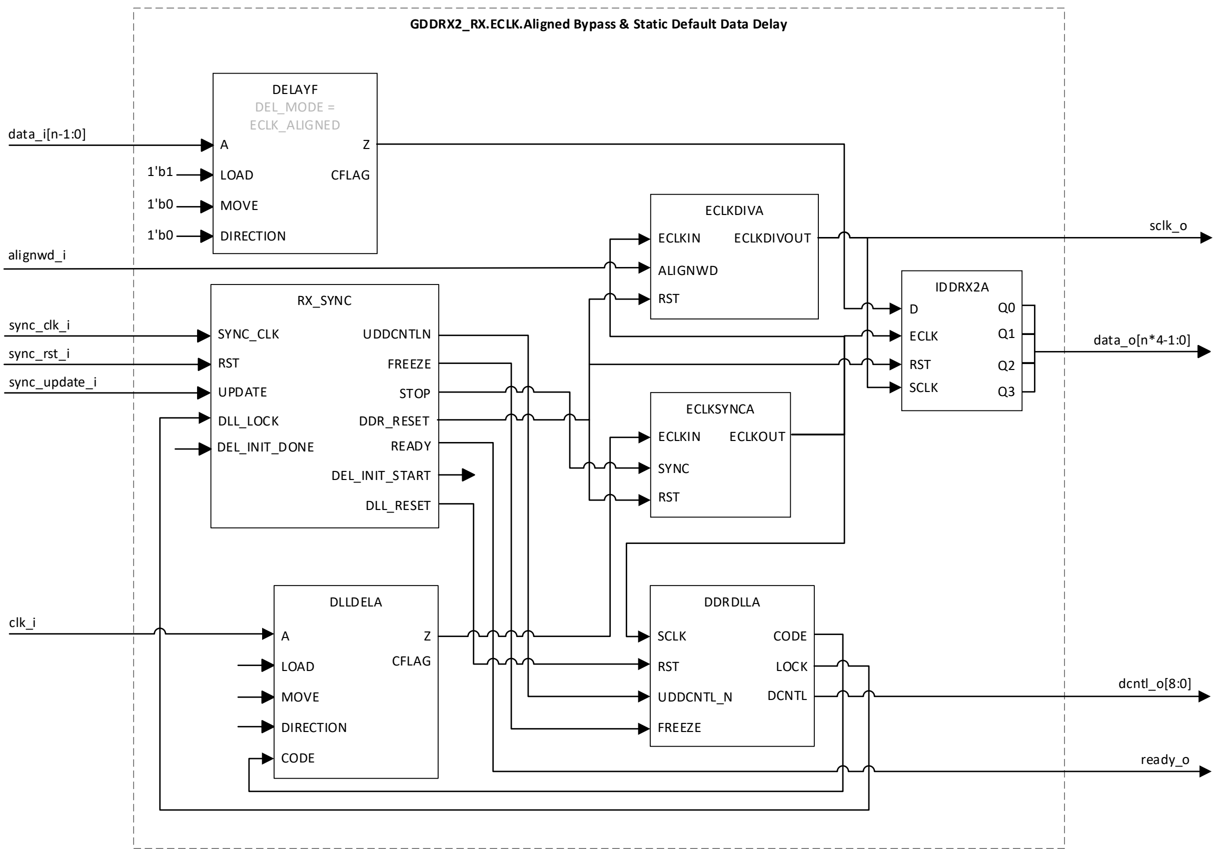


Figure 3.15. GDDRX2_RX.ECLK.Aligned (Bypass and Static Default Data Delay)

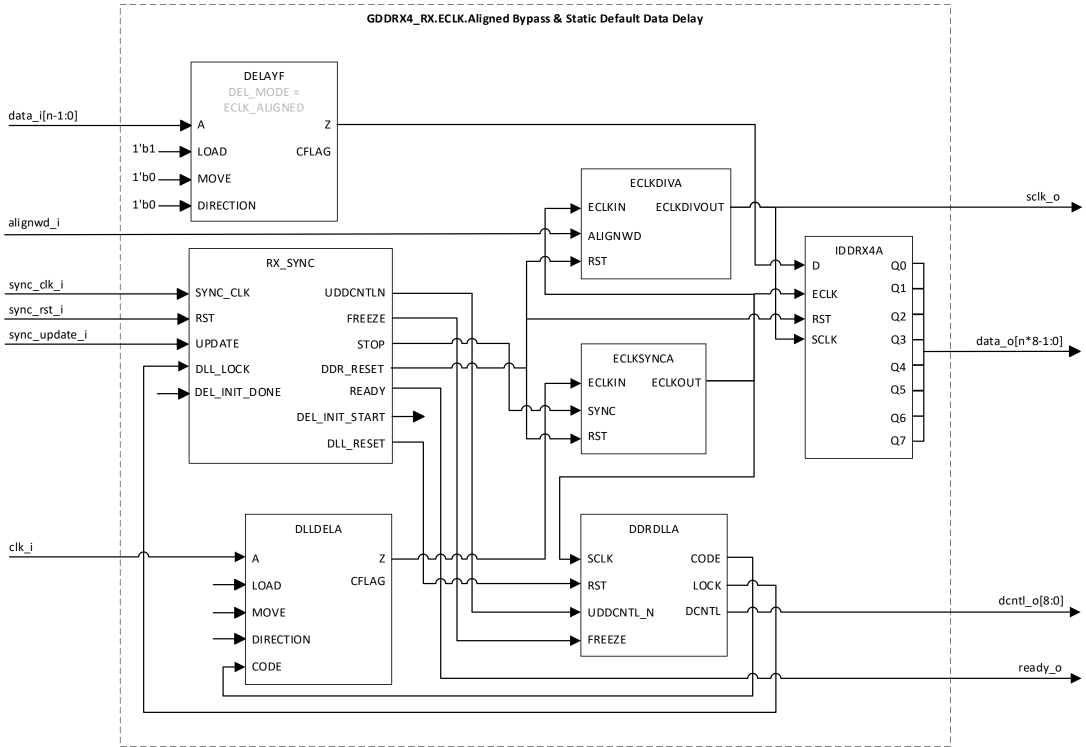


Figure 3.16. GDDRX4_RX.ECLK.Aligned (Bypass and Static Default Data Delay)

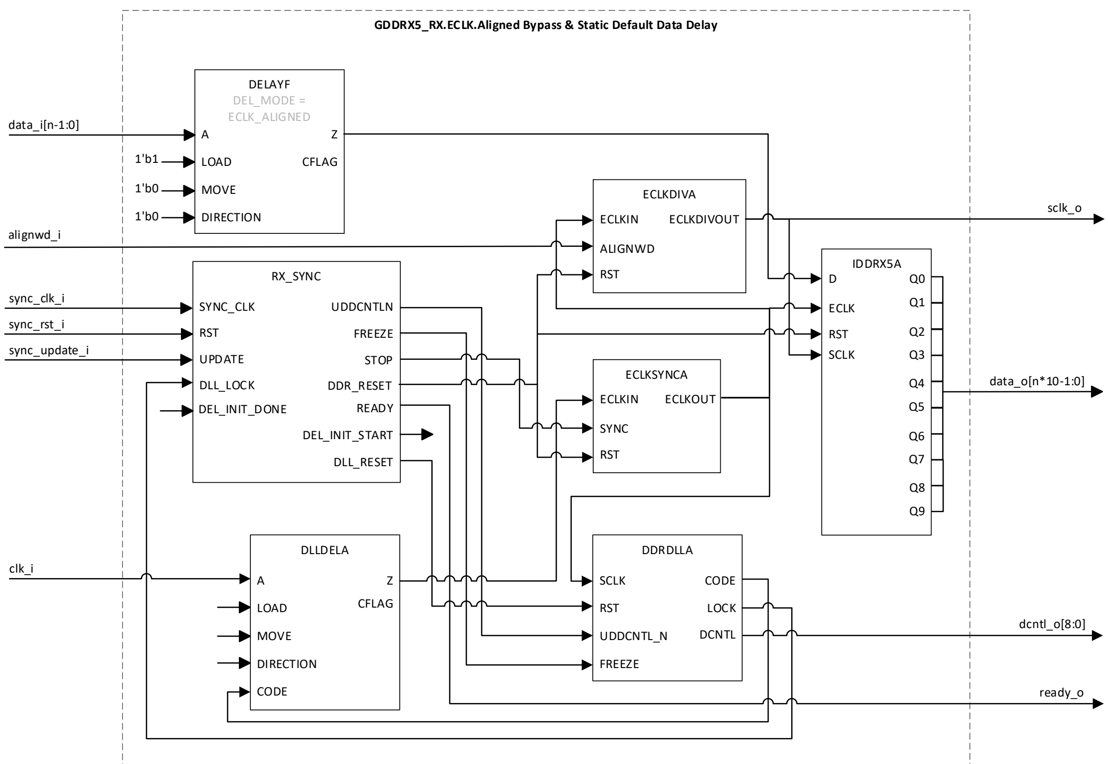


Figure 3.17. GDDRX5_RX.ECLK.Aligned (Bypass and Static Default Data Delay)

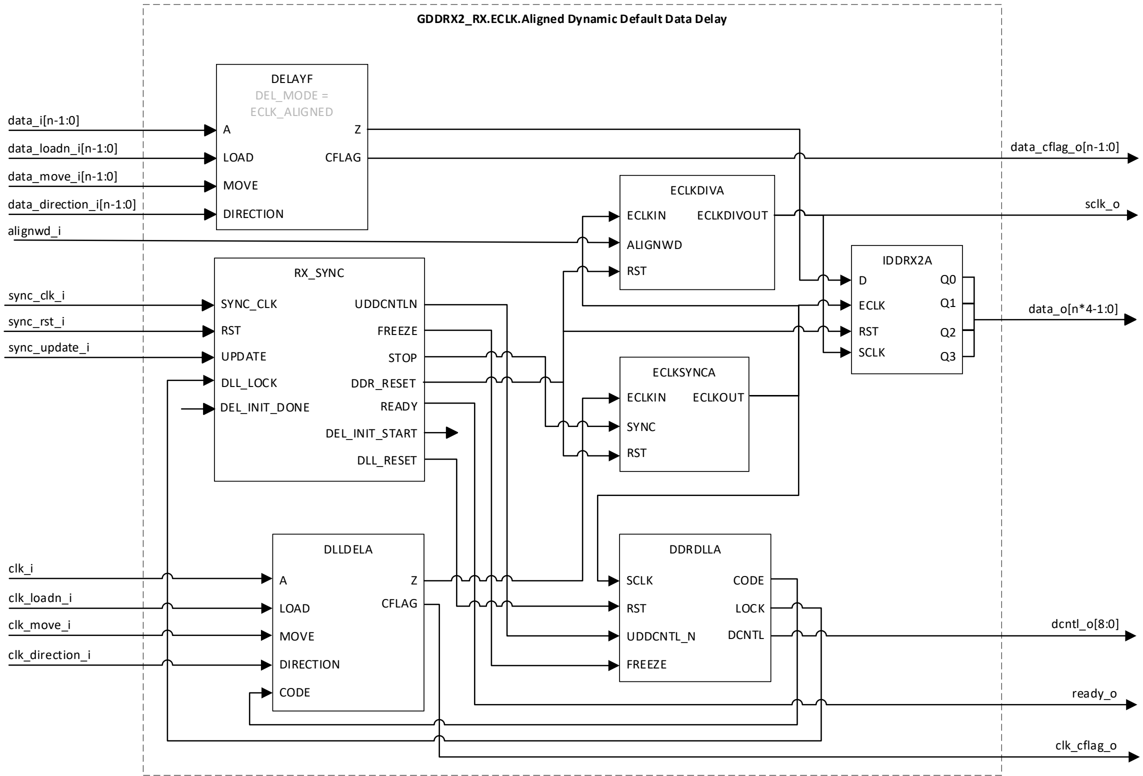


Figure 3.18. GDDRX2_RX.ECLK.Aligned (Dynamic Default Data Delay)

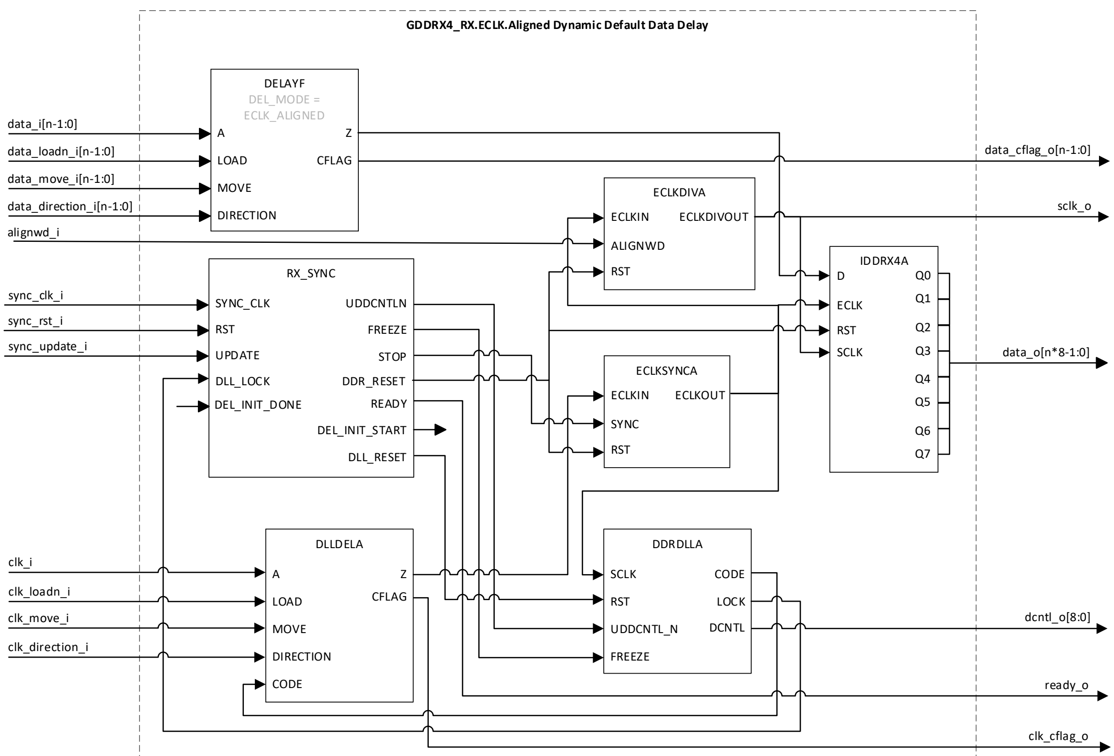


Figure 3.19. GDDRX4_RX.ECLK.Aligned (Dynamic Default Data Delay)

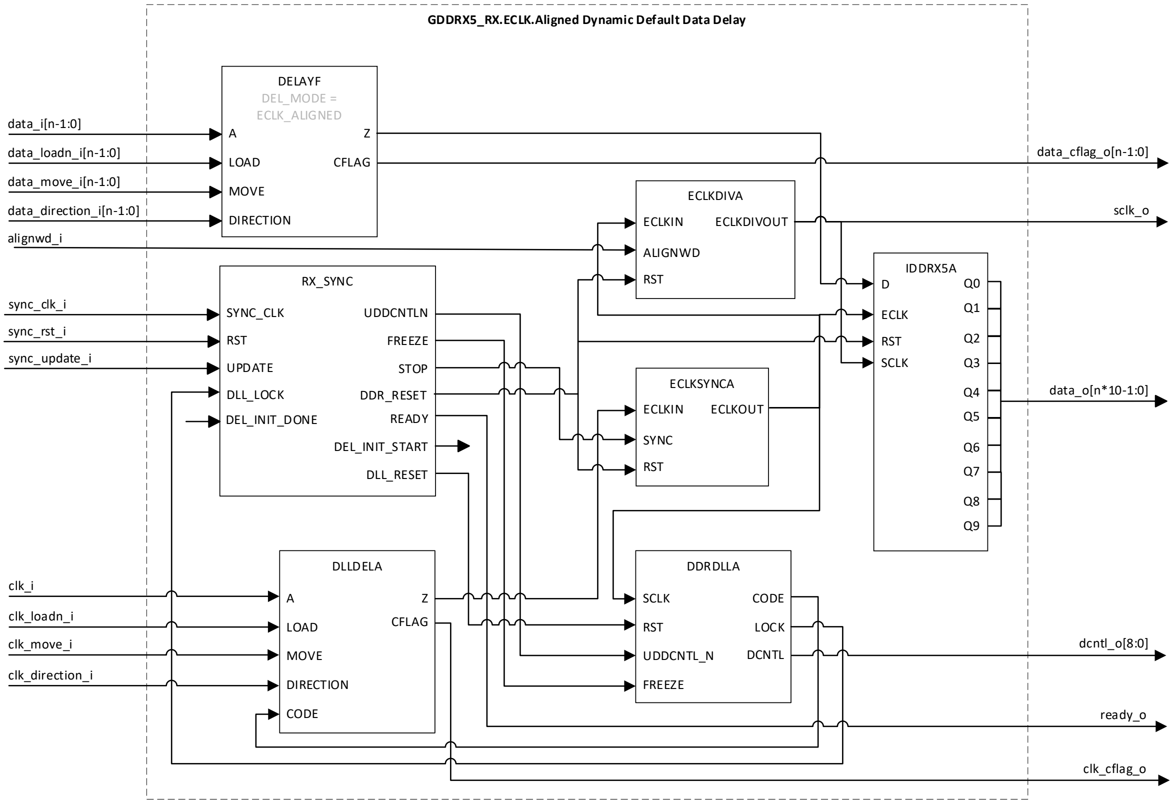


Figure 3.20. GDDRX5_RX.ECLK.Aligned (Dynamic Default Data Delay)

3.2.5. GDDR71_RX.ECLK

This section describes the Generic DDR 7:1 Receive interface with bit word alignment and data delay control. The Generic DDR 7:1 interface can be used for speeds up to 150 MHz. These DDR interfaces use ECLK and are made up of the following modules:

- The IDDR71A primitives are registers that are used to capture data both at the positive (Q0/Q2/Q4/Q6) and negative (Q1/Q3/Q5) edges of the clock.
- The dynamic delay primitive DELAYF is used when data delay control is enabled. DELAYF allows you to override the set input delay on the data through the Module/IP Block Wizard. The DEL_MODE attribute is used with the DELAYF primitive to indicate the interface type so that the correct delay value can be set in the delay element. USER_DEFINED is used for Generic DDR 7:1 receive interfaces. Note that the MOVE input on the DELAYF primitive needs to meet a 6 ns minimum pulse width requirement.
- The input clock is routed through the ECLKSYNCA and ECLKDIVA modules to the ECLK tree. The clock is divided by 3.5 and is center-aligned relative to the data. The SCLK output of the ECLKDIVA module is routed through the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.
- The synchronization soft IP (GDDR_SYNC) is required and automatically generated to account for the skew between the sync input to ECLKSYNCA and the reset input to ECLKDIVA and IDDR71A primitives.
- Bit and Word Alignment soft IP (BW_ALIGN) can be optionally enabled to align the PLL generated edge clock to the center of the DDR clock and data window. A second IDDR71A primitive is used with the data port connected to the clock input to generate a 7-bit clock phase for bit and word alignment.

Figure 3.21 shows the default option for this interface (without BW_ALIGN and data delay control). Figure 3.22 shows the BW Align option for this interface, and Figure 3.23 shows the BW Align and data delay control options for this interface.

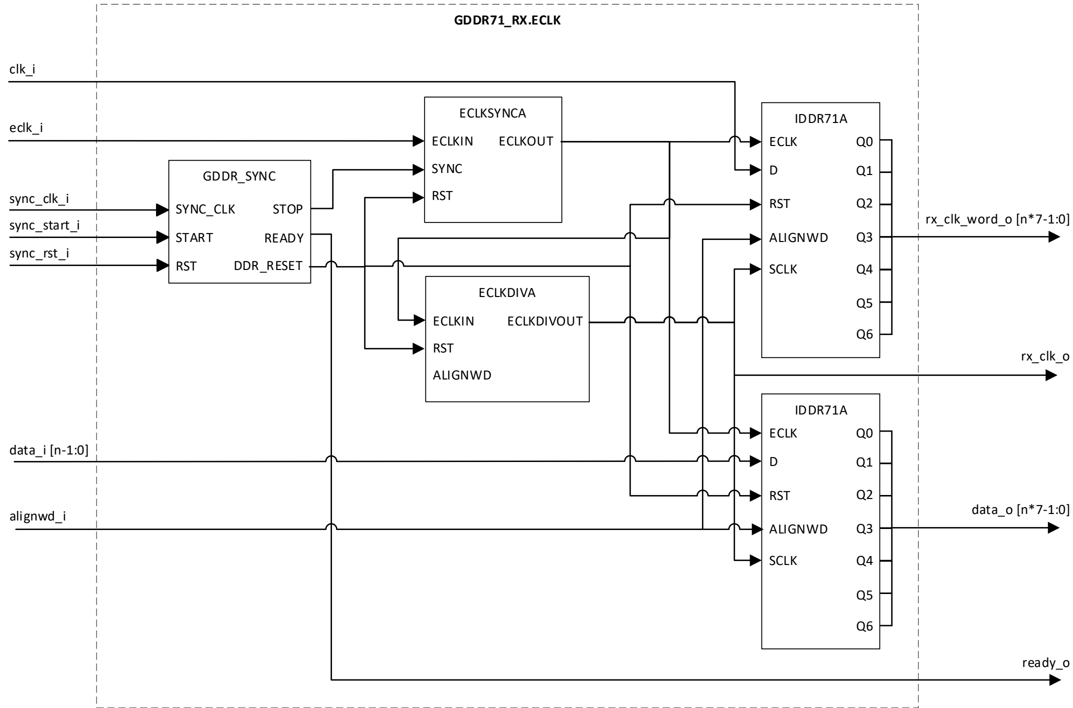


Figure 3.21. GDDR71_RX.ECLK

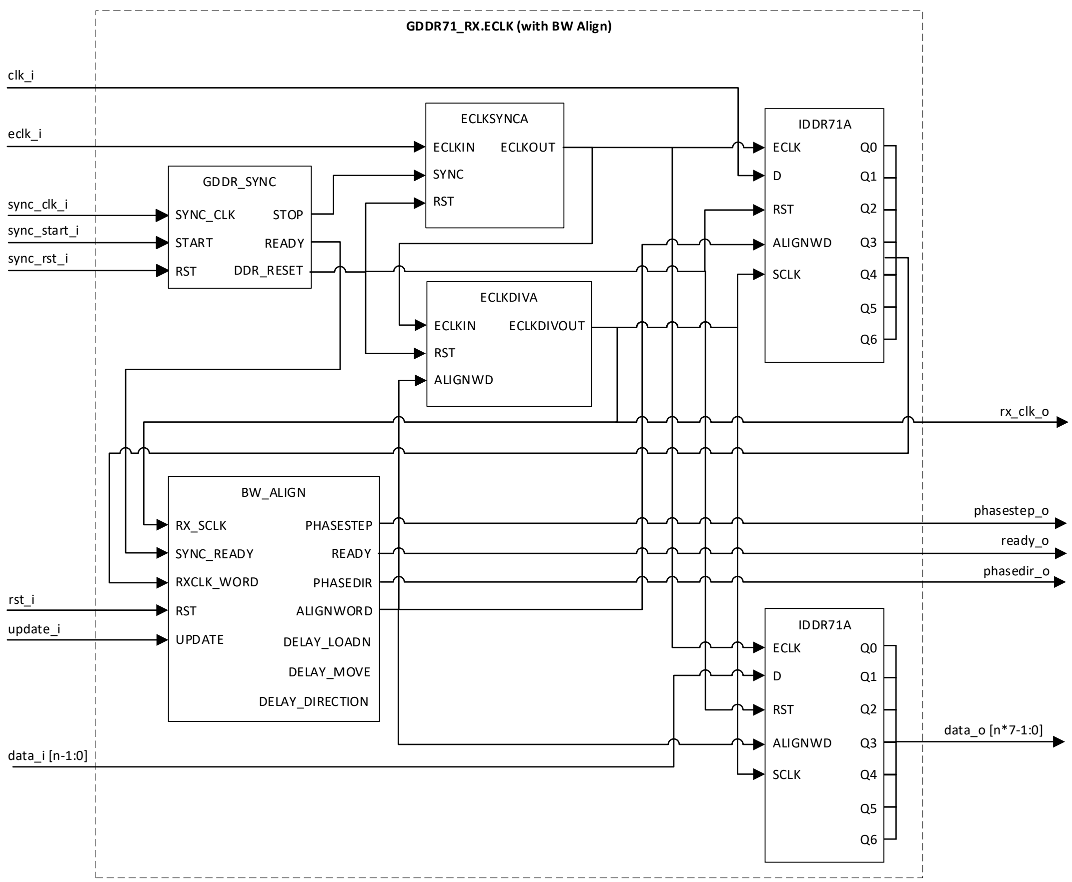


Figure 3.22. GDDR71_RX.ECLK (with Bit and Word Alignment)

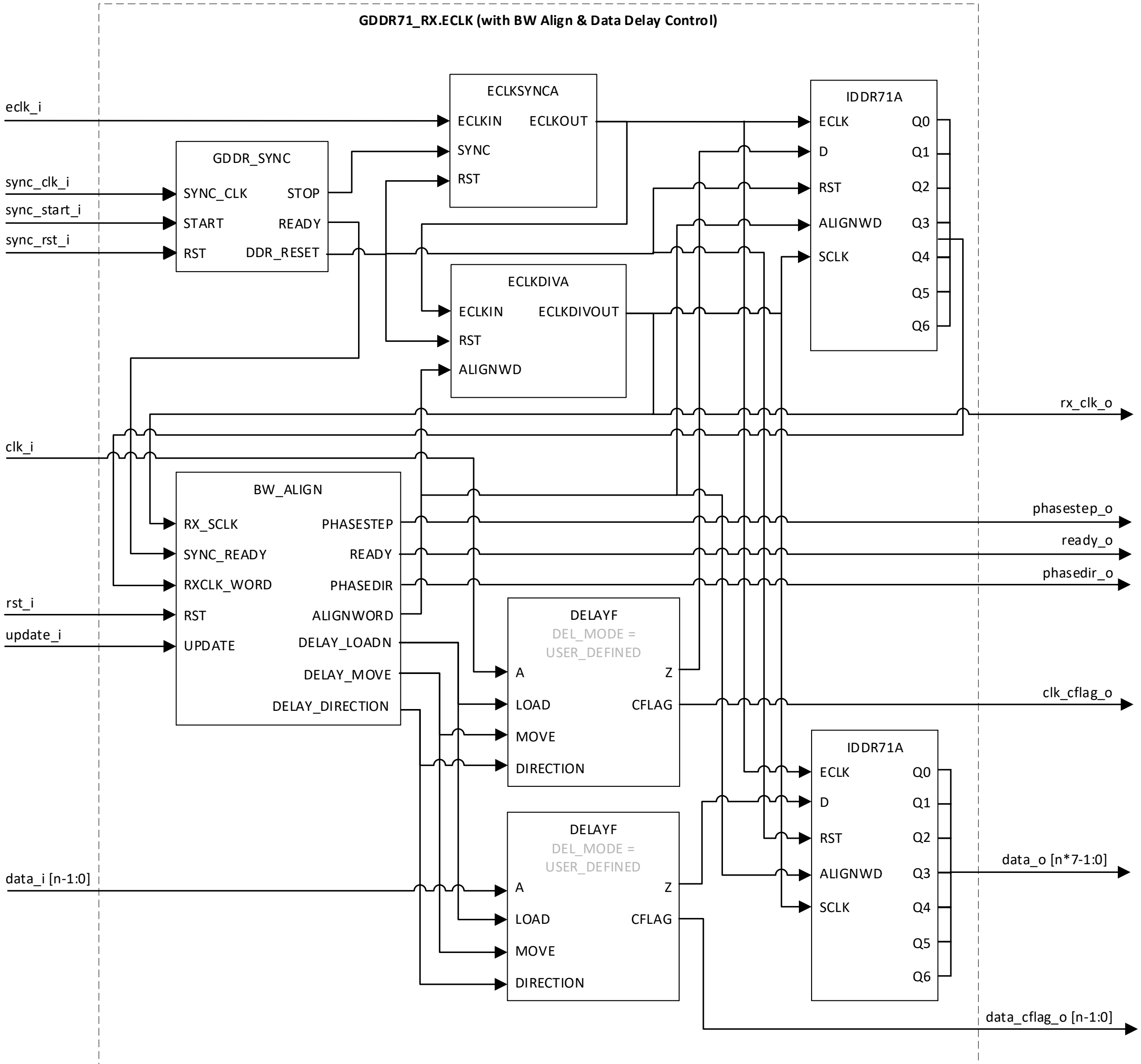


Figure 3.23. GDDR71_RX.ECLK (With Bit and Word Alignment and Data Delay)

3.3. Soft MIPI D-PHY Receive Interfaces

The Lattice Semiconductor Mobile Industry Processor Interface (MIPI) D-PHY Module provides receive interfaces up to 1,500 Mbps and supports bus widths up to 4. The MIPI D-PHY interface uses 8x gearing, supports speeds up to 750 MHz, and supports Camera Serial Interface 2 (CSI-2) and Display Serial Interface (DSI) protocols.

These MIPI D-PHY interfaces use ECLK and are made up of the following modules:

- The IDDRX4A primitives are registers used to capture data both at the positive (Q0/Q2/Q4/Q6) and negative (Q1/Q3/Q5/Q7) edges of the clock.
- The dynamic delay primitive DELAYF allows you to override the set input delay on the data through the Module/IP Block Wizard. The DEL_MODE attribute is used with the DELAYF primitive to indicate the interface type so that the correct delay value can be set in the delay element. ECLK_CENTERED is used for MIPI D-PHY receive interfaces.

- The input clock (clk_p_io/clk_n_io) is routed through the ECLKSYNCA and ECLKDIVA modules to the ECLK tree and is centered relative to data. The ECLKSYNCA module provides clock alignment, and the ECLKDIVA module provides a divided down frequency clock to drive the IDDRX4A SCLK input signals.
- The GDDR_SYNC input reset is an active-high asynchronous reset.
- The synchronization soft IP (GDDR_SYNC) is required to account for the skew between the sync input to ECLKSYNCA and the reset input to ECLKDIVA and IDDRX4A primitives. The DDR_RESET output for GDDR_SYNC is left unconnected on MIPI D-PHY receive interfaces.
- The MIPIA hardware primitives are used to receive MIPI data and clock. The TP and TN ports provide tri-state control to switch LP ports, which can be bidirectional.

Figure 3.24 shows a block diagram of the MIPI D-PHY receive interface.

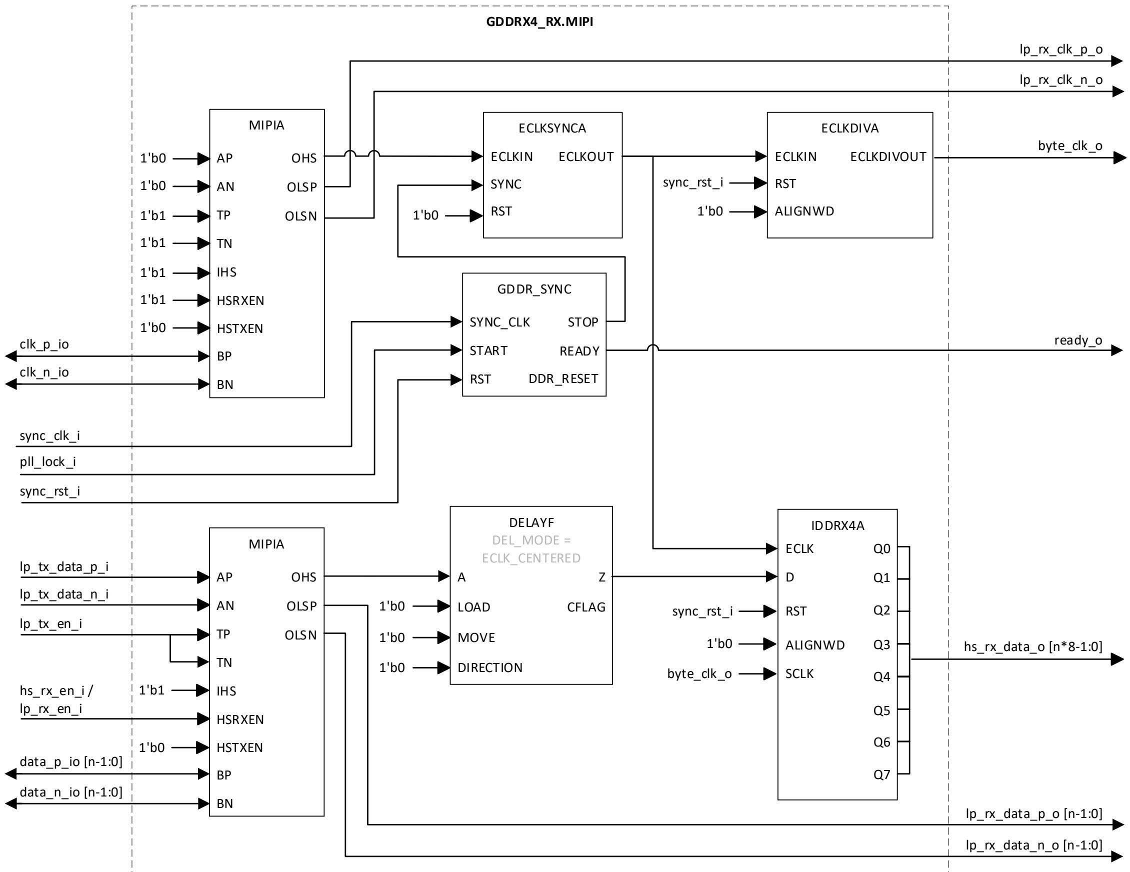


Figure 3.24. GDDRX4_RX.MIPI

3.4. Generic SDR Transmit Interfaces

The Lattice Semiconductor Single Data Rate Input/Output (SDR I/O) Module provides transmit interfaces up to 250 Mbps and supports up to 256-bit data bus widths with single-ended or differential signaling.

3.4.1. GIREG_TX.SCLK Bypass Delay

This section describes the Generic SDR Transmit interface with bypass delay. These interfaces can be used for speeds up to 250 MHz. This SDR interface uses SCLK and is made up of the following modules:

- The OFD1P3DX primitive is a positive edge-triggered register that is used to capture the data.
- The input clock is routed through the ODDRX1A primitive to the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.
- Tri-state control can be optionally enabled on the output data and clock via the OBZ primitive.

Figure 3.25 shows the bypass data delay option for this interface.

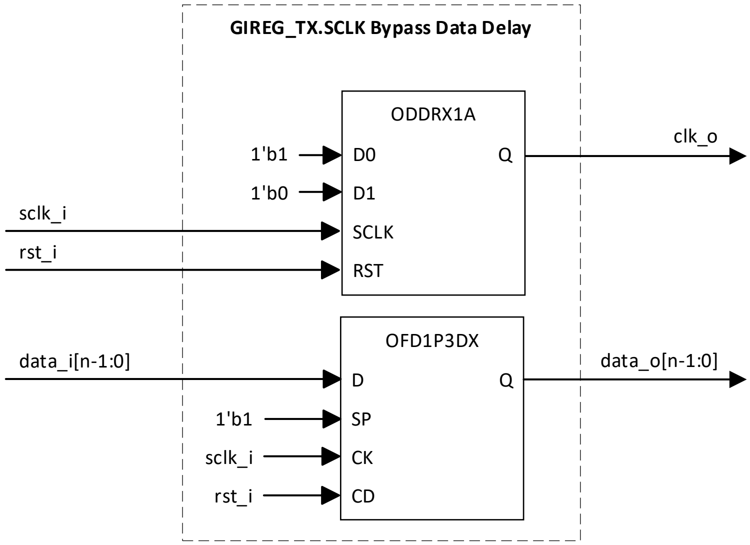


Figure 3.25. GIREG_TX.SCLK (Bypass Data Delay)

3.4.2. GIREG_TX.SCLK Static User-Defined Delay

This section describes the Generic SDR Transmit interface with a static user-defined delay. These interfaces can be used for speeds up to 250 MHz. This SDR interface uses SCLK and is made up of the following modules:

- The OFD1P3DX primitive is a positive edge-triggered register that is used to capture the data.
- The static delay primitive DELAYE is used to delay the incoming data to remove clock injection time.
- The DEL_MODE attribute is used with the DELAYE primitive to indicate the interface type so that the correct delay value can be set in the delay element. USER_DEFINED is used for the static user-defined delay mode.
- The input clock is routed through the ODDRX1A primitive to the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.
- Tri-state control can be optionally enabled on the output data and clock via the OBZ primitive.

Figure 3.26 shows the static user-defined data delay option for this interface.

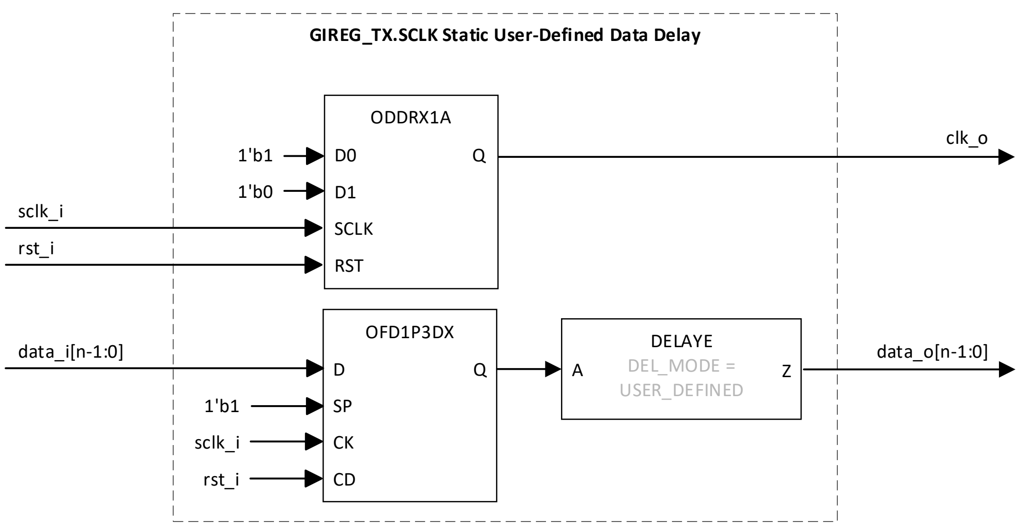


Figure 3.26. GIREG_TX.SCLK (Static User-Defined Data Delay)

3.4.3. GIREG_TX.SCLK Dynamic User-Defined Delay

This section describes the Generic SDR Transmit interface with a dynamic user-defined delay. These interfaces can be used for speeds up to 250 MHz. This SDR interface uses SCLK and is made up of the following modules:

- The OFD1P3DX primitive is a positive edge-triggered register that is used to capture the data.
- The dynamic delay primitive DELAYD is used to control the delay on the data dynamically. DELAYD allows you to override the set input delay through the Module/IP Block Wizard. Note that the MOVE input on the DELAYD primitive needs to meet a 6 ns minimum pulse width requirement.
- The DEL_MODE attribute is used with the DELAYD primitive to indicate the interface type so that the correct delay value can be set in the delay element. USER_DEFINED is used for the dynamic user-defined delay mode.
- The input clock is routed through the ODDRX1A primitive to the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.
- Tri-state control can be optionally enabled on the output data and clock via the OBZ primitive.

Figure 3.27 shows the dynamic user-defined data delay option for this interface.

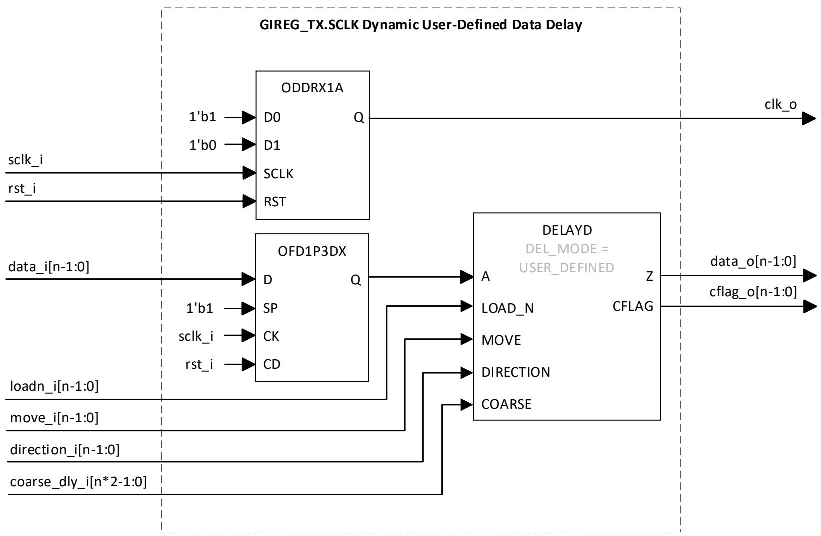


Figure 3.27. GIREG_TX.SCLK (Dynamic User-Defined Data Delay)

3.5. Generic DDR Transmit Interfaces

The Lattice Semiconductor Generic Double Data Rate Input/Output (GDDR I/O) Module provides transmit interfaces up to 1,800 Mbps and supports up to 256-bit data bus widths with x1, x2, x4, and x5 gearing. The Lattice Semiconductor Generic Double Data Rate 7:1 Input/Output (GDDR 7:1 I/O) Module provides transmit interfaces up to 1,050 Mbps and supports up to 16-bit data bus widths.

3.5.1. GDDRX1_TX.SCLK.Centered

This section describes the Generic DDR 1x Centered Transmit interface with bypass and static/dynamic user-defined delays. These interfaces can be used for speeds up to 250 MHz. This DDR interface uses SCLK and is made up of the following modules:

- The ODDRX1A primitive is a register that is used to capture the data at the positive and negative edges of a clock.
- The static delay primitive DELAYE is used in static user-defined delay modes to delay the incoming data to remove clock injection time. The type of delay required can be selected through the Module/IP Block Wizard.
- The dynamic delay primitive DELAYD is used in dynamic user-defined delay modes. DELAYD allows you to override the set input delay on the data dynamically through the Module/IP Block Wizard. Note that the MOVE input on the DELAYD primitive needs to meet a 6 ns minimum pulse width requirement.

- The DEL_MODE attribute is used with the DELAYE and DELAYD primitives to indicate the interface type so that the correct delay value can be set in the delay element. USER_DEFINED is used for static/dynamic user-defined delay modes.
- The input clock (clk_i) is routed through the ODDRX1A primitive to the primary (SCLK) clock tree and is centered relative to data.
- The input reset is an active-high asynchronous reset.

Figure 3.28, Figure 3.29, and Figure 3.30 show the bypass and static/dynamic user-defined delay options for this interface.

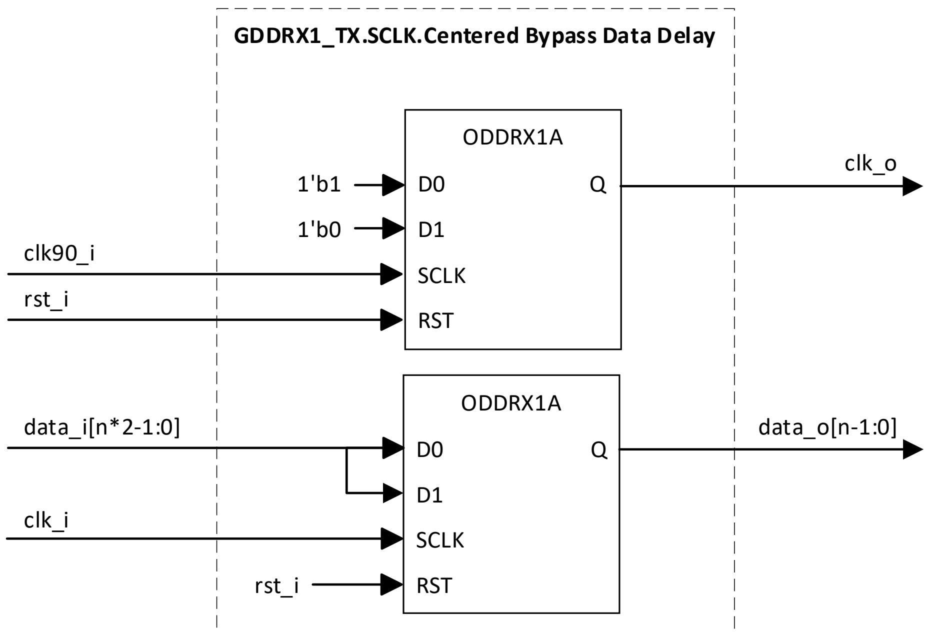


Figure 3.28. GDDRX1_TX.SCLK.Centered (Bypass Data Delay)

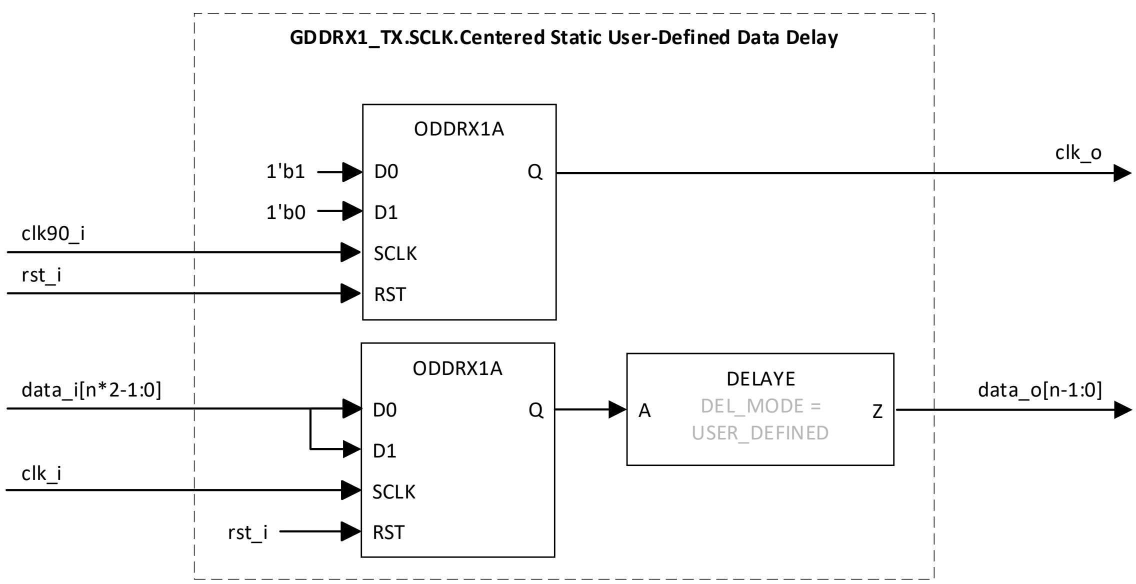


Figure 3.29. GDDRX1_TX.SCLK.Centered (Static User-Defined Data Delay)

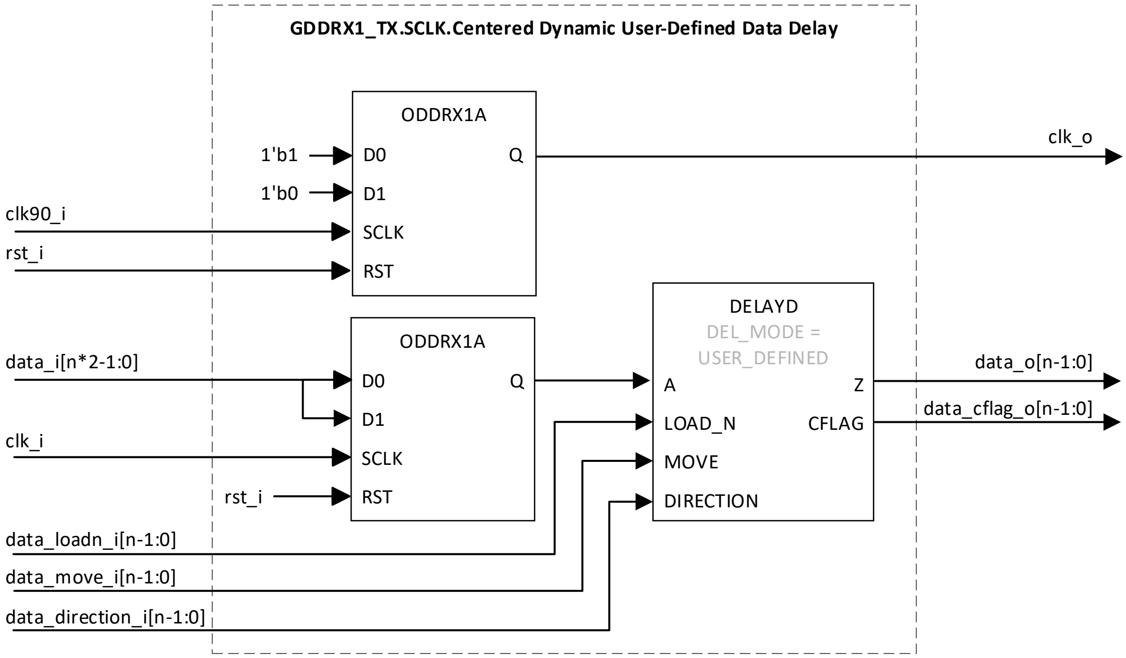


Figure 3.30. GDDRX1_TX.SCLK.Centered (Dynamic User-Defined Data Delay)

3.5.2. GDDRX1_TX.SCLK.Aligned

This section describes the Generic DDR 1x Aligned Transmit interface with bypass and static/dynamic user-defined delays. These interfaces can be used for speeds up to 250 MHz. This DDR interface uses SCLK and is made up of the following modules:

- The ODDRX1A primitive is a register that is used to capture the data at the positive and negative edges of a clock.
- The static delay primitive DELAYE is used in static user-defined delay modes to delay the incoming data to remove clock injection time. The type of delay required can be selected through the Module/IP Block Wizard.
- The dynamic delay primitive DELAYD is used in dynamic user-defined delay modes. DELAYD allows you to override the set input delay on the data dynamically through the Module/IP Block Wizard. Note that the MOVE input on the DELAYD primitive needs to meet a 6 ns minimum pulse width requirement.
- The DEL_MODE attribute is used with the DELAYE and DELAYD primitives to indicate the interface type so that the correct delay value can be set in the delay element. USER_DEFINED is used for static/dynamic user-defined delay modes.
- The input clock (clk_i) is routed through the ODDRX1A primitive to the primary (SCLK) clock tree and is edge-aligned relative to data.
- The input reset is an active-high asynchronous reset.

Figure 3.31, Figure 3.32, and Figure 3.33 show the bypass and static/dynamic user-defined delay options for this interface.

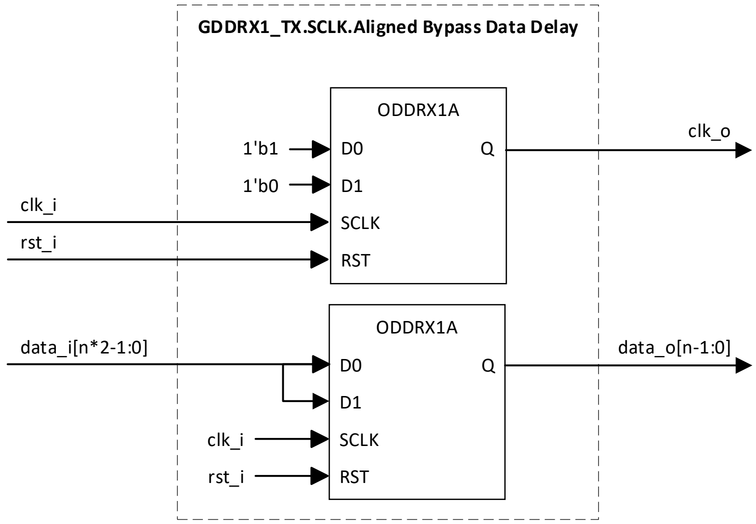


Figure 3.31. GDDRX1_TX.SCLK.Aligned (Bypass Data Delay)

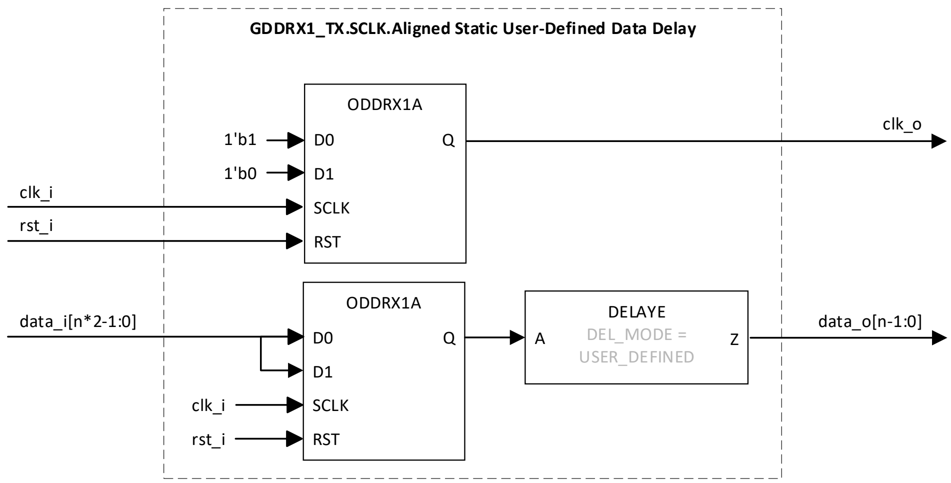


Figure 3.32. GDDRX1_TX.SCLK.Aligned (Static User-Defined Data Delay)

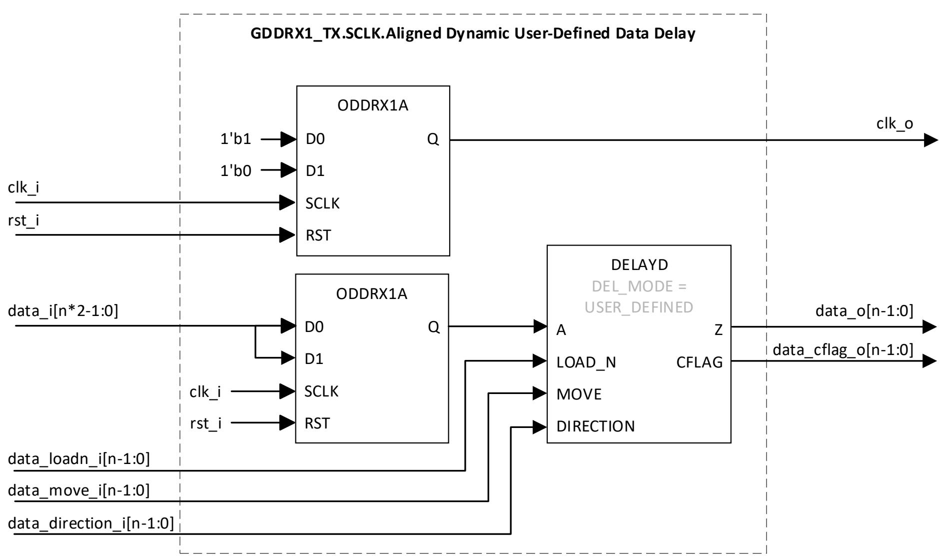


Figure 3.33. GDDRX1_TX.SCLK.Aligned (Dynamic User-Defined Data Delay)

3.5.3. GDDRX2_TX.ECLK.Centered/GDDRX4_TX.ECLK.Centered/GDDRX5_TX.ECLK.Centered

This section describes the Generic DDR 2x/4x/5x Centered Transmit interface with bypass and static/dynamic user-defined delays. The Generic 2x interfaces can be used for speeds up to 600 MHz and the Generic 4x/5x interfaces can be used for speeds up to 900 MHz. These DDR interfaces use ECLK and are made up of the following modules:

- The ODDRX2A/ODDRX4A/ODDRX5A primitives are registers that are used to capture data both at the positive (Q0/Q2/Q4/Q6/Q8) and negative (Q1/Q3/Q5/Q7/Q9) edges of the clock.
- The dynamic delay primitive DELAYF is used in static/dynamic user-defined delay modes. DELAYF allows you to override the set input delay on the data through the Module/IP Block Wizard. The CFLAG output for DELAYF is left unconnected for the static user-defined delay mode. Note that the MOVE input on the DELAYF primitive needs to meet a 6 ns minimum pulse width requirement.
- The DEL_MODE attribute is used with the DELAYF primitive to indicate the interface type so that the correct delay value can be set in the delay element. USER_DEFINED is used for static/dynamic user-defined delay modes.
- The input clock (eclk_i) is routed through the ECLKSYNCA module to the ECLK tree and is centered relative to the data. The SCLK output of the ECLKDIVA module is routed through the primary (SCLK) clock tree. Optionally, PLL instantiation can be enabled to generate ECLKIN input for ECLKSYNCA.
- The GDDR_SYNC input reset is an active-high asynchronous reset.
- The synchronization soft IP (GDDR_SYNC) is required for bypass and static modes to account for the skew between the sync input to ECLKSYNCA and the reset input to ECLKDIVA and ODDRX2A/ODDRX4A/ODDRX5A primitives.
- Tri-state control can be optionally enabled on the output data and clock via the OBZ primitive in dynamic user-defined delay modes.

Figure 3.34, Figure 3.35, and Figure 3.36 show the bypass delay options for this interface. Figure 3.37, Figure 3.38, and Figure 3.39 show the static user-defined delay options for this interface. Figure 3.40, Figure 3.41, and Figure 3.42 show the dynamic user-defined delay options for this interface.

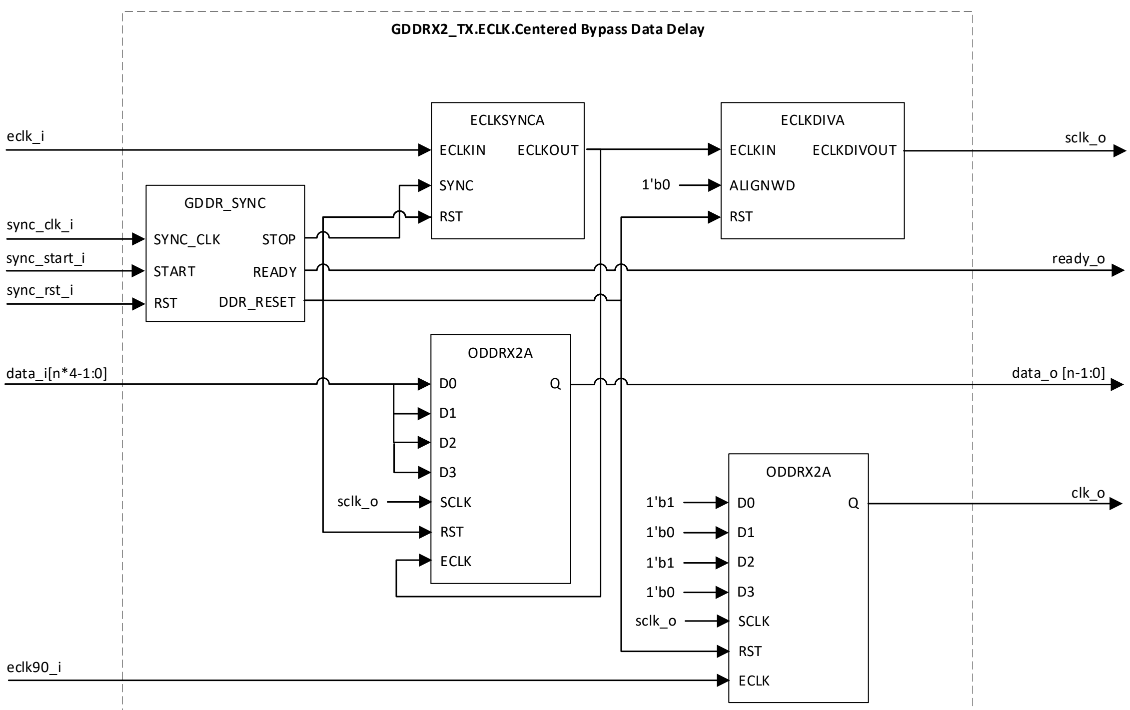


Figure 3.34. GDDRX2_TX.ECLK.Centered (Bypass Data Delay)

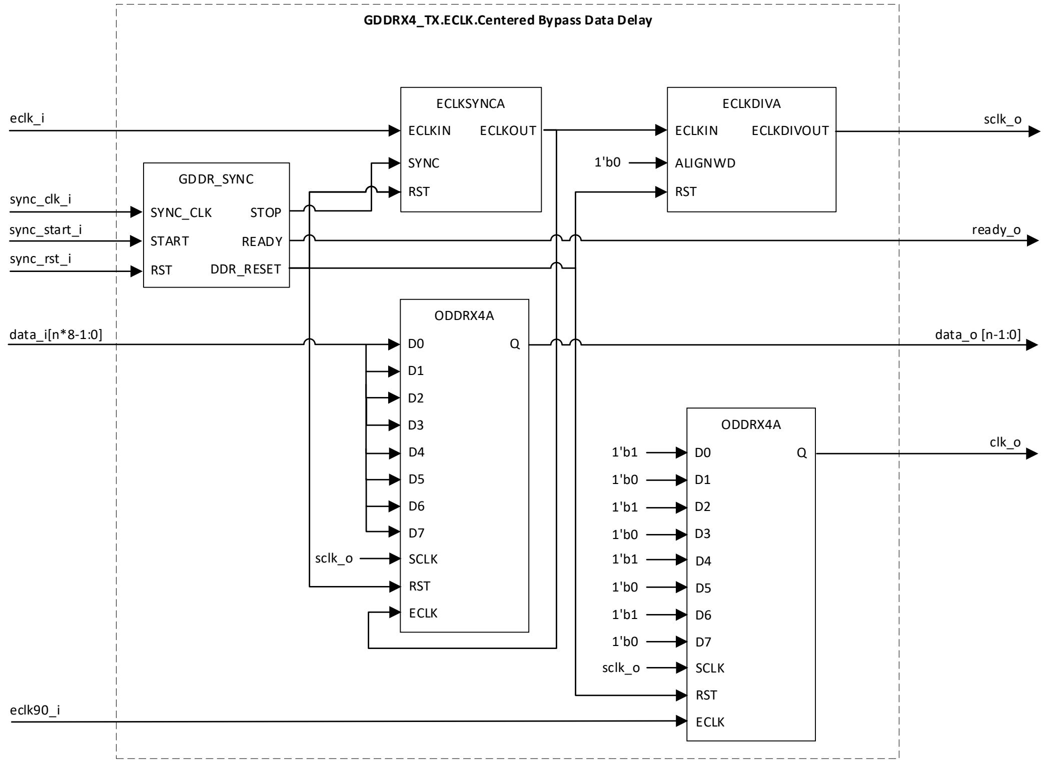


Figure 3.35. GDDRX4_TX.ECLK.Centered (Bypass Data Delay)

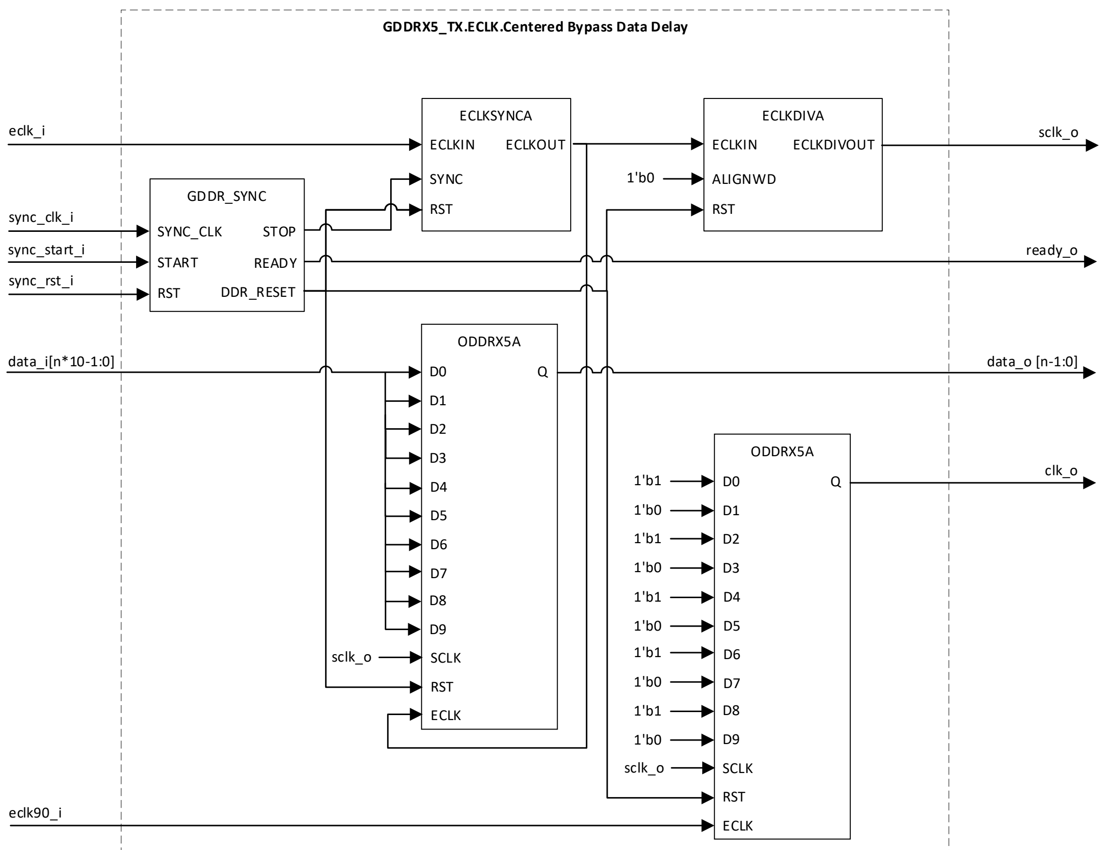


Figure 3.36. GDDRX5_TX.ECLK.Centered (Bypass Data Delay)

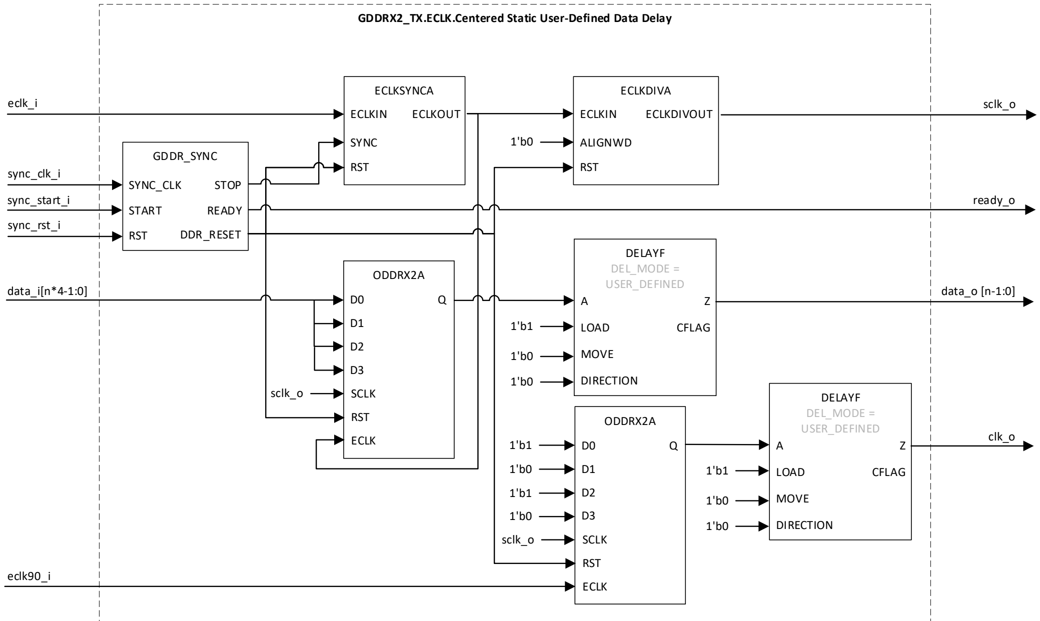


Figure 3.37. GDDRX2_TX.ECLK.Centered (Static User-Defined Data Delay)

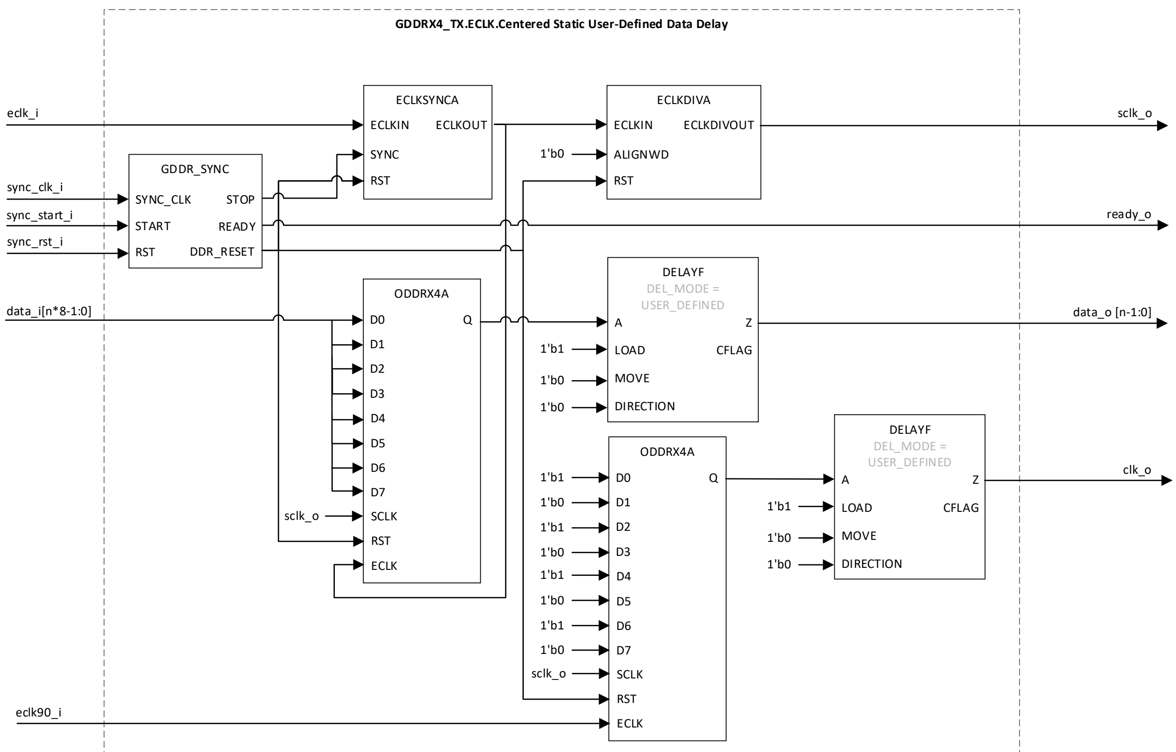


Figure 3.38. GDDRX4_TX.ECLK.Centered (Static User-Defined Data Delay)

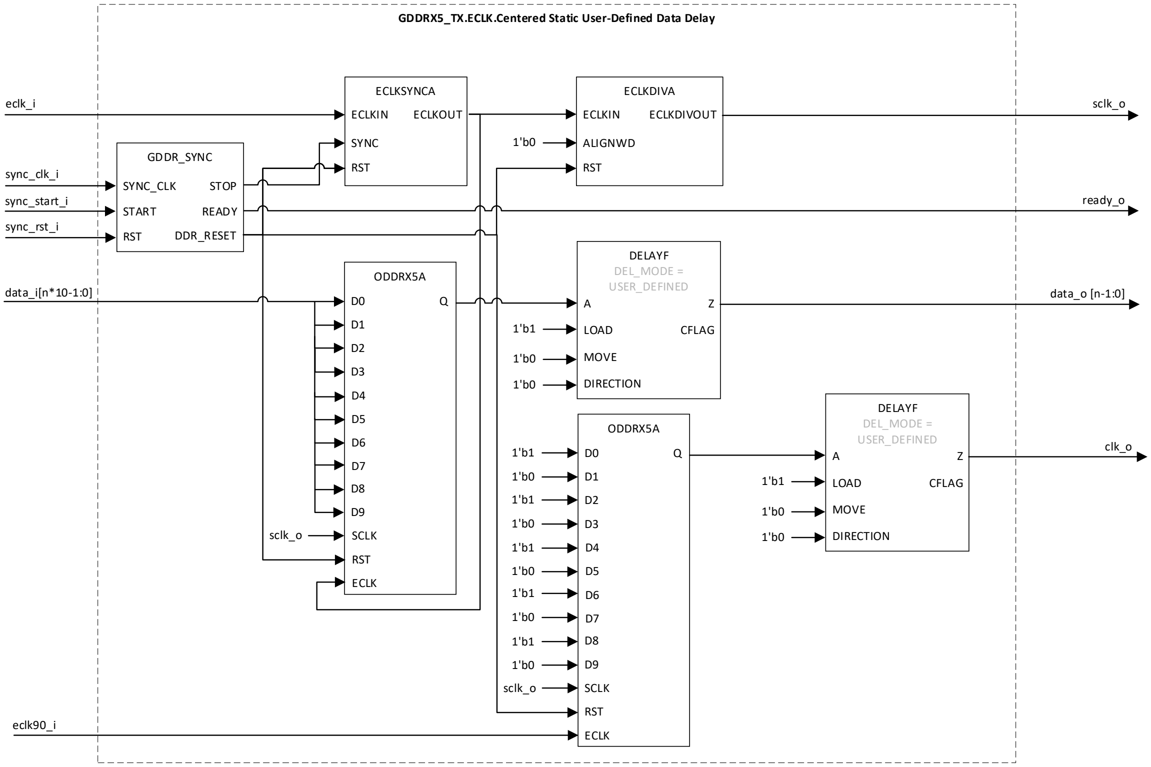


Figure 3.39. GDDRX5_TX.ECLK.Centered (Static User-Defined Data Delay)

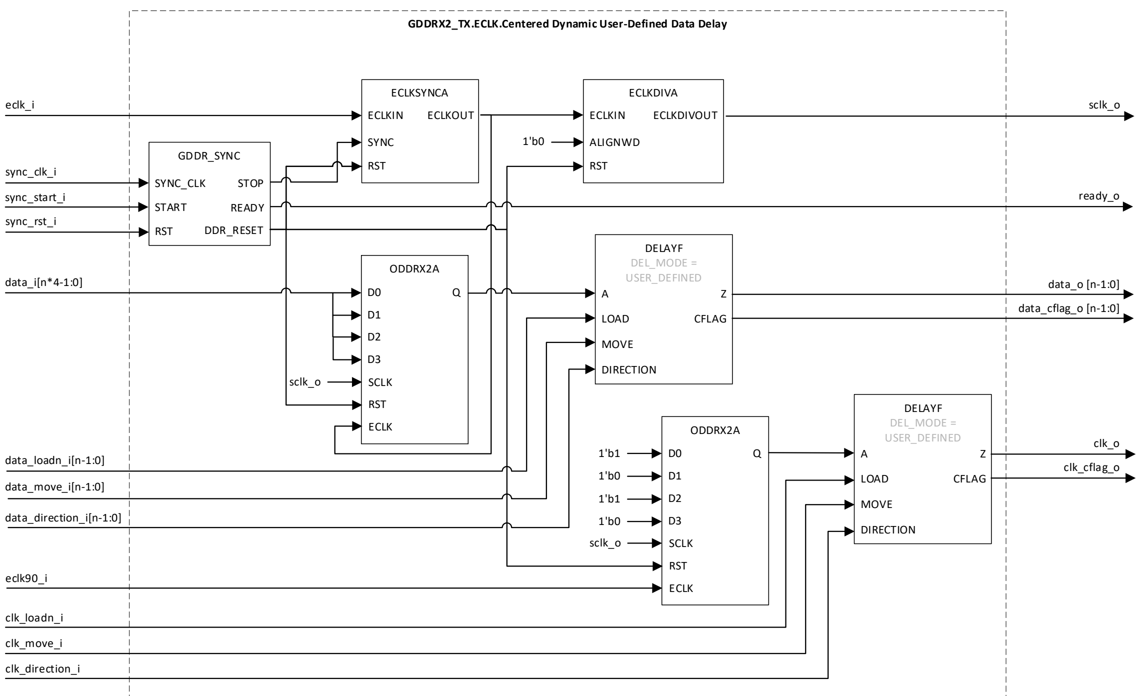


Figure 3.40. GDDRX2_TX.ECLK.Centered (Dynamic User-Defined Data Delay)

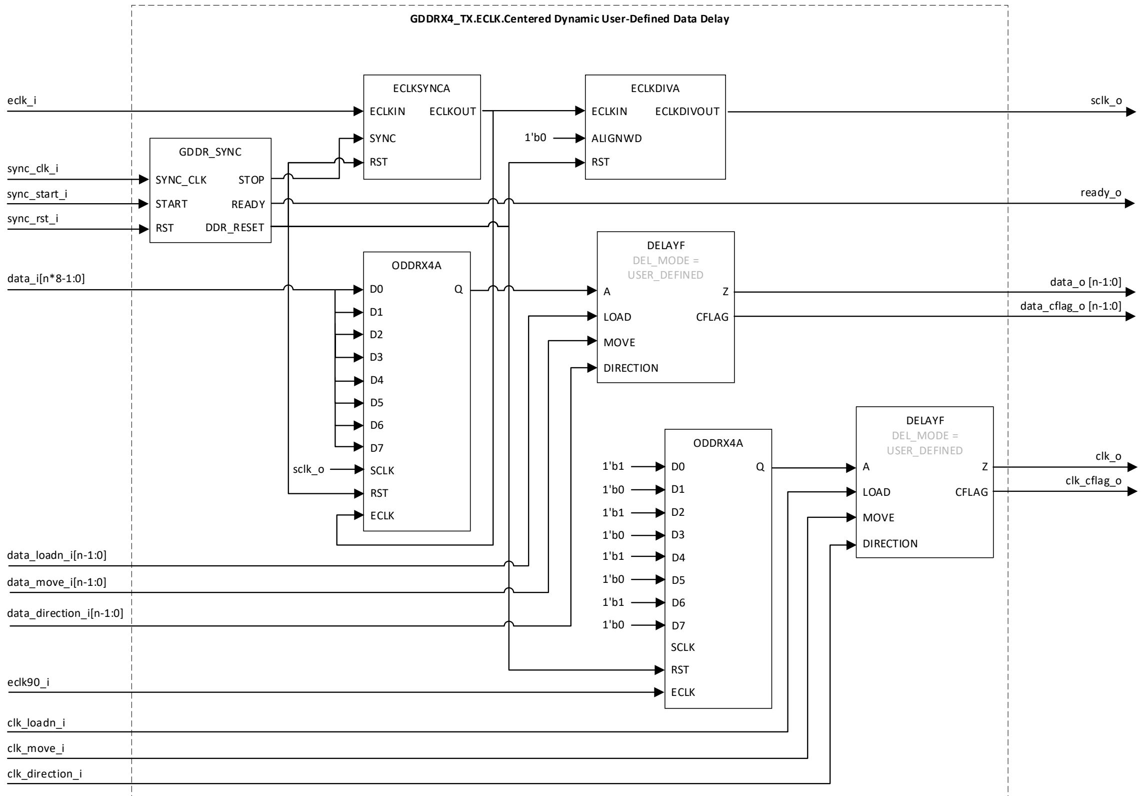


Figure 3.41. GDDRX4_TX.ECLK.Centered (Dynamic User-Defined Data Delay)

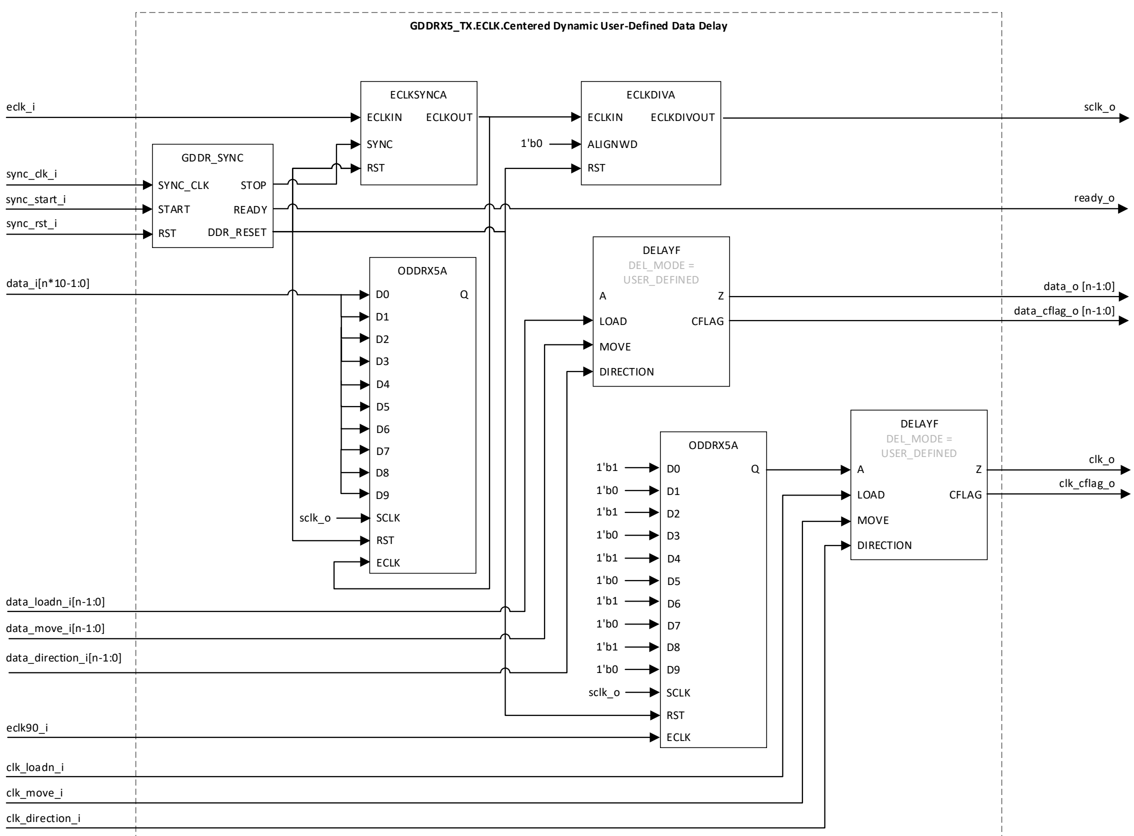


Figure 3.42. GDDRX5_TX.ECLK.Centered (Dynamic User-Defined Data Delay)

3.5.4. GDDRX2_TX.ECLK.Aligned/GDDRX4_TX.ECLK.Aligned/GDDRX5_ TX.ECLK.Aligned

This section describes the Generic DDR 2x/4x/5x Aligned Transmit interface with bypass and static/dynamic user-defined delays. The Generic 2x interfaces can be used for speeds up to 600 MHz and the Generic 4x/5x interfaces can be used for speeds up to 900 MHz. These DDR interfaces use ECLK and are made up of the following modules:

- The ODDRX2A/ODDRX4A/ODDRX5A primitives are registers that are used to capture data both at the positive (Q0/Q2/Q4/Q6/Q8) and negative (Q1/Q3/Q5/Q7/Q9) edges of the clock.
- The dynamic delay primitive DELAYF is used in static/dynamic user-defined delay modes. DELAYF allows you to override the set input delay on the data through the Module/IP Block Wizard. The CFLAG output for DELAYF is left unconnected for the static user-defined delay mode. Note that the MOVE input on the DELAYF primitive needs to meet a 6 ns minimum pulse width requirement.
- The DEL_MODE attribute is used with the DELAYF primitive to indicate the interface type so that the correct delay value can be set in the delay element. USER_DEFINED is used for static/dynamic user-defined delay modes.
- The input clock (clk_i) is routed through the ECLKSYNCA module to the ECLK tree and is edge-aligned relative to the data. The SCLK output of the ECLKDIVA module is routed through the primary (SCLK) clock tree. Optionally, PLL instantiation can be enabled to generate ECLKIN input for ECLKSYNCA.
- The GDDR_SYNC input reset is an active-high asynchronous reset.
- The synchronization soft IP (GDDR_SYNC) is required for bypass and static modes to account for the skew between the sync input to ECLKSYNCA and the reset input to ECLKDIVA and ODDRX2A/ODDRX4A/ODDRX5A primitives.
- Tri-state control can be optionally enabled on the output data and clock via the OBZ primitive in bypass and static user-defined delay modes.

Figure 3.43, Figure 3.44, and Figure 3.45 show the bypass delay options for this interface. Figure 3.46, Figure 3.47, and Figure 3.48 show the static user-defined delay options for this interface. Figure 3.49, Figure 3.50, and Figure 3.51 show the dynamic user-defined delay options for this interface.

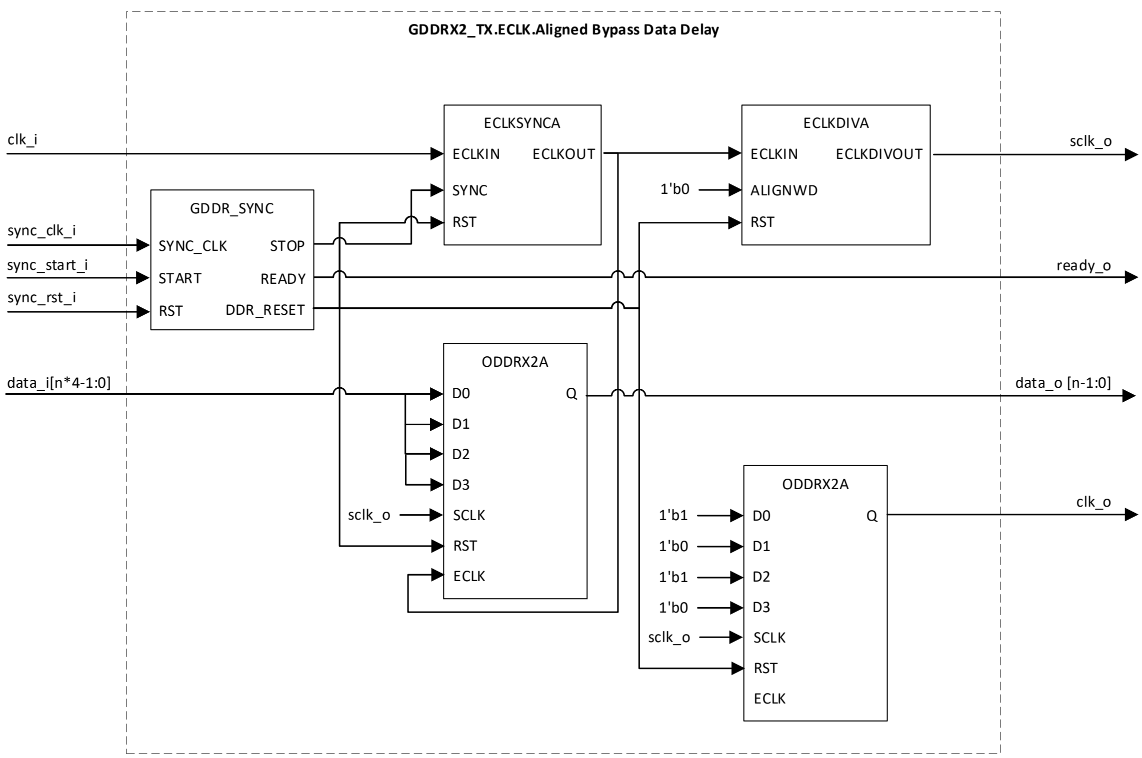


Figure 3.43. GDDRX2_TX.ECLK.Aligned (Bypass Data Delay)

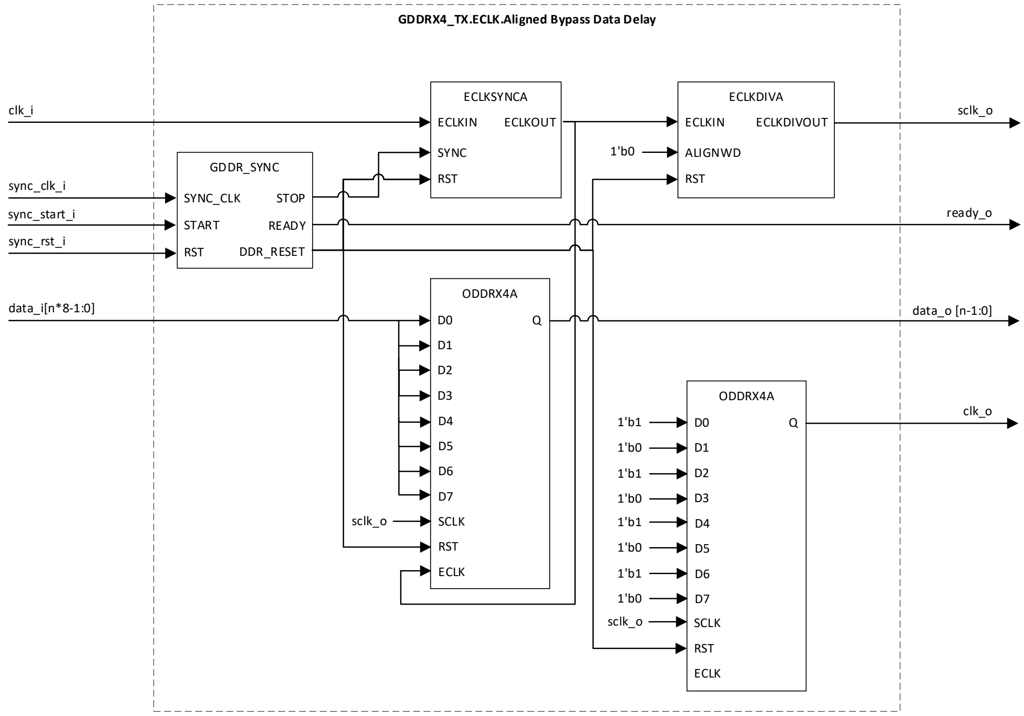


Figure 3.44. GDDRX4_TX.ECLK.Aligned (Bypass Data Delay)

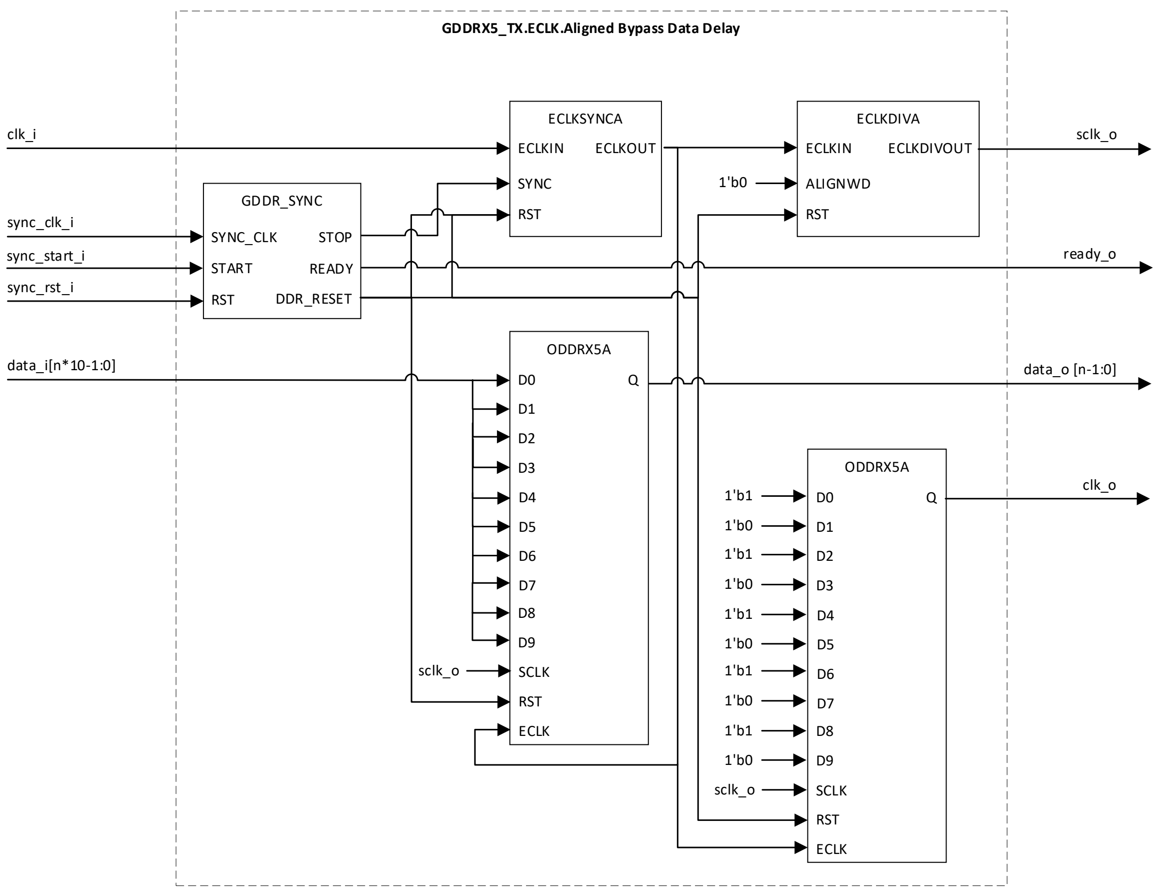


Figure 3.45. GDDRX5_TX.ECLK.Aligned (Bypass Data Delay)

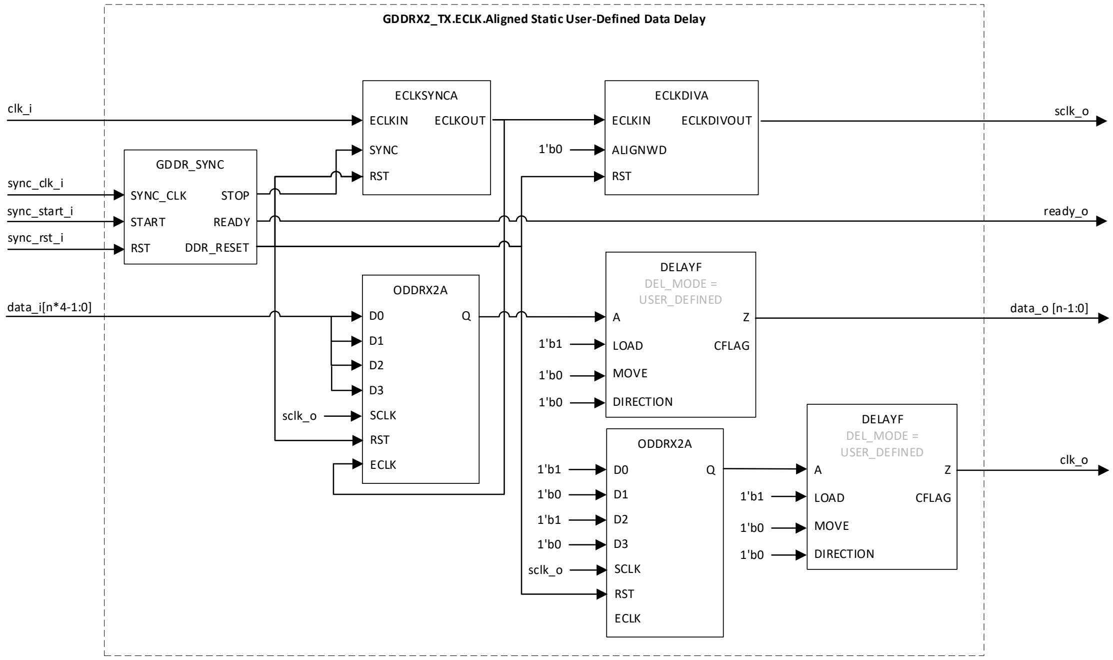


Figure 3.46. GDDRX2_TX.ECLK.Aligned (Static User-Defined Data Delay)

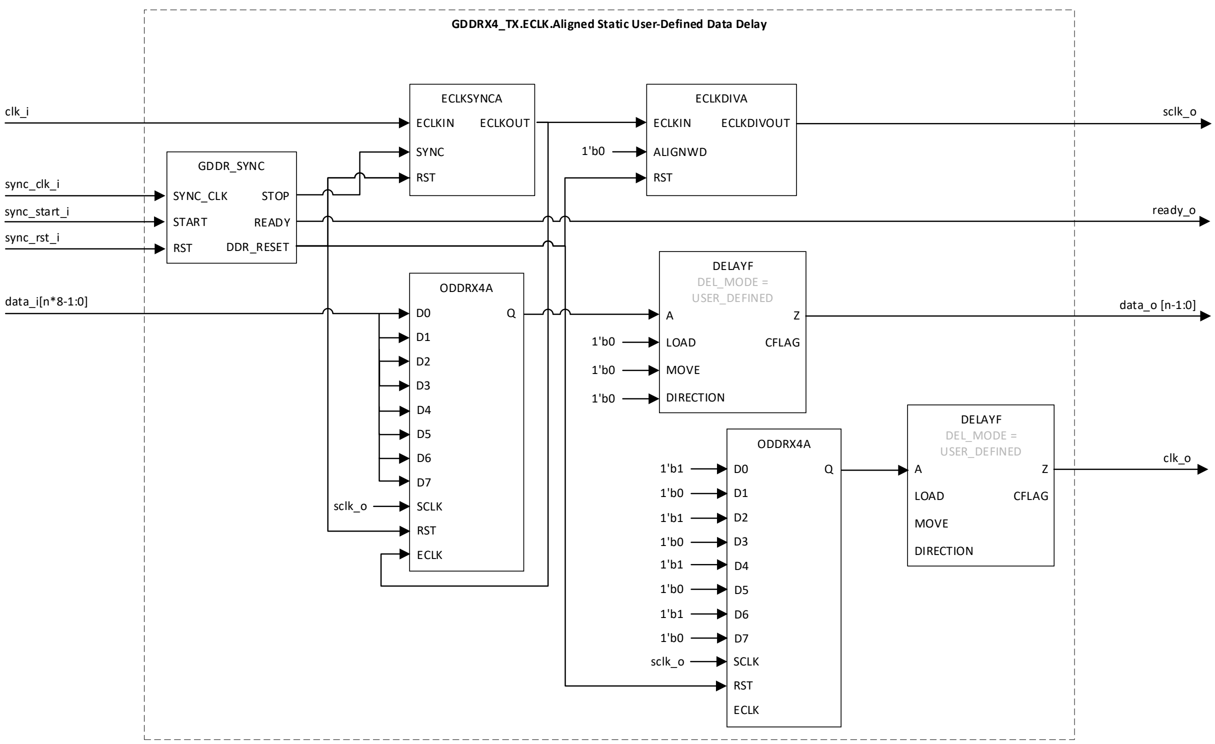


Figure 3.47. GDDRX4_TX.ECLK.Aligned (Static User-Defined Data Delay)

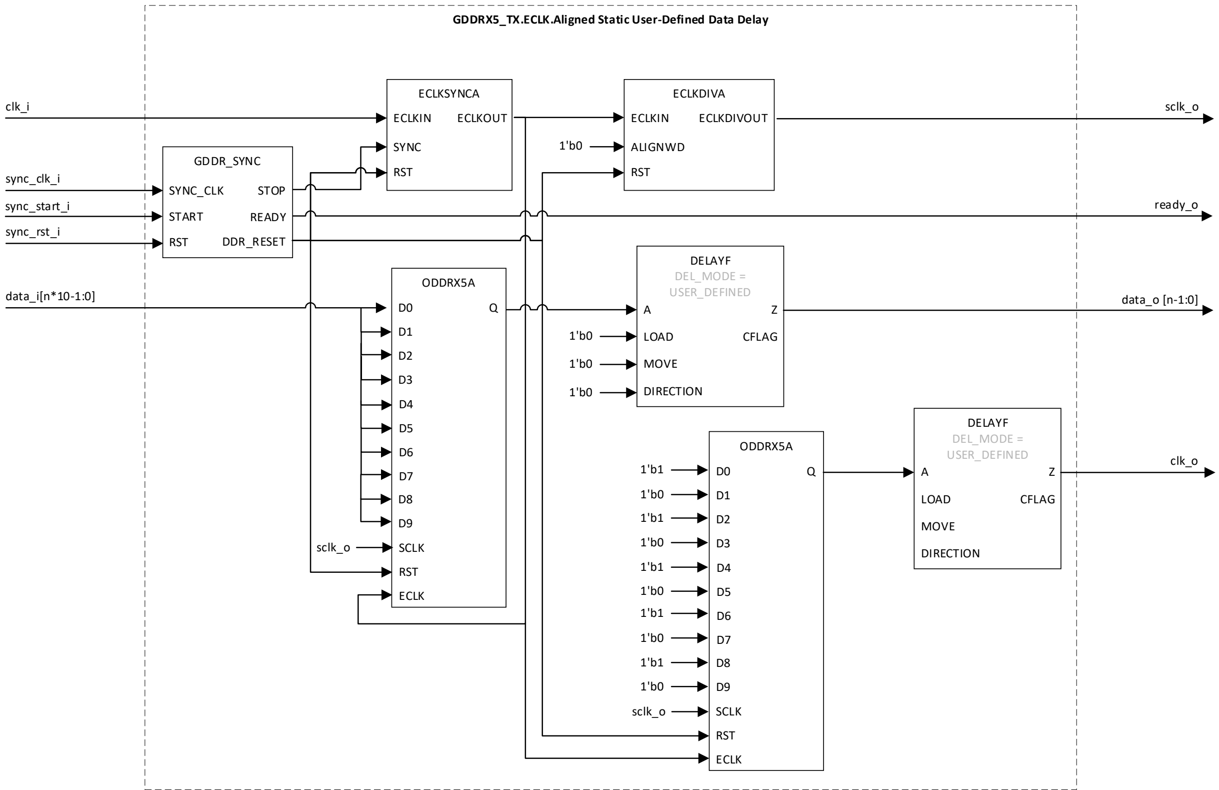


Figure 3.48. GDDRX5_TX.ECLK.Aligned (Static User-Defined Data Delay)

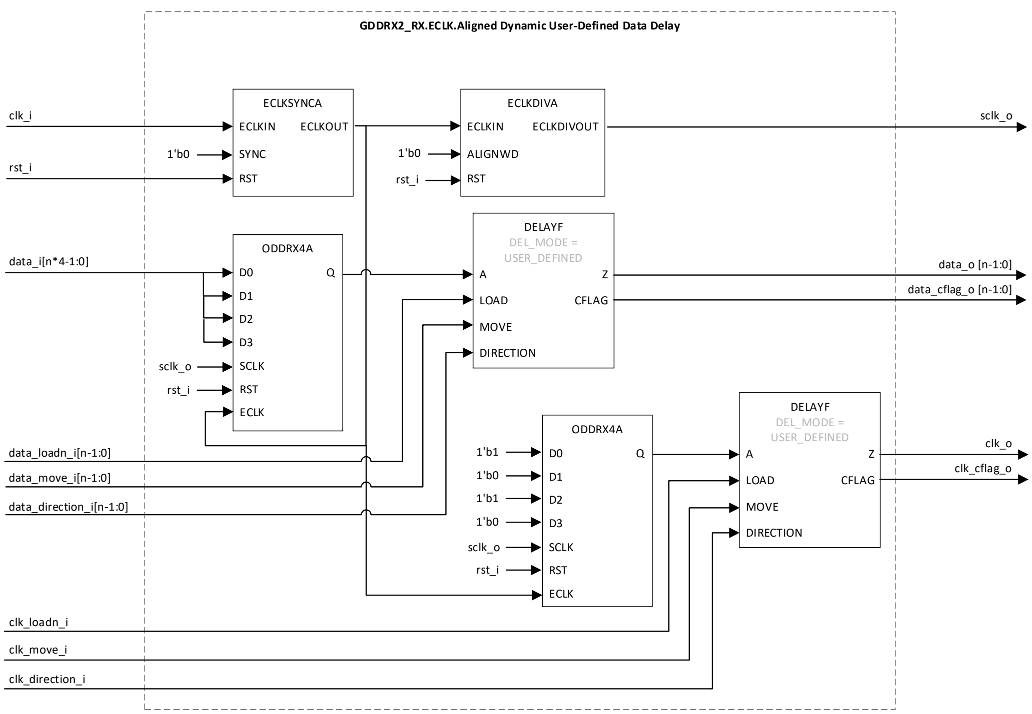


Figure 3.49. GDDRX2_TX.ECLK.Aligned (Dynamic User-Defined Data Delay)

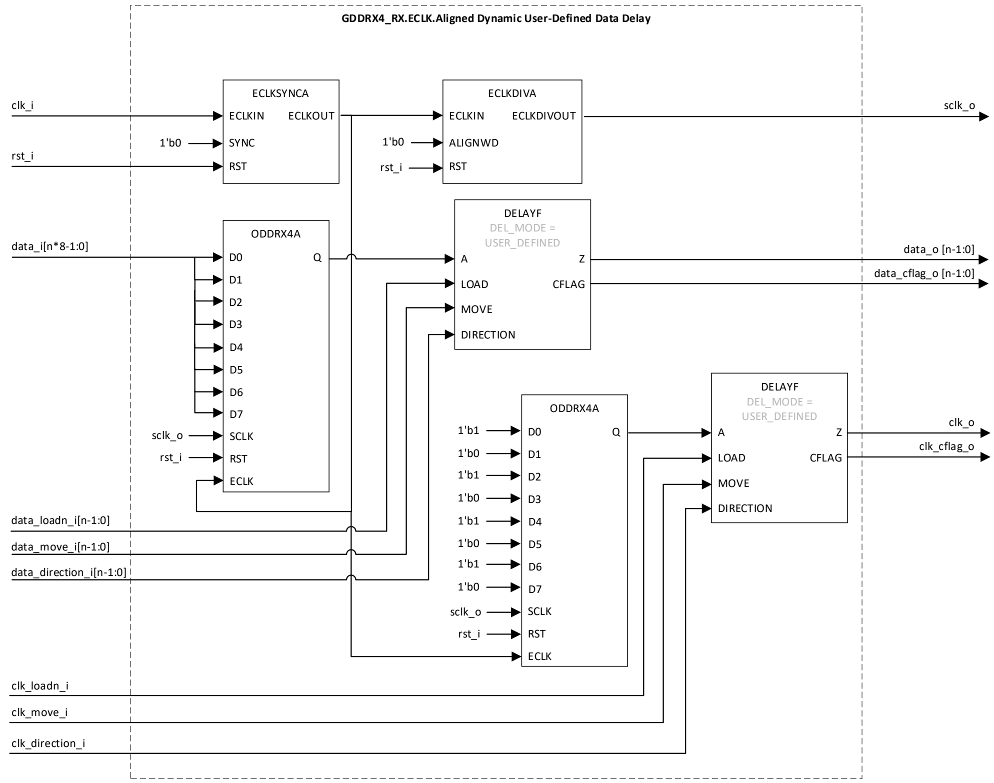


Figure 3.50. GDDRX4_TX.ECLK.Aligned (Dynamic User-Defined Data Delay)

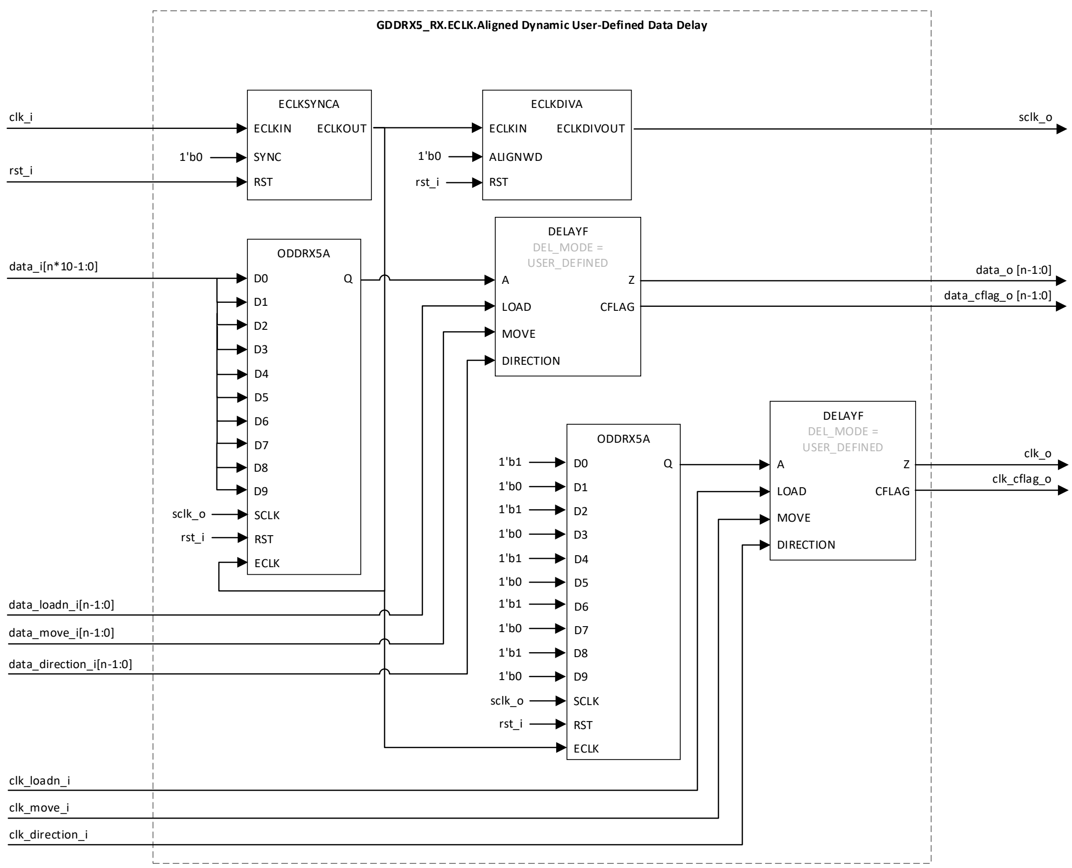


Figure 3.51. GDDRX5_TX.ECLK.Aligned (Dynamic User-Defined Data Delay)

3.5.5. GDDR71_TX.ECLK

This section describes the Generic DDR 7:1 Transmit interface. The Generic DDR 7:1 interface can be used for speeds up to 150 MHz. These DDR interfaces use ECLK and are made up of the following modules:

- The ODDR71A primitives are registers that are used to capture data both at the positive (Q0/Q2/Q4/Q6) and negative (Q1/Q3/Q5) edges of the clock.
- The input clock (eclk_i) is routed through the ECLKSYNCA and ECLKDIVA modules to the ECLK tree. The clock is divided by 3.5 and is center-aligned relative to the data. The SCLK output of the ECLKDIVA module is routed through the primary (SCLK) clock tree.
- The input reset is an active-high asynchronous reset.
- The synchronization soft IP (GDDR_SYNC) is required to account for the skew between the sync input to ECLKSYNCA and the reset input to ECLKDIVA and ODDR71A primitives.

Figure 3.52. GDDR71_TX.ECLKshows the Generic DDR 7:1 transmit interface.

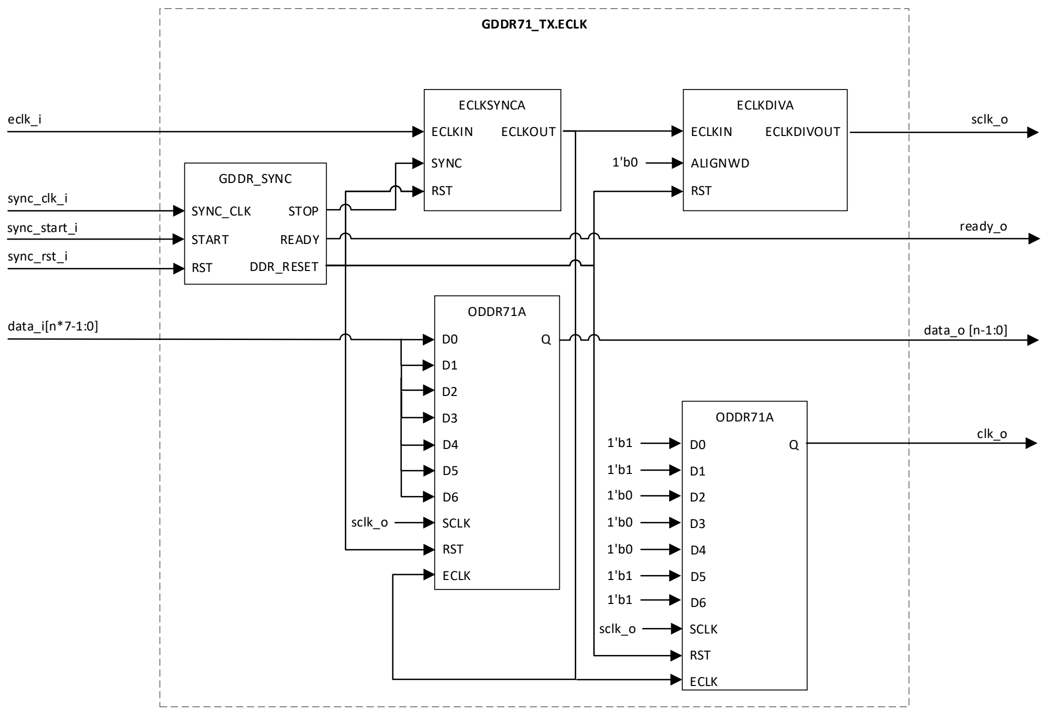


Figure 3.52. GDDR71_TX.ECLK

3.6. Soft MIPI D-PHY Transmit Interfaces

The Lattice Semiconductor Mobile Industry Processor Interface (MIPI) D-PHY Module provides transmit interfaces up to 1,500 Mbps and supports bus widths up to 4. The MIPI D-PHY interface uses 8x gearing, supports speeds up to 750 MHz, and supports Camera Serial Interface 2 (CSI-2) and Display Serial Interface (DSI) protocols.

These MIPI D-PHY interfaces use ECLK and are made up of the following modules:

- The ODDRX4A primitives are registers used to capture data both at the positive (Q0/Q2/Q4/Q6) and negative (Q1/Q3/Q5/Q7) edges of the clock.
- The input clock (pll_clkop_i) is routed through the ECLKSYNCA and ECLKDIVA modules to the ECLK tree and is centered relative to data. The ECLKSYNCA module provides clock alignment, and the ECLKDIVA module provides a divided down frequency clock to drive the ODDRX4A SCLK input signals.
- The GDDR_SYNC input reset is an active-high asynchronous reset.
- The synchronization soft IP (GDDR_SYNC) is required to account for the skew between the sync input to ECLKSYNCA and the reset input to ECLKDIVA and ODDRX4A primitives.
- The MIPIA hardware primitives are used to receive MIPI data and clock. The TP and TN ports provide tri-state control to switch LP ports, which can be bidirectional.

Figure 3.53 shows a block diagram of the MIPI D-PHY transmit interface.

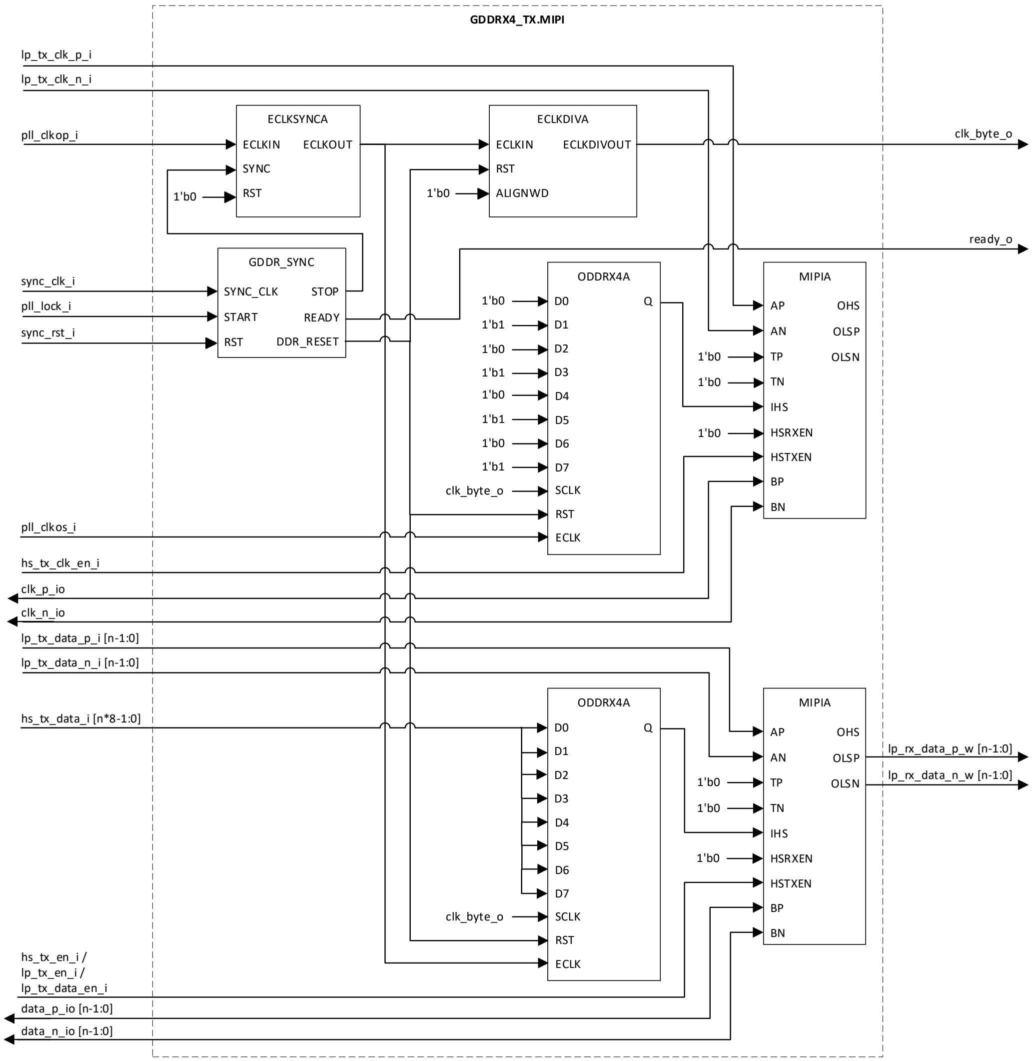


Figure 3.53. GDDRX4_TX.MIPI

3.7. Design Rules and Guidelines

This section describes the various design guidelines used for building generic high-speed DDR interfaces in The Lattice Nexus 2 devices. In addition to these guidelines, it is also required to follow the interface rules described for each type of interface. Refer to the Generic High-Speed I/O Interfaces section of this technical note to find the interface you are building.

3.7.1. Pin and Resource Planning

The I/O banks that support each of the Generic High-Speed I/O Interfaces depend on the following factors:

- Type of interface
- Selected I/O standard

The following table summarizes the bank support for each type of interface, where Top refers to WRIO banks and Bottom refers to HPIO banks. Depending on the selected I/O standard, the banks that can support an interface may vary. Refer to the Lattice Nexus 2 Platform Datasheet Overview (FPGA-02122) for details on which I/O standards are supported by WRIO and HPIO banks.

Table 3.2. Generic High-Speed I/O Interface Bank Support

Mode	Interface Name	Supported Banks
Receive SDR	GIREG_RX.SCLK	Top and Bottom
RX DDRX1 Centered	GDDRX1_RX.SCLK.Centered	Top and Bottom
RX DDRX1 Aligned	GDDRX1_RX.SCLK.Aligned	Top and Bottom
RX DDRX2 Centered	GDDRX2_RX.ECLK.Centered	Bottom Only
RX DDRX2 Aligned	GDDRX2_RX.ECLK.Aligned	Bottom Only
RX DDRX4 Centered	GDDRX4_RX.ECLK.Centered	Bottom Only
RX DDRX4 Aligned	GDDRX4_RX.ECLK.Aligned	Bottom Only
RX DDRX5 Centered	GDDRX5_RX.ECLK.Centered	Bottom Only
RX DDRX5 Aligned	GDDRX5_RX.ECLK.Aligned	Bottom Only
RX DDR71	GDDR71_RX.ECLK	Bottom Only
RX DDRX4 MIPI	GDDRX4_RX.MIPI	Bottom Only
Transmit SDR	GOREG_TX.SCLK	Top & Bottom
TX DDRX1 Centered	GDDRX1_TX.SCLK.Centered	Top & Bottom
TX DDRX1 Aligned	GDDRX1_TX.SCLK.Aligned	Top & Bottom
TX DDRX2 Centered	GDDRX2_TX.ECLK.Centered	Bottom Only
TX DDRX2 Aligned	GDDRX2_TX.ECLK.Aligned	Bottom Only
TX DDRX4 Centered	GDDRX4_TX.ECLK.Centered	Bottom Only
TX DDRX4 Aligned	GDDRX4_TX.ECLK.Aligned	Bottom Only
TX DDRX5 Centered	GDDRX5_TX.ECLK.Centered	Bottom Only
TX DDRX5 Aligned	GDDRX5_TX.ECLK.Aligned	Bottom Only
TX DDR71	GDDR71_TX.ECLK	Bottom Only
TX DDRX4 MIPI	GDDRX4_TX.MIPI	Bottom Only

3.7.2. Clocking

- Only the Primary Clock (PCLK) resources are used when implementing x1 receive or transmit interfaces.
- The Edge Clock (ECLK) and Primary Clock resources are used when implementing 2×, 4×, or 5× receive or transmit interfaces.
- Each ECLK can only span up to one side of the device, hence, all the data bits in the ×2 interface must be locked to one side of the device.
- The Edge Clock to DDR registers can be accessed through dedicated PCLK pins, GPLL outputs, and DDRDLL outputs.
- The Primary Clock to DDR registers can be accessed through dedicated PCLK pins, GPLL outputs, and CLKDIV outputs.
- None of the clocks going to the DDR registers can come from internal general routing.

Refer to the Lattice Nexus 2 System Clock PLL Design User Guide (FPGA-TN-02364) for more information regarding the PCLK and ECLK resources.

3.7.3. Receiver Interfaces

- A differential DDR interface can be implemented on the bottom side of the device.
- Each HPIO bank on the bottom side of the device has four Edge Clocks that can be used to generate either a center or aligned interface. This means you can implement four different GDDRX2 RX interfaces per bank as each 2×, 4×, or 5× gearing requires an ECLKDIVA module to generate a slower SCLK.
- There is one DDRDLL per HPIO bank on the bottom side of the device.
- There are four DLLDELs per HPIO bank, and each one is associated with an ECLK. The delay code from DDRDLL is applied to all the DLLDEL modules within each HPIO bank.
- The Receive clock input must be placed on a dedicated PCLK input pin. The PCLK pin has direct access to the Edge Clock tree for a centered interface, and it also has a direct connection to the DLLDEL when implementing an aligned interface.

- When implementing the IDDRX71 interface, the complementary PAD is not available for other functions since the IDDRX71 uses the I/O registers of the complementary PAD as well.
- It is recommended that clock inputs be located on the same side as data pins.
- The top left and right sides of the device do not have Edge Clocks. Hence, these can only be used to receive lower-speed interfaces (<250 MHz) that use 1× gearing. These can be used for single-ended interfaces only.
- Interfaces using the ×1 gearing use the primary clock resource. You can use as many interfaces as the number of primary clocks supported by the device.
- In addition to the dedicated PCLK pins, The Lattice Nexus 2 devices have GR_PCLK pins. These pins use the shortest general route path to get to the primary clock tree. These pins are not recommended for use with DDR interfaces. They can be used for SDR or other generic FPGA designs.

3.7.4. Transmitter Interfaces

- Use PADA and PADB for all TX using true LVDS interfaces.
- When implementing Transmit Centered interface, two ECLKs are required. One to generate the Data Output and the other to generate the CLK Output.
- When implementing Transmit Aligned interface, only one ECLK is required for both Data output and Clock output.
- Each bank on the bottom side of the device has four ECLKDIVA modules, which means that you can implement up to four different GDDRX2 TX interfaces per bank since as 2×, 4×, or 5× gearing requires an ECLKDIVA module to generate a slower SCLK.
- The top left and right sides of the device do not have Edge Clocks. Hence, it can only be used to receive lower-speed interfaces (<250 MHz) that use 1X gearing. It can be used for single-ended interfaces only.
- Interfaces using the ×1 gearing use the primary clock resource. You can use as many interfaces as the number of primary clocks supported by the device.

4. External Memory Interfaces

The Lattice Nexus 2 device contains a hardened DDRPHY, which enables DDR3L, DDR4, and LPDDR4 external memory interface support.

4.1. Overview

The Lattice Nexus 2 offers two possible solutions when interfacing with external SDRAM:

- DDRPHY IP
- Memory Controller IP

4.1.1. DDRPHY IP (PHY-Only Interface)

The DDRPHY IP is an implementation of the DDR PHY Interface (DFI) v4.0 specification that provides the following:

- Physical interface between Memory Controller and external SDRAM.
- Supports DDR3L, DDR4, and LPDDR4 memory standards.
- Supports the APB interface for register configuration.

For more information regarding the APB protocol, refer to the AMBA APB Protocol Specification.

4.1.2. Memory Controller IP

The Memory Controller IP is an integrated IP that consists of a Memory Controller and DDRPHY and provides the following:

- Supports DDR3L, DDR4, and LPDDR4 memory standards.
- Supports the AXI4 interface for data access.
- Supports the APB interface for register configuration.

For more information regarding the AXI4 and APB protocols, refer to the AMBA AXI Protocol Specification and the AMBA APB Protocol Specification.

4.2. Design Rules and Guidelines

4.2.1. Pin and Resource Planning

The Lattice Nexus 2 device contains dedicated I/O functions for supporting DDR memory interfaces. External memory interfaces are supported in the DDRPHY, where each DDRPHY is composed of three High Performance I/O (HPIO) banks. These banks are labeled as HIGHSPEED in the device pinout tables. As The Lattice Nexus 2 leverages a hardened PHY, the external memory interface pinouts (DQ, DM, and DQS) are in fixed locations, specified under DDRPHY in the device pinout tables. Dedicated clock routing within HPIO banks is represented as PLL or PCLK, and dedicated reference voltage pins are represented as VREF, in the device pinout tables. Refer to the Hardened DDR Physical Layer (DDRPHY) section of this user guide and the pinout files located on the Lattice Nexus 2 web pages on www.latticesemi.com for more information.

The following pinout rules must be followed to effectively use the dedicated I/O functions:

- All associated pins for a single external memory interface (DQ, DM, and DQS) must reside within the same DDRPHY located at the bottom of the Lattice Nexus 2 device.
  - Example: For a 64-bit LPDDR4 interface, all DQ, DM, and DQS pins must be placed in either DDRPHY0, DDRPHY1, or DDRPHY2. The signals cannot be split or rearranged.
- A DQS# pad is auto placed when a differential I/O type is selected and must not be manually assigned (SSTL135D in DDR3L, POD12D in DDR4, POD11D in and LVSTL11D in LPDDR4).
- The input reference clock to the PLL must be differential and assigned to use dedicated clock routing (PLL or PCLK). It is strongly recommended to place the input reference clock on the pin closest to the dedicated PLL within each DDRPHY (labeled as PLL under DDRPHY in device pinout tables). This ensures better performance by minimizing jitter/routing.

- At least one VREF pin per HPIO bank where the DQ, DMI, and DQS pads are located must be available and used as a reference voltage input.
- Unused pins within a DDRPHY can be used as general-purpose I/O if the following restrictions are observed:
  - For the banks that do not contain any external memory interface signals, the available pins can be used as GPIO. Refer to Figure 4.1 for more details, where each lane marked in green is available as a GPIO.
  - For banks that contain external memory interface signals. Any remaining pins cannot be used as GPIO. Refer to Figure 4.1 for more details, where each lane marked in red is not available as GPIO.

Lattice Nexus 2 FPGAs can support up to a x72 interfaces (including ECC), which would occupy 2 HPIO banks. Unused I/O can be configured as GPIO if the voltage matches that of the SDRAM standard. Pin mapping for the DQ/DQS and CA lanes is shown in Figure 4.1, along with the pin assignments for a x72 interface in Table 4.1.

					Bank 11				Bank 10				Bank 9			
Type	CA_Lanes	Mem_Width	Lane_CNT	Bank_CNT	12	12	12	12	12	12	12	12	12	12	12	12
DDR3L/DDR4	3	16	5	2				CA2	CA1	CA0	8	8				
	3	32	7	2		8	8	CA2	CA1	CA0	8	8				
	3	40	8	2	8	8	8	CA2	CA1	CA0	8	8				
	3	64	11	3	8	8	8	CA2	CA1	CA0	8	8	8	8	8	
	3	72	12	3	8	8	8	CA2	CA1	CA0	8	8	8	8	8	8
LPDDR4	2	16	4	1					CA1	CA0	8	8				
	2	32	6	2		8	8		CA1	CA0	8	8				
	2	64	10	3	8	8	8		CA1	CA0	8	8	8	8	8	
					x72											
D/CA					8	8	8	CA2	CA1	CA0	8	8	8	8	8	8
Bank/DQS Group Mapping			MH20/CT20		x72 DDRx Memory Interface + 3 HPIO Banks											

Notes:
Green colored lanes can support GPIO
Red colored lanes cannot support GPIO

Figure 4.1. Lattice Nexus 2 Memory Interface Mapping Diagram

Table 4.1. x72 DIMM Pin Assignments

Lane	Bank	Group#(L→R)	I/O Port
DQ / DQS Lane 0	Bank# 10	Group# 2	DQ[7:0] DQS[0] / DQS_n[0] DM[0]
DQ / DQS Lane 1	Bank# 10	Group# 3	DQ[15:8] DQS[0] / DQS_n[1] DM[1]
DQ / DQS Lane 2	Bank# 11	Group# 1	DQ[23:16] DQS[0] / DQS_n[2] DM[2]
DQ / DQS Lane 3	Bank# 11	Group# 2	DQ[31:24] DQS[0] / DQS_n[3] DM[3]
DQ / DQS Lane 4	Bank# 11	Group# 0	DQ[39:32] DQS[0] / DQS_n[4] DM[4]
DQ / DQS Lane 5	Bank# 9	Group# 0	DQ[47:40] DQS[5] / DQS_n[5] DM[5]
DQ / DQS Lane 6	Bank# 9	Group# 1	DQ[55:48] DQS[6] / DQS_n[6] DM[6]

Lane	Bank	Group#(L→R)	I/O Port
DQ / DQS Lane 7	Bank# 9	Group# 2	DQ[63:56] DQS[7] / DQS_n[7] DM[7]
DQ / DQS Lane 8	Bank# 9	Group# 3	DQ[71:64] DQS[8] / DQS_n[8] DM[8]
CA Lane 0	Bank# 10	Group# 0	A[8:0] CLK0_p / CLK0_n
CA Lane 1	Bank# 10	Group# 1	A[13:9] CLK1_p / CLK1_n ODT[1:0] CS_n[1:10]
CA Lane 2	Bank# 11	Group# 3	A[15:14] CLK2_p / CLK2_n CLK3_p / CLK3_n BA[2:0] WE_n RAS_n

4.2.2. FPGA Input Reference Clock

The reference oscillator voltage cannot exceed the VCCIO voltage of 1.35 V/1.2 V/1.1 V for DDR3L/DDR4/LPDDR4. To drive the reference clock for DDR3L/DDR4/LPDDR4, a DC-coupled HCSL oscillator is recommended as shown in the figure below.

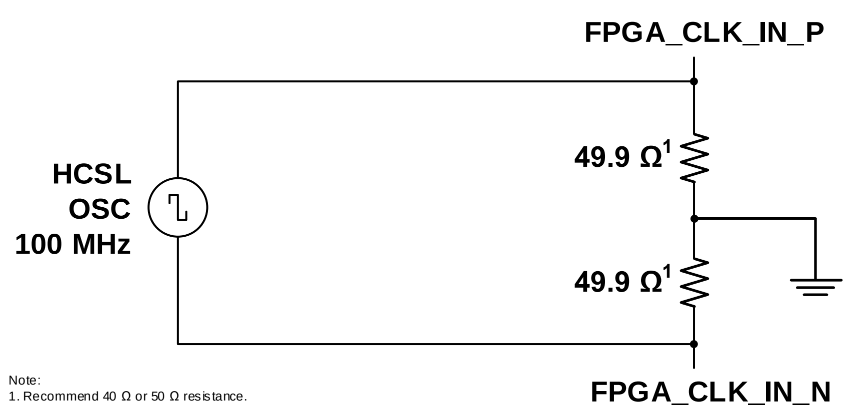


Figure 4.2. HCSL Reference Clock Circuit

Alternatively, an AC-coupled LVDS oscillator with a DC bias termination circuit close to the FPGA clock pins can be used as shown in the Figure 4.3.

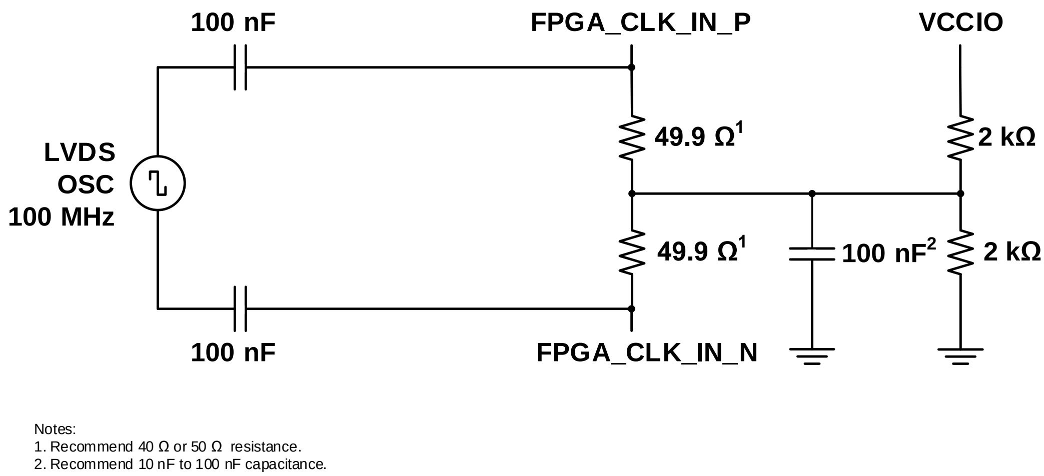


Figure 4.3. LVDS Reference Clock Circuit

The LVDS clock circuit can also be used to support HSCL oscillators in a single PCB footprint as shown in Figure 4.4. It is recommended to use 0201-size resistors and capacitors to allow placement beneath the FPGA (opposite side of the PCB) and close to the clock input pins.

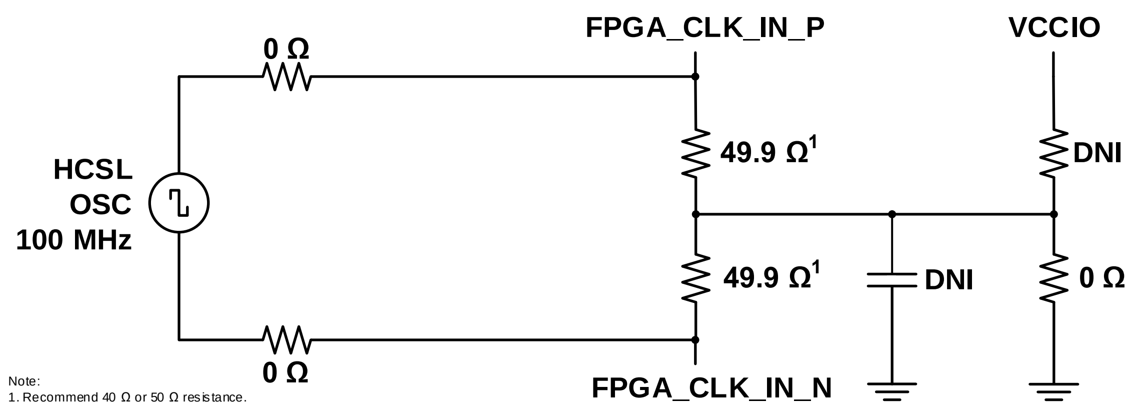


Figure 4.4. HCSL Oscillator on LVDS PCB Footprint

4.2.3. DDRPHY Trace and Skew Requirements

The following table defines combined board and package skew requirements for DQS, DQ, DM, and Clock, Address, and Control signals based on operating frequency. Skew is defined as the maximum and minimum time difference, or propagation delay, among signals.

Table 4.2. Board and Package Trace/Skew Requirements

	1200 MHz Combined Skew (ps)	1066 MHz Combined Skew (ps)	933 MHz Combined Skew (ps)	800 MHz Combined Skew (ps)	666 MHz Combined Skew (ps)	533 MHz Combined Skew (ps)	400 MHz Combined Skew (ps)
Skew on DQ/DM/DQS per byte lane	24	28	30	45	60	90	120
Differential Skew on DQS/DQSB/CLK/CLKB per byte lane	6	8	12	15	15	15	15
Skew on Clock, Address and Control signals to every DRAM part	30	35	40	55	75	90	110
Delay difference between CLK and DQS	1000	1000	1000	1000	1000	1000	1000
Skew on DQ/DM/DQS per byte lane across each CS	25	28	30	35	45	60	75
Skew on Address and Control signals per byte Lane across each CS	25	28	30	35	45	60	75

To ensure proper operation of external memory interfaces, the trace and skew requirements captured in must be met. To extract the package skew for a given Radiant design, refer to the following steps:

1. Launch the **Device Constraint Editor** in the Lattice Radiant software by selecting **Tools > Device Constraint Editor**. Note that the Radiant design must first be synthesized.

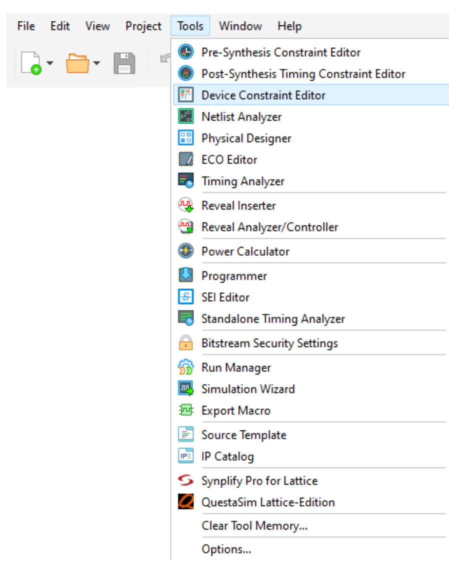


Figure 4.5. Launching Device Constraint Editor

2. Within the Device Constraint Editor, export the pin layout by selecting **File** > **Export** > **Pin Layout File**.

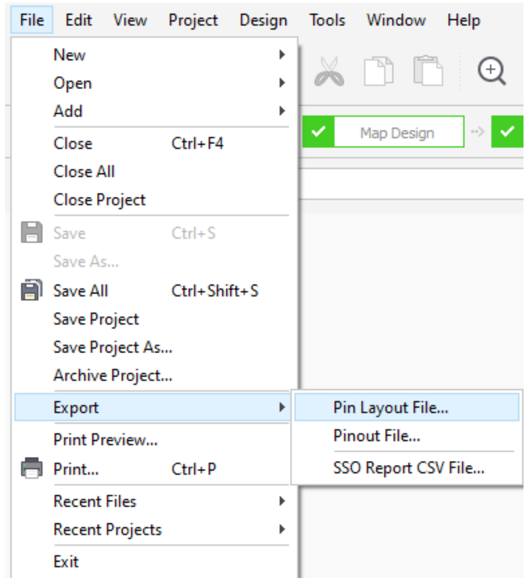


Figure 4.6. Exporting Pin Layout

3. In the **Export Pin Layout** window, enable the **Trace Length** and **Package Delay** options. Click **Ok**.

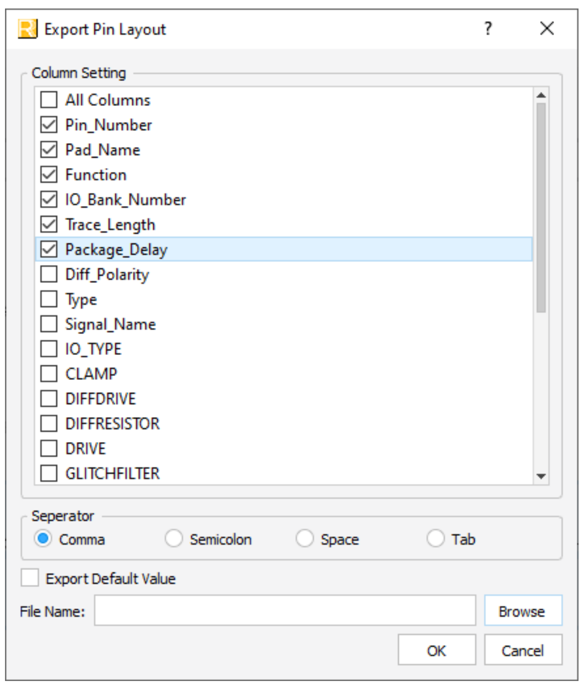


Figure 4.7. Exporting Trace Length and Package Delay

4. Click on **Browse**, specify a name for the file, and click **Ok** twice.

This will by default, export a file in CSV format with associated data for all ports in the Radiant design, where the package skew corresponds to the package delay column. Boards need to be designed so that the combination of package skew and board skew does not exceed the skew requirements in Table 4.2.

4.2.4. DDR3L Board Design Guidelines

4.2.4.1. External Resistance for PVT Calibration

All supported HPIO standards have calibration capability. High-speed I/O must be calibrated to meet performance requirements (DDR, LVDS, SLVS, etc.). Calibration is recommended, but optional for low-speed standards (LVCMOS).

When using HPIO banks, an external reference resistor is needed for PVT calibration.

- DDR3L interfaces require a 240 Ω ±1% tied to GND for PVT calibration.

The Lattice Nexus 2 can contain up to three DDRPHY's, where each DDRPHY consists of up to 3 HPIO banks. There are two methods of providing this external resistance:

- Each HPIO bank uses its own resistors.
- Each DDRPHY can have the middle HPIO banks (4, 7, 10) use its resistor to drive all three banks used by the DDRPHY interface.

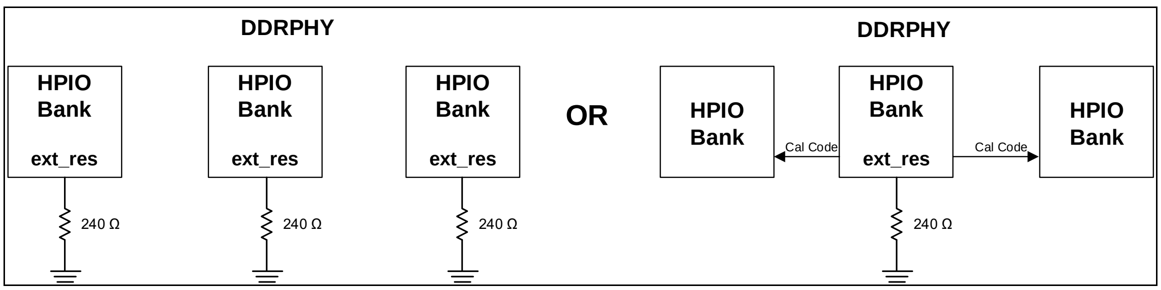


Figure 4.8. External Reference Resistor for SSTL (DDR3L)

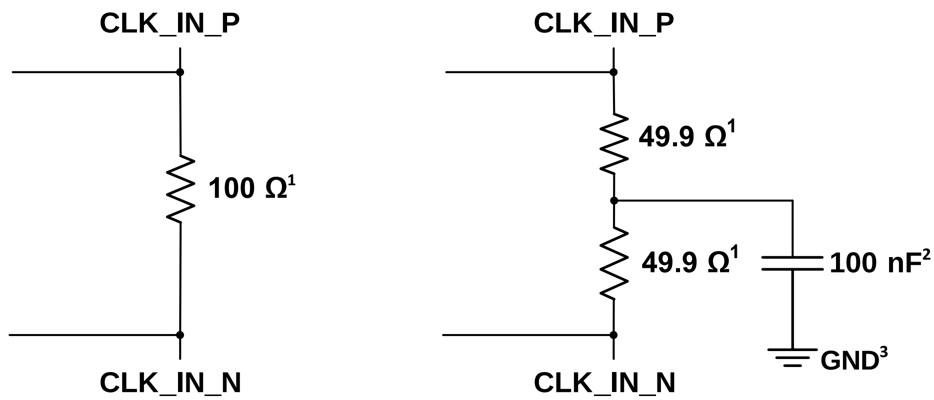
4.2.4.2. Termination for DQ, DQS, and DM

- Memory side termination on DQ, DQS, and DM is dynamically controlled by the DDR3L SDRAM ODT function (no external termination components are needed).
- The Lattice Nexus 2 device has an internal termination on DQ and DQS, which is dynamically controlled by the FPGA's ODT function (no external termination components are needed).
- Use the TERMINATION preference for DQ and DQS pads to enable the internal parallel termination to V_{CCIO}/2. The TERMINATION preference has options: OFF, 40 Ω, 50 Ω, and 60 Ω.

4.2.4.3. Termination for CK

Differential DDR memory clocks require differential termination. Use SSTL135D to drive the clock signals for DDR3L memory. 100 Ω differential termination on the memory side can be achieved using one of the following methods:

- Place a 100 Ω resistor between the positive and negative clock signals.
- Connect one end of an Rtt resistor to the positive pin and one end of another Rtt to the negative pin of a CK pair. Connect the other ends of the two Rtt resistors and return to VDD or GND through Ctt capacitance.
- The JEDEC CK termination scheme defined in the DIMM specifications uses 36 Ω for Rtt, with 0.1 µF Ctt returning to VDD. For non-DIMM applications, 50 Ω Rtt with 0.1 µF Ctt can be used.



Notes:
1. For differential etches, recommend 80 Ω or 100 Ω resistance. For single-ended etches, recommend 40 Ω or 50 Ω resistance.
2. Recommend 10 nF to 100 nF capacitance.
3. For clock routed referenced to VTT plane, connect to VTT. For clock routed referenced to GND plane, connect to GND.

Figure 4.9. Memory Differential Termination for DDR3L

The use of series termination resistors on the FPGA side is not recommended. When fly-by wiring is used in DDR3L, the CK termination resistor must be located after the last DDR3L SDRAM device for non-DIMM applications.

4.2.4.4. Routing/Termination for Address, Command, and Control

- Place a 40 Ω or 50 Ω parallel-to-VTT resistor to match trace impedance, or a best-known resistance obtained from SI simulations, on each address, command, and control line on the memory side.
- When fly-by wiring is used in DDR3L, the address, command, and control termination resistors must be located after the last DDR3L SDRAM device.

4.2.4.5. Termination for DIMM

The DDR3L DIMMs incorporate internal termination following the requirements defined by the JEDEC DIMM specification. For this reason, the DDR3L DIMM termination requirements are different from DDR3L SDRAM devices:

- Memory side differential termination on CK is built into the DIMM memory module (no external termination components are needed).

4.2.5. DDR4 Board Design Guidelines

4.2.5.1. External Resistance for PVT Calibration

All supported HPIO standards have calibration capability. High-speed I/O must be calibrated to meet performance requirements (DDR, LVDS, SLVS, etc.). Calibration is recommended, but optional for low-speed standards (LVCMOS). When using HPIO banks, an external reference resistor is needed for PVT calibration

- For DDR4 interfaces, a 240 Ω ±1% tied to GND is required for PVT calibration.

The Lattice Nexus 2 devices can contain up to three DDRPHY's, where each DDRPHY consists of up to 3 HPIO banks. There are two methods of providing this external resistance:

- Each HPIO bank uses its own resistors.
- Each DDRPHY can have the middle HPIO banks (4, 7, 10) use its resistor to drive all three banks used by the DDRPHY interface.

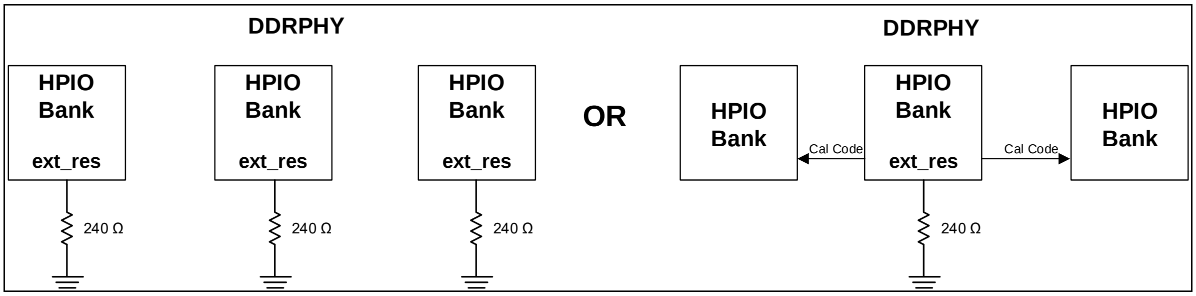


Figure 4.10. External Reference Resistor for POD (DDR4)

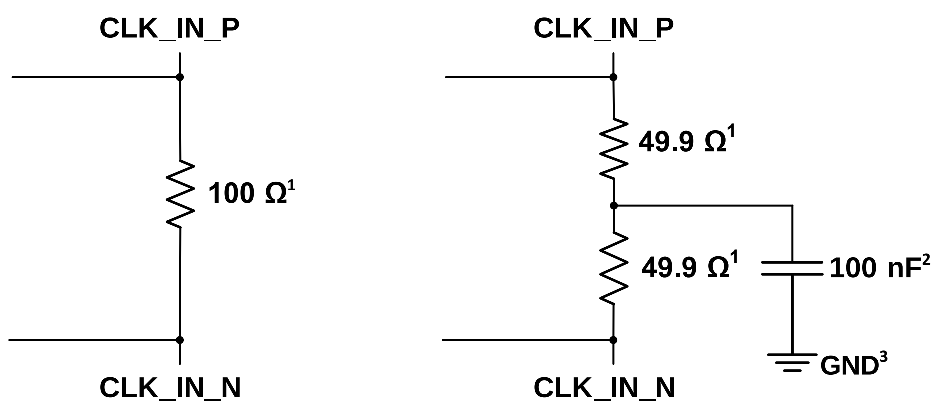
4.2.5.2. Termination for DQ, DQS, and DN

- Memory side termination on DQ, DQS, and DM is dynamically controlled by the DDR4 SDRAM ODT function (no external termination components are needed).
- The Lattice Nexus 2 device has internal termination on DQ and DQS, which is dynamically controlled by the FPGA's ODT function (no external termination components are needed)
- Use the TERMINATION preference for DQ and DQS pads to enable the internal parallel termination to GND. The TERMINATION preference has options: OFF, 34 Ω, 40 Ω, 48 Ω, 60 Ω, 80 Ω, 120 Ω, and 240 Ω.

4.2.5.3. Termination for CK

Differential DDR memory clocks require differential termination, where SSTL12D is used to drive the clock signals for DDR3L memory. Differential termination on the memory side can be achieved using one of the following methods:

- Connect a single Rtt resistor across the positive and negative pins for the clock signal.
- Place AC termination using split Rtt resistors and a center-tapped Ctt capacitor returning to VTT or GND.



Notes:
1. For differential etches, recommend 80 Ω or 100 Ω resistance. For single-ended etches, recommend 40 Ω or 50 Ω resistance.
2. Recommend 10 nF to 100 nF capacitance.
3. For clock routed referenced to VTT plane, connect to VTT. For clock routed referenced to GND plane, connect to GND.

Figure 4.11. Memory Differential Termination for DDR4

The use of series termination resistors on the FPGA side is not recommended. When fly-by wiring is used in DDR4, the CK termination resistor must be located after the last DDR4 SDRAM device for non-DIMM applications.

4.2.5.4. Routing/Termination for Address, Command, and Control

For DDR4 external memory interfaces, it is recommended to adopt fly-by topology for Address, Command, and Control signals.

- Place a 40 Ω or 50 Ω parallel-to-VTT resistor to match trace impedance, or a best-known resistance obtained from SI simulations, on each address, command, and control line on the memory side.
- When fly-by wiring is used in DDR4, the address, command, and control termination resistors must be located after the last DDR4 SDRAM device.

4.2.5.5. Termination for DIMM

The DDR4 DIMMs incorporate internal termination following the requirements defined by the JEDEC DIMM specification. For this reason, DDR4 DIMM termination requirements are different from DDR4 SDRAM devices:

- Memory side differential termination on CK is built into the DIMM memory module (no external termination components are needed).

4.2.6. LPDDR4 Board Design Guidelines

4.2.6.1. External Resistance for PVT Calibration

All supported HPIO standards have calibration capability. High-speed I/O must be calibrated to meet performance requirements (DDR, LVDS, SLVS, etc.). Calibration is recommended, but optional for low-speed standards (LVCMOS). When using HPIO banks, an external reference resistor is needed for PVT calibration.

- For LPDDR4 interfaces using LVSTL_I (VCCIO/3), a 240 Ω ±1% tied to V_{CCIO} is required for PVT calibration.
- For LPDDR4 interfaces using LVSTL_II (VCCIO/2.5), a 180 Ω ±1% tied to V_{CCIO} is required for PVT calibration.

The Lattice Nexus 2 devices can contain up to three DDRPHY's, where each DDRPHY consists of up to 3 HPIO banks. There are two methods of providing this external resistance:

- Each HPIO bank uses its own resistors.
- Each DDRPHY can have the middle HPIO banks (4, 7, 10) use its resistor to drive all three banks used by the DDRPHY interface.

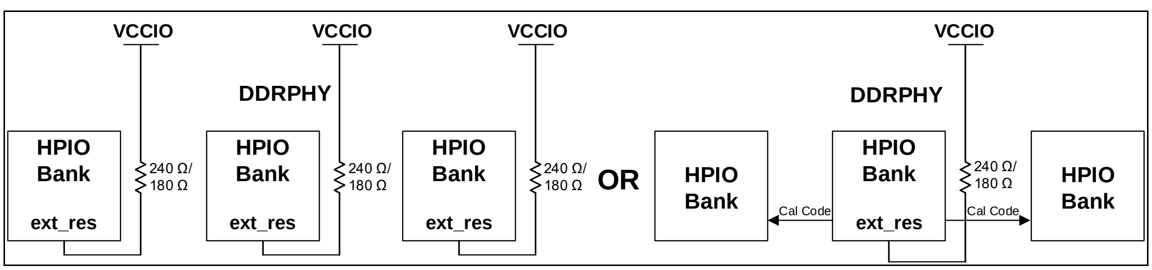


Figure 4.12. External Reference Resistor for LVSTL_I/LVSTL_II (LPDDR4)

4.2.6.2. Termination for DQ, DQS, and DM

- Memory side termination on DQ, DQS, and DM is dynamically controlled by the LPDDR4 SDRAM ODT function (no external termination components are needed).
- The Lattice Nexus 2 device has internal termination on DQ and DQS, which is dynamically controlled by the FPGA's ODT function (no external termination components are needed).
- Use the TERMINATION preference for DQ and DQS pads to enable the internal parallel termination to GND. The TERMINATION preference has options: OFF, 34 Ω, 40 Ω, 48 Ω, 60 Ω, 80 Ω, 120 Ω, and 240 Ω.

4.2.6.3. Termination for CK

No board termination is required for the LPDDR4 CK as it has ODT termination built into the memory.

4.2.6.4. Routing/Termination for Address, Command, and Control

For LPDDR4 external memory interfaces, it is recommended to adopt T-topology for Address, Command, and Control signals. No board termination is required on the LPDDR4 CA bus and control as it has ODT termination built into the memory.

5. User Primitives and Attributes

This section describes the user primitives for I/O logic that can be used to implement all SDR/DDR interfaces, MIPI interfaces, and external memory interfaces (DDR3L/DDR4/LPDDR4).

Table 5.1. Software Primitives

Type	Primitive	Description
DDR Delay-Locked Loop	DDRDLLA	Dedicated Delay Lock Loop for DDR
DLL Delay	DLLDELA	Delay for DDRDLL in Generic DDR RX modes
Delay	DELAYD	Dynamic Input/Output Delay Element for SCLK Interface
	DELAYE	Static Input/Output Delay Element for SCLK Interface
	DELAYF	Dynamic Input/Output Delay Element for ECLK Interface
	IMONDELAY	Dynamic Input Delay with Edge Monitoring (IMON) for ECLK Interface
SDR/DDR Input	IFD1P3DX	Input D Flip-Flop for SDR RX modes
	IDDRX1A	Generic ×1 IDDR
	IDDRX2A	Generic ×2 IDDR
	IDDRX4A	Generic ×4 IDDR
	IDDRX5A	Generic ×5 IDDR
	IDDR71A	7:1 LVDS IDDR
DDR Data Output	OFD1P3DX	Output D Flip-Flop for SDR TX modes
	ODDRX1A	Generic ×1 ODDR
	IDDRFIFOX1DQ	Used for Octal SPI operation
	IDDRFIFOX1DS	Used for Octal SPI operation
	IDDRX1B	Used for Octal SPI operation
	ODDRX1B	Used for Octal SPI operation
	ODDRX2A	Generic ×2 ODDR
	ODDRX4A	Generic ×4 ODDR
	ODDRX5A	Generic X5 ODDR
	ODDR71A	7:1 LVDS ODDR
MIPI	MIPIA	Soft version of bidirectional MIPI
DDR3L/DDR4	DDRPHY16D	16-bit wide DDR3L/DDR4 Interface
	DDRPHY32D	32-bit wide DDR3L/DDR4 Interface
	DDRPHY40D	40-bit wide DDR3L/DDR4 Interface
	DDRPHY64D	64-bit wide DDR3L/DDR4 Interface
	DDRPHY72D	72-bit wide DDR3L/DDR4 Interface
LPDDR4	DDRPHY16E	16-bit wide LPDDR4 Interface
	DDRPHY32E	32-bit wide LPDDR4 Interface
	DDRPHY64E	64-bit wide LPDDR4 Interface

5.1. DDRDLLA

The DDRDLL is a dedicated DLL for creating a 90 degrees clock delay. The DDRDLL outputs delay codes that are used in:

- Generic DDR interfaces delay the data input.
- DLLDEL module to delay the input clock of GDDR.
- External Memory interfaces are used to delay the DQS input and DQ output.

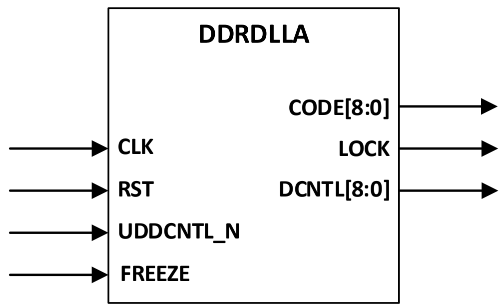


Figure 5.1. DDRDLLA Ports

Table 5.2. DDRDLLA Ports

Port		Range	Type	Active State	SW Default	Description
Input	CLK	1	Data	Rising Edge	—	Clock input
	RST	1	Control	High	0	Reset input
	UDDCNTL_N	1	Control	Low	0	Update control to update the delay code. 0: the delay code out of the DDRDLL is updated. Should not be active during a read or a write cycle
	FREEZE	1	Control	High	0	Releases the DDRDLL input clock
Output	CODE	8:0	Data	—	—	The delay codes from the DDRDLL to be used to delay input of DQS, DLLDEL, and output of external memory interfaces
	LOCK	1	Data	—	—	Lock output to indicate the DDRDLL has valid delay output
	DCNTL	8:0	Data	—	—	The delay codes from the DDRDLL are available for the user IP. Each step size is about 12.5ps[1]

Note:

1. In simulation, the delay step size is about ~10ps due to timescale precision limitations.

Table 5.3. DDRDLLA Parameters

Name	Values	Default	Description
CODE_LOCK_GENERATION	DLL_LOOP FORCE_MAX	DLL_LOOP	DLL control code forcing. DLL_LOOP: Code/lock generated from DLL loop. FORCE_MAX: Force lock and code are set to maximum of 255 for lower input frequency mode
ROUNDOFF	ENABLED DISABLED	ENABLED	DLL counter round-off select. ENABLED: Enable round-off, more stable code. DISABLED: Disable round-off
DIV	1 2	1	Clock divider
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.2. DLLDELA

The target delay element is used to support generic DDR RX modes by shifting the input clock phase by 90 degrees.

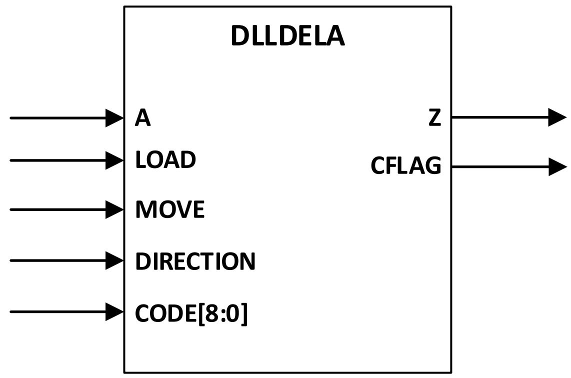


Figure 5.2. DLLDELA Ports

Table 5.4. DLLDELA Ports

Port		Range	Type	Active State	SW Default	Description
Input	A	1	Data	High	0	Data input from pin
	LOAD	1	Control	High	1	LOAD is set to 1 to reset delay to: CODE + DEL_VALUE if DEL_ADJ=PLUS. CODE – DEL_VALUE if DEL_ADJ=MINUS
	MOVE	1	Control	High	0	Pulse MOVE to: Increment delay setting by 1 if DIRECTION is 0. Decrement delay setting by 1 if DIRECTION is 1. MOVE needs to meet a 6ns minimum pulse width requirement
	DIRECTION	1	Control	High	0	Controls whether MOVE increases or decreases the delay setting. 0 to increment 1 to decrement
	CODE	9	Control	High		Delay code from DDRDLL
Output	Z	1	Data	—	—	Delayed data to input register block
	CFLAG	1	Flag	—	—	Flag indicating the delay counter has reached max (when moving up) or min (when moving down) value

Table 5.5. DLLDELA Parameters

Name	Values	Default	Description
DEL_ADJ	PLUS MINUS	PLUS	DLL control code forcing. DLL_LOOP: Code/lock generated from DLL loop. FORCE_MAX: Force lock and code are set to maximum of 511 for lower input frequency mode
DEL_VALUE	0 – 511	0	Offset for target delay adjustment
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.3. Input/Output Delay User Primitives

The input and output delay user primitives are available for both WRIO and HPIO, except for input delay with IMON, which is available for HPIO only. All delay user primitives are not inferable by synthesis tools. The following 4 delay primitives support static or dynamic delay and the IMON feature.

5.3.1. DELAYD

DELAYD can be used as either input delay or output delay for SCLK interfaces with dynamic delay control.

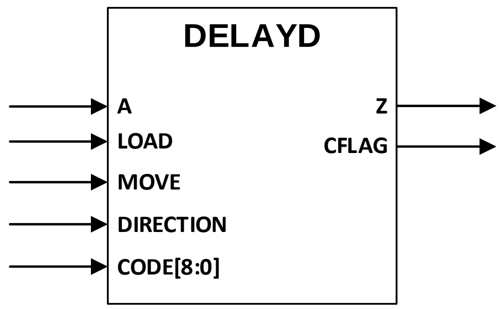


Figure 5.3. DELAYD Ports

Table 5.6. DELAYD Ports

Port		Range	Type	Active State	SW Default	Description
Input	A	1	Data	High	0	Data input from pin
	LOAD	1	Control	High	1	LOAD is set to 1 to reset to delay setting (DEL_VALUE)
	MOVE	1	Control	High	0	Pulse MOVE to: Increment delay setting by 1 if DIRECTION is 0. Decrement delay setting by 1 if DIRECTION is 1. MOVE needs to meet a 6ns minimum pulse width requirement
	DIRECTION	1	Control	High	0	Controls whether MOVE increases or decreases the delay setting. 0 to increment 1 to decrement
	CODE	8:0	Data	High	0	Delay Code from DDRDLL
Output	Z	1	Data	—	—	Delay Output
	CFLAG	1	Flag	—	—	Flag indicating the delay counter has reached max (when moving up) or min (when moving down) value

Table 5.7. DELAYD Parameters

Name	Values	Default	Description
DEL_MODE	USER_DEFINED SCLK_ZEROHOLD SCLK_ALIGNED SCLK_CENTERED	USER_DEFINED	Set delay mode
DEL_VALUE	0 – 511	0	Static Delay Value
DEL_SOURCE	DEL_VALUE DDRDLL	DEL_VALUE	Static Delay Value Source

5.3.2. DELAYE

DELAYE can be used as either an input delay or an output delay for SCLK interfaces with static delay.

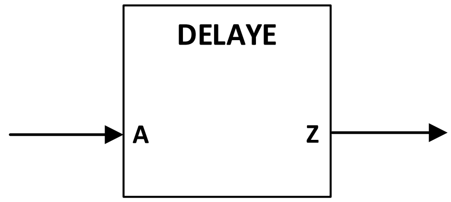


Figure 5.4. DELAYE Ports

Table 5.8. DELAYE Ports

Port		Range	Type	Active State	SW Default	Description
Input	A	1	Data	High	0	Data input from pin
Output	Z	1	Data	—	—	Delay Output

Table 5.9. DELAYE Parameters

Name	Values	Default	Description
DEL_MODE	USER_DEFINED SCLK_ZEROHOLD SCLK_ALIGNED SCLK_CENTERED	USER_DEFINED	Set delay mode
DEL_VALUE	0 – 511	0	Static Delay Value

5.3.3. DELAYF

DELAYF can be used as either an input delay or an output delay for ECLK interfaces with dynamic delay control.

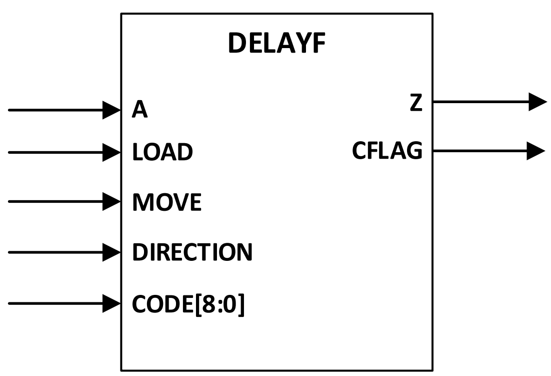


Figure 5.5. DELAYF Ports

Table 5.10. DELAYF Ports

Port		Range	Type	Active State	SW Default	Description
Input	A	1	Data	High	0	Data input from pin
	LOAD	1	Control	High	1	LOAD is set to 0 to reset to delay setting (DEL_VALUE)
	MOVE	1	Control	High	0	Pulse MOVE to: Increment delay setting by 1 if DIRECTION is 0. Decrement delay setting by 1 if DIRECTION is 1. MOVE needs to meet a 6ns minimum pulse width requirement
	DIRECTION	1	Control	High	0	Controls whether MOVE increases or decreases the delay setting. 0 to increment 1 to decrement
	CODE	8:0	Data	High	0	Delay Code from DDRDLL
Output	Z	1	Data	—	—	Delay Output
	CFLAG	1	Flag	—	—	Flag indicating the delay counter has reached max (when moving up) or min (when moving down) value

Table 5.11. DELAYF Parameters

Name	Values	Default	Description
DEL_MODE	USER_DEFINED ECLK_ALIGNED ECLK_CENTERED	USER_DEFINED	Set delay mode
DEL_VALUE	0 – 511	0	Static Delay Value
DEL_SOURCE	DEL_VALUE DDRDLL	DEL_VALUE	Static Delay Value Source

5.3.4. IMONDELAY

IMONDELAY is for ECLK interfaces with dynamic control delay and edge monitoring (IMON).

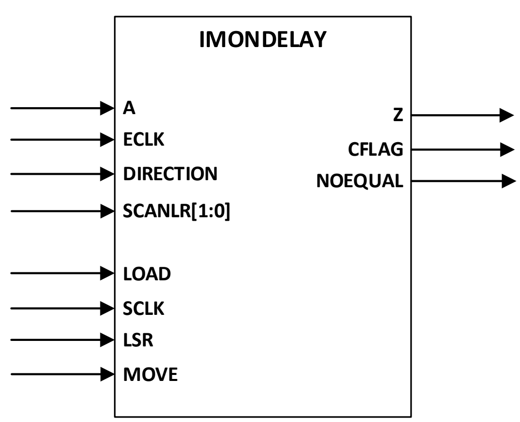


Figure 5.6. IMONDELAY Ports

Table 5.12. IMONDELAY Ports

Port		Range	Type	Active State	SW Default	Description
Input	A	1	Data	High	0	Data input from pin
	SCLK	1	Clock	High	—	Primary clock for IREG/IDDR
	ECLK	1	Clock	Rising Edge	—	Fast Edge Clock
	LSR	1	Reset		—	Local Set/Reset
	LOAD	1	Control	High	—	If SCANLR==2'b00: Sets the DATA Delay to DEL_VALUE If SCANLR==2'b01: Sets TEST Delay to current DATA delay + DEL_ADJUST If SCANLR==2'b10: Sets TEST Delay to current DATA delay - DEL_ADJUST
	DIRECTION	1	Control	High	0	0 to make MOVE increase delay 1 to make MOVE decrease delay
	MOVE	1	Control	—	—	2 SCLK MPW to guarantee proper IMON functionality. If SCANLR==2'b00: Adjusts the delay of DATA path. If SCANLR==2'b01 or 2'b10: Adjusts the delay of TEST path.
	SCANLR	1:0	Control	High	00	Controls whether DATA path or TEST path is being dynamically adjusted. 2'b00: Control DATA path 2'b01 or 2'b10: Control TEST path
Output	Z	1	Data	—	—	Delayed data to input register block
	CFLAG	1	Flag	—	—	Flag indicating the delay counter has reached max (when moving up) or min (when moving down) value
	NOEQUAL	1	Flag	—	—	Flag indicating either the DATA or TEST path is too close to the active edge of ECLK. This flag is constantly evaluating in the background. It is expected to go low briefly after each MOVE pulse to prevent evaluations while the delay is being adjusted.

Table 5.13. IMONDELAY Parameters

Name	Values	Default	Description
DEL_MODE	USER_DEFINED ECLK_ALIGNED ECLK_CENTERED	USER_DEFINED	Set delay mode
DEL_VALUE	0 – 511	0	Static Delay Value
DEL_ADJUST	0 – 63	0	Test path window adjustment. See LOAD description.

5.4. Input SDR/DDR User Primitives

The following are the primitives used to implement various generic SDR/DDR input configurations.

5.4.1. IFD1P3DX

The IFD1P3DX primitive is used to receive generic SDR.

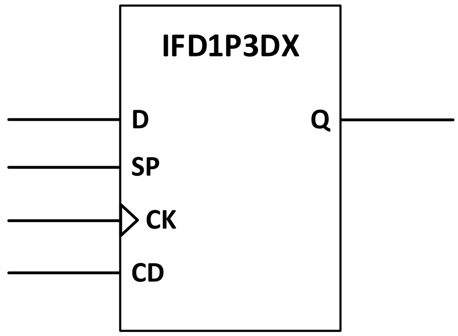


Figure 5.7. IFD1P3DX Ports

Table 5.14. IFD1P3DX Ports

Port		Range	Type	Active State	SW Default	Description
Input	D	1	Data	High	0	Data Input
	CK	1	Clock	Rising Edge	—	Clock Input
	SP	1	Control	High	—	Positive Clock Enable
	CD	1	Control	High	—	Asynchronous Reset
Output	Q	1	Data	—	—	Data Output

Table 5.15. IFD1P3DX Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.4.2. IDDRX1A

The IDDRX1A primitive is used to receive generic DDR with 1X gearing.

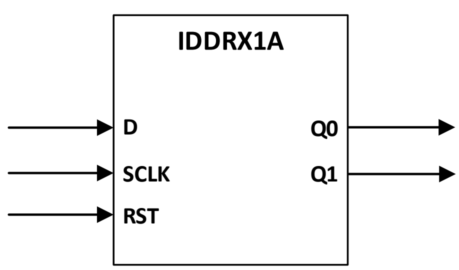


Figure 5.8. IDDRX1A Ports

Table 5.16. IDDRX1A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D	1	Data	High	0	DDR data input
	SCLK	1	Clock	Rising Edge	—	Primary clock input
	RST	1	Control	High	0	Reset to DDR registers
Output	Q0	1	Data	—	—	Data at the positive edge of the clock
	Q1	1	Data	—	—	Data at the negative edge of the clock

Table 5.17. IDDRX1A Parameter

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.4.3. IDDRX2A

The IDDRX2A primitive is used to receive generic DDR with 2X gearing.

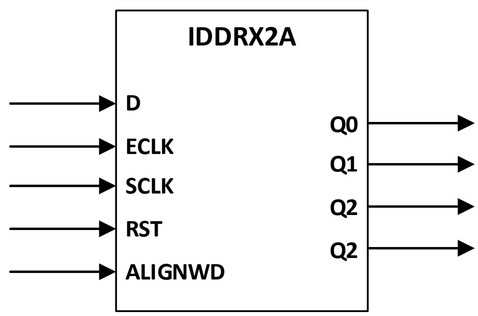


Figure 5.9. IDDRX2A Ports

Table 5.18. IDDRX2A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D	1	Data	High	0	DDR data input
	ECLK	1	Clock	Rising Edge	—	Fast edge clock
	SCLK	1	Clock	Rising Edge	—	Primary clock input (divide-by-2 of ECLK)
	RST	1	Control	High	0	Reset to DDR registers
	ALIGNWD	1	Control	High	0	This signal is used for word alignment. It shifts the word by one bit.
Output	Q0, Q2	2	Data	—	—	Data at the positive edge of the clock
	Q1, Q3	2	Data	—	—	Data at the negative edge of the clock

Table 5.19. IDDRX2A Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.4.4. IDDRX4A

The IDDRX4A primitive is used to receive generic DDR with 4X gearing.

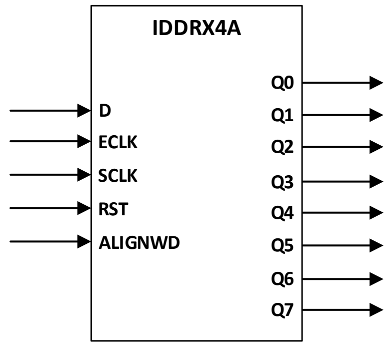


Figure 5.10. IDDRX4A Ports

Table 5.20. IDDRX4A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D	1	Data	High	0	DDR data input
	ECLK	1	Clock	Rising Edge	—	Fast edge clock
	SCLK	1	Clock	Rising Edge	—	Primary clock input (divide-by-4 of ECLK)
	RST	1	Control	High	0	Reset to DDR registers
	ALIGNWD	1	Control	High	0	This signal is used for word alignment. It shifts the word by one bit.
Output	Q0, Q2, Q4, Q6	4	Data	—	—	Data at the positive edge of the clock
	Q1, Q3, Q5, Q7	4	Data	—	—	Data at the negative edge of the clock

Table 5.21. IDDRX4A Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.4.5. IDDRX5A

The IDDRX5A primitive is used to receive generic DDR with 5X gearing.

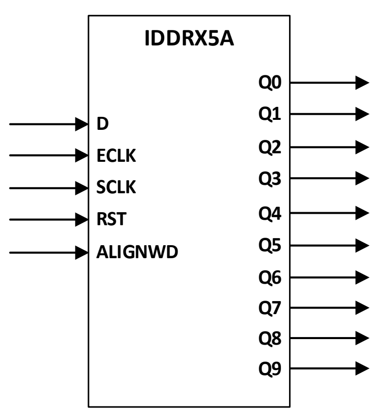


Figure 5.11. IDDRX5A Ports

Table 5.22. IDDRX5A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D	1	Data	High	0	DDR data input
	ECLK	1	Clock	Rising Edge	—	Fast edge clock
	SCLK	1	Clock	Rising Edge	—	Primary clock input (divide-by-5 of ECLK)
	RST	1	Control	High	0	Reset to DDR registers
	ALIGNWD	1	Control	High	0	This signal is used for word alignment. It shifts the word by one bit.
Output	Q0, Q2, Q4, Q6, Q8	5	Data	—	—	Data at the positive edge of the clock
	Q1, Q3, Q5, Q7, Q9	5	Data	—	—	Data at the negative edge of the clock

Table 5.23. IDDRX5A Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.4.6. IDDR71A

The IDDR71A primitive is used for 7:1 LVDS input-side implementation.

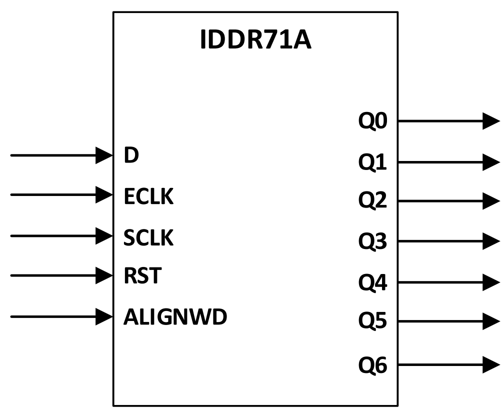


Figure 5.12. IDDR71A Ports

Table 5.24. IDDR71A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D	1	Data	High	0	DDR data input
	ECLK	1	Clock	Rising Edge	—	Fast edge clock
	SCLK	1	Clock	Rising Edge	—	Primary clock input (divide-by-3.5 of ECLK)
	RST	1	Control	High	0	Reset to DDR registers
	ALIGNWD	1	Control	High	0	This signal is used for word alignment. It shifts the word by one bit.
Output	Q0, Q2, Q4, Q6	4	Data	—	—	Data at the positive edge of the clock
	Q1, Q3, Q5	3	Data	—	—	Data at the negative edge of the clock

Table 5.25. IDDR71A Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.5. Output SDR/DDR User Primitives

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

5.5.1. OFD1P3DX

The OFD1P3DX primitive is used for generic TX SDR implementation.

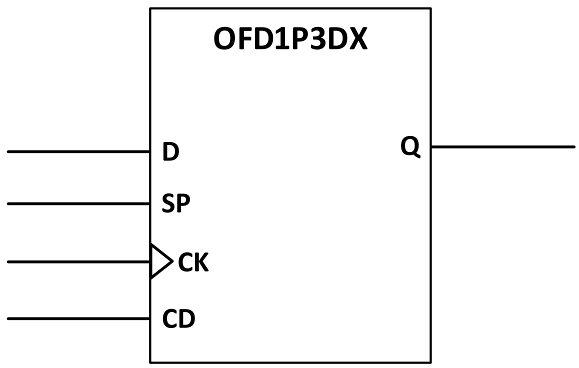


Figure 5.13. OFD1P3DX Ports

Table 5.26. OFD1P3DX Ports

Port		Range	Type	Active State	SW Default	Description
Input	D	1	Data	High	0	Data Input
	SP	1	Clock	Rising Edge	—	Clock Input
	CK	1	Control	High	—	Positive Clock Enable
	CD	1	Control	High	—	Asynchronous Reset
Output	Q	1	Data	—	—	Data out to IO

Table 5.27. OFD1P3DX Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.5.2. ODDRX1A

This primitive is used for Generic x1 ODDR implementation.

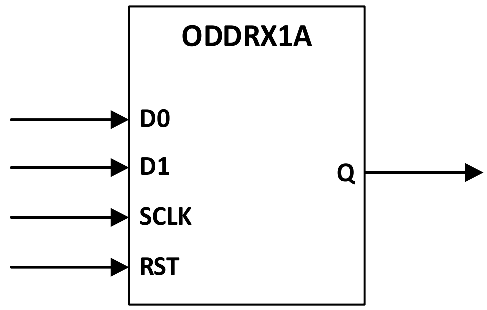


Figure 5.14. ODDRX1A Ports

Table 5.28. ODDRX1A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D0	1	Data	High	0	DDR data input
	D1	1	Data	High	0	DDR data input
	SCLK	1	Clock	Rising Edge	—	Primary clock input
	RST	1	Control	High	0	Reset to DDR registers
Output	Q	1	Data	—	—	DDR data output on both edges of SCLK D0: Positive edge of SCLK D1: Negative edge of SCLK

Table 5.29. ODDRX1A Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.5.3. ODDRX2A

This primitive is used for Generic x2 ODDR implementation.

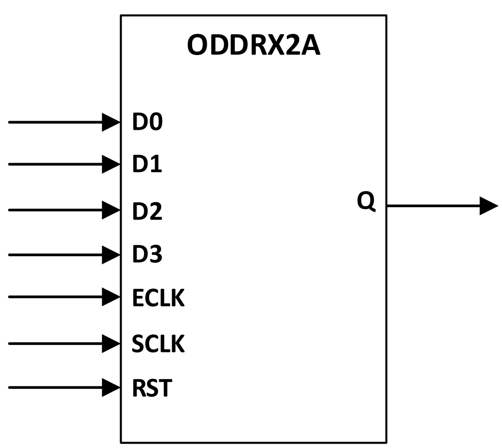


Figure 5.15. ODDRX2A Ports

Table 5.30. ODDRX2A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D0	1	Data	High	0	DDR data input
	D1	1	Data	High	0	DDR data input
	D2	1	Data	High	0	DDR data input
	D3	1	Data	High	0	DDR data input
	ECLK	1	Clock	Rising Edge	—	Fast edge clock
	SCLK	1	Clock	Rising Edge	—	Primary clock input (divide-by-2 of ECLK)
	RST	1	Control	High	0	Reset to DDR registers
Output	Q	1	Data	—	—	DDR data output on both edges of ECLK D0, D2: Positive edge of ECLK D1, D3: Negative edge of ECLK

Table 5.31. ODDRX2A Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.5.4. ODDRX4A

This primitive is used for Generic x4 ODDR implementation.

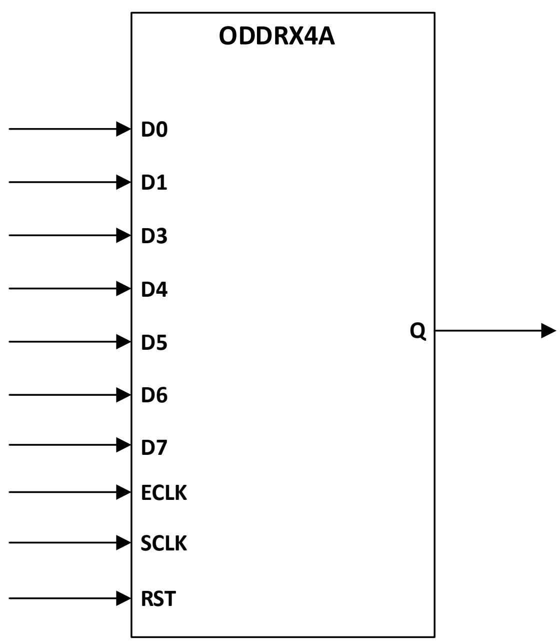


Figure 5.16. ODDRX4A Ports

Table 5.32. ODDRX4A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D0	1	Data	High	0	DDR data input
	D1	1	Data	High	0	DDR data input
	D2	1	Data	High	0	DDR data input
	D3	1	Data	High	0	DDR data input
	D4	1	Data	High	0	DDR data input
	D5	1	Data	High	0	DDR data input
	D6	1	Data	High	0	DDR data input
	D7	1	Data	High	0	DDR data input
	ECLK	1	Clock	Rising Edge	—	Fast edge clock
	SCLK	1	Clock	Rising Edge	—	Primary clock input (divide-by-4 of ECLK)
	RST	1	Control	High	0	Reset to DDR registers
Output	Q	1	Data	—	—	DDR data output on both edges of ECLK D0, D2, D4, D6: Positive edge of ECLK D1, D3, D5, D7: Negative edge of ECLK

Table 5.33. ODDRX4A Parameter

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.5.5. ODDRX5A

This primitive is used for Generic x5 ODDR implementation.

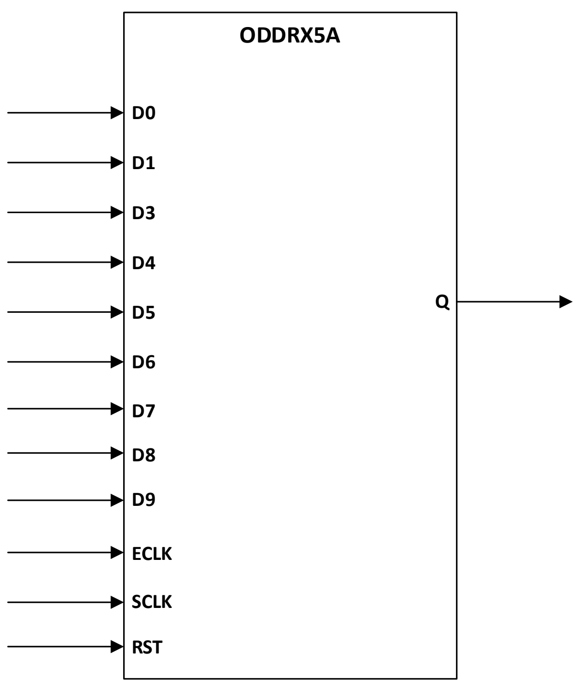


Figure 5.17. ODDRX5A Ports

Table 5.34. ODDRX5A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D0	1	Data	High	0	DDR data input
	D1	1	Data	High	0	DDR data input
	D2	1	Data	High	0	DDR data input
	D3	1	Data	High	0	DDR data input
	D4	1	Data	High	0	DDR data input
	D5	1	Data	High	0	DDR data input
	D6	1	Data	High	0	DDR data input
	D7	1	Data	High	0	DDR data input
	D8	1	Data	High	0	DDR data input
	D9	1	Data	High	0	DDR data input
	ECLK	1	Clock	Rising Edge	—	Fast edge clock
	SCLK	1	Clock	Rising Edge	—	Primary clock input (divide-by-5 of ECLK)
	RST	1	Control	High	0	Reset to DDR registers
Output	Q	1	Data	—	—	DDR data output on both edges of ECLK D0, D2, D4, D6, D8: Positive edge of ECLK D1, D3, D5, D7, D9: Negative edge of ECLK

Table 5.35. ODDRX5A Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.5.6. ODDR71A

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

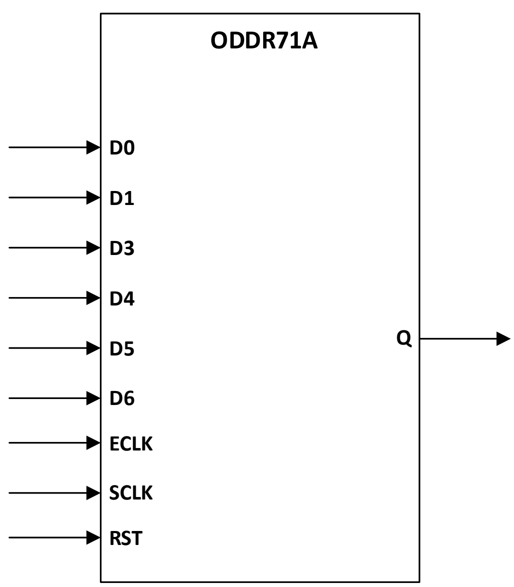


Figure 5.18. ODDR71A Ports

Table 5.36. ODDR71A Ports

Port		Range	Type	Active State	SW Default	Description
Input	D0	1	Data	High	0	DDR data input
	D1	1	Data	High	0	DDR data input
	D2	1	Data	High	0	DDR data input
	D3	1	Data	High	0	DDR data input
	D4	1	Data	High	0	DDR data input
	D5	1	Data	High	0	DDR data input
	D6	1	Data	High	0	DDR data input
	ECLK	1	Clock	Rising Edge	—	Fast edge clock
	SCLK	1	Clock	Rising Edge	—	Primary clock input (divide-by-3.5 of ECLK)
	RST	1	Control	High	0	Reset to DDR registers
Output	Q	1	Data	—	—	DDR data output on both edges of ECLK D0, D2, D4, D6: Positive edge of ECLK D1, D3, D5: Negative edge of ECLK

Table 5.37. ODDR71A Parameters

Name	Values	Default	Description
GSR	ENABLED DISABLED	ENABLED	Enable or disable Global Set/Reset feature

5.6. MIPIA

The following primitive is used to implement bidirectional MIPI interfaces.

Table 5.38. MIPIA Ports

Port		Range	Type	Active State	SW Default	Description
Input	BP	1	Data	—	—	IOPAD for P-port
	BN	1	Data	—	—	IOPAD for N-port
	AP	1	Data	—	—	LP Data for P-port
	AN	1	Data	—	—	LP Data for N-port
	TP	1	Control	—	—	LP Tri-state control for P-port
	TN	1	Control	—	—	LP Tri-state control for N-port
	HSRXEN	1	Control	—	—	HSRXEN for MIPI RX 0: Disable 1: Enable
	HSTXEN	1	Control	—	—	HSRXEN for MIPI TX to support BIDI 0: Disable 1: Enable
	IHS	1	Data	—	—	Data
Output	OLSP	1	Clock	—	—	Low Power MIPI output for P-port
	OLSN	1	Clock	—	—	Low Power MIPI output for N-port
	OHS	1	Clock	—	—	High-Speed output

Table 5.39. MIPIA Parameters

Name	Values	Default	Description
MIPI_ID	Any Integer	0	The MIPI_ID is used to place 2 PIO in a pair of PIO. 0: Not used for MIPI >0: For MIPI pair

5.7. DDR3L/DDR4 User Primitives

This section describes the primitives used to build DDR3L and DDR4 memory interfaces. These primitives support 4:1 (x2) and 8:1 (x4) gearing and are for DQ/DQS and CA signals.

Table 5.40. DDRPHY16D, DDRPHY32D, DDRPHY40D, DDRPHY64D, and DDRPHY72D Parameters

Name	Values	Default	Description
RATE	HALF FULL	HALF	DDR Rate
PROTOCOL	DDR3L DDR4	DDR3L	DDR Protocol
EXT_RES	PER_BANK PER_DDRPHY	PER_BANK	External Resistor

5.8. LPDDR4 User Primitives

This section describes the primitives used to build LPDDR4 memory interfaces. These primitives support 4:1 (x2) and 8:1 (x4) gearing and are for DQ/DQS and CA signals.

Table 5.41. DDRPHY16E, DDRPHY32E, and DDRPHY64E Parameters

Name	Values	Default	Description
RATE	HALF FULL	HALF	DDR Rate
PROTOCOL	LPDDR4	LPDDR4	DDR Protocol
EXT_RES	PER_BANK PER_DDRPHY	PER_BANK	External Resistor

6. Soft IP Modules

The following soft IP modules are available for use with the Generic DDR interfaces. All the soft IP modules can be generated using the IP Catalog.

Table 6.1. Supported Soft IPs

Soft IP Name	Function	Required
RX_SYNC	Used to break up the DDRDLL to DLLDEL clock loop for Aligned Interfaces.	Yes
GDDR_SYNC	Needed to tolerate large skew between stop and reset input.	Yes
7:1 LVDS Bit and Word Alignment (BW_ALIGN)	The soft IP is used to perform bit and word alignment using PLL dynamic phase shift interface and aligned input of IDDR71A.	Optional

Table 6.2. Soft IP Used in Each Interface

Interface	Soft IP
GIREG_RX.SCLK	None
GDDRX1_RX.SCLK.Centered	None
GDDRX1_RX.SCLK.Aligned	RX_SYNC
GDDRX2_RX.ECLK.Centered	GDDR_SYNC
GDDRX2_RX.ECLK.Aligned	RX_SYNC
GDDRX4_RX.ECLK.Centered	GDDR_SYNC
GDDRX4_RX.ECLK.Aligned	RX_SYNC
GDDRX5_RX.ECLK.Centered	GDDR_SYNC
GDDRX5_RX.ECLK.Aligned	RX_SYNC
GDDR71_RX.ECLK	GDDR_SYNC, BW_ALIGN
GDDRX4_RX.MIPI	GDDR_SYNC
GOREG_TX.SCLK	None
GDDRX1_TX.SCLK.Centered	None
GDDRX1_TX.SCLK.Aligned	None
GDDRX2_TX.ECLK.Centered	GDDR_SYNC
GDDRX2_TX.ECLK.Aligned	GDDR_SYNC
GDDRX4_TX.ECLK.Centered	GDDR_SYNC
GDDRX4_TX.ECLK.Aligned	GDDR_SYNC
GDDRX5_TX.ECLK.Centered	GDDR_SYNC
GDDRX5_TX.ECLK.Aligned	GDDR_SYNC
GDDR71_TX.ECLK	GDDR_SYNC
GDDRX4_TX.MIPI	GDDR_SYNC

6.1. RX_SYNC

This module is needed for the startup of the DDRDLL and DLLDEL clock loops on RX-aligned interfaces.

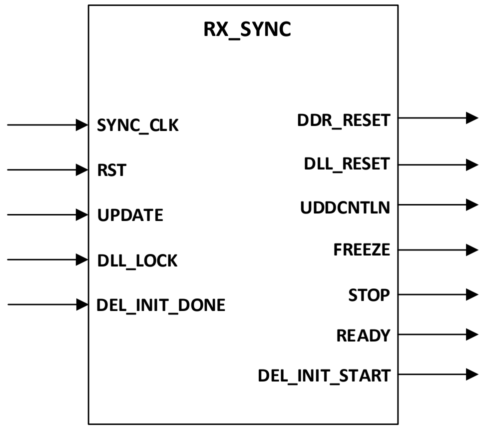


Figure 6.1. RX_SYNC Ports

Table 6.3. GDDR_SYNC Ports Description

Port	Direction	Description
SYNC_CLK	Input	Startup clock. It can be a continuously running low-speed clock, for example, an oscillator clock. It cannot be the RX_CLK or divided version coming from clock sources that causes this module to stop or reset.
RST	Input	Active high asynchronous reset to this circuit. When RST=1: STOP=0, FREEZE=0, UDDCNTLN=1, DLL_RESET=1, DDR_RESET=1, READY=0
UPDATE	Input	UPDATE can be used to restart the sync process. READY goes low and waits for the sync process to be completed before going high again. This can only be performed when no traffic is present
DLL_LOCK	Input	LOCK output from DDRDLL
DEL_INIT_DONE	Input	DLLDEL delay counter done
DDR_RESET	Output	Resets all IDDRX components and CLKDIV
DLL_RESET	Output	Resets to DDRDLL
UDDCNTLN	Output	Connects to DDRDLL.UDDCNTLN
FREEZE	Output	Connects to DDRDLL.FREEZE
STOP	Output	Connects to ECLKSYNC.STOP
READY	Output	Indicates that startup is finished, and RX circuit is ready to operate
DEL_INIT_START	Output	Starts DLLDEL initialization and delays counter

6.2. GDDR_SYNC

This module is needed for the startup of all RX centered and all TX interfaces with 2×, 4×, or 5× gearing.

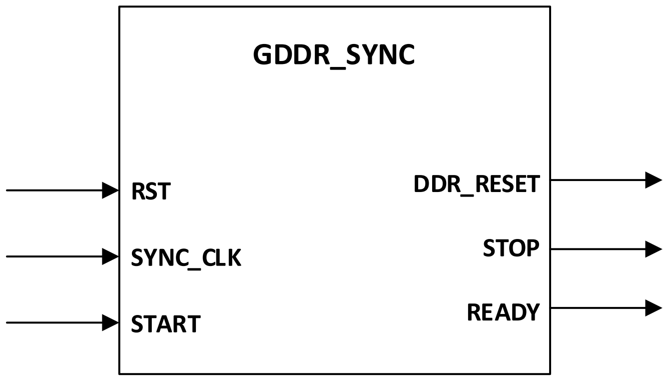


Figure 6.2. GDDR_SYNC Ports

Table 6.4. GDDR_SYNC Ports Description

Port	Direction	Description
RST	Input	Active high asynchronous reset to this circuit. When RST=1: STOP=0, DDR_RESET=1, READY=0
SYNC_CLK	Input	Startup clock. It can be a continuously running low-speed clock, for example, an oscillator clock. It cannot be the RX_CLK or divided version coming from clock sources that causes this module to stop or reset.
START	Input	Starts sync process. This is used to wait for PLL lock, then start sync process in 7:1 LVDS interface
DDR_RESET	Output	Resets to all IDDRX or ODDRX components and CLKDIV
STOP	Output	Connects to ECLKSYNC.STOP
READY	Output	Indicates that startup is finished, and RX circuit is ready to operate

6.3. BW_ALIGN

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

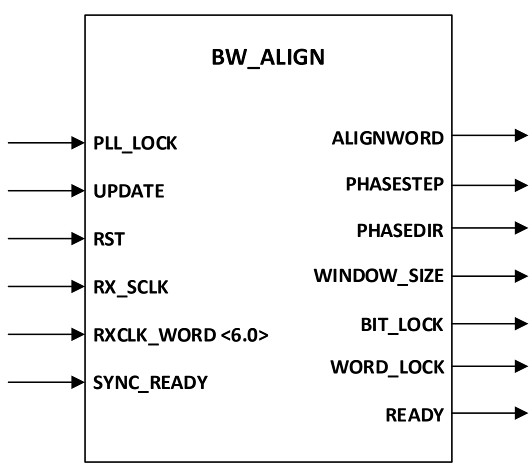


Figure 6.3. BW_ALIGN Ports

Table 6.5. BW_ALIGN Port Description

Port	Direction	Description
PLL_LOCK	Input	Connects to PLL LOCK output. Starts the alignment procedures after PLL lock goes high
UPDATE	Input	Starts the procedure, or re-starts to optimize again
RST	Input	Active high asynchronous reset to this circuit. When RST=1: All outputs=0
RX_SCLK	Input	Divided RX clock from the 7:1 RX interface, produced by ECLKDIV
RXCLK_WORD<6:0>	Input	Parallel data output from the 2nd IDDRX71 attached to RX CLK Input
SYNC_READY	Input	Starts bit/word alignment or re-starts to align again
ALIGNWORD	Output	Connects to IDDRX71.ALIGNWORD, for word rotation
PHASESTEP	Output	Rotates phase for PLL
PHASEDIR	Output	Phases rotation direction for PLL, fixed to forward (0) for this design
WINDOW_SIZE	Output	Final valid window size
BIT_LOCK	Output	Status output, bit lock is achieved
WORD_LOCK	Output	Status output, word lock is achieved
READY	Output	Indicates that alignment procedure is finished, and RX circuit is ready to operate

The goal of bit alignment is to place the Edge Clock (under PLL dynamic phase shift control) in the center of the valid window for the clock word and data word. The PLL-phase rotation goes through all 16 phases. The PLL high-speed output is used to sample the RX input clock. Transitions are detected on the second IDDR71 output, where the RX clock and phases close to transition are identified. The IP chooses the phase furthest away from transition as the final phase to use.

The low-speed clock has two transitions per 7-bit word. It is not the worst case in terms of inter-symbol interference, but samples 8 points per bit period. A minimum eye-opening of 3/8 UI is needed to achieve lock. Jitter tolerance is around 0.25 UI or, about 300 ps at 756 Mbps.

After bit alignment is achieved, word alignment is needed so video data (in 7-bit words) can be processed in the core. The IP uses the ALIGNWD function of the IDDRX71A primitive for word alignment. Each pulse on ALIGNWD rotates the 7-bit bus by 2 bits. For the maximum of seven ALIGNWD operations, the word loops through all seven possibilities. The goal is to get 7’b1100011 (7’h63) in the clock word. The clock word is the clock (4-bit 1 and 3-bit 0) converted to parallel data, exactly as the video data traffic. For the 7:1 video, the RX input clock serves as:

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

References

- Lattice Certus-N2 web page
- Lattice Nexus 2 Platform Data Sheet Overview (FPGA-DS-02122)
- Lattice Nexus 2 Platform Data Sheet Specifications (FPGA-DS-02121)
- Lattice Nexus 2 System Configuration User Guide (FPGA-TN-02370)
- Lattice Nexus 2 System Clock PLL Design User Guide (FPGA-02364)
- SGMII and Gb Ethernet PCS IP User Guide (FPGA-IPUG-02077)
- AMBA APB Protocol Specification
- AMBA AXI Protocol Specification
- Lattice Insights for Lattice Semiconductor training courses and learning plans.

Technical Support Assistance

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

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/en/Support/AnswerDatabase.

Revision History

Revision 0.81, December 2024

Section	Change Summary

External Memory	• Updated Figure 4.1. Lattice Nexus 2 Memory Interface Mapping Diagram.
	  ◦ Changed the *DDR3/DDR4* to *DDR3L/DDR4*.
	  ◦ Replaced *x40* with *x72* memory interface in the Pin and Resource Planning subsection.
	  ◦ Added x64 and x72 in the DDR3L/DDR4 memory width.
	  ◦ Added x64 in the LPDDR4 memory width.
	• Changed the table title of Table 4.1 from *x40 Interface Pin Assignment* to *x72 DIMM Pin Assignment*.
	• Added DQ/DQS Lane 5 to DQ/DQS Lane 8 in Table 4.1. x72 DIMM Pin Assignments.
	• Updated the values of CA Lane 0 to CA Lane 2 in Table 4.1. x72 DIMM Pin Assignments.
	• Removed DDR3L, DDR4 and LPDDR4 Input Reference Clock subsections and changed to FPGA Input Reference Clock subsection.
	• Added DDRPHY Trace and Skew Requirements subsection.
	• Reworked DDR3L Board Design Guidelines, DDR4 Board Design Guidelines and LPDDR4 Board Design Guidelines subsections.

Revision 0.80, August 2024

Section	Change Summary
All	Preliminary release.

www.latticesemi.com