Lattice Nexus 2 sysCLOCK PLL Design and User Guide

Preliminary Technical Note

FPGA-TN-02364-0.81

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

Contents 3
Abbreviations in This Document 7
1. Introduction 8
2. Lattice Nexus 2 Clocks 9
2.1. Lattice Nexus 2 Clock Networks and Elements 11
2.2. Dedicated Clock Pins 12
2.3. Global Clock (GCLK) Network 13
2.3.1. Global Clock Sources 13
2.3.2. Global Clock Routing 14
2.4. Regional Clock (RCLK) Network 15
2.4.1. Regional Clock Sources 16
2.4.2. Regional Clock Routing 17
2.5. Edge Clock (ECLK) Network 18
2.5.1. Edge Clock Sources 18
2.5.2. Edge Clock Routing 19
2.6. PHY Clock (PHYCLK) Network 20
2.6.1. PHYCLK Sources 21
2.6.2. PHYCLK Routing 21
2.7. Edge Clock Synchronizer and Divider (ECLKSYNCA and ECLKDIVA) 22
2.7.1. ECLKSYNCA Component Definition 22
2.7.2. ECLKSYNCA Usage in VHDL 23
2.7.3. ECLKSYNCA Usage in Verilog 23
2.7.4. ECLKDIVA Component Definition 23
2.7.5. ECLKDIVA Usage in VHDL 24
2.7.6. ECLKDIVA Usage in Verilog 25
2.7.7. ECLKDIVA Divide Ratio Options 25
2.8. DLLDEL 26
2.8.1. DLLDELA Component Definition 27
2.8.2. DLLDELA Usage in VHDL 27
2.8.3. DLLDELA Usage in Verilog 28
2.9. Dynamic Clock Select (DCS) 29
2.9.1. DCS Timing Diagrams 29
2.9.2. DCSA Component Definition 34
2.9.3. DCSA Usage in VHDL 34
2.9.4. DCSA Usage in Verilog 35
2.10. Dynamic Clock Control (DCC) 35
2.10.1. DCCA Component Definition 37
2.10.2. DCCA Usage in VHDL 37
2.10.3. DCCA Usage in Verilog 37
2.11. Internal Oscillator (OSCE) 38
2.11.1. OSCE Component Definition 38
2.11.2. OSCE Usage in VHDL 38
2.11.3. OSCE Usage in Verilog 39
2.12. General Routing for Clocks 39
3. sysCLOCK PLL 41
3.1. sysCLOCK PLL Overview 41
3.1.1. PLL Input Sources 42
3.2. sysCLOCK PLL Component Definition 43
3.3. PLLREFCSA Component Definition 46
3.4. Functional Description 47
3.4.1. Refclk (CLKI) Divider 47
3.4.2. Feedback Loop (CLKFB) Divider 47

3.4.3. Output Clock Dividers (CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5) 47
3.4.4. Phase Adjustment (Static Mode) 47
3.4.5. Phase Adjustment (Dynamic Mode) 47
3.5. PLL Inputs and Outputs 47
3.5.1. CLKI Input 47
3.5.2. CLKFB Input 47
3.5.3. RST Input 48
3.5.4. Dynamic Clock Enables 48
3.5.5. Dynamic Phase Shift Inputs 48
3.5.6. PHASESEL Input 49
3.5.7. PHASEDIR Input 49
3.5.8. PHASESTEP Input 49
3.5.9. PHASELOADREG Input 49
3.5.10. PLL Clock Outputs 49
3.5.11. LOCK Output 50
3.5.12. Dynamic Phase Adjustment 50
3.5.13. VCO Phase Shift 50
3.5.14. Divider Phase Shift 51
3.5.15. Total Phase Shift 52
3.6. Fractional-N Synthesis Operation 52
3.7. Spread Spectrum Clock Generation 52
3.8. Low Power Features 53
3.8.1. Dynamic Clock Enable 53
4. PLL LMMI Operation 54
4.1. PLL Architecture 54
4.2. LMMI Register Map 54
4.3. Example 1: Dynamic Reconfigure Output Frequency 58
4.4. Example 2: Post-Divider Phase Shift 58
References 60
Technical Support Assistance 61
Revision History 62

Figures

Figure 2.1. Clock Regions (LN2-CT-20 Devices) 9
Figure 2.2. Clock Regions (LN2-CT-16 Devices) 9
Figure 2.3. Clock Regions (LN2-CT-10 Devices) 10
Figure 2.4. Clock Regions (LN2-CT-06 Devices) 10
Figure 2.5. High-Level View of Clock Networks and Elements (LN2-CT-20 Devices) 12
Figure 2.6. Global Clock Routing Architecture for LN2-CT-20 Devices 14
Figure 2.7. DCSCMUX 15
Figure 2.8. Multi-Region Clock Formations (LN2-CT-20 Devices) 16
Figure 2.9. Generic RGMUX Implementation 18
Figure 2.10. Edge Clock Sources per Bank 19
Figure 2.11. ECLKSYNCDIV 20
Figure 2.12. Edge Clock Sources Bridged to Multiple Banks 20
Figure 2.13. PHYCLK Network 21
Figure 2.14. ECLKSYNCA Component Symbol 22
Figure 2.15. ECLKSYNCA Functional Waveform 22
Figure 2.16. ECLKDIVA Component Symbol 23
Figure 2.17. High-Level Implementation of DLLDEL and Code Control 26
Figure 2.18. DLLDELA Component Symbol 27
Figure 2.19. DCSCMUX 29
Figure 2.20. Posedge DCS Switch from SEL: 0 ≥ 1 30
Figure 2.21. Posedge DCS Switch from SEL: 1 ≥ 0 30
Figure 2.22. Negedge DCS Switch from SEL: 0 ≥ 1 31
Figure 2.23. Negedge DCS Switch from SEL: 1 ≥ 0 31
Figure 2.24. DCSMODE = *CLK0* or *CLK1* 32
Figure 2.25. DCSMODE = *CLK0_LOW* or *CLK0_HIGH* 32
Figure 2.26. DCSMODE = *CLK1_LOW* or *CLK1_HIGH* 33
Figure 2.27. MODESEL = *1* DCS Clock Switch 33
Figure 2.28. DCSA Component Symbol 34
Figure 2.29. TMID Multiplexer, DCC Elements, and Top GCLK Feedlines 36
Figure 2.30. BMID Multiplexer, DCC Elements, and Bottom GCLK Feedlines 36
Figure 2.31. Glitchless DCC Functional Waveform 36
Figure 2.32. DCCA Component Symbol 37
Figure 2.33. OSCE Component Symbol 38
Figure 2.34. Gated Clock to the Primary Clock Routing 40
Figure 2.35. Gated Clock to Small Logic Domain 40
Figure 3.1. Lattice Nexus 2 PLL Block Diagram 41
Figure 3.2. PLL Component Instance 43
Figure 3.3. PLLREFCSA Component Instance 47
Figure 3.4. RST Input Timing Diagram 48
Figure 3.5. PLL Phase Shifting Using the PHASESTEP Signal 51
Figure 3.6. Divider Phase Shift Timing Diagram 51
Figure 3.7. Down Spread Profile 52
Figure 3.8. Dynamic Clock Enable for PLL Outputs 53
Figure 4.1. Example of PLL Output Frequency Reconfiguration Using PLL LMMI Operation 58
Figure 4.2. Example of Overwriting the Default Value to New Value 58
Figure 4.3. Example of Post Divider 90 Degree Phase Shift (Lagging) Implementation Using PLL LMMI Operation 59

Tables

Table 1.1. Number of PLLs, Edge Clocks, and Clock Synchronizers and Dividers 8
Table 2.1. Number of Dedicated Clock Pins for Lattice Nexus 2 Devices 12
Table 2.2. Global Clock Sources 13
Table 2.3. High-Performance I/O (HPIO) Regional Clock Sources 16
Table 2.4. Wide-Range I/O (WRIO) Regional Clock Sources 17
Table 2.5. Edge Clock Sources 18
Table 2.6. PHYCLK Sources 21
Table 2.7. ECLKSYNCA Component Port Definition 22
Table 2.8. ECLKDIVA Component Port Definition 24
Table 2.9. ECLKDIVA Component Attribute Definition 24
Table 2.10. ECLKDIVA pre-divby2 and ECLKDIVA Options 25
Table 2.11. DLLDELA Component Port Definition 27
Table 2.12. DLLDELA Component Attribute Definition 27
Table 2.13. DCSA Component Port Definition 34
Table 2.14. DCSA DCSMODE Attribute 34
Table 2.15. DCCA Component Port Definition 37
Table 2.16. OSCE Component Port Definition 38
Table 2.17. OSCE Component Attribute Definition 38
Table 3.1. General Purpose PLLs (GPLLs) 42
Table 3.2. PLL Input Sources, Feedback Sources, and Output Clocks 42
Table 3.3. PLL Component Port Definition 43
Table 3.4. PLL Component Attribute 44
Table 3.5. PLLREFCSA Component Port Definition 46
Table 3.6. PLL Clock Output Enable Signal List 48
Table 3.7. PHASESEL Signal Settings Definition 49
Table 3.8. PHASEDIR Signal Settings Definition 49
Table 3.9. PLL Clock Outputs and ECLK Connectivity 50
Table 4.1. PLL Data Bus Port Definition 54
Table 4.2. LMMI Offset Address Locations for PLL Registers 54
Table 4.3. PLL Registers Descriptions[1, 2, 3] 55

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
ATM	Anti-Tampering Monitor
BMID	Bottom Mid Multiplexer
CKR	Clock Region
DCC	Dynamic Clock Control
DCS	Dynamic Clock Select
DCSCMUX	Dynamic Clock Selector Center Multiplexer
DDR	Double Data Rate
DLL	Delay Locked Loop
DSP	Digital Signal Processor
EBR	Embedded Block RAM
ECLK	Edge Clock
FPGA	Field Programmable Gate Array
GCLK	Global Clock
GSR	Global Set Reset
HPIO	High-Performance I/O
LC	Logic Cell
LMMI	Lattice Memory Mapped Interface
PFU	Programmable Functional Unit
PHYCLK	PHY Clock
PLL	Phase Locked Loop
RCLK	Regional Clock
RGMUX	Regional Multiplexer
SED	Soft Error Detect
SERDES	Serializer/Deserializer
SSC	Spread Spectrum Clock
TMID	Top Mid Multiplexer
VHDL	VHSIC Hardware Description Language
WRIO	Wide-Range I/O

1. Introduction

This user guide describes the clock resources available in the Lattice Nexus™ 2 architecture. Details are provided for Global Clocks, Regional Clocks, PHY Clocks, Edge Clocks, PLLs, Internal Oscillator, and clocking elements such as Clock Synchronizers, Clock Dividers, Clock Multiplexers, and Clock Enable/Disable blocks available in the Lattice Nexus 2 device.

The number of PLLs, Edge Clocks, and Clock Synchronizers and Dividers for each device is listed in the following table.

Table 1.1. Number of PLLs, Edge Clocks, and Clock Synchronizers and Dividers

Parameter	Description	LN2-CT-06/10	LN2-CT-16/20
Number of PLLs	General purpose phase locked loops. One per wide-range I/O and high-performance I/O region.	4	7
Number of Dedicated Clock Pins	Clock input pins that drive the different clock networks. 4 per wide-range I/O and high-performance I/O region.	16	28
Number of SERDES RX/TX Clocks	RX and TX clocks for high-speed serial I/O interfaces. 4 RX and 4 TX clocks each per SERDES quad region.	8	24
Number of Global Clocks	Global clocks that drive the regional clock network.	24	24
Number of Regional Clocks (per Clock Region)	Regional clocks that drive all synchronous elements in the clock regions.	16	16
Number of Edge Clocks	Edge clocks for high-speed I/O interfaces. 4 per high-performance I/O bank.	8	20
Number of Edge Clock Synchronizers and Dividers	Edge clock synchronizers and dividers (ECLKSYNCA and ECLKDIVA) for high-speed I/O interfaces. 4 ECLKSYNCA and 4 ECLKDIVA per high-performance I/O bank.	8	20
Number of PHYCLKs	PHYCLKs generated from high-performance PLLs can drive the DDRPHY and other clock networks. 1 PHYCLK per high-performance I/O bank.	2	5
Number of DDRDLLs	DDRDLL used for high-speed I/O interfaces. 1 DDRDLL per high-performance I/O bank.	2	5

It is important to validate the device pinout using the Lattice Radiant™ software tool to avoid implementation issues.

2. Lattice Nexus 2 Clocks

The Lattice Nexus 2 device core architecture is constructed of a number of similar sized Clock Regions (CKR). Each CKR comprises blocks such as PFUs, EBRs, DSPs, and a Regional Clock Network. Each CKR is about 11k–22k Logic Cells (LCs) depending on the device density, and is associated with an I/O bank. An I/O bank may be a group of high-speed SERDES I/O, a High-Performance I/O (HPIO) bank, or a Wide-Range I/O (WRIO) bank. The Clock Regions are arranged in two rows and multiple columns depending on the density of the device.

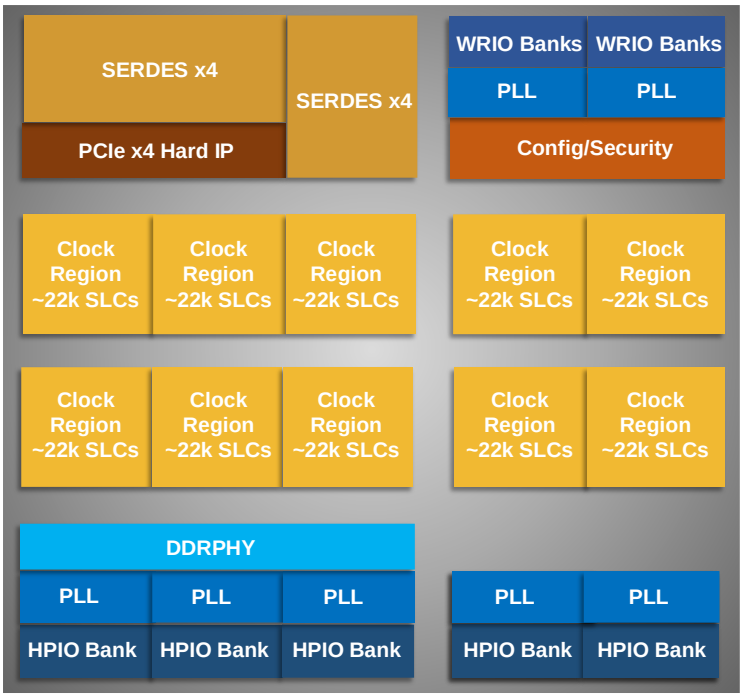


Figure 2.1. Clock Regions (LN2-CT-20 Devices)

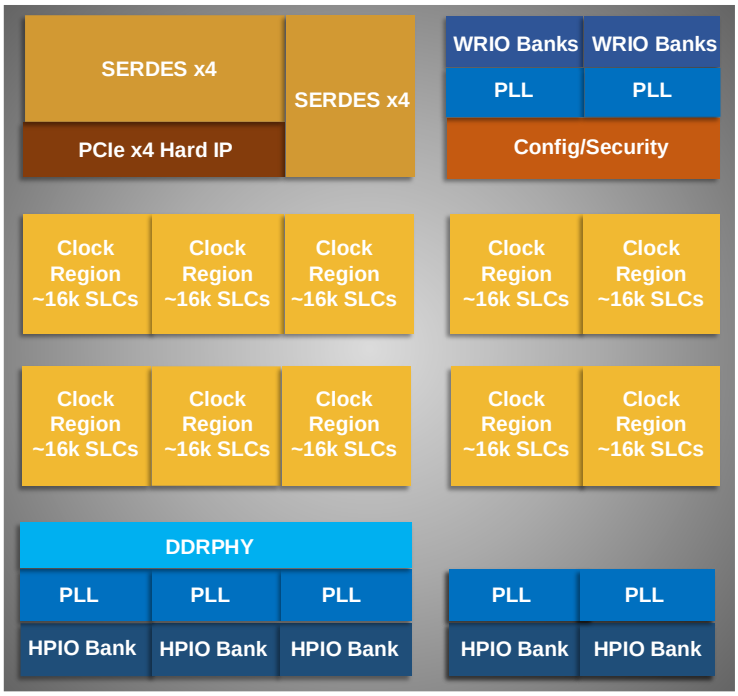


Figure 2.2. Clock Regions (LN2-CT-16 Devices)

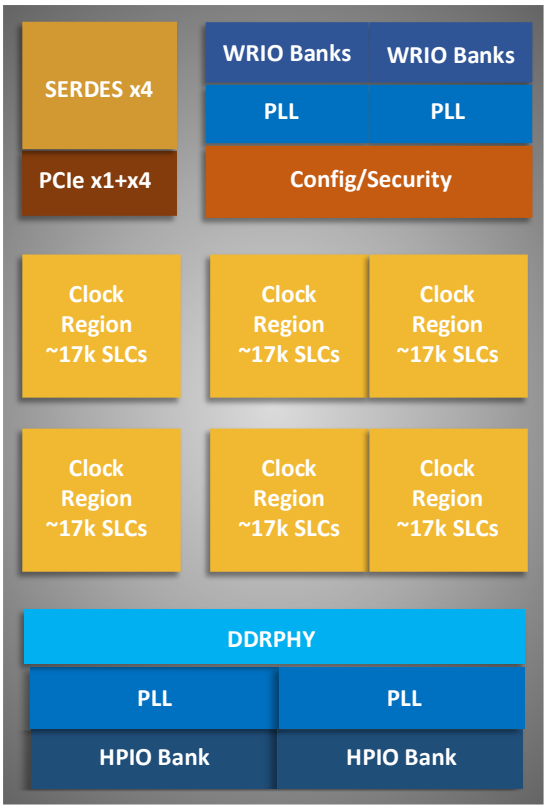


Figure 2.3. Clock Regions (LN2-CT-10 Devices)

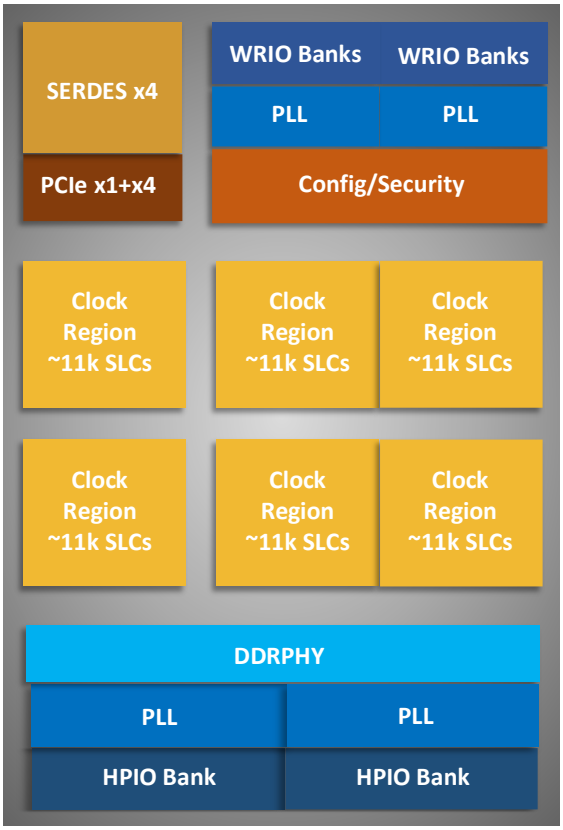


Figure 2.4. Clock Regions (LN2-CT-06 Devices)

2.1. Lattice Nexus 2 Clock Networks and Elements

The Lattice Nexus 2 Clock Network consists of four different clock structures: Global Clock (GCLK) Network, Regional Clock (RCLK) Network, Edge Clock (ECLK) Network, and PHY Clock (PHYCLK) Network.

- The GCLK network provides the clock sources to drive the RCLK network of all Clock Regions in the Lattice Nexus 2 devices.
- The RCLK network provides the clock sources to drive all blocks (PFU, EBR, DSP) within a Clock Region. The RCLK network can bridge to adjacent Clock Regions to form a multi-region RCLK network.
- The ECLK network provides the clock sources for the high-speed DDR I/O interfaces. Similar to the RCLK, the ECLK network can bridge to adjacent Clock Regions to form a wider ECLK. The ECLK can also serve as clock sources to the GCLK network. The ECLKs go through ECLK dividers to provide a divided clock to the GCLK network.
- The PHYCLK network provides a special high frequency clock to support special interfaces such as the DDRPHY interface in the HPIO bank.
- The Lattice Nexus 2 devices also support internally generated clocks (Fabric_CLK) from the PFU blocks.

Apart from the main clock networks, there are other clock components that form the overall clock architecture.

- Dedicated PCLKT pins that enable external clock sources into the FPGA.
- Dynamic Clock Switching (DCS) block that enables dynamic, glitchless switching between two independent clock sources.
- Dynamic Clock Control (DCC) block that enables dynamic enabling and disabling of the GCLK Feedlines.
- Delay cells, DLLDEL, that provide the necessary clock phase shift for certain applications. The DLLDEL adjusts the phase of the clock and feeds the ECLK and GCLK networks (through the ECLK).
- Clock multiplexers:
  - MIDMUX — TMID (Top Mid Multiplexer) and BMID (Bottom Mid Multiplexer) take inputs from various clock sources to drive the GCLK Feedlines that feed into the DCSCMUX.
  - CENTERMUX — The DCSCMUX (Dynamic Clock Selector Center Multiplexer) takes the GCLK Feedlines, the fabric clocks from the FPGA core to generate two sets of 24 GCLKs. A set of four DCS blocks reside within the DCSCMUX. The generated GCLKs can be sourced from the clocks coming out of the DCS blocks, as shown in the following diagram.
  - RGMUX — Regional Multiplexer takes the GCLKs and other Regional Clock sources (including SERDES generated clocks (if present), dedicated PCLKT pins, PLL outputs, fabric clocks from the FPGA core) to generate a set of sixteen RCLKs. The RGMUX clock sources can also be other Regional Clock sources from adjacent Clock Regions.
  - BRIDGEMUX — The Regional, Edge, and PHYCLK Clock Bridge Multiplexers (RSBRGMUX, ECLKBRGMUX, and PHYBRGMUX respectively) enable clocks in the current region to drive adjacent clock regions to form a wider clock.
  - ECLKINMUX — ECLK Input Multiplexer takes various clock sources to drive the ECLK network.
  - PHYCLKBRGMUX — PHYCLKBRGMUX takes PHYCLKs from current and adjacent clock regions to drive the DDRPHY and High-Speed I/O Regions.
  - **Note:** When CFGDONE is asserted, the output of all Clock Multiplexers default to 0.
- ECLK Synchronizer (ECLKSYNCA) and ECLK Divider (ECLKCDIVA) provide synchronized clocks to support DDR bus synchronization and divided clocks for high frequency applications.

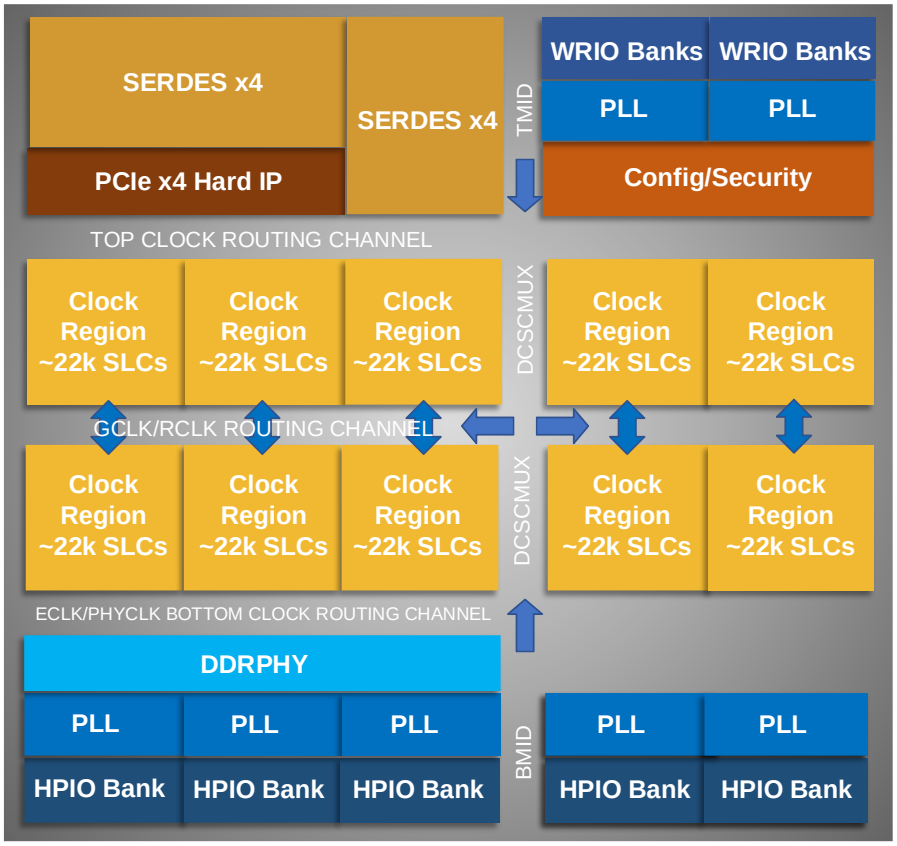


Figure 2.5. High-Level View of Clock Networks and Elements (LN2-CT-20 Devices)

The following sections detail the different clock structures and elements in the Lattice Nexus 2 family of devices.

2.2. Dedicated Clock Pins

The Lattice Nexus 2 device has dedicated clock pins, called PCLKT pins, which enable external clock sources into the FPGA. These clock pins route directly to GCLK network, ECLK network, and RCLK network. A dedicated PCLKT pin must always be used to route an external clock source to the FPGA.

Table 2.1. Number of Dedicated Clock Pins for Lattice Nexus 2 Devices

The number of dedicated clock pins are package dependent.

Device Densities	LN2-CT-06/10	LN2-CT-16/20
PCLKT0 [0:3]	4	4
PCLKT3 [0:3]	—	4
PCLKT4 [0:3]	—	—
PCLKT5 [0:3]	—	—
PCLKT6 [0:3]	—	—
PCLKT7 [0:3]	—	—
PCLKT8 [0:3]	—	4
PCLKT9 [0:3]	—	4
PCLKT10 [0:3]	4	4
PCLKT11 [0:3]	4	4
PCLKT12 [0:3]	4	4
Total PCLKT Pins	16	28

2.3. Global Clock (GCLK) Network

GCLK network provides the main clock sources in the Lattice Nexus 2 devices. The GCLK network drives the RCLK networks of all Clock Regions (CKRs) in the device. The GCLK structure has two clock domains for all device densities, left half and right half. Each domain takes all available GCLK sources and generates 24 independent GCLKs. These 24 GCLKs, combined with other RCLK sources provide 16 RCLKs to drive each row within a Clock Region.

2.3.1. Global Clock Sources

The GCLKs are sourced from multiple inputs, referred to as Global Clock sources. The Global Clock sources that can drive the GCLKs are as follows:

- Dedicated Clock Pins (PCLKT pins)
- PLL (WRIO) outputs (CLKOP, CLKOS, CLKOS2, CLKOS3)
- PLL (HPIO) outputs (CLKOP, CLKOS, CLKOS2, CLKOS3)
- SERDES RX/TX Clocks
- JTAG Clock
- OSC Clock
- Edge Clocks (through ECLKDIVA)
- Internally generated clocks Fabric_CLK

Table 2.2. Global Clock Sources

Device Densities	LN2-CT-06/10	LN2-CT-16/20	To
Clock Input Pins			
PCLKT0 [0:1]	Yes	Yes	TMID
PCLKT12 [0:1]	Yes	Yes	TMID
PCLKT3 [0:1]	No	Yes	BMID
PCLKT4 [0:1]	No	No	BMID
PCLKT5 [0:1]	No	No	BMID
PCLKT6 [0:1]	No	No	BMID
PCLKT7 [0:1]	No	No	BMID
PCLKT8 [0:1]	No	Yes	BMID
PCLKT9 [0:1]	No	Yes	BMID
PCLKT10 [0:1]	Yes	Yes	BMID
PCLKT11 [0:1]	Yes	Yes	BMID
PLL Outputs			
PLL0 (WRIO)	clkop, clkos, clkos2, clkos3	clkop, clkos, clkos2, clkos3	TMID
PLL12 (WRIO)	clkop, clkos, clkos2, clkos3	clkop, clkos, clkos2, clkos3	TMID
PLL (HPIO) [1:n]	n = 2 clkop, clkos, clkos2, clkos3	n = 5 clkop, clkos, clkos2, clkos3	BMID
Others			
SERDES [0:n]	n = 1 rxclk0, rxclk1, rxclk2, rxclk3, txclk	n = 3 rxclk0, rxclk1, rxclk2, rxclk3, txclk	TMID
JTAGCLK	Yes	Yes	TMID
OSC	Yes	Yes	TMID
ECLKDIVA ECLKDIVA[0:1]	 Yes	 Yes	 BMID
Internally Generated Clocks (Fabric_CLK)			
Fabric clock for GCLK Fabric_CLK [0:3] Fabric_CLK [0:3] Fabric_CLK [0:7]	 Yes Yes Yes	 Yes Yes Yes	 TMID BMID DCSCMUX

2.3.2. Global Clock Routing

The Global Clock Network is made up of low skew clock routing resources with connectivity to every synchronous element of the device. Global Clock sources are selected at the MIDMUX (TMID and BMID), then selected at the CENTERMUX (DCSCMUX) to clock the synchronous elements in the Clock Regions. For the Lattice Nexus 2 family, the Global Clock routing network is divided into left and right regions.

The following diagram shows the clock sources that feed into the GCLK network through the MIDMUX (TMID and BMID) and CENTERMUX (DCSCMUX). Note that not all available clock sources feed into the GCLK network.

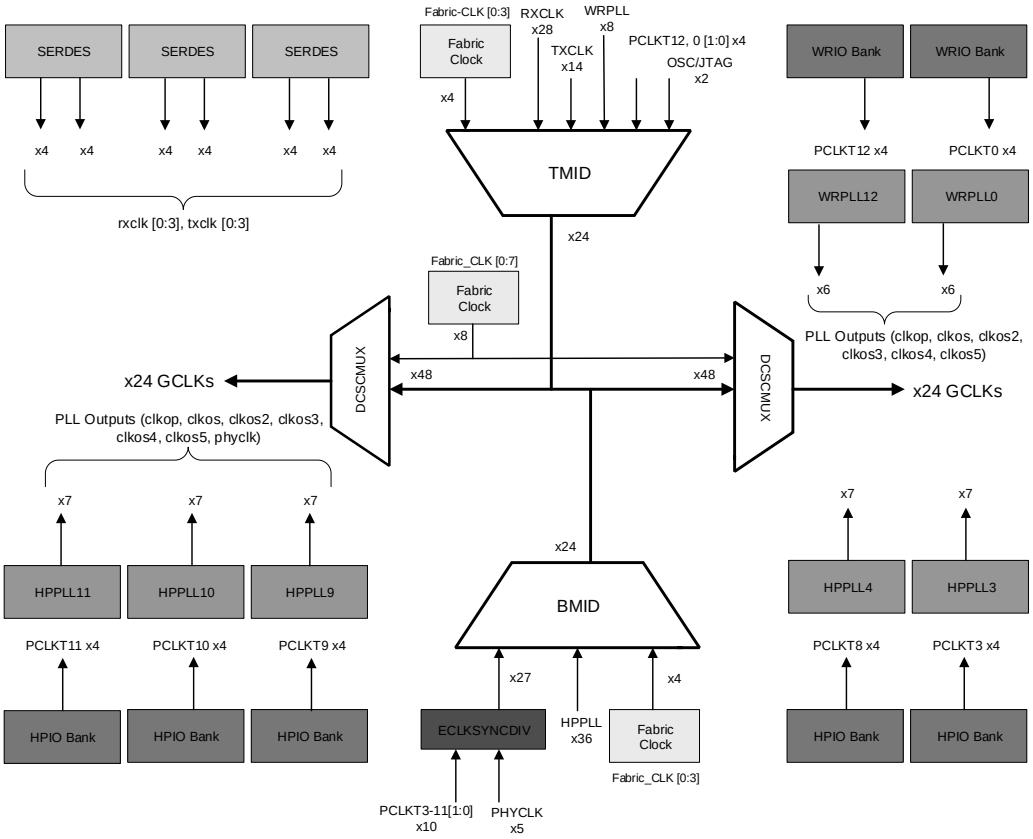


Figure 2.6. Global Clock Routing Architecture for LN2-CT-20 Devices

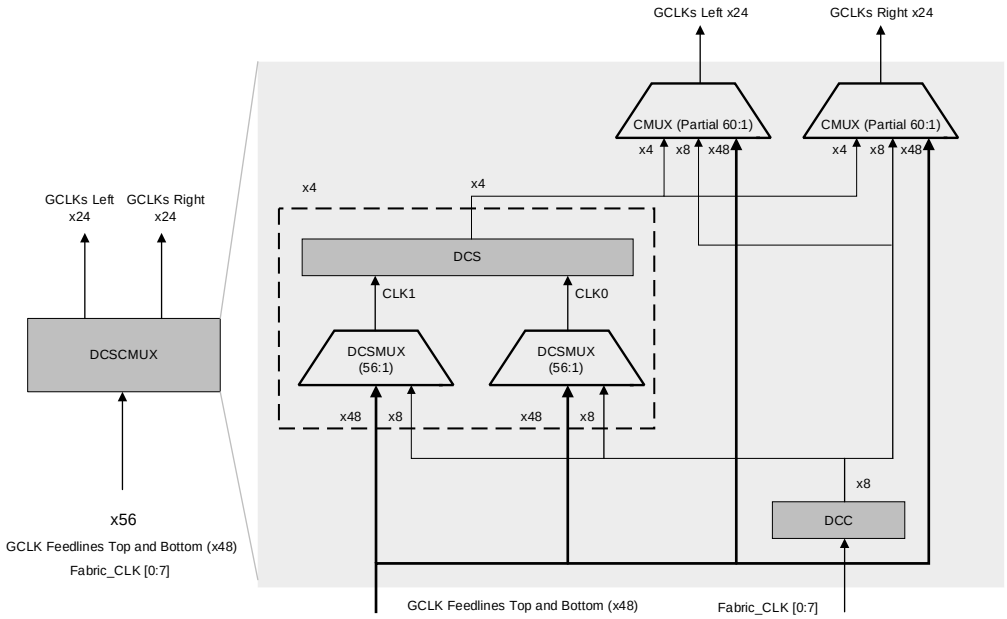


Figure 2.7. DCSCMUX

These 24 GCLKs are combined with the Regional Clock sources to provide 16 RCLKs to drive each row within a Clock Region.

2.4. Regional Clock (RCLK) Network

RCLK network is the main clock network within a Clock Region. RCLK is driven by the Global Clock Network and provides clock sources to all blocks (PFU, EBR, DSP) within a Clock Region. The RCLK network also provide clock sources for other blocks such as the I/O banks, PLLs, SERDES (when present), and fabric clocks from the FPGA.

The RCLK network can bridge to adjacent Clock Regions to form Multi-Region Clock Networks. Specifically, RCLK can bridge to one other Clock Region either to the left, right, top, or bottom of the current Clock Region. The multi-region clock can be formed in the following topology: 1×1, 1×2, 2×1, 2×2, or 3×2 (column × row).

Depending on the location of the Clock Region, some of the Regional Clock sources within the Clock Region generate five sets of RSBRG_* [0:3] bridging clocks. One set drives to the left, one drives to right, one drives to top or bottom, and two drive diagonally (lower left or right and upper left or right). These RSBRG_* [0:3] clocks support the multi-Clock Regions clock function.

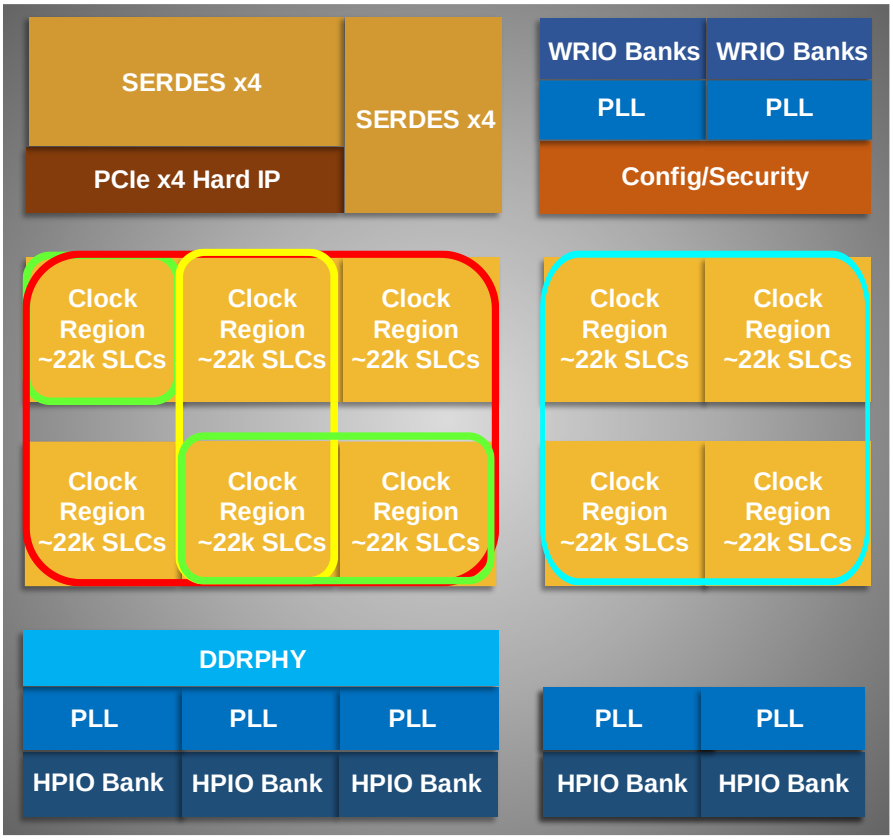


Figure 2.8. Multi-Region Clock Formations (LN2-CT-20 Devices)

2.4.1. Regional Clock Sources

The RCLK networks are sourced from multiple inputs, referred to as Regional Clock sources. The Regional Clock sources that can drive the RCLK networks are as follows:

- Dedicated Clock Pins (PCLKT pins)
- GCLKs
- SERDES Clocks
- PLL (WRIO) outputs (CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5)
- PLL (HPIO) outputs (CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5)
- Regional Bridge clocks (RSBRG_* [0:3])
- Internally generated clocks (Fabric_CLK)

The Regional Clock sources depend on which I/O bank the Clock Region is associated with, namely a WRIO bank, or a HPIO bank.

Table 2.3. High-Performance I/O (HPIO) Regional Clock Sources

Device Densities	LN2-CT-06/10	LN2-CT-16/20
GCLKs		
GCLKs [0:23]	Yes	Yes
SERDES [0:n]	n = 1 rxclk0, rxclk1, rxclk2, rxclk3, txclk	n = 3 rxclk0, rxclk1, rxclk2, rxclk3, txclk
Clock Input Pins		
PCLKT3 [0:3]	No	Yes
PCLKT4 [0:3]	No	No
PCLKT5 [0:3]	No	No
PCLKT6 [0:3]	No	No

Device Densities	LN2-CT-06/10	LN2-CT-16/20
PCLKT7 [0:3]	No	No
PCLKT8 [0:3]	No	Yes
PCLKT9 [0:3]	No	Yes
PCLKT10 [0:3]	Yes	Yes
PCLKT11 [0:3]	Yes	Yes
PLL Outputs		
PLL (HPIO)[1:n]	n = 2 clkop, clkos, clkos2, clkos3, clkos4, clkos5	n = 5 clkop, clkos, clkos2, clkos3, clkos4, clkos5
Internally Generated Clocks (Fabric_CLK)		
Fabric clock for Clock Region Fabric_CLK [0:11][1]	Yes	Yes
Regional Bridge Clocks (From Adjacent Clock Regions)		
RSBRG_* [0:3][2]	Yes	Yes

Notes:
1. Because of jitter performance, it is not recommended to use Fabric_CLK for high performance design.
2. Bridge clocks come from left, right, top, upper left, and upper right.

Table 2.4. Wide-Range I/O (WRIO) Regional Clock Sources

Device Densities	LN2-CT-06/10	LN2-CT-16/20
Clock Input Pins		
PCLKT0 [0:3]	Yes	Yes
PCLKT12 [0:3]	Yes	Yes
PLL Outputs		
PLL0 (WRIO)	clkop, clkos, clkos2, clkos3, clkos4, clkos5	clkop, clkos, clkos2, clkos3, clkos4, clkos5
PLL12 (WRIO)	clkop, clkos, clkos2, clkos3, clkos4, clkos5	clkop, clkos, clkos2, clkos3, clkos4, clkos5
Internally Generated Clocks Fabric_CLK		
Fabric clock for Clock Region Fabric_CLK [0:11][1]	Yes	Yes
Regional Bridge Clocks (From Adjacent Clock Regions)		
RSBRG_* [0:3][2]	Yes	Yes

Notes:
1. Because of jitter performance, it is not recommended to use Fabric_CLK for high performance design.
2. Bridge clocks come from left, right, bottom, lower left, and lower right.

2.4.2. Regional Clock Routing

The Regional Clock network is comprised of Regional Multiplexers (RGMUX) and RCLKs. The RCLK network takes the GCLK clocks and clock sources for each Clock Region to generate 16 RCLKs. The 16 RCLKs drive the PFU and fabric in the clock region.

The RGMUX is a two-stage multiplexer, with the region clock sources divided into two groups. The first group is the GCLKs, and the second group is RSCLK. RSCLK is sourced from a combination of different clock sources in the Clock Region. Each group generates 16 outputs. There is a DCC block on each RSCLK.

Note: The clock sources from I/O, TX, RX, or PLL are gated off by default to save dynamic power when they are not used.

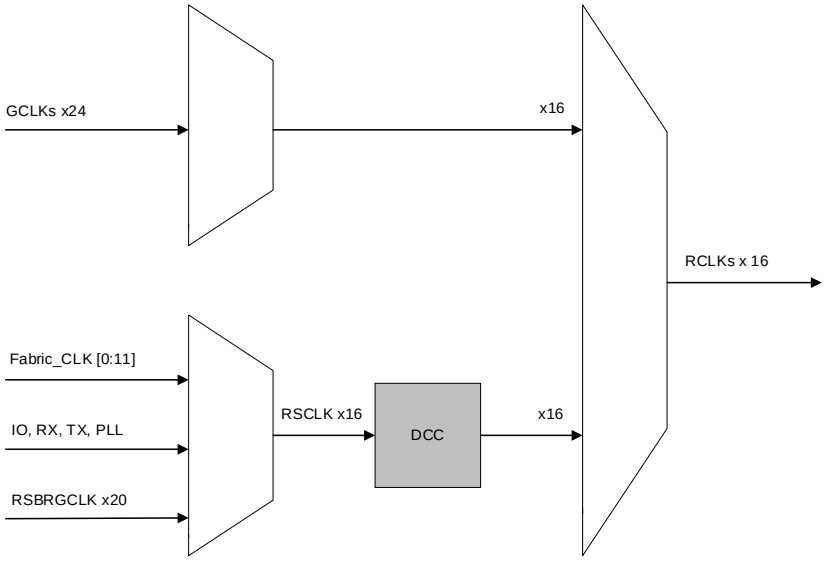


Figure 2.9. Generic RGMUX Implementation

2.5. Edge Clock (ECLK) Network

ECLKs are low skew, high speed clock resources used to clock data into and out of the high-speed DDR I/O interface of Lattice Nexus 2 devices. The ECLK network is associated with the high-performance I/O (HPIO) banks located at the bottom side of the device. These clocks, which have low injection time and skew, are suitable to drive the high-speed I/O interfaces with high fan-out capability.

Each HPIO bank has four Edge Clocks supporting Clock Region (CKR) associated with the clocks. The ECLK network is also able to bridge to adjacent left or right Clock Regions to form a wider ECLK network.

2.5.1. Edge Clock Sources

The ECLK networks are sourced from multiple inputs, referred to as Edge Clock sources. The Edge Clock sources that can drive the ECLK networks are as follows:

- Dedicated Clock Pins (PCLKT pins)
- DLLDEL outputs
- PLL (HPIO) outputs (CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5)
- ECLK Bridge clocks (EBRG_* [0:3])
- Internally generated clocks (Fabric_CLK)

Table 2.5. Edge Clock Sources

Device Densities	LN2-CT-06/10	LN2-CT-16/20
Clock Input Pins		
PCLKT3 [0:3]	No	Yes
PCLKT4 [0:3]	No	No
PCLKT5 [0:3]	No	No
PCLKT6 [0:3]	No	No
PCLKT7 [0:3]	No	No
PCLKT8 [0:3]	No	Yes
PCLKT9 [0:3]	No	Yes
PCLKT10 [0:3]	Yes	Yes
PCLKT11 [0:3]	Yes	Yes
PLL Outputs		
PLL (HPIO) [1:n]	n = 2 clkop, clkos, clkos2, clkos3, clkos4, clkos5	n = 5 clkop, clkos, clkos2, clkos3, clkos4, clkos5

Device Densities	LN2-CT-06/10	LN2-CT-16/20
Internally Generated Clocks (Fabric_CLK)		
Fabric_CLK [0:3]	No	No
Regional Bridge Clocks (From Adjacent Clock Regions)		
EBRG_* [0:3][1]	Yes	Yes

Note:
1. Bridge clocks come from left and right.

2.5.2. Edge Clock Routing

The ECLK network contains the following blocks: the ECLKINMUX, ECLKSYNCA, ECLKDIVA, ECLKBRGMUX, and DLLDEL. The ECLKINMUX collects all clock sources available as shown in the following diagram. Each ECLK input multiplexer generates a total of four ECLK clock sources, driving the high-speed I/O Interface of each I/O bank. ECLK can also be divided down before ECLK drives the BMID multiplexer (PCLK2BMID) or the RGMUX (PCLK2RCLK). In the first path through the BMID multiplexer, the ECLKs (ECLKDIVA) drive the GCLK network to RCLK network to fabric clock to the high-speed I/O interface. In the second path through the RGMUX, the ECLKs (ECLKDIVA) drive the fabric clock to the high-speed I/O interface.

Note: Only two of the four ECLKs (ECLKDIVA) can drive the BMID multiplexer.

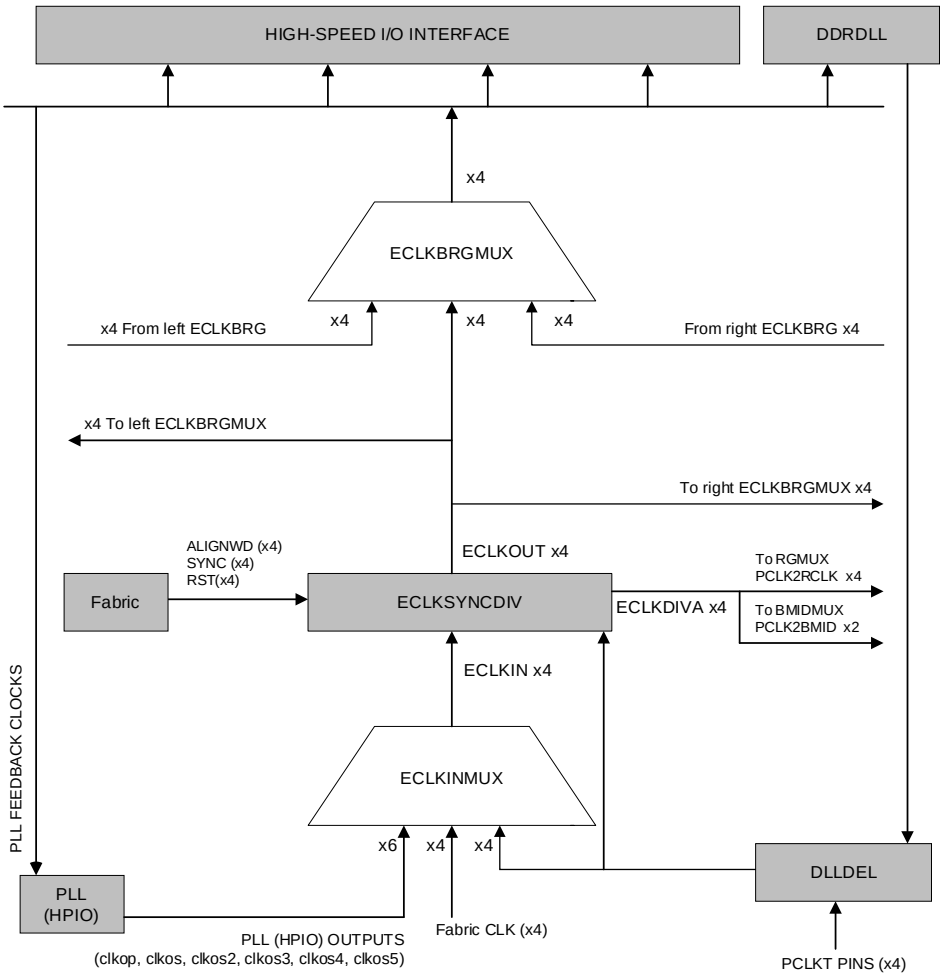


Figure 2.10. Edge Clock Sources per Bank

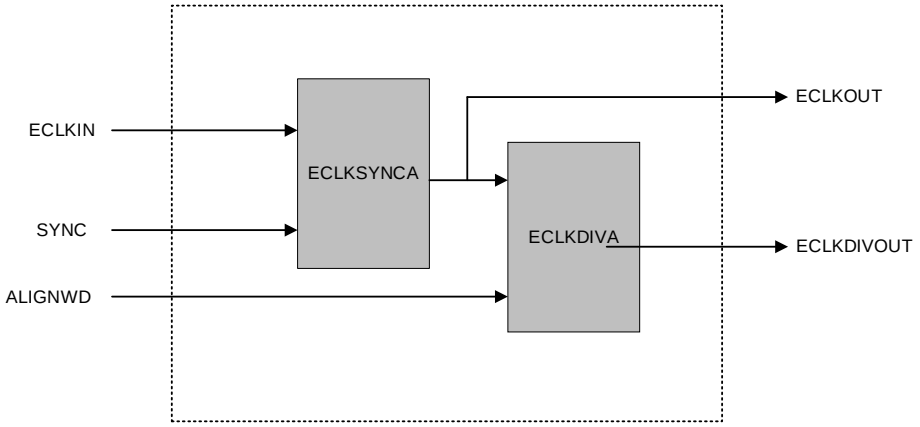
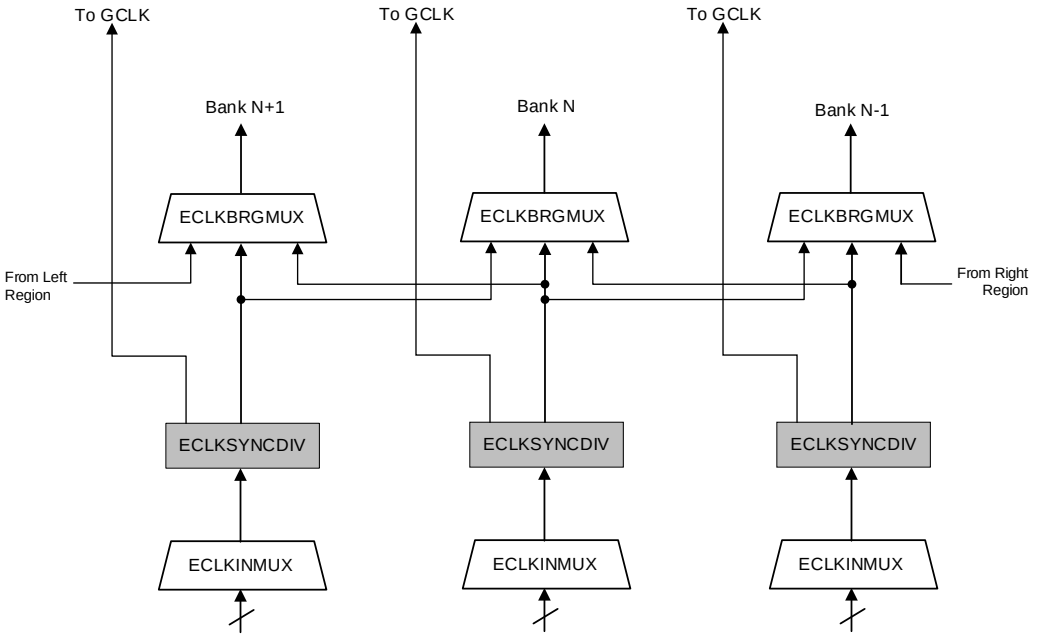


Figure 2.11. ECLKSYNCDIV

The four ECLK structures of each HPIO sector can bridge to two adjacent HPIO sectors, left and right, together to form a wider high-performance interface. To enable bridging, the DDRDLL, DLLDEL, PLL (HPIO), and the dedicated clock pins (PCLKT) must come from the same HPIO sector. To bridge across three HPIO sectors, the middle HPIO sector clock sources and PLL must be used.



Note: For HPIO Bank 3 and Bank 11, the inputs *From Left Region* and *From Right Region* are tied to ground because there is no region after that.

Figure 2.12. Edge Clock Sources Bridged to Multiple Banks

2.6. PHY Clock (PHYCLK) Network

The PHYCLK network is a special high frequency clock network that is used to support high frequency clocks for interface protocols such as DDRPHY interface for fmax up to 2.4 GHz. The PHYCLK network also generates a divided down clock that drives the Clock Region for single Clock Region interface or drives the adjacent Clock Region (left and right) through the BMID to support a wider Multi-Clock Region interface.

2.6.1. PHYCLK Sources

The PHYCLK is driven only by a special output of the PLL (HPIO) and there is one PHYCLK per I/O bank.

Table 2.6. PHYCLK Sources

Device Densities	LN2-CT-06/10	LN2-CT-16/20
PHYCLK		
PLL (HPIO) [1:n]	n = 2 phyclk	n = 5 phyclk
PHYCLK Bridge Clocks (From Adjacent Clock Regions)		
PHYCLKBRG_*[1]	Yes	Yes

Note:

1. Bridge clocks come from left and right.

2.6.2. PHYCLK Routing

The PHYCLK drives the High-Speed I/O interfaces and the DDRPHY hard IP at the same time. The DDRPHY hard IP spans three HPIO sectors and the full rate PHYCLK drives the DDRPHY full-rate clock port located at the center HPIO sector.

The PHYCLK also drives ECLKDIVA module to provide quad-rate clock, divided by 4, frequency clock. The divided down clock drives the Regional Clock Network (PHYCLK2RCLK), the slow clock domain of the DDRPHY (PHYCLKBY4), and the BMID (PHYCLK2BMID).

Unlike the ECLKs, there is no option for the PHYCLK to be the feedback clock to the HPPLLs.

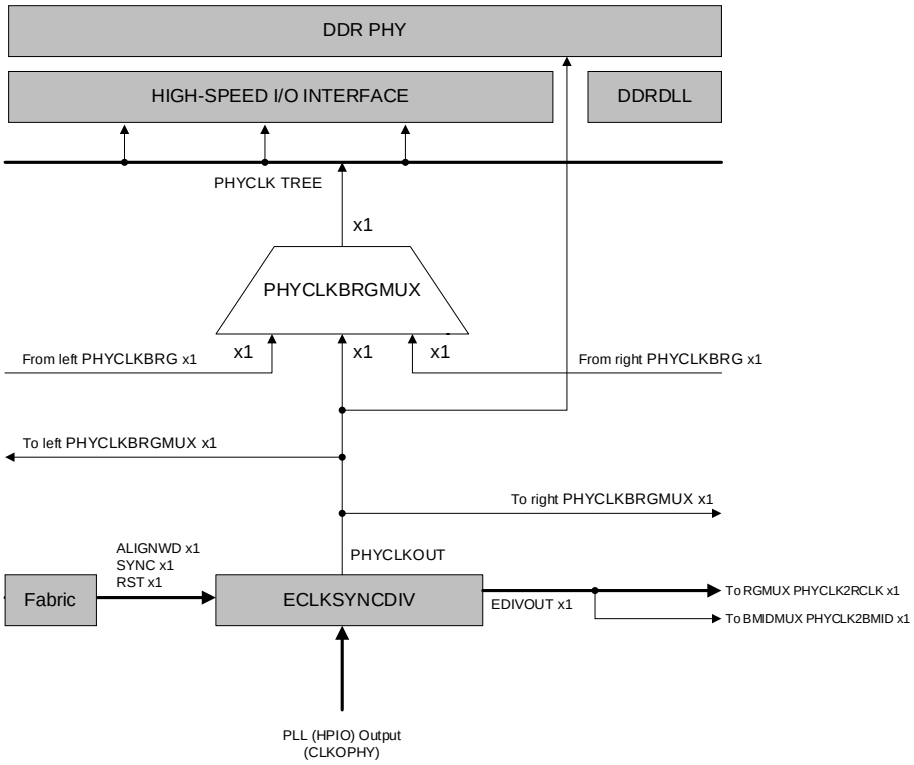


Figure 2.13. PHYCLK Network

The PHYCLKs can bridge from one HPIO Sector to the adjacent left and right HPIO Sector in the same manner as the ECLKs. However, as each DDRPHY hard IP spans three HPIO sectors, only the PHYCLK and PHYCLKBY4 of the middle HPIO drives the DDRPHY hard IP. The PHYCLK2BMID, PHYCLK2RCLK, and PHYCLKBRG CLKs are disabled when they are not used to save power.

2.7. Edge Clock Synchronizer and Divider (ECLKSYNCA and ECLKDIVA)

Edge Clock Synchronizer and Divider provide clock synchronization and clock divider functions in Lattice Nexus 2 devices.

The Edge Clock Synchronizer allows each Edge Clock to be disabled or enabled glitchlessly from core logic if desired. This allows the Edge Clock to be synchronized to an event or external signal if desired. The Edge Clock Synchronizer also allows the design to dynamically disable a clock and the associated logic in the design when clock is not needed and thus, save power. The clock synchronizer is commonly used in high-speed DDR applications such as DDR2, DDR3, and 7:1 LVDS for display.

The Edge Clock Divider provides divided down clocks used for the I/O multiplexer and demultiplexer gearing logic and they drive the Global Clock network. The inputs to the Clock Dividers are the Edge Clocks, PLL outputs, and Dedicated Clock Input pins. The outputs of the Clock Divider drive region clock network and Global Clock network (through BMID), and are mainly used for DDR I/O domain crossing. The outputs can be bypassed or divided by 2, 3.5, 4, or 5 with respect to the inputs. In Lattice Nexus 2 device, there is also a pre-divide by 2 feature for the clock source to support half rate of input clock frequency before the clock drives the ECLKDIVA divider module.

There are five ECLKSYNCA and ECLKDIVA modules available in each HPIO bank at the bottom of the device (four for Edge clocks, one for PHYCLK). The ECLKSYNCA and ECLKDIVA components can be instantiated in the source code of a design as defined in this section.

2.7.1. ECLKSYNCA Component Definition

The ECLKSYNCA component can be instantiated in the source code of a design as defined in this section. Verilog and VHDL instantiations are included.

Control signal SYNC is synchronized with ECLKIN when asserted. When control signal SYNC is asserted, the clock output is forced to low after the fourth falling edge of the input ECLKIN. When the SYNC signal is released, the clock output starts to toggle at the fourth rising edge of the input ECLKIN clock.

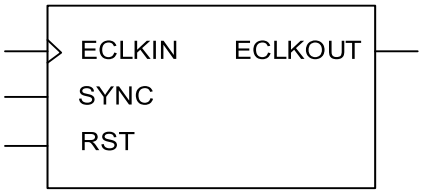


Figure 2.14. ECLKSYNCA Component Symbol

Table 2.7. ECLKSYNCA Component Port Definition

Port Name	Type	Description
ECLKIN	Input	Edge clock input port (clock signal after DLLDEL)
SYNC	Input	Signal is used to enable or disable the synchronizer
RST	Input	Reset input — Active High, asynchronously forces all outputs low. RST = 0 Clock outputs are active RST = 1 Clock outputs are OFF
ECLKOUT	Output	Non-divided clock output port

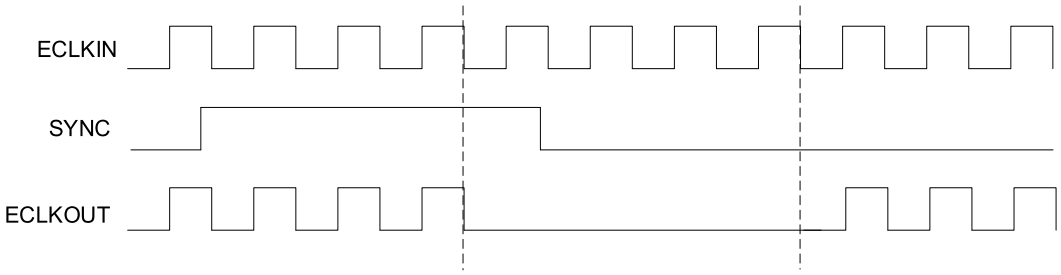


Figure 2.15. ECLKSYNCA Functional Waveform

2.7.2. ECLKSYNCA Usage in VHDL

Component Instantiation

```
Library lattice;
use lattice.components.all;
```

Component and Attribute Declaration

```
component ECLKSYNCA
port (RST : in STD_LOGIC;
      ECLKIN: in STD_LOGIC;
      SYNC : in STD_LOGIC;
      ECLKOUT : out STD_LOGIC);
end component;
```

ECLKSYNCA Instantiation

```
I1: ECLKSYNCA
port map (RST => RST,
          ECLKIN => ECLKIN,
          SYNC => SYNC,
          ECLKOUT => ECLKOUT);
```

2.7.3. ECLKSYNCA Usage in Verilog

Component and Attribute Declaration

```
module ECLKSYNCA (RST, ECLKIN, SYNC, ECLKOUT);

input  RST, ECLKIN, SYNC;
output ECLKOUT;
endmodule
```

ECLKSYNCA Instantiation

```
ECLKSYNCA I1 (
   .RST (RST),
   .ECLKIN (ECLKIN),
   .SYNC  (SYNC),
   .ECLKOUT (ECLKOUT));
```

2.7.4. ECLKDIVA Component Definition

The ECLKDIVA component can be instantiated in the source code of a design as defined in this section. Verilog and VHDL instantiations are included.

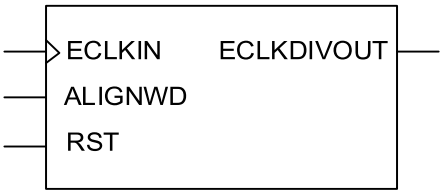


Figure 2.16. ECLKDIVA Component Symbol

Table 2.8. ECLKDIVA Component Port Definition

Port Name	Type	Description
ECLKIN	Input	Edge clock input port
ALIGNWD	Input	Signal is used for word alignment. When enabled, the signal slips the output one cycle relative to the input clock
RST	Input	Reset input — Active High, asynchronously forces all outputs low. RST = 0 Clock outputs are active RST = 1 Clock outputs are OFF
ECLKDIVOUT	Output	Divided clock output port

Table 2.9. ECLKDIVA Component Attribute Definition

Name	Value	Default	Description
CLK_DIV	2 3P5 4 5	2	ECLK DIVIDE Ratio selection (*3P5* = 3.5)
ALIGN	ENABLED DISABLED	ENABLED	Enable ECLK Synchronization
RATE	HALF FULL	HALF	Select DDR Rate
PHYBY4_SEL	ENABLED DISABLED	ENABLED	Enable PHYCLK Divide by 4
GSR	ENABLED DISABLED	ENABLED	Enable Global Set Reset

The ALIGNWD input is intended for use with high-speed data interfaces such as DDR or 7:1 LVDS Video.

2.7.5. ECLKDIVA Usage in VHDL

Component Instantiation

```
Library lattice;
use lattice.components.all;
```

Component and Attribute Declaration

```
component ECLKDIVA
generic  (CLK_DIV : string;
          ALIGN : string;
          RATE : string;
          PHYBY4_SEL : string;
          GSR : string);
port (RST : in STD_LOGIC;
      ECLKIN: in STD_LOGIC;
      ALIGNWD: in STD_LOGIC;
      ECLKDIVOUT : out STD_LOGIC);
end component;
```

ECLKDIVA Instantiation

```
attribute CLK_DIV : string;
attribute CLK_DIV of I1 : label is “2”;
attribute ALIGN : string;
attribute ALIGN of I1 : label is “ENABLED”;
```

```
attribute RATE : string;
attribute RATE of I1 : label is “HALF”;
attribute PHYBY4_SEL : string;
attribute PHYBY4_SEL of I1 : label is “ENABLED”;
attribute GSR : string;
attribute GSR of I1 : label is “DISABLED”;

I1: ECLKDIVA
generic map (CLK_DIV => “2”,
             ALIGN => “ENABLED”,
             RATE => “HALF”,
             PHYBY4_SEL => “ENABLED”,
             GSR => “DISABLED”);
port map (RST => RST,
          ECLKIN => ECLKIN,
          ALIGNWD => ALIGNWD,
          ECLKDIVOUT => ECLKDIVOUT);
```

2.7.6. ECLKDIVA Usage in Verilog

Component and Attribute Declaration

```
module ECLKDIVA (SYNC, CLK_DIV, GSR);

parameter CLK_DIV = “2”;
parameter ALIGN = “ENABLED”
parameter RATE = “HALF”
parameter PHYBY4_SEL = “ENABLED”
parameter GSR = “DISABLED”;

input  RST, ECLKIN, ALIGNWD;
output ECLKDIVOUT;
endmodule
```

ECLKDIVA Instantiation

```
defparam I1.CLK_DIV = “2”;
defparam I1.ALIGN = “ENABLED”;
defparam I1.RATE = “HALF”;
defparam I1.PHYBY4_SEL = “ENABLED”;
defparam I1.GSR = “DISABLED”;
ECLKDIVA I1 (
   .RST (RST),
   .ECLKIN (ECLKIN),
   .ALIGNWD (ALIGNWD),
   .ECLKDIVOUT (ECLKDIVOUT));
```

2.7.7. ECLKDIVA Divide Ratio Options

Table 2.10. ECLKDIVA pre-divby2 and ECLKDIVA Options

Interface	Type	Rate	Gearing	Fabric	Full/Half	CLK	Freq	Divider	ECLKDIVA
LVDS/DPHY	Tx	800	4	200	Half	ECLK	400	2	200
	Rx	800	4	200	Half	ECLK	400	2	200

Interface	Type	Rate	Gearing	Fabric	Full/Half	CLK	Freq	Divider	ECLKDIVA
LVDS/DPHY	Tx	1200	4	300	Half	PHYCLK/ECLK	600	2	300
	Rx	1200	4	300	Half	ECLK	600	2	300
LVDS	Tx	1600	8	200	Half	PHYCLK/ECLK	800	4	200
	Rx	1600	8	200	Half	ECLK	800	4	200
DPHY	Tx	1800	8	225	Half	PHYCLK/ECLK	900	4	225
	Rx	1800	8	225	Half	ECLK	900	4	225
LVDS71	Tx	1050	7	150	Half	PHYCLK/ECLK	525	3.5	150
	Rx	1050	7	150	Half	ECLK	525	3.5	150
SGMII	Tx	1250	10	125	Half	PHYCLK/ECLK	625	5	125
	Rx	1250	10	125	Half	ECLK	625	5	125

2.8. DLLDEL

DLLDEL is a passive delay component to provide necessary clock phase shift for the dedicated clock pins before driving the GCLK and ECLK network. The adjusted phase can be dynamic or static controlled. In dynamic control mode, the delay code comes from the associated DDRDLL available on the device. In static control mode, the delay control is set by software. The delay element inside the DLLDEL can be bypassed if delay is not used.

There are four DLLDEL elements in each HPIO bank. Each associate with one ECLK. There is only one DLLDEL code common to all DLLDEL modules, provided by one DDRDLL within each HPIO section.

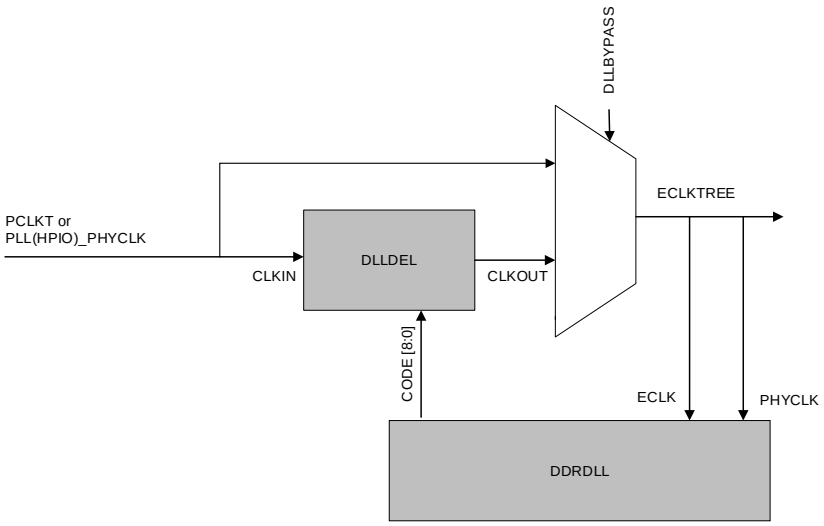


Figure 2.17. High-Level Implementation of DLLDEL and Code Control

2.8.1. DLLDELA Component Definition

The DLLDELA component can be instantiated in the source code of a design as defined in this section.

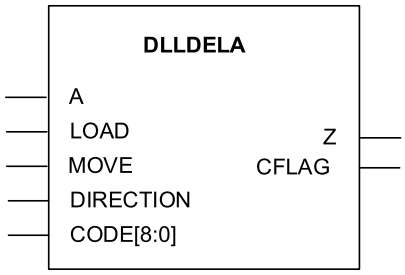


Figure 2.18. DLLDELA Component Symbol

Table 2.11. DLLDELA Component Port Definition

Port Name	Type	Description
A	Input	Data input from pin.
LOAD	Input	LOAD is set to 1 to set the delay as follows: • CODE + DEL_VALUE if DEL_ADJ is *PLUS*. • CODE - DEL_VALUE if DEL_ADJ is *MINUS*.
MOVE	Input	Pulses MOVE for the following settings: • Increment delay setting by 1 if DIRECTION is 0. • Decrement delay setting by 1 if DIRECTION is 1. MOVE needs to meet a 6 ns minimum pulse width requirement.
DIRECTION	Input	Controls whether MOVE increases or decreases the delay setting as follows: • 0 to increase. • 1 to decrease.
CODE	Input	Delay code from DDRDLL.
Z	Output	Delayed data to input register block.
CFLAG	Output	Flag indicating the delay counter has reached max (when moving up) or min (when moving down) value.

Table 2.12. DLLDELA Component Attribute Definition

Name	Value	Default	Description
DEL_VALUE	0	0	Offset for target delay adjustment (0-511).
DEL_ADJ	PLUS MINUS	PLUS	Determines the adjusted delay setting when using the LOAD port. See the description for LOAD.
GSR	ENABLE DISABLED	ENABLED	Enable or disable GSR feature.

2.8.2. DLLDELA Usage in VHDL

Component Instantiation

```
Library lattice;
use lattice.components.all;
```

Component and Attribute Declaration

```
component DLLDELA
generic (DEL_ADJ : string;
DEL_VALUE : string;
GSR : string);
```

```
port (
A : in std_logic;
LOAD : in std_logic;
MOVE : in std_logic;
DIRECTION : in std_logic;
CODE : in std_logic;
Z : out std_logic;
CFLAG : out std_logic
);
end component;
```

DLLDELA Instantiation

```
DLLDELA_0 : DLLDELA
generic map (
  DEL_ADJ  => "PLUS",
  DEL_VALUE => "0",
  GSR => "ENABLED"
)
port map (
  A         => A ,
  LOAD      => LOAD,
  MOVE      => MOVE,
  DIRECTION => DIRECTION,
  CODE      => CODE,
  Z         => Z,
  CFLAG     => CFLAG
```

2.8.3. DLLDELA Usage in Verilog

Component and Attribute Declaration

```
module DLLDELA(A, LOAD, MOVE, DIRECTION, CODE, Z, CFLAG);
input A;
input LOAD;
input MOVE;
input DIRECTION;
input [8:0] CODE;
output Z;
output CFLAG;
```

DLLDELA Instantiation

```
DLLDELA
#(
  .DEL_ADJ   ("PLUS"),
  .DEL_VALUE ("0"),
  .GSR       ("ENABLED")
 )DLLDEL_0 (
  .A         (A),
  .LOAD      (LOAD),
  .MOVE      (MOVE),
  .DIRECTION (DIRECTION),
  .CODE      (CODE),
  .Z         (Z),
```

```
  .CFLAG     (CFLAG)
);

endmodule
```

2.9. Dynamic Clock Select (DCS)

The Dynamic Clock Select feature provides a dynamic, run-time selectable glitchless selection between two independent clock sources. If the DCS feature is not used, clock switching may be glitchy.

There are four dynamic clock select blocks in LN2-CT-20 devices. The DCSCMUX consists of DCSMUX, DCS, and CMUX to generate two set of GCLKs. One set drives the left clock regions and one drives right clock regions.

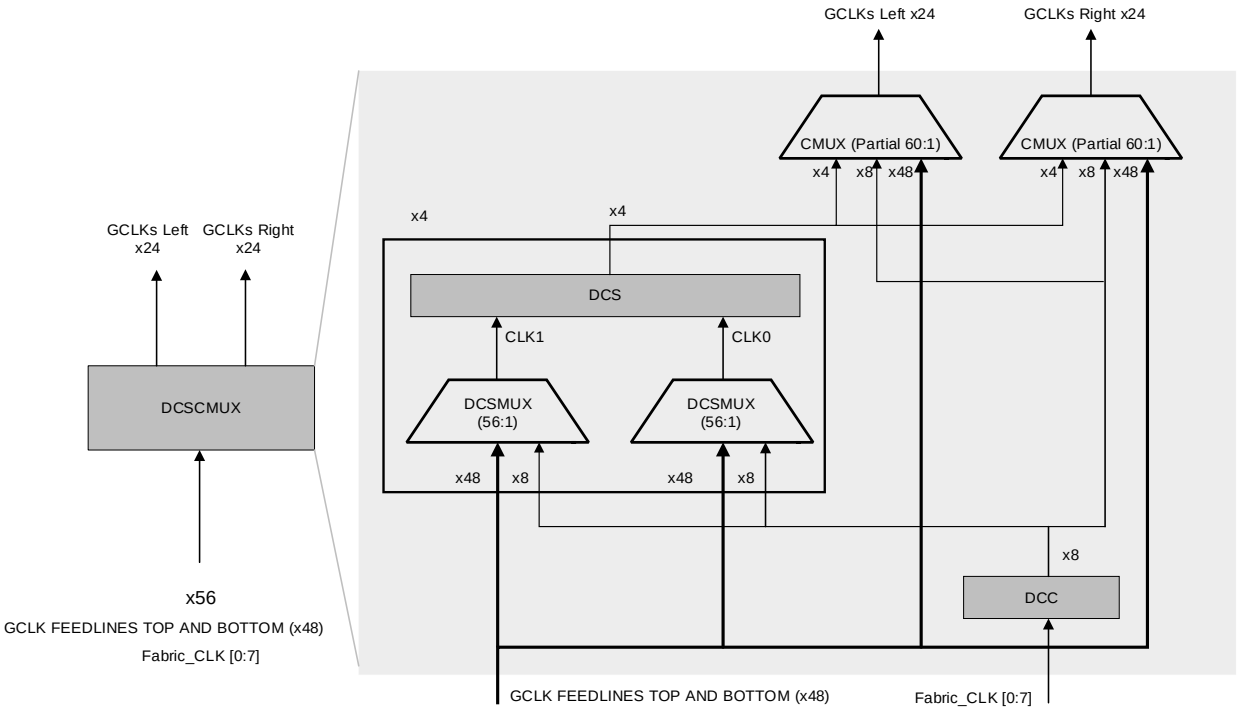


Figure 2.19. DCSCMUX

The DCS block allows dynamic and glitchless selection between two PCLK clock sources. The DCS block shares the same clock resource as any PCLK CMUX, enabling the DCS function to be used on any two primary clock sources. The inputs to the DCS block come from the GCLK Feedlines (such as outputs of TMID and BMID multiplexers) and 8 Fabric clocks. The output of the DCS feeds the CMUXes and are selected to drive the GCLKs.

Reset signals disable the DCS function during power up to avoid any toggling of the GCLK network.

The *DCSMODE* attribute sets the behavior of the DCS output. The DCS attributes are described in Table 2.14.

2.9.1. DCS Timing Diagrams

The DCS block allows dynamic and glitchless selection between two PCLK clock sources. The DCS block shares the same clock resource as any PCLK CMUX. Therefore, the DCS function can be performed on any two primary clock sources. Figure 2.20, Figure 2.21, Figure 2.22, and Figure 2.23 show the DCS in glitchless operation in conjunction with the DCSMODE attribute. Figure 2.27 shows the non-glitchless bypass operation scenario.

2.9.1.1. Functionality – DCSMODE = *POS*

The selection switches from current clock to target clock. For posedge configuration, the latch state is low i.e. output clock is held at low state during transition. The following sequence of events occur when SEL toggles:

1. Current clock must see posedge then negedge, then is deactivated.
2. Target clock must see posedge then negedge, then output is successfully switched over.

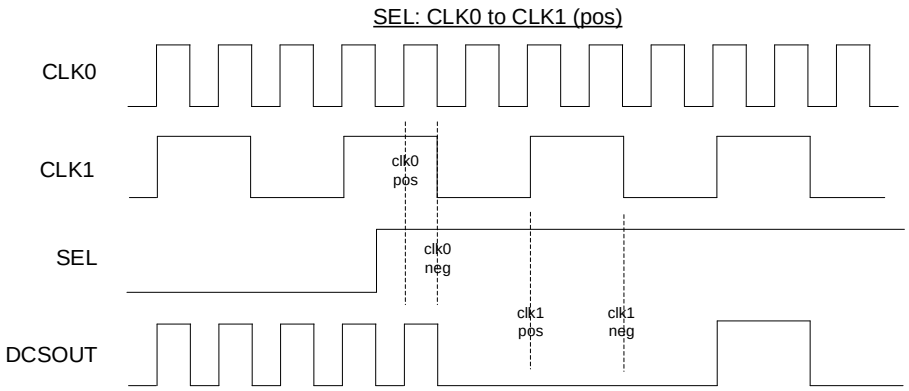


Figure 2.20. Posedge DCS Switch from SEL: 0 ≥ 1

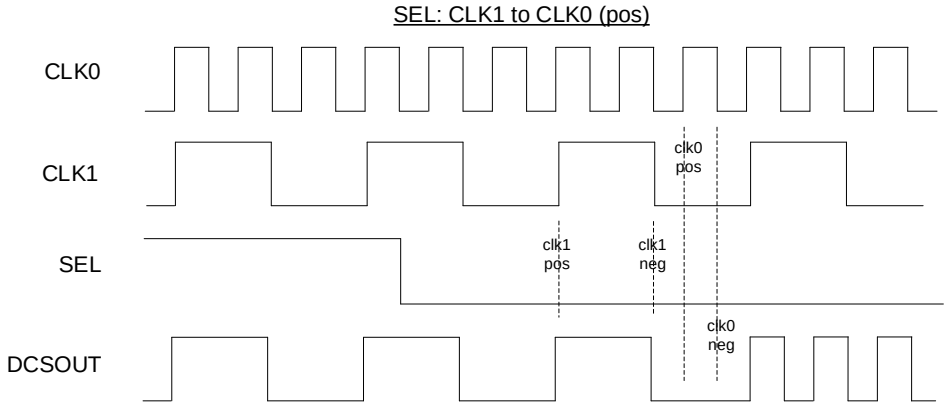


Figure 2.21. Posedge DCS Switch from SEL: 1 ≥ 0

2.9.1.2. Functionality – DCSMODE = *NEG*

The selection switches from current clock to target clock. For negedge configuration, the latch state is high. The following sequence of events occur when SEL toggles:

1. Current clock must see negedge then posedge, then is deactivated.
2. Target clock must see negedge then posedge, then output is successfully switched over.

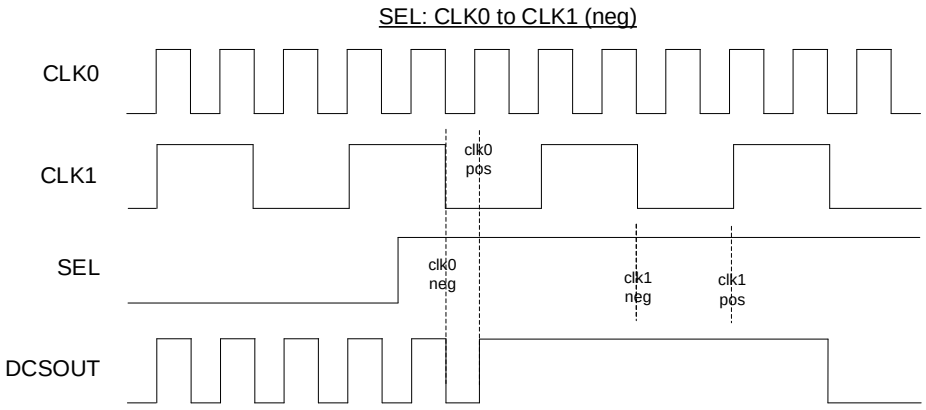


Figure 2.22. Negedge DCS Switch from SEL: 0 ≥ 1

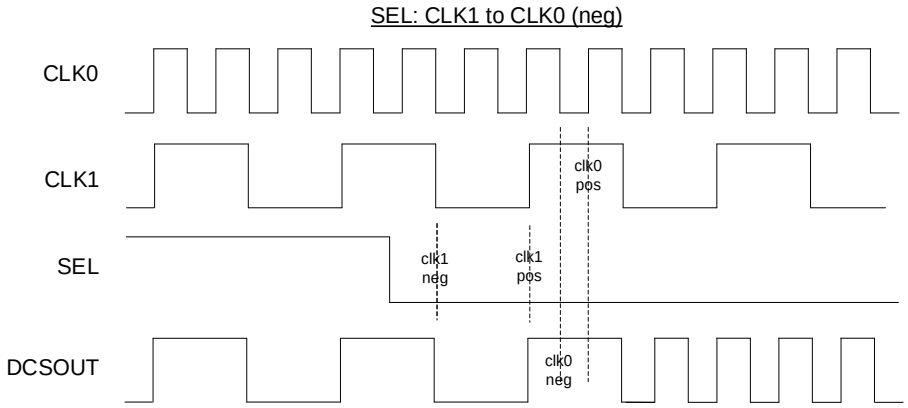


Figure 2.23. Negedge DCS Switch from SEL: 1 ≥ 0

2.9.1.3. Functionality – DCSMODE = *CLK0* or *CLK1*

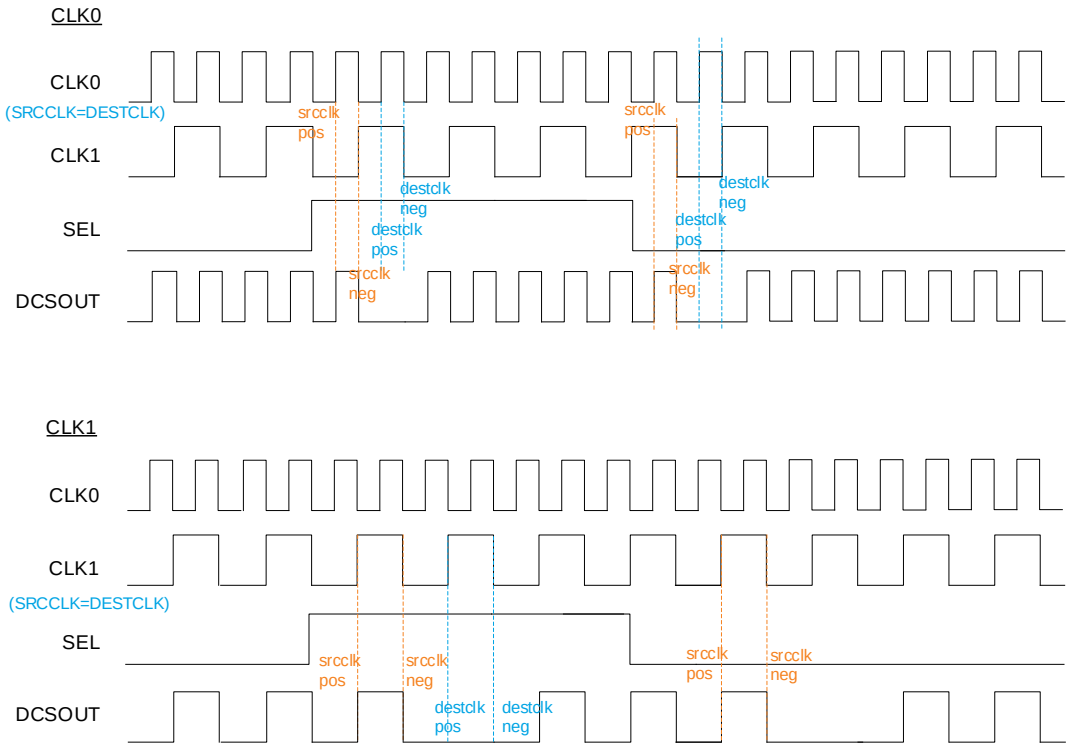


Figure 2.24. DCSMODE = *CLK0* or *CLK1*

2.9.1.4. Functionality – DCSMODE = *CLK0_LOW* or *CLK0_HIGH*

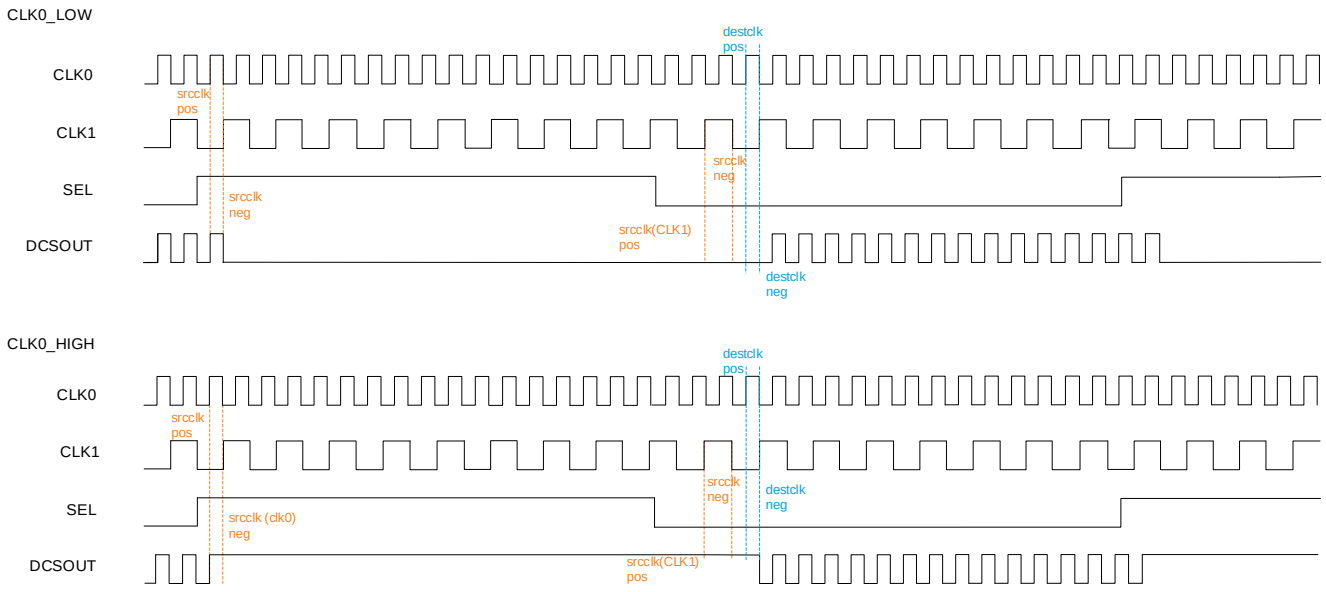


Figure 2.25. DCSMODE = *CLK0_LOW* or *CLK0_HIGH*

2.9.1.5. Functionality – DCSMODE = *CLK1_LOW* or *CLK1_HIGH*

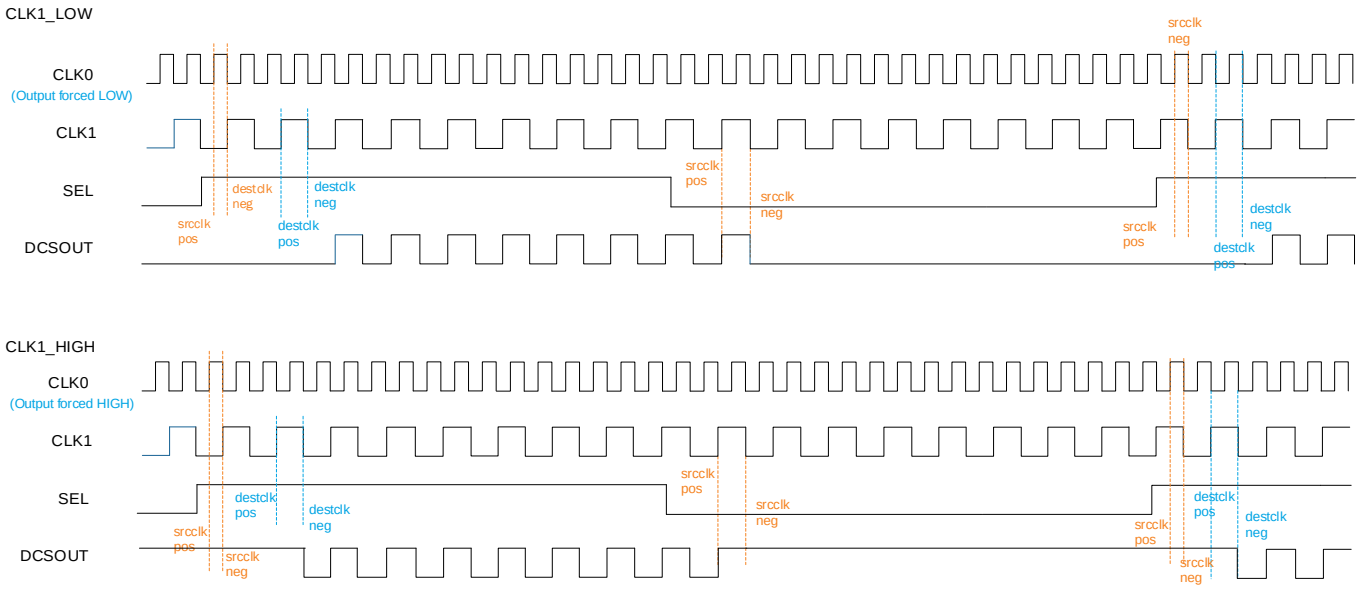


Figure 2.26. DCSMODE = *CLK1_LOW* or *CLK1_HIGH*

2.9.1.6. Functionality – MODELSEL = *1*, DCS in Bypass Mode

When MODESEL is high, the switch is in bypass mode. The output clock transitions immediately from the current clock to the target clock and may have glitches.

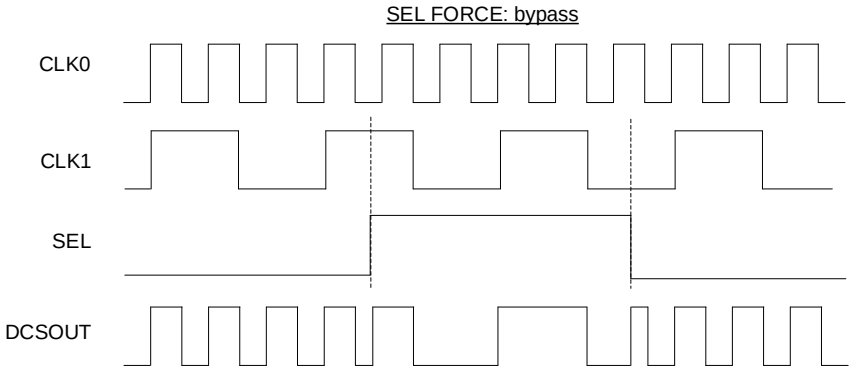


Figure 2.27. MODESEL = *1* DCS Clock Switch

2.9.2. DCSA Component Definition

The DCSA component can be instantiated in the source code of a design as defined in this section.

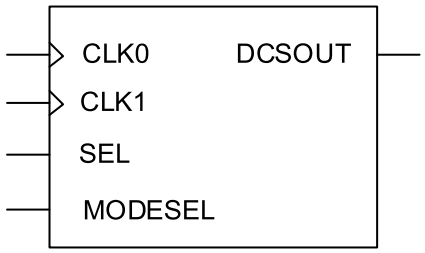


Figure 2.28. DCSA Component Symbol

Table 2.13. DCSA Component Port Definition

Port Name	Type	Description
CLK0	Input	Clock Input port 0 — Default
CLK1	Input	Clock Input port 1
SEL	Input	Input Clock Select
MODESEL	Input	1 = DCS function disabled 0 = DCS function enabled Selects Glitchless (*0*) or Non-Glitchless (*1*) behavior
DCSOUT	Output	Clock Output Port

The following table provides the behavior of the DCS output based on the setting of the *DCSMODE* attribute and the MODESEL pin input. The MODESEL pin is dynamic and can toggle during operation. The glitchless switching is only achievable when MODESEL = *0*.

Table 2.14. DCSA DCSMODE Attribute

Attribute Name	Attribute Value	Output		Description
		SEL = 0	SEL = 1	
DCSMODE MODESEL = 0	CLK0 (default)	CLK0	CLK0	Buffer for CLK0.
	CLK1	CLK1	CLK1	Buffer for CLK1.
	POS	CLK0	CLK1	Rising edge triggered. Latched state is high.
	NEG	CLK0	CLK1	Falling edge triggered. Latched state is low.
	CLK1_LOW	0	CLK1	SEL is active high. Disabled output is low.
	CLK1_HIGH	1	CLK1	SEL is active high. Disabled output is high.
	CLK0_LOW	CLK0	0	SEL is active low. Disabled output is low.
	CLK0_HIGH	CLK0	1	SEL is active low. Disabled output is high.
	HIGH	1	1	Output is always high.
	LOW	0	0	Output is always low.
MODESEL = 1	Non-Glitchless	CLK0	CLK1	—

2.9.3. DCSA Usage in VHDL

Component Instantiation

```
Library lattice;
use lattice.components.all;
```

Component and Attribute Declaration

```
component DCSA
```

```
generic (DCSMODE : string := “LOW”);
port (CLK0 : in STD_LOGIC;
      CLK1 : in STD_LOGIC;
      SEL : in STD_LOGIC;
      MODESEL : in STD_LOGIC;
     DCSOUT : out STD_LOGIC);
end component;
```

DCS Instantiation

```
attribute DCSMODE : string;
attribute DCSMODE of DCSinst0 : label is "LOW";
I1: DCS
generic map (
   DCSMODE => “VCC”);
port map (
   CLK0 => CLK0,
   CLK1 => CLK1,
   SEL => SEL,
   MODESEL => MODESEL,
   DCSOUT => DCSOUT);
```

2.9.4. DCSA Usage in Verilog

Component and Attribute Declaration

```
module DCSA (CLK0, CLK1, SEL, MODESEL, DCSOUT);
input CLK0;
input CLK1;
input SEL;
input MODESEL;
output DCSOUT;
endmodule
```

DCSA Instantiation

```
defparam DCSInst0.DCSMODE = “LOW”;
DCSA DCSInst0 (
  .CLK0 (CLK0),
  .CLK1 (CLK1),
  .SEL (SEL),
  .MODESEL (MODESEL),
  .DCSOUT (DCSOUT));
```

2.10. Dynamic Clock Control (DCC)

The Lattice Nexus 2 device has a power-saving feature known as Dynamic Clock Control. This feature allows internal logic to dynamically enable or disable the GCLK networks, thus enabling overall dynamic power consumption of the device. This gating function does not create glitches or increase the clock latency to the Global Clock network. There is a DCC element associated with each 48 GCLK networks.

The Lattice Nexus 2 device clock architecture allows both DCC and DCS to function at the same time. Care must be taken when CLKO is used as input to the PLL. The DCC must remain enabled, otherwise if the PLL input clock stops toggling, the PLL loses locked and the PLL output clock also stops toggling.

Dynamic Clock Control allows dynamic clock to enable and disable the MIDMUX Feed Line and the fabric clocks from the core. When a Feed Line is disabled, all the logic and clock signals that are fed by this Feed Line do not toggle. Hence, Dynamic Clock Control reduces the overall dynamic power consumption of the device.

The following diagrams show how DCC is available for all clock sources driving the GCLKs.

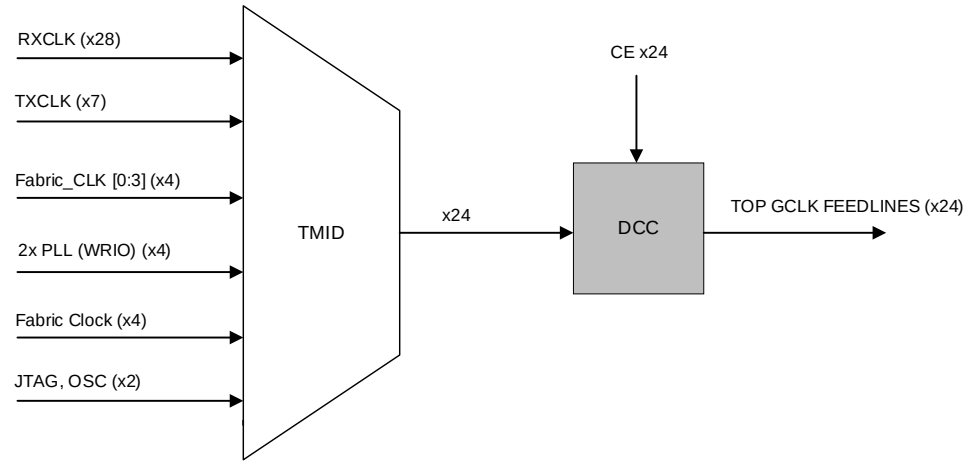


Figure 2.29. TMID Multiplexer, DCC Elements, and Top GCLK Feedlines

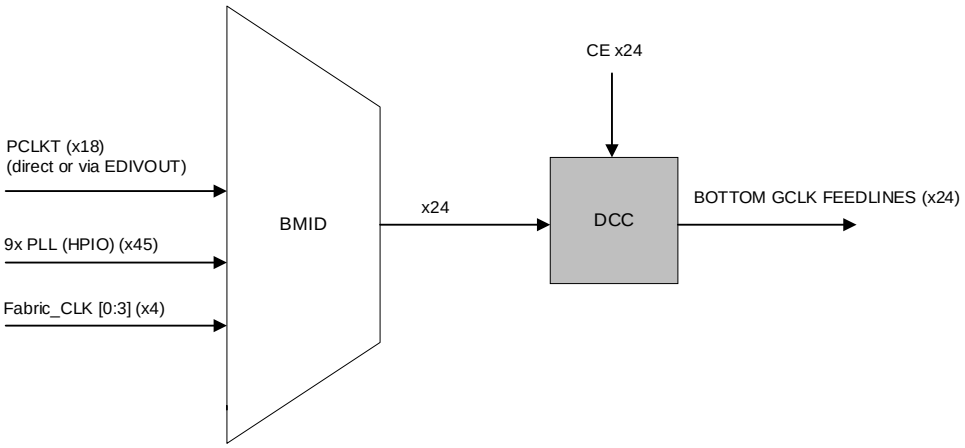


Figure 2.30. BMID Multiplexer, DCC Elements, and Bottom GCLK Feedlines

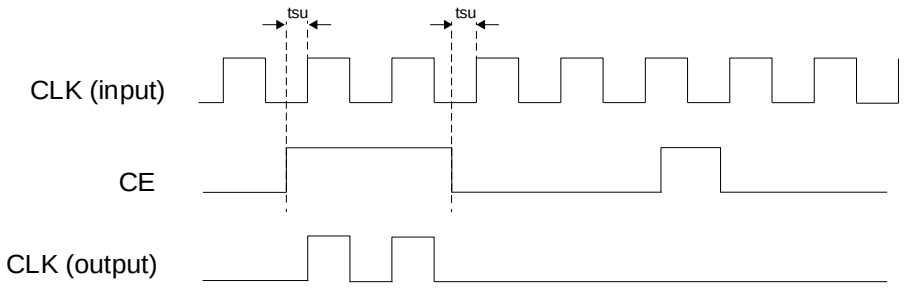


Figure 2.31. Glitchless DCC Functional Waveform

2.10.1. DCCA Component Definition

The DCCA component can be instantiated in the source code of a design as defined in this section.

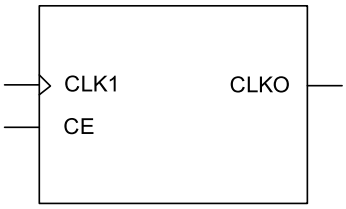


Figure 2.32. DCCA Component Symbol

Table 2.15. DCCA Component Port Definition

Port Name	Type	Description
CLKI	Input	Clock input port.
CE	Input	Clock enable port. • CE = 0 CLKO is disabled (CLKO = 0). • CE = 1 CLKO is enabled (CLKO = CLKI).
CLKO	Output	Clock output port.

2.10.2. DCCA Usage in VHDL

Component and Attribute Declaration

```
library lattice;
use lattice.components.all;
component DCCA
port (CLKI : in STD_LOGIC;
      CE : in STD_LOGIC;
      CLKO : out STD_LOGIC);
end component;
```

DCCA Instantiation

```
I1: DCCA
port map (
  CLKI => CLKI,
  CE => CE,
  CLKO => CLKO);
```

2.10.3. DCCA Usage in Verilog

Component and Attribution Declaration

```
module DCCA (CLKI,CE,CLKO);
input CLKI;
input CE;
output CLKO;
endmodule
```

DCCA Instantiation

```
DCCA DCCAInst0 (
.CLKI (CLKI),
.CE (CE),
.CLKO (CLKO));
```

2.11. Internal Oscillator (OSCE)

An internal programmable rate oscillator is provided on all Lattice Nexus 2 devices. The oscillator can be used for FPGA configuration, system monitoring/ATM (Anti-Tampering Monitor) block, and as a user logic clock source that is available after FPGA configuration. There is one OSCE on Lattice Nexus 2 devices. The oscillator clock output is routed directly to primary clocking.

The oscillator output is not a high-accuracy clock, having a ±10% variation in the output frequency. Oscillator is mainly used for circuits that do not require a high degree of clock accuracy. Examples of usage are asynchronous logic blocks such as a timer, reset generator, or other logic that require a constantly running clock.

The oscillator performs multiple functions on the Lattice Nexus 2 device. An internal programmable rate oscillator is provided on all Lattice Nexus 2 devices. The oscillator can be used for FPGA configuration, system monitoring or ATM block, and as a user logic clock source that is available after FPGA configuration. Lattice Nexus 2 devices has one OSCE where the oscillator clock output is routed directly to primary clocking.

The oscillator performs multiple functions on the Lattice Nexus 2 device. The OSCE generates the user clock that drives the FPGA clock tree; this user clock is dynamically selectable between 400 MHz or 320 MHz, with 1–256 programmable divider options.

2.11.1. OSCE Component Definition

The OSCE component can be instantiated in the source code of a design as defined in this section.

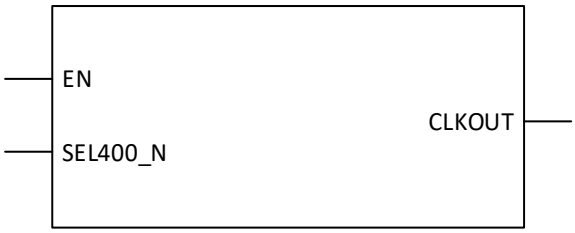


Figure 2.33. OSCE Component Symbol

Table 2.16. OSCE Component Port Definition

Port Name	Type	Description
EN	Input	Enable CLKOUT
SEL400_N	Input	Switch between 400 MHz and 320 MHz
CLKOUT	Output	Output clock to fabric

Table 2.17. OSCE Component Attribute Definition

Name	Value	Default	Description
CLK_DIV	1–256	1	Clock divider for CLKOUT

2.11.2. OSCE Usage in VHDL

Component Instantiation

```
Library lattice;
use lattice.components.all;
```

Component and Attribute Declaration

```
component OSCE
generic (CLK_DIV : string);
port (
  EN : in std_logic;
```

```
  SEL400_N: in std_logic;
  CLKOUT : out std_logic;
);
```

OSCE Instantiation

```
attribute CLK_DIV : string;
attribute CLK_DIV of I1 : label is “1.0”;

I1: OSCE
generic map (CLK_DIV => “1.0”);
port map (
  EN => EN,
  SEL400_N =>SEL400_N,
  CLKOUT => CLKOUT,
);
```

2.11.3. OSCE Usage in Verilog

Component and Attribute Declaration

```
module OSCE(EN, SEL400_N, CLKOUT);
input EN;
inputSEL400_N;
output CLKOUT;
endmodule
```

OSCE Instantiation

```
defparam OSCEInst0.CLK_DIV = “1.0”;

OSCE Inst0
(
.EN (EN),
.SEL400_N (SEL400_N),
.CLKOUT (CLKOUT),
);
```

2.12. General Routing for Clocks

The Lattice Nexus 2 device architecture supports the ability to use general routing for a clock. This capability is intended to be used for small areas of the design to allow additional flexibility in linking dedicated clocking resources and building very small clock trees. General routing cannot be used for Edge Clocks for applications that use the DDR registers in the I/O components of the FPGA.

Software limits the distance of a general routing based (gated) clock to one PFU in distance to a primary clock entry point. If the software cannot place the clock gating logic close enough to a primary clock entry point, the following error occurs:

- ERROR-par – Unable to reach a primary clock entry point for general route clock <net> in the minimum required distance of one PFU.

There are multiple entry points to the Primary clock routing throughout the Lattice Nexus 2 device fabric. In this case, it is recommended to add a preference for this gated clock to use primary routing. The PFU generated clock can reach the neighboring left and right PFUs.

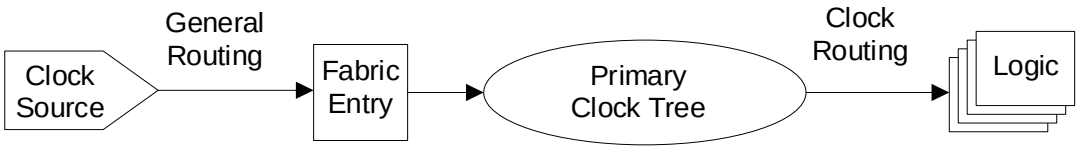


Figure 2.34. Gated Clock to the Primary Clock Routing

For a very small clock domain, the distance of a general routing based (gated) clock can be limited to one PFU in distance to the logic it clocks. This is done by grouping this logic (UGROUP) with a *BBOX = 1, 1* (see **Lattice Radiant Help > Constraints Reference Guide > Preferences > UGROUP**) and specify a *PROHIBIT PRIMARY* on the generated clock. If the software cannot place the logic tree within the BBOX, an error occurs.

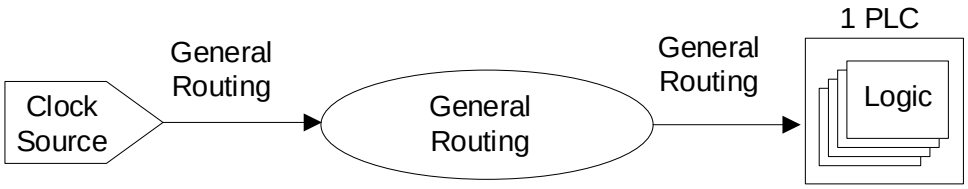


Figure 2.35. Gated Clock to Small Logic Domain

3. sysCLOCK PLL

3.1. sysCLOCK PLL Overview

The Lattice Nexus 2 sysCLOCK™ PLLs can be used for a variety of clock management applications such as frequency synthesis (multiplication and division of a clock), clock injection delay removal, clock phase adjustment, and clock timing adjustment.

The Lattice Nexus 2 PLL supports frequency synthesis by enabling the input reference clock to be multiplied up or divided down. The reference clock and feedback clock can come from various sources. The Lattice Nexus 2 PLL also supports multiple output clock selections, with each output clock having a different frequency. You can then dynamically enable or disable each clock output.

The Lattice Nexus 2 PLL supports the clock injection delay removal feature where delays associated with the PLL and clock tree are removed. This feature is typically used to reduce clock path delays and is performed by aligning the PLL input clock with a feedback clock from the clock tree.

The Lattice Nexus 2 PLL further supports a clock phase adjustment feature. This feature provides the ability to set a specific phase offset between the outputs of the PLL. The clock output selected as feedback cannot use the phase adjustment feature which can cause the PLL to unlock. Each clock output can support an independent phase shift value. The following diagram shows the native signals of the Lattice Nexus 2 PLL block diagram. These signals are abstracted in the PLL module in the IP Catalog of the Lattice Radiant software.

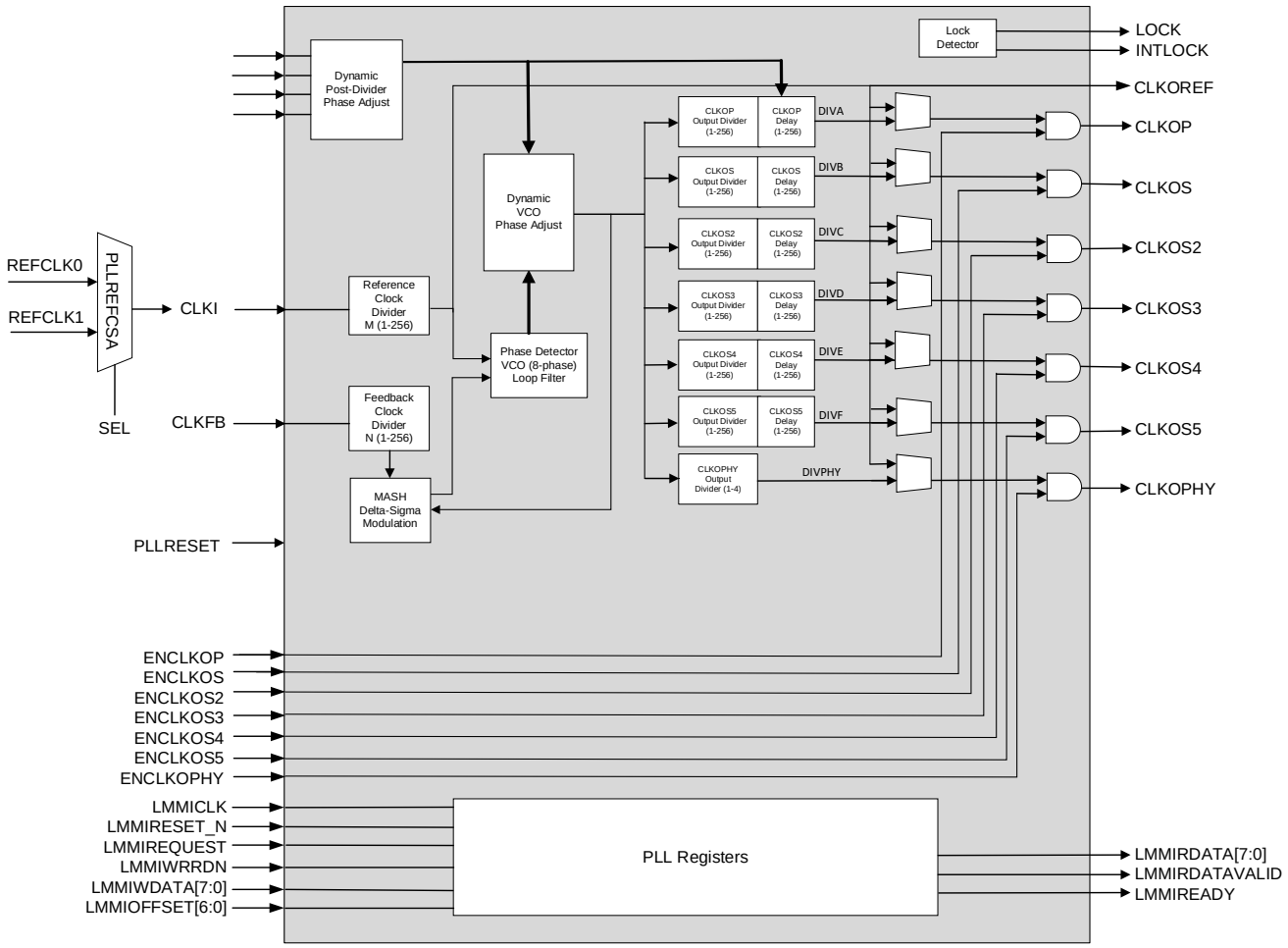


Figure 3.1. Lattice Nexus 2 PLL Block Diagram

The following table shows the number of PLLs available in the different Lattice Nexus 2 devices.

Table 3.1. General Purpose PLLs (GPLLs)

Parameter	Description	LN2-CT-06/10	LN2-CT-16/20
Number of PLLs (WRIO)	General purpose Phase Locked Loops. One per Wide-Range I/O.	2	2
Number of PLLs (HPIO))	General purpose Phase Locked Loops. One per High-Performance I/O Region.	2	5

3.1.1. PLL Input Sources

The PLL Input sources are as follows:

- Dedicated PLL Input Clock Pins (PLLT pins)
- Dedicated Clock Pins (PCLK pins)
- Global Clock Routing
- Regional Clock Routing
- Edge Clock Routing

Table 3.2. PLL Input Sources, Feedback Sources, and Output Clocks

PLL	Input Sources	Feedback Sources	Output Clocks
PLL (WRIO)	PCLKT0 [0:3] PCLKT12[0:3] PLLT0 PLLT12 Fabric_CLKs (from GCLK)	PLLFBK Fabric_CLKs (from GCLK) Edge Clocks	clkop, clkos, clkos2, clkos3, clkos4, clkos5
PLL (HPIO)	PCLKT3-11[0:3] PLLT3-11 Fabric_CLKs (from GCLK)	ECLKs Fabric_CLKs (from GCLK)	clkop, clkos, clkos2, clkos3, clkos4, clkos5, CLKOPHY

Every PLL has a dedicated low skew input that routes directly to the corresponding reference clock input. There are two reference clocks per PLL. These are the recommended inputs for a PLL. You can route a PLL input from the Global Clock routing, but the routing incurs more clock input injection delays, which are not natively compensated for using feedback, compared to a dedicated PLL input.

3.2. sysCLOCK PLL Component Definition

The PLL component can be instantiated in the source code of a design as defined in this section.

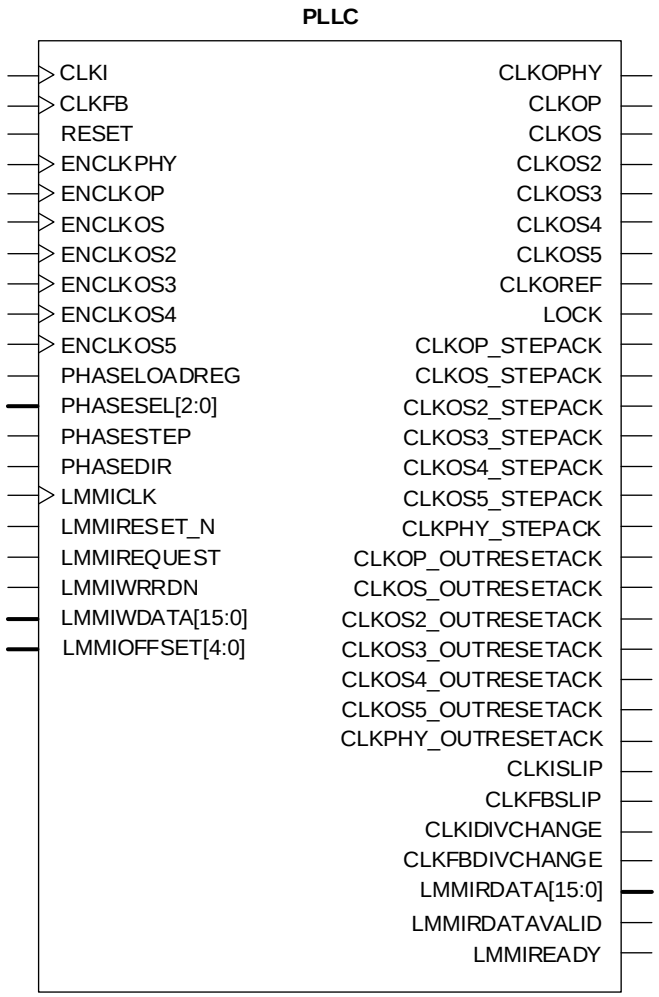


Figure 3.2. PLL Component Instance

Table 3.3. PLL Component Port Definition

Signal	Type	Description
CLKI	Input	Input Clock to PLL.
CLKFB	Input	Feedback Clock.
RESET	Input	Resets the entire PLL.
ENCLKOP	Input	Enable PLL output CLKOP.
ENCLKOS	Input	Enable PLL output CLKOS.
ENCLKOS2	Input	Enable PLL output CLKOS2.
ENCLKOS3	Input	Enable PLL output CLKOS3.
ENCLKOS4	Input	Enable PLL output CLKOS4.
ENCLKOS5	Input	Enable PLL output CLKOS5.
ENCLKPHY	Input	Enable PLL output PHYCLK.
CLKOP	Output	PLL main output clock.
CLKOS	Output	PLL output clock.
CLKOS2	Output	PLL output clock2.
CLKOS3	Output	PLL output clock3.

Signal	Type	Description
CLKOS4	Output	PLL output clock4.
CLKOS5	Output	PLL output clock5.
CLKOPHY	Output	PLL output PHYCLK – drives the PHYCLK tree.
LOCK	Output	Indicates PLL is now locked to CLKI. Active high indicates PLL is locked.
CLKOREF	Output	REFCLK output (bypass PLL).
LMMICLK	Input	Fabric LMMI interface clock.
LMMIRESET_N	Input	Fabric LMMI interface reset, active low.
LMMIREQUEST	Input	Fabric LMMI interface request signal.
LMMIWDATA[15:0]	Input	Fabric LMMI interface write data.
LMMIWRRDN	Input	Fabric LMMI interface Write/Read control; 1=write, 0=read.
LMMIOFFSET[4:0]	Input	Fabric LMMI interface address offset (LSB of address bus).
LMMIRDATA[15:0]	Output	Fabric LMMI interface read data.
LMMIRDATA_VALID	Output	Fabric LMMI interface read data valid signal.
LMMIREADY	Output	Fabric LMMI interface ready signal.
Dynamic Phase Adjustment		
PHASESTEP	Input	Dynamic Phase adjustment step.
PHASEDIR	Input	Dynamic Phase adjustment direction. 0: Rotates to later phase; 1: Rotates to earlier phase.
PHASELOADREG	Input	Load dynamic phase adjustment values into PLL.
PHASESEL[2:0]	Input	Select the divider output affected by Dynamic Phase adjustment.
CLKOP_STEPACK	Output	CLKOP VCS PS status.
CLKOS_STEPACK	Output	CLKOS VCS PS status.
CLKOS2_STEPACK	Output	CLKOS2 VCS PS status.
CLKOS3_STEPACK	Output	CLKOS3 VCS PS status.
CLKOS4_STEPACK	Output	CLKOS4 VCS PS status.
CLKOS5_STEPACK	Output	CLKOS5 VCS PS status.
CLKPHY_STEPACK	Output	CLKPHY VCS PS status.
CLKOP_OUTRESETACK	Output	CLKOP reset status.
CLKOS_OUTRESETACK	Output	CLKOS reset status.
CLKOS2_OUTRESETACK	Output	CLKOS2 reset status.
CLKOS3_OUTRESETACK	Output	CLKOS3 reset status.
CLKOS4_OUTRESETACK	Output	CLKOS4 reset status.
CLKOS5_OUTRESETACK	Output	CLKOS5 reset status.
CLKPHY_OUTRESETACK	Output	CLKPHY reset status.
CLKISLIP	Output	Reference clock slip output.
CLKFBSLIP	Output	Feedback clock slip output.
CLKIDIVCHANGE	Output	Reference divider count change enable output.
CLKFBDIVCHANGE	Output	Feedback divider count change enable output.

Table 3.4. PLL Component Attribute

Name	Values	Description
FCLKI	100 *(default)*	None
FVCO	1600 *(default)*	None
CLKOP_OUT_SEL	DIVA *(default)*	Select output to CLKOP 0: DIVA, 1: CLKI
CLKOS_OUT_SEL	DIVB *(default)*	Select output to CLKOS 0: DIVB, 1: CLKI
CLKOS2_OUT_SEL	DIVC *(default)*	Select output to CLKOS2 0: DIVC, 1: CLKI
CLKOS3_OUT_SEL	DIVD *(default)*	Select output to CLKOS3 0: DIVD, 1: CLKI
CLKOS4_OUT_SEL	DIVE *(default)*	Select output to CLKOS4 0: DIVE, 1: CLKI
CLKOS5_OUT_SEL	DIVF *(default)*	Select output to CLKOS5 0: DIVF, 1: CLKI

Name	Values	Description
CLKPHY_OUT_SEL	DIVPHY *(default)*	Select output to CLKPHY 0: DIVPHY, 1: CLKI
SYNC_CLKOP	DISABLED *(default)*	Enable output(s) sync with CLKOP 0: DISABLED; 1: ENABLED
PHASE_SOURCE	STATIC *(default)*	Select DYN (Fabric signals) for dynamic phase shift
STATIC_PHASE_SEL	CLKOP *(default)*	Select which output clocks is phase shifted
STATIC_PHASE_LOADREG	NO *(default)*	Immi equivalent of Fabric load signal
STATIC_VCO_PHASE_STEP	NO *(default)*	Trigger for VCO phase rotation 0: NO, 1: YES
STATIC_VCO_PHASE_DIR	DELAYED *(default)*	VCO phase rotation direction
CLKOP_FPHASE	1 *(default)*	VCO phase A step control
CLKOS_FPHASE	1 *(default)*	VCO phase B step control
CLKOS2_FPHASE	1 *(default)*	VCO phase C step control
CLKOS3_FPHASE	1 *(default)*	VCO phase D step control
CLKOS4_FPHASE	1 *(default)*	VCO phase E step control
CLKOS5_FPHASE	1 *(default)*	VCO phase F step control
CLKPHY_FPHASE	1 *(default)*	VCO phase PHY step control
CLKOP_CPHASE	1 *(default)*	Delay A section output DELA VCO clock cycles with respect to VCO phase 0, or REFBUF if in VCO bypass mode (post-divider phase-shift)
CLKOS_CPHASE	1 *(default)*	Delay B section output DELB VCO clock cycles with respect to VCO phase 0, or REFBUF if in VCO bypass mode (post-divider phase-shift)
CLKOS2_CPHASE	1 *(default)*	Delay C section output DELC VCO clock cycles with respect to VCO phase 0, or REFBUF if in VCO bypass mode (post-divider phase-shift)
CLKOS3_CPHASE	1 *(default)*	Delay D section output DELD VCO clock cycles with respect to VCO phase 0, or REFBUF if in VCO bypass mode (post-divider phase-shift)
CLKOS4_CPHASE	1 *(default)*	Delay E section output DELE VCO clock cycles with respect to VCO phase 0, or REFBUF if in VCO bypass mode (post-divider phase-shift)
CLKOS5_CPHASE	1 *(default)*	Delay F section output DELF VCO clock cycles with respect to VCO phase 0, or REFBUF if in VCO bypass mode (post-divider phase-shift)
CLKOPHY_CPHASE	1 *(default)*	Delay F section output DELPHY VCO clock cycles with respect to VCO phase 0, or REFBUF if in VCO bypass mode (post-divider phase-shift)
FAST_LOCK	ENABLED *(default)*	Fast lock enable 0: DISABLED; 1: ENABLED. This attribute is set to default by GPLL IP setting script.
LOSS_LOCK_DETECTION	ENABLED *(default)* DISABLED	Enable cycle slip detection. This attribute is set to default by GPLL IP setting script. This is the HW fuse/bit setup. Set the bit to 0.
CLKI_DIV	8 *(default)*	Reference clock divider; NR=CLKR[5:0]+1, CLKR[0] is LSB
CLKI_SEL	REFMUX0 *(default)*	Static setting (Immi version) of refin_sel (Fabric) from REFMUX0 or REFMUX1
CLKFB_DIV	2 *(default)*	Feedback clock divider
FRACTIONAL_FBK	0b00000000000000 *(default)*	Precise and SSC fractional multiplication
CLKFB_PATH	EXTERNAL *(default)*	Feedback path 0: External feedback, 1: Internal feedback
EXT_FB_DELAY	0b00 *(default)*	Determine the output enable delayed time for external feedback mode
LOOP_BW	0b000000000000 *(default)*	Loop BW control: NB=BWADJ[11:0]+1, BWADJ[0] is LSB
CLKV_SSC_SLOPE	0b00000000000000000000000000 000 *(default)*	SSC slope multiplier
CLKS_SSC_RATE	0b000000000000 *(default)*	SSC rate multiplier
SCC_SS	DISABLED *(default)*	Enable or disable Spread Spectrum 0: DISABLED, 1: ENABLED
SCC_FRACTIONAL	DISABLED *(default)*	Enable fractional accumulation 0: DISABLED, 1: ENABLED

Name	Values	Description
CLKOP_DIV	8 *(default)*	CLKOP Divider Setting
CLKOS_DIV	8 *(default)*	CLKOS Divider Setting
CLKOS2_DIV	8 *(default)*	CLKOS2 Divider Setting
CLKOS3_DIV	8 *(default)*	CLKOS3 Divider Setting
CLKOS4_DIV	8 *(default)*	CLKOS4 Divider Setting
CLKOS5_DIV	8 *(default)*	CLKOS5 Divider Setting
CLKPHY_DIV	1 *(default)*	CLKPHY Divider Setting
INT_CLK7_DIV	8 *(default)*	CLK7 Divider Setting
SATURATION	ENABLED *(default)*	Enable saturation behavior 0: DISABLED, 1: ENABLED
EN_PLL	DISABLED *(default)*	Select GPLL IP 0: DISABLED, 1: ENABLED
EN_PLLRESET	DISABLED *(default)*	Enable pllreset control (Fabric port) 0: DISABLED, 1: ENABLED
EN_CLKOP	YES *(default)*	Enable CLKOP (DIVA) clock output 0: DISABLED (NO); 1: ENABLED (YES)
EN_CLKOS	YES *(default)*	Enable CLKOP (DIVA) clock output 0: DISABLED (NO); 1: ENABLED (YES)
EN_CLKOS2	YES *(default)*	Enable CLKOS2 (DIVC) clock output 0: DISABLED (NO); 1: ENABLED (YES)
EN_CLKOS3	YES *(default)*	Enable CLKOS3 (DIVD) clock output 0: DISABLED (NO); 1: ENABLED (YES)
EN_CLKOS4	YES *(default)*	Enable CLKOS4 (DIVE) clock output 0: DISABLED (NO); 1: ENABLED (YES)
EN_CLKOS5	YES *(default)*	Enable CLKOS5 (DIVF) 0: DISABLED (NO); 1: ENABLED (YES)
EN_CLKPHY	YES *(default)*	Enable CLKPHY (DIVPHY) 0: DISABLED (NO); 1: ENABLED (YES)
EN_CLKOP_OUT	OFF *(default)*	Output CLKOP 0: ON, 1: OFF
EN_CLKOS_OUT	OFF *(default)*	Output CLKOS 0: ON, 1: OFF
EN_CLKOS2_OUT	OFF *(default)*	Output CLKOS2 0: ON, 1: OFF
EN_CLKOS3_OUT	OFF *(default)*	Output CLKOS3 0: ON, 1: OFF
EN_CLKOS4_OUT	OFF *(default)*	Output CLKOS4 0: ON, 1: OFF
EN_CLKOS5_OUT	OFF *(default)*	Output CLKOS5 0: ON, 1: OFF
EN_CLKPHY_OUT	OFF *(default)*	Output CLKPHY 0: ON, 1: OFF

3.3. PLLREFCSA Component Definition

The PLLREFCSA is a 2:1 multiplexer that can dynamically select between using REFCLK0 or REFCLK1 as the CLKI input to the PLL Component. An external reset is required after switching the CLKI input.

Table 3.5. PLLREFCSA Component Port Definition

Signal	Type	Description
CLK0	Input	Input CLK0.
CLK1	Input	Input CLK1.
SEL	Input	Multiplexer Select between CLK0 (SEL = 0) and CLK1 (SEL = 1).
PLLCSOUT	Output	Clock Output (feed into CLKI of the PLL component).

Figure 3.3. PLLREFCSA Component Instance

3.4. Functional Description

3.4.1. Refclk (CLKI) Divider

The CLKI divider is used to control the input clock frequency into the phase detector. The valid PLL input frequency range is specified in the device data sheet.

3.4.2. Feedback Loop (CLKFB) Divider

The CLKFB divider is used to divide the feedback signal, effectively multiplying the output clock. The VCO block increases the output frequency until the divided feedback frequency equals the input frequency. The output of the feedback divider must be within the phase detector frequency range specified in the device data sheet. This port is only available for your configuration when CLKOUT* is selected for feedback clock. Otherwise, this port is connected by the tool to the appropriate signal you selected in the software.

3.4.3. Output Clock Dividers (CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5)

The output Clock Dividers allow the VCO frequency to be scaled up to the maximum range to minimize jitter. Each of the output dividers is independent of the other dividers and each uses the VCO as the source by default. Each of the output dividers can be set to a value of 1 to 256.

3.4.4. Phase Adjustment (Static Mode)

The CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, and CLKOS5 outputs can be phase adjusted relative to the enabled unshifted output clock at 45° increments. The clock output selected as the feedback cannot use the static phase adjustment feature as the feedback causes the PLL to unlock.

3.4.5. Phase Adjustment (Dynamic Mode)

The phase adjustments can also be controlled in a dynamic mode using the PHASESEL, PHASEDIR, PHASESTEP, and PHASELOADREG ports.

See the Dynamic Phase Adjustment section for usage details. The clock output selected as the feedback cannot use the dynamic phase adjustment feature as the feedback causes the PLL to unlock. Similar restrictions apply to other clocks.

3.5. PLL Inputs and Outputs

3.5.1. CLKI Input

The CLKI signal is the reference clock for the PLL. The signal must conform to the specifications in the data sheet for the PLL to operate correctly. The CLKI signal can come from a dedicated PLL input pin or from internal routing. The dedicated dual-purpose I/O pin provides a low skew input path and is the recommended source for the PLL. The reference clock can be divided by the input (M) divider to create one input to the phase detector of the PLL.

3.5.2. CLKFB Input

The CLKFB signal is the feedback signal to the PLL. The feedback signal is used by the Phase Detector inside the PLL to determine if the output clock needs adjustment to maintain the correct frequency and phase. The CLKFB signal can come from PLL clock outputs (CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, and CLKOS5) or internal VCO. Feedback clock is internal

VCO if fractional-N or spread spectrum clock is enabled. The feedback clock signal is divided by the feedback (N) divider to create an input to the phase detector of the PLL. A bypassed PLL output cannot be used as the feedback signal.

3.5.3. RST Input

At power-up, an internal power-up reset signal from the configuration block resets the PLL. Additionally, an active high, asynchronous, user-controlled reset port can be optionally added to the PLL. The RST signal can be driven by an internally generated reset function or by an I/O pin. This RST signal resets the PLL core (VCO, phase detector, and charge pump) and the output dividers, which causes the outputs to be logic *0*. In bypass mode, the output does not reset.

After the RST signal is deasserted, the PLL starts the lock-in process and takes tLOCK time, about 500 div. reference cycles (maximum), to complete. The RST signal is active high. The RST signal is optional. Refer to the *sysCLOCK PLL Timing* section in the Lattice Nexus 2 Platform – Specifications Data Sheet (FPGA-DS-02121) for Trst pulse width.

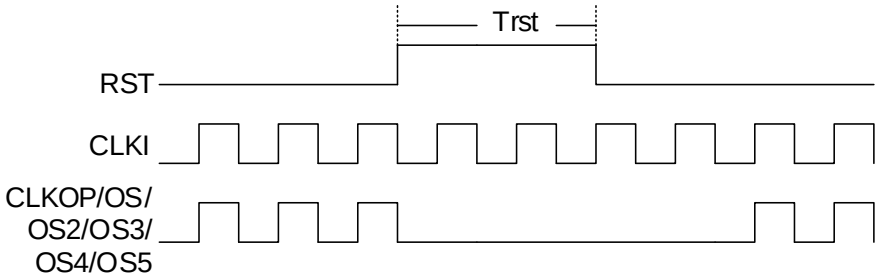


Figure 3.4. RST Input Timing Diagram

3.5.4. Dynamic Clock Enables

Each PLL output has a user input signal to dynamically enable and disable the output clock glitchlessly. This clock enable signal is ENCLKO*.

Table 3.6. PLL Clock Output Enable Signal List

Clock Enable Signal Name	Corresponding PLL Output	IP Block Wizard Option Name
ENCLKOP	CLKOP	CLKOUT 0 Enable Port
ENCLKOS	CLKOS	CLKOUT 1 Enable Port
ENCLKOS2	CLKOS2	CLKOUT 2 Enable Port
ENCLKOS3	CLKOS3	CLKOUT 3 Enable Port
ENCLKOS4	CLKOS4	CLKOUT 4 Enable Port
ENCLKOS5	CLKOS5	CLKOUT 5 Enable Port
ENCLKOPHY	CLKOPHY	CLKOUT 6 Enable Port

This dynamic block enable function allows you to save power by stopping the corresponding output clock when not in use. When the clock enable signal is set to logic *0*, the corresponding output clock is held to logic *0*. When the clock enable signal is set to logic 1, the output clock becomes active. The clock enable signals are optional and are only available if the corresponding option is enabled in the IP Block Wizard. If a clock enable signal is not requested, the corresponding output is active at all time when the PLL is instantiated unless the PLL is placed into standby mode. The clock enable signals cannot be accessed in the IP Block Wizard when the PLL output is used for external feedback to avoid shutting off the feedback clock.

3.5.5. Dynamic Phase Shift Inputs

The Lattice Nexus 2 PLL has five ports to allow for dynamic phase adjustment from FPGA logic. The Dynamic Phase Adjustment section elaborates on how you can drive these ports.

3.5.6. PHASESEL Input

The PHASESEL[2:0] inputs are used to specify which PLL output port is affected by the dynamic phase adjustment ports. The settings available are shown in the Dynamic Phase Adjustment section. The PHASESEL signal must be stable for Tphasesel_setup (refer to the *sysCLOCK PLL Timing* section in the Lattice Nexus 2 Platform – Specifications Data Sheet (FPGA-DS-02121)) before the PHASESTEP signal is pulsed. The PHASESEL signal is optional and is available if the *Enable Dynamic Phase Ports* option is selected in the IP Block Wizard.

Table 3.7. PHASESEL Signal Settings Definition

PHASESEL[2:0]	PLL Output Shifted
000	CLKOP
001	CLKOS
010	CLKOS2
011	CLKOS3
100	CLKOS4
101	CLKOS5

3.5.7. PHASEDIR Input

The PHASEDIR input is used to specify which direction the dynamic phase shift occurs, advanced (leading) or delayed (lagging). When PHASEDIR = 0, then the phase shift is delayed. When PHASEDIR = 1, then the phase shift is advanced. The PHASEDIR signal must be stable for Tphasedir_setup (refer to the *sysCLOCK PLL Timing* section in the Lattice Nexus 2 Platform – Specifications Data Sheet (FPGA-DS-02121)) before the PHASESTEP signal is pulsed. The PHASEDIR signal is optional and is available if the *Enable Dynamic Phase Ports* option is selected in the IP Block Wizard.

Table 3.8. PHASEDIR Signal Settings Definition

PHASEDIR	Direction
0	Delayed (lagging)
1	Advanced (leading)

3.5.8. PHASESTEP Input

The PHASESTEP signal is used to initiate a VCO dynamic phase shift for the clock output port and in the direction specified by the PHASESEL and PHASEDIR inputs. This phase adjustment is done by changing the phase of the VCO in 45° increments. The VCO phase changes on the negative edge of the PHASESTEP input after four VCO cycles. This is an active low signal and the minimum pulse width (both high and low) of PHASESTEP pulse is Tphasestep_pulse (refer to the *sysCLOCK PLL Timing* section in the Lattice Nexus 2 Platform – Specifications Data Sheet (FPGA-DS-02121)) cycles. The PHASESTEP signal is optional and is available if the *Enable Dynamic Phase Ports* option is selected in the IP Block Wizard.

3.5.9. PHASELOADREG Input

The PHASELOADREG signal is used to initiate a post-divider dynamic phase shift. A post-divider dynamic phase shift is started on the falling edge of the PHASELOADREG signal and there is a minimum pulse width of Tphaseloadreg_pulse (refer to the *sysCLOCK PLL Timing* section in the Lattice Nexus 2 Platform – Specifications Data Sheet (FPGA-DS-02121)) from assertion to desertion. The PHASELOADREG signal is optional and is available if the *Enable Dynamic Phase Ports* option is selected in the IP Block Wizard.

3.5.10. PLL Clock Outputs

The PLL has six outputs. All six outputs can be routed to the Global Clock routing of the FPGA. All six outputs can be phase shifted statically or dynamically if external feedback on the clock is not used. The outputs can come from their output divider or the reference clock input (PLL bypass). In PLL bypass mode, the output divider can be bypassed or used to divide the reference clock.

Table 3.9. PLL Clock Outputs and ECLK Connectivity

Clock Output Name	Edge Clock Connectivity	Selectable Output
CLKOP	ECLK Connection	Always Enabled
CLKOS	ECLK Connection	Selectable through IP Block Wizard
CLKOS2	No ECLK Connection	Selectable through IP Block Wizard
CLKOS3	No ECLK Connection	Selectable through IP Block Wizard
CLKOS4	No ECLK Connection	Selectable through IP Block Wizard
CLKOS5	No ECLK Connection	Selectable through IP Block Wizard

3.5.11. LOCK Output

The LOCK output provides information about the status of the PLL. When lock is achieved, the PLL LOCK signal is asserted. During the locked condition, the positive edges of the PLL feedback and reference clock are phase aligned in normal operation. The LOCK signal can be set in the IP Block Wizard by selecting the Provide PLL Lock Signal option. There is also an Enable Fast Lock option in the IP Block Wizard to set the behavior of PLL lock signal as well. When this option is selected, fast lock is enabled.

3.5.12. Dynamic Phase Adjustment

Dynamic phase adjustment of the PLL output clocks can be done without reconfiguring the FPGA by using the dedicated dynamic phase-shift ports of the PLL.

All six output clocks, CLKOP, CLKOS, CLKOS2, CLKOS3, CLKOS4, and CLKOS5 have the dynamic phase adjustment feature but only one output clock can be adjusted at a time. Table 3.7 shows the output clock selection settings available for the PHASESEL[2:0] signal. The PHASESEL signal must be stable for Tphasesel_setup before the PHASESTEP signal is pulsed.

The selected output clock phase is either advanced or delayed depending upon the value of the PHASEDIR port. Table 3.8 shows the PHASEDIR settings available. The PHASEDIR signal must be stable for Tphasedir_setup before the PHASESTEP signal is pulsed.

The PLL provides the static and dynamic output phase shifting through VCO phase shift and divider phase shift. The PLL parameters allow you to set VCO phases and divider phase shifts operation. The VCO phase shift provides 8 VCO phases with 45 degree per rotation or step. The divider phase shift provides the output delay from 1 to 256. Dynamic VCO and divider phase shifts are controlled by user signals.

3.5.13. VCO Phase Shift

When the PHASESEL and PHASEDIR have been set, a VCO phase adjustment is made by toggling the PHASESTEP signal from the current setting. Each pulse of the PHASESTEP signal generates a phase step based on this equation:

```
VCO Shifted Phase per step = [1 / (8 × (DIVO<n>_ACTUAL_STR + 1))] × 360°
```

Where <n> is the clock output specified by PHASESEL (CLKOP/OS/OS2/OS3/OS4/OS5). Values for DIVO<n>_ACTUAL_STR are located in the HDL source file generated by the IP Block Wizard.

The PHASESTEP signal is latched in on the rising edge. One step size is the smallest phase shift that can be generated by the PLL in one pulse. The dynamic phase adjustment results in a glitch free adjustment when delaying the output clock, but glitches may result when advancing the output clock.

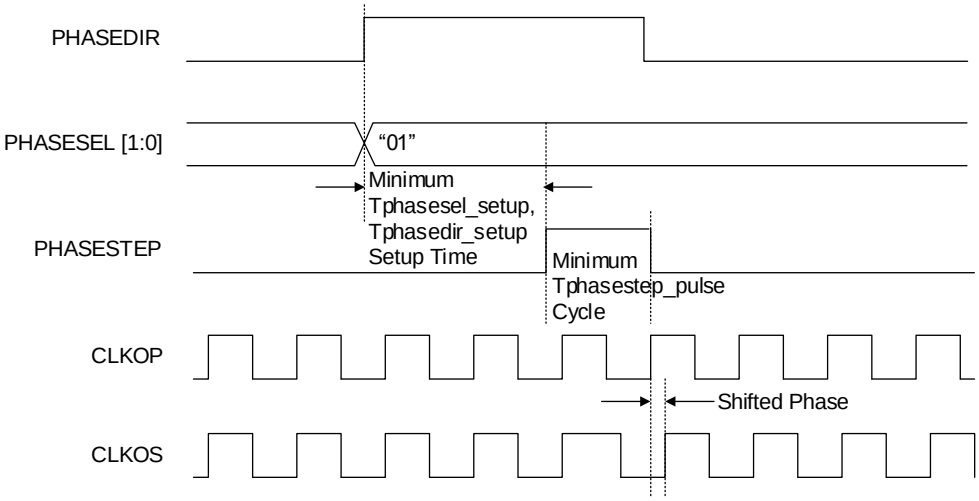


Figure 3.5. PLL Phase Shifting Using the PHASESTEP Signal

For example:

PHASESEL[2:0]=3'b001 to select CLKOS for phase shift

PHASEDIR =1'b0 for selecting delayed (lagging) phase

Assume the output is divided by 2, DIVOS_ACTUAL_STR = 1

The above signals need to be stable for Tphasesel_setup and Tphasedir_setup before the rising edge of PHASESTEP and the minimum pulse width of PHASESTEP must be one VCO clock cycle.

For each toggling of PHASESTEP, there is [1/(8×2)]×360 = 22.5° phase shift (delayed).

3.5.14. Divider Phase Shift

A post-divider phase adjustment is made by toggling the PHASELOADREG signal. You can calculate the desired post-divider phase using the following equation:

```
Post-Divider Phase = [(DEL<n> – DIVO<n>_ACTUAL_STR) / (DIVO<n>_ACTUAL_STR + 1)] × 360°
```

Where <n> is the clock output (CLKOP/OS/OS2/OS3/OS4/OS5). Values for DEL<n> and DIVO<n>_ACTUAL_STR are located in the HDL source file generated by IP Block Wizard. Note that if these values are both *1*, no shift is made. See the Example 2: Post-Divider Phase Shift section for usage details.

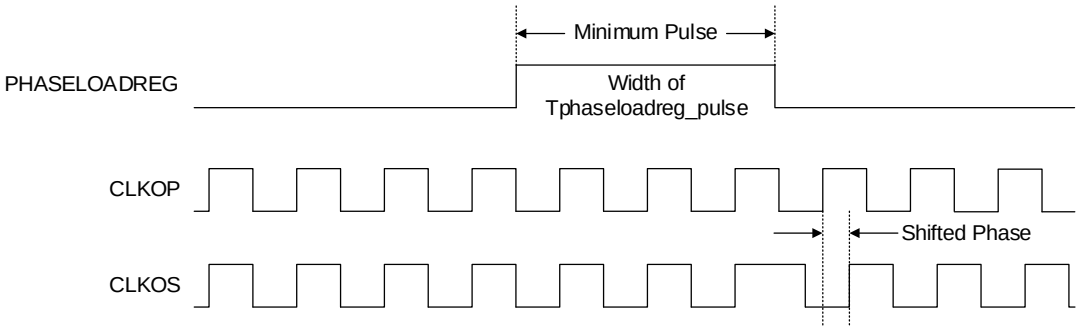


Figure 3.6. Divider Phase Shift Timing Diagram

3.5.15. Total Phase Shift

For the total phase shift calculation for each PLL output section:

```
Total Phase Shift = VCO Phase <n> + Post Divider Phase <n>
Post divider Phase <n> = 360 × (DEL<n> – DIV<n>)/(DIV<n> + 1)
VCO Phase <n> = 45 × ((PHI<n> + m – n)/(DIV<n> + 1)); where PHI is the initial Phase shift; m
is the number of toggles when dir=0; n is the number of toggles when dir=1;
```

3.6. Fractional-N Synthesis Operation

The GPLL is designed to multiply an input clock signal by a value between 2 and 4095 with 14-bit of fractional resolution (precise fractional N). The GPLL does not provide any deskew functionality as the GPLL uses internal feedback clock. The GPLL contains a 1–64 divider at the reference clock input, a 1–4096 divider in the internal feedback path, and a 1-256 divider at the output.

The output frequency is given by the equation:

$$F_{out} = \frac{F_{CLKI}}{\mathrm{M} \times \mathrm{O}} \times (\mathrm{N} + \frac{\mathrm{F}}{16384})$$

Where:

F_{out} is the output *Frequency Actual Value*.

F_{CLKI} is the CLKI input frequency.

M is the CLKI *Divider Desired Value*.

N is the CLKFB *FBK Divider Desired Value (Integer)*.

F is the CLKFB *FBK Divider Desired Value (Fractional)*.

O is the output *Divider Actual Value.*

3.7. Spread Spectrum Clock Generation

The Lattice Nexus 2 PLL supports Spread spectrum clock generation. The spread spectrum function is integrated with the Fractional-N controls and supports *Down Spread*, triangle wave, from 0% to 10% with modulation frequency range from 15 kHz to 4 MHz. When enabled, spread spectrum characteristics is applied to all active PLL outputs.

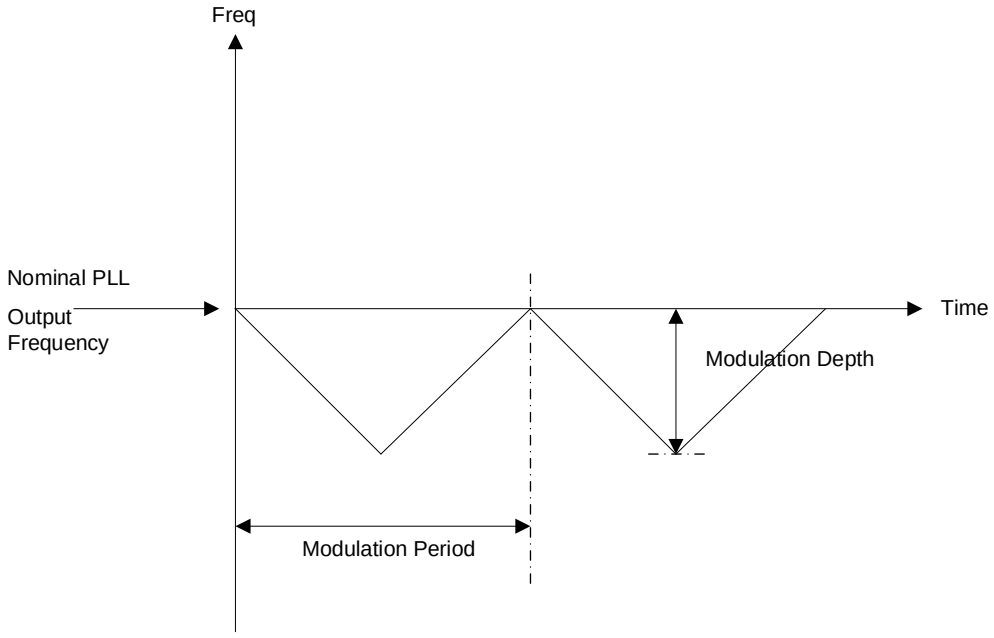


Figure 3.7. Down Spread Profile

3.8. Low Power Features

The Lattice Nexus 2 PLL contains several features that allows you to reduce the power usage of a design. When unused, you can drive the Dynamic Clock Enable feature.

3.8.1. Dynamic Clock Enable

The Dynamic Clock Enable feature supports a glitchless enable and disable of selected output clocks during periods when not used in the design. A disabled output clock is logic *0*. Re-enabled clocks start on the falling edge of the associated clocks. To support this feature, each output clock has an independent Output Enable signal that can be selected. The Output Enable signals are ENCLKOP, ENCLKOS, ENCLKOS2, ENCLKOS3, ENCLKOS4, ENCLKOS5, and ENPHYCLK. Each clock enable port has an option in the IP Block Wizard to bring the signal to the top-level ports of the PLL. If external feedback is used on a port or if the clock output is not enabled, dynamic clock enable port is unavailable.

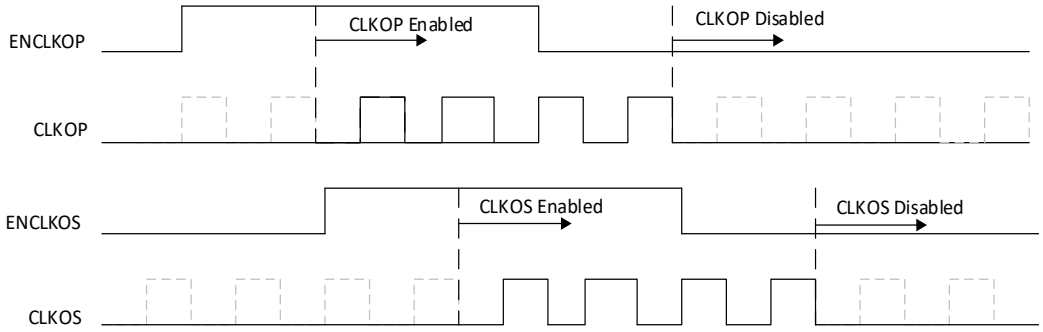


Figure 3.8. Dynamic Clock Enable for PLL Outputs

4. PLL LMMI Operation

The Nexus 2 PLL operating parameters can be changed dynamically through the LMMI bus or ABP bus. This section uses LMMI nomenclature. All addresses and bit definitions in PLL Architecture and LMMI Register Map sections are used identically for APB interface applications. A hard-wired LMMI bus is used to communicate between the LMMI host and the PLL. See Lattice Memory Mapped Interface (LMMI) and Lattice Interrupt Interface (LINTR) User Guide (FPGA-UG-02039) for more information about the LMMI bus.

The LMMI bus on the PLL module provides support for functional operation and simulation. You must connect the LMMI bus to the LMMI host in their HDL design to make the operand and simulation working properly. The LMMI bus ports and the corresponding LMMI connections are listed in the following table.

Table 4.1. PLL Data Bus Port Definition

PLL Port Name	Type	Description
lmmi_clk_i	Input	LMMI clock.
lmmi_resetn_i	Input	LMMI reset signal (active low). Only reset the bus, not register value.
lmmi_offset_i[4:0]	Input	LMMI offset address.
lmmi_wr_rdn_i	Input	LMMI WR/RD signal (write-high/read-low).
lmmi_request_i	Input	LMMI request signal.
lmmi_wdata_i[15:0]	Input	LMMI write data.
lmmi_ready_o	Output	LMMI ready signal.
lmmi_rdata_o[15:0]	Output	LMMI read data.
lmmi_rdata_valid	Output	LMMI read data valid signal.

4.1. PLL Architecture

The Nexus 2 PLL has six output sections with flexible configuration settings to support a variety of applications. IP Catalog can support most of the common PLL configurations. For more advanced support options, you can change the PLL configurations using the LMMI bus.

Each of the six PLL output sections have similar configuration options. Each output section is assigned a letter designator as follows:

- A for the CLKOP output
- B for the CLKOS output
- C for the CLKOS2 output
- D for the CLKOS3 output
- E for the CLKOS4 output
- F for the CLKOS5 output section

The LMMI addressable PLL registers are defined in Table 4.2.

4.2. LMMI Register Map

Table 4.2. LMMI Offset Address Locations for PLL Registers

Addr (Hex)	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	lmmi_reserved1[3:0}				lmmi_sel1	lmmi_refsel	lmmi_sel_fbk[3:0]				lmmi_sel_ref2[2:0]			lmmi_sel_ref2[2:0]		
1	lmmi_clkf[9:0]										lmmi_clkr[5:0]					
2	lmmi_clkf[25:10]															
3	lmmi_clkv[3:0]				lmmi_bwadj[11:0]											
4	lmmi_clkv[19:4]															
5	lmmi_clks[9:0]										lmmi_clkv[25:20]					

Addr (Hex)	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
6	lmmi_clkod1[0]	lmmi_clkod0[10:0]											lmmi_dithen	lmmi_ssen	lmmi_clks[11:10]	
7	lmmi_clkod2[5:0]						lmmi_clkod1[10:1]									
8	lmmi_clkod3[10:0]											lmmi_clkod2[10:6]				
9	lmmi_clkod5[4:0]					lmmi_clkod4[10:0]										
A	lmmi_clkod6[9:0]										lmmi_clkod5[10:5]					
B	lmmi_phia[0]	lmmi_test	lmmi_ensat	lmmi_fasten	lmmi_clkod7[10:0]											lmmi_clkod6[10]
C	lmmi_phif[1:0]		lmmi_phie[2:0]			lmmi_phid[2:0]			lmmi_phic[2:0]			lmmi_phib[2:0]			lmmi_phia[2:1]	
D	lmmi_reserved2[0]	lmmi_bypass[7:0]								lmmi_phiclk7[2:0]			lmmi_phiphy[2:0]			lmmi_phif[2]
E	lmmi_counts[2:0]			lmmi_loss_lock	lmmi_cycle_num[1:0]		lmmi_extfbk_delay[1:0]		lmmi_reserved2[8:1]							
F	lmmi_enable_clkphy[1:0]	lmmi_enable_clk[5:0						lmmi_dyn_source	lmmi_load_reg	lmmi_direction	lmmi_rotate	lmmi_dyn_sel[2:0]			lmmi_pllreset_en	lmmi_pll_en
10	Reserved3[5:0]						lmmi_intfb	lmmi_clkos7_off	lmmi_clkphy_off	lmmi_clkos5_off	lmmi_clkos4_off	lmmi_clkos3_off	lmmi_clkos2_off	lmmi_clkos_off	lmmi_clkop_off	lmmi_enable_clk7
11	lmmi_dela[7:0]								lmmi_diva[7:0]							
12	lmmi_delb[7:0]								lmmi_divb[7:0]							
13	lmmi_delc[7:0]								lmmi_divc[7:0]							
14	lmmi_deld[7:0]								lmmi_divd[7:0]							
15	lmmi_dele[7:0]								lmmi_dive[7:0]							
16	lmmi_delf[7:0]								lmmi_divf[7:0]							
17	lmmi_delphy[7:0]								lmmi_divphy[7:0]							
18	lmmi_delclk7[7:0]								lmmi_divclk7[7:0]							
19	lmmi_reserved4]	lmmi_mfgout2[2:0]			lmmi_mfgout1[2:0]			lmmi_enable_sync	lmmi_bypassclk7	lmmi_bypassphy	lmmi_bypassf	lmmi_bypasse	lmmi_bypassd	lmmi_bypassc	lmmi_bypassb	lmmi_bypassa

Table 4.3. PLL Registers Descriptions[1, 2, 3]

Register Name	Register Addr (HEX)	Size (Bit)	Description	User Access
lmmi_sel_ref1[2:0]	00[2:0]	3	Select refclk1 clock set	R/W
lmmi_sel_ref2[2:0]	00[5:3]	3	Select refclk2 clock set	R/W
lmmi_sel_fbk[3:0]	00[9:6]	4	Select feedback clocks	R/W
lmmi_refsel	00[10]	1	Determine use Fabric or LMMI bit control	R/W
lmmi_sel1	00[11]	1	lmmi version of refin_sel (Fabric)	R/W
lmmi_reserved1[3:0]	00[15:12]	4	LMMI reserved bits	RO
lmmi_clkr[5:0]	01[5:0]	6	reference clock divider; NR=CLKR[5:0] + 1, CLKR[0] is LSB	R/W
lmmi_clkf[25:0]	01[15:6] 02[15:0]	26	feedbck clock divider; NF=CLKF[25:0]/16384, CLKF[0] is LSB	R/W
lmmi_bwadj[11:0]	03[11:0]	12	Loop BW adj.: NB=BWADJ[11:0]+1,BWADJ[0] is LSB	R/W

Register Name	Register Addr (HEX)	Size (Bit)	Description	User Access
lmmi_clkv[25:0]	03[15:12] 04[15:0] 05[5:0]	26	Spreading slope: NV=CLKV[25:0]/16384. CLKV[0] is LSB	R/W
lmmi_clks[11:0]	0515:6] 06[1:0]	12	Spreading freq.: NS=CLKS[11:0], CLKS[0] is LSB	R/W
lmmi_ssen	06[2]	1	Active-high spread spectrum enabled	R/W
lmmi_dithen	06[3]	1	Enable fractional accumulation when high	R/W
lmmi_clkod0[10:0]	06[14:4]	11	3rd party IP output divider0: OD[0]=CLKOD[0][10:0] + 1; CLKOD[0][0] is LSB	R/W
lmmi_clkod1[10:0]	06[15] 07[9:0]	11	3th party IP output divider0: OD[1]=CLKOD[1][10:0] + 1; CLKOD[1][0] is LSB	R/W
lmmi_clkod2[10:0]	07[15:10] 08[4:0]	11	3th party IP output divider0: OD[2]=CLKOD[2][10:0] + 1; CLKOD[2][0] is LSB	R/W
lmmi_clkod3[10:0]	08[15:5]	11	3th party IP output divider0: OD[3]=CLKOD[3][10:0] + 1; CLKOD[3][0] is LSB	R/W
lmmi_clkod4[10:0]	09[10:0]	11	3th party IP output divider0: OD[4]=CLKOD[4][10:0] + 1; CLKOD[4][0] is LSB	R/W
lmmi_clkod5[10:0]	09[15:11] 0A[5:0]	11	3th party IP output divider0: OD[5]=CLKOD[5][10:0] + 1; CLKOD[5][0] is LSB	R/W
lmmi_clkod6[10:0]	0A[15:6] 0B[0]	11	3th party IP output divider0: OD[6]=CLKOD[6][10:0] + 1; CLKOD[6][0] is LSB	R/W
lmmi_clkod7[10:0]	0B[11:1]	11	3th party IP output divider0: OD[7]=CLKOD[7][10:0] + 1; CLKOD[7][0] is LSB	R/W
lmmi_fasten	0B[12]	1	Enable fast locking circuit (default 1'b0)	RO
lmmi_ensat	0B[13]	1	Enable saturation behavior (normal high; default 1'b1)	R/W
lmmi_test	0B[14]	1	Reference-to-counters-to_output bypass when high (Test mode)	RO
lmmi_phia[2:0]	0B[15] 0C[1:0]	3	VCO phase A step control	R/W
lmmi_phib[2:0]	0C[4:2]	3	VCO phase B step control	R/W
lmmi_phic[2:0]	0C[7:5]	3	VCO phase C step control	R/W
lmmi_phid[2:0]	0C[10:8]	3	VCO phase D step control	R/W
lmmi_phie[2:0]	0C[13:11]	3	VCO phase E step control	R/W
lmmi_phif[2:0]	0C[15:14] 0D[0]	3	VCO phase F step control	R/W
lmmi_phiphy[2:0]	0D[3:1]	3	VCO phase PHY step control	R/W
lmmi_phiclk7[2:0]	0D[6:4]	3	VCO phase PHY step control *Test feature	R/W
lmmi_bypass[7:0]	0D[14:7]	8	1'b1 to bypass TCI IP clock	R/W
lmmi_reserved2[8:0]	0D[15] 0E[7:0]	9	LMMI reserved bits	RO
lmmi_extfbk_delay[1:0]	0E[9:8]	2	Determine the output enable delayed time for external feedback mode.	R/W
lmmi_cycle_num[1:0]	0E[11:10]	2	Determine the occurrence of cycle slips	R/W
lmmi_loss_lock	0E[12]	1	1'b1 enable cycle slip detection.	RO
lmmi_counts[2:0]	0E[15:13]	3	Internal reset pulse duration selection.	R/W
lmmi_pllen	0F[0]	1	1'b1 to select GPLL IP	R/W
lmmi_pllreset_en	0F[1]	1	1'b1 to enable pllreset control	R/W
lmmi_dyn_sel[2:0]	0F[4:2]	3	To select which output clocks is phase shifted	R/W
lmmi_rotate	0F[5]	1	Trigger for VCO phase rotation,	R/W
lmmi_direction	0F[6]	1	VCO phase rotation direction, 1'b0 for phase lagging, 1'b1 for leading	R/W
lmmi_load_reg	0F[7]	1	To initiate output phase shift on the falling edge of lmmi_load_reg	R/W
lmmi_dyn_source	0F[8]	1	1'b1 select Fabric signals for dynamic phase shift	R/W

Register Name	Register Addr (HEX)	Size (Bit)	Description	User Access
lmmi_enable_clk[5:0]	0F[14:9]	6	1'b1 of [5:0] to enable clk[os5, os4, os3, os2, os, op] respectively	R/W
lmmi_enable_clkphy	0F155]	1	1'b1 to enable clkphy	R/W
lmmi_enable_clk7	10[0]	1	1'b1 to enable clk7 *Test feature	RO
lmmi_clkop_off	10[1]	1	1'b0 to statically disable clkop output	R/W
lmmi_clkos_off	10[2]	1	1'b0 to statically disable clkos output	R/W
lmmi_clkos2_off	10[3]	1	1'b0 to statically disable clkos2 output	R/W
lmmi_clkos3_off	10[4]	1	1'b0 to statically disable clkos3 output	R/W
lmmi_clkos4_off	10[5]	1	1'b0 to statically disable clkos4 output	R/W
lmmi_clkos5_off	10[6]	1	1'b0 to statically disable clkos5 output	R/W
lmmi_clkphy_off	10[7]	1	1'b0 to statically disable clkphy output	R/W
lmmi_clk7_off	10[8]	1	1'b0 to statically disable clk7 output *Test feature	RO
lmmi_intfb	10[9]	1	1'b1 indicates the internal feedback is selected	R/W
lmmi_reserved3[5:0]	10[15:10]	6	LMMI reserved bits	RO
lmmi_diva[7:0]	11[7:0]	8	clkop divider; ND= lmmi_diva[7:0] +1; lmmi_diva[0] is LSB	R/W
lmmi_dela[7:0]	11[15:8]	8	clkop phase control	R/W
lmmi_divb[7:0]	12[7:0]	8	clkos divider; ND= lmmi_divb[7:0] +1; lmmi_divb[0] is LSB	R/W
lmmi_delb[7:0]	12[15:8]	8	clkos phase control	R/W
lmmi_divc[7:0]	13[7:0]	8	clkos2 divider; ND= lmmi_divc[7:0] +1; lmmi_divc[0] is LSB	R/W
lmmi_delc[7:0]	13[15:8]	8	clkos2 phase control	R/W
lmmi_divd[7:0]	14[7:0]	8	clkos3 divider; ND= lmmi_divd[7:0] +1; lmmi_divd[0] is LSB	R/W
lmmi_deld[7:0]	14[15:8]	8	clkos3 phase control	R/W
lmmi_dive[7:0]	15[7:0]	8	clkos4 divider; ND= lmmi_dive[7:0] +1; lmmi_dive[0] is LSB	R/W
lmmi_dele[7:0]	15[15:8]	8	clkos4 phase control	R/W
lmmi_divf[7:0]	16[7:0]	8	clkos5 divider; ND= lmmi_divf[7:0] +1; lmmi_divf[0] is LSB	R/W
lmmi_delf[7:0]	16[15:8]	8	clkos5 phase control	R/W
lmmi_divphy[7:0]	17[7:0]	8	clkphy divider (the divider selection is limited; 1~4)	R/W
lmmi_delphy[7:0]	17[15:8]	8	clkphy phase control (dummy, for implementation only, no phase control needed)	R/W
lmmi_divclk7[7:0]	18[7:0]	8	clkphy phase control *Test feature	RO
lmmi_delclk7[7:0]	18[15:8]	8	clkphy phase control *Test feature	RO
lmmi_bypassa	19[0]	1	clkop reference bypass; 1'b1 refclk -> clkop	R/W
lmmi_bypassb	19[1]	1	clkos reference bypass; 1'b1 refclk -> clkos	R/W
lmmi_bypassc	19[2]	1	clkos2 reference bypass; 1'b1 refclk -> clkos2	R/W
lmmi_bypassd	19[3]	1	clkos3 reference bypass; 1'b1 refclk -> clkos3	R/W
lmmi_bypasse	19[4]	1	clkos4 reference bypass; 1'b1 refclk -> clkos4	R/W
lmmi_bypassf	19[5]	1	clkos5 reference bypass; 1'b1 refclk -> clkos5	R/W
lmmi_bypassphy	19[6]	1	clkphy reference bypass; 1'b1 refclk -> clkphy	R/W
lmmi_bypassclk7	19[7]	1	clkphy reference bypass; 1'b1 refclk -> clk7 *Test feature	RO
lmmi_enable_sync	19[8]	1	1'b1 enable output(s) sync with clkop	R/W
lmmi_mfgout1[2:0]	19[11:9]	3	Selections for pll_mfgout1	RO
lmmi_mfgout2[2:0]	19[14:12]	3	Selections for pll_mfgout2	RO
lmmi_reserved4	19[15]	1	LMMI reserved bit	RO

Notes:

1. The dynamic programming through PLL LMMI operation is done in user mode. You can control the dynamic configuration without the assistance of the PLL module IP GUI.

2. There are many LMMI register bits that can affect the PLL close loop, that are reference clock selection, feedback clock selection, feedback mode selection (internal/external), reference clock divider, feedback clock divider, output divider, and bandwidth settings. Any change to these LMMI register bits can cause the PLL to lose lock and fail to produce the expected outputs.
3. It is recommended to perform the checks and optimizations using the PLL Module IP GUI before making any dynamic programming to the LMMI register bits.

4.3. Example 1: Dynamic Reconfigure Output Frequency

This example shows the reconfiguration of the PLL output frequency using PLL LMMI operation. The LMMI bus first reads the lmmi_divb[7:0] of clkos2 at offset address 0x12, then overwrites the default value 0x63 to new value 0xC7 to change the output frequency from 25 MHz to 12.5 MHz when the VCO frequency is 2,500 MHz.

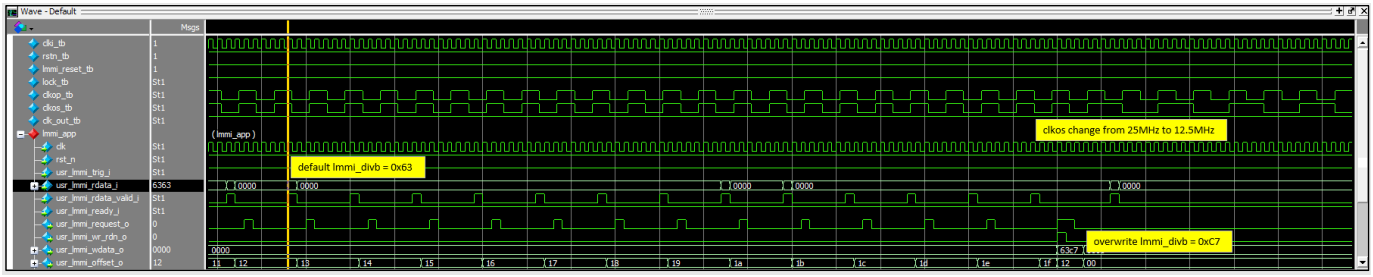


Figure 4.1. Example of PLL Output Frequency Reconfiguration Using PLL LMMI Operation

4.4. Example 2: Post-Divider Phase Shift

This example shows the implementation of the post-divider 90-degree phase shift (lagging) using PLL LMMI operation. The LMMI bus first reads the lmmi_delf[7:0] of clkos5 at offset address 0x16, then overwrites the default value 0x63 to new value 0x7C, and initiates a divider phase shift on the falling edge of phaseloadreg.

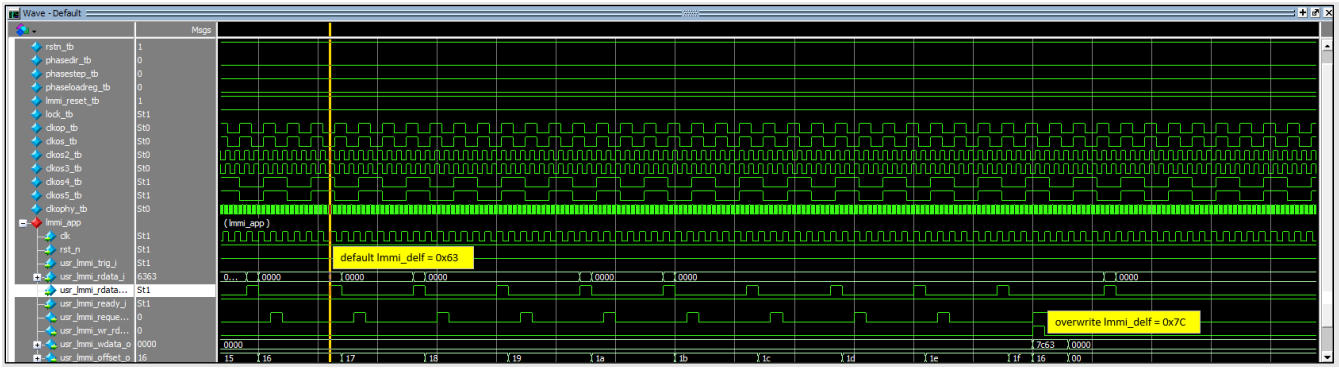


Figure 4.2. Example of Overwriting the Default Value to New Value

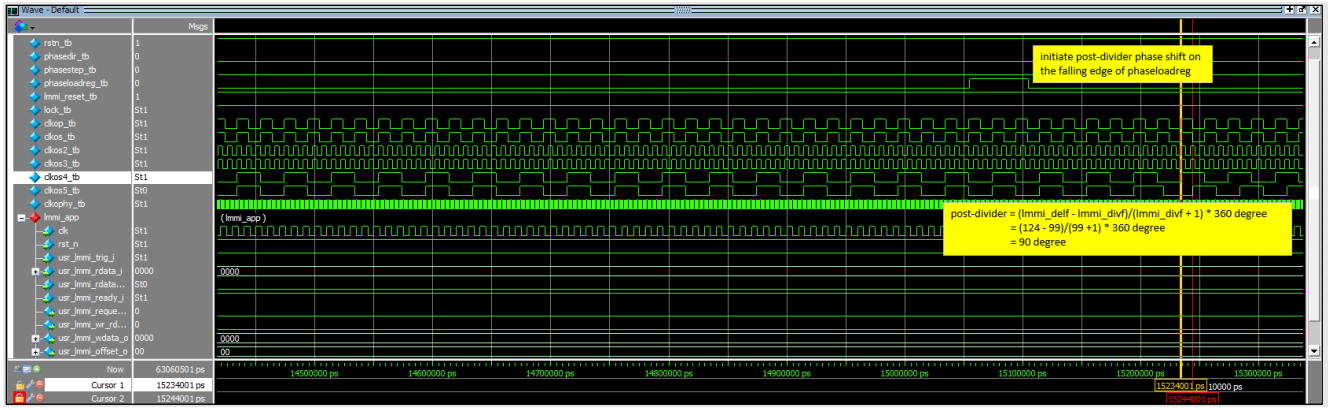


Figure 4.3. Example of Post Divider 90 Degree Phase Shift (Lagging) Implementation Using PLL LMMI Operation

Post-divider phase shift = (lmmi_delf – lmmi_divf) / (lmmi_divf + 1) × 360°

= (124 – 99) / (99 + 1) × 360°

= 90°

References

- Lattice Nexus 2 Platform – Overview Data Sheet (FPGA-DS-02122)
- Lattice Nexus 2 Platform – Specifications Data Sheet (FPGA-DS-02121)
- Lattice Nexus 2 Embedded Memory User Guide (FPGA-TN-02366)
- Lattice Nexus 2 sysCONFIG User Guide (FPGA-TN-02370)
- Lattice Nexus 2 sysDSP User Guide (FPGA-TN-02362)
- Lattice Nexus 2 sysI/O User Guide (FPGA-TN-02365)
- Sub-LVDS Signaling Using Lattice Devices (FPGA-TN-02028)
- Lattice Memory Mapped Interface (LMMI) and Lattice Interrupt Interface (LINTR) User Guide (FPGA-UG-02039)
- Certus-N2 web page
- Lattice Radiant Software web page
- 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/Support/AnswerDatabase.

Revision History

Revision 0.81, December 2024

Section	Change Summary
Introduction	Updated the number of dedicated clock pins for LN2-CT-16/20 from 24 to 28 in Table 1.1. Number of PLLs, Edge Clocks, and Clock Synchronizers and Dividers.

Lattice Nexus 2 Clocks	• Updated the maximum logic cells from 20k to 22k in the Lattice Nexus 2 Clocks section.
	• Updated the following figures:
	  • Figure 2.1. Clock Regions (LN2-CT-20 Devices)
	  • Figure 2.5. High-Level View of Clock Networks and Elements (LN2-CT-20 Devices)
	  • Figure 2.8. Multi-Region Clock Formations (LN2-CT-20 Devices)
	• Updated PCLKT3 [0:3], PCLKT8 [0:3], PCLKT10 [0:3], and PCLKT11 [0:3] for the LN2-CT-06/10 devices in the following tables:
	  • Table 2.1. Number of Dedicated Clock Pins for Lattice Nexus 2 Devices
	  • Table 2.2. Global Clock Sources
	  • Table 2.3. High-Performance I/O (HPIO) Regional Clock Sources
	  • Table 2.5. Edge Clock Sources
	• Updated Fabrick_CLK [0:11] and added a note in the following tables:
	  • Table 2.3. High-Performance I/O (HPIO) Regional Clock Sources
	  • Table 2.4. Wide-Range I/O (WRIO) Regional Clock Sources

sysCLOCK PLL	• Updated the PLL input resources in the PLL Input Sources section. • Removed the *CONFIG_WAIT_FOR_LOCK* attribute in Table 3.4. PLL Component Attribute.
References	Updated references.

Revision 0.80, August 2024

Section	Change Summary
All	Preliminary release.

www.latticesemi.com