UltraScale Architecture Clocking Resources

User Guide

UG572 (v1.10) August 28, 2020

XILINX

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
08/28/2020	1.10	Chapter 3: Updated Figure 3-23. Updated the table for register 15 in MMCM Registers.
10/31/2019	1.9	Chapter 3: Updated Table 3-4 footnote. Updated Spread-Spectrum Clock Generation section with new content and equations. Updated Table 3-12. Updated Ultrascale+ note on page 32.
12/19/2018	1.8	Chapter 3: Updated the example in Determine the Input Frequency section. Added the new sections Dynamic Reconfiguration Port and Clocking Guidelines.
04/09/2018	1.7	Chapter 2: Updated the BUFG_GT and BUFG_GT_SYNC section. Chapter 3: In Table 3-4, updated note 3.
06/06/2017	1.6	Chapter 3: In Table 3-4, updated the description of BUF_IN for the COMPENSATION attribute on page 55.
03/15/2017	1.5	Chapter 2: Updated the discussion on page 15. Added clarification to the BUFG_GT and BUFG_GT_SYNC section. Chapter 3: Updated the Dynamic Phase Shift Interface in the MMCM section. Added Table 3-6 and Table 3-8. In Table 3-12, updated the descriptions for CLKOUT[0:1]_PHASE and CLKFBOUT_PHASE.
12/12/2016	1.4	Chapter 1: Updated the discussion on page 9 about the differences between clock capable and global clock pins. Chapter 2: Added clarification to the Global Clock Inputs section. Added further information following Figure 2-3. Updated the BUFGCE_DIV section. Revised the BUFG_GT_SYNC description on page 33 to include the UltraScale+ devices. Chapter 3: Added the UltraScale+ device MMCME4 and PLLE4 primitives to the MMCM Primitives and PLL Primitives sections. Updated the description of PSCLK cycles in the Dynamic Phase Shift Interface in the MMCM section. Added a *Recommended* note on page page 51. Updated the CLKINSTOPPED – Input Clock Status section. Added CLKFBOUT and CLKFBIN to Table 3-11 and their descriptions below the table. Updated the CLKOUTPHYEN – PHY Clock Enable description. Added Figure 3-17 and Figure 3-18. In Table 3-12, updated the DIVCLK_DIVIDE allowed values and added PHY_ALIGN to the COMPENSATION attribute. Updated the Please Read: Important Legal Notices section.

Date	Version	Revision
11/24/2015	1.3	Under Introduction to UltraScale Architecture, page 5, added new introductory text for UltraScale+ devices. Added ninth bullet under Key Differences from 7 Series FPGAs, page 9. Updated first paragraph under Global Clock Inputs, page 10 to include information about HDGC pins. Updated first paragraph under Clock Structure, page 12. Added *Important* note under Clock Buffers, page 16. Added second paragraph under BUFCE_LEAF Clock Buffer, page 30. Added first two sentences under BUFG_GT and BUFG_GT_SYNC, page 32. Added BUFG_PS, page 34 section. Updated Frequency Synthesis Using Fractional Divide in the MMCM, page 40, by changing 0.125 degrees to 0.125. Revised the heading Static Phase Shift Mode (MMCM and PLL), page 41 by adding (MMCM and PLL). Revised the heading MMCM Clock Divide Dynamic Change, page 44 by adding MMCM. Added *Important* note under CLKFBIN – Feedback Clock Input, page 48. In Table 3-4, added a row of UltraScale+ device MMCM attributes for CLKFBOUT_MULT_F(1), page 53, changed default value for COMPENSATION, page 55 from ZHOLD to AUTO, revised the COMPENSATION Description, added note 4, and note 5. In Table 3-12, added a row of UltraScale+ device PLL attributes for CLKFBOUT_MULT, page 78, revised the COMPENSATION Description, and added note 1. Added Dynamic Reconfiguration Port, page 81 section. Updated References, page 102.
02/23/2015	1.2	In Table 3-4, changed the Allowed Values attribute for CLKIN1_PERIOD and CLKIN2_PERIOD, page 54. In Table 3-12, changed the Allowed Values attribute for CLKIN_PERIOD, page 79.
08/21/2014	1.1	Replaced clock-capable with global clock in Global Clock Inputs. Updated Byte Clock Inputs. Added BUFG_GT_SYNC to BUFG_GT and BUFG_GT_SYNC. Updated Figure 3-3 and added tip for Table 3-3. Updated Figure 3-11.
12/10/2013	1.0	Initial Xilinx release.

Table of Contents

Overview

Introduction to UltraScale Architecture

The Xilinx® UltraScale™ architecture is a revolutionary approach to creating programmable devices capable of addressing the massive I/O and memory bandwidth requirements of next-generation applications while efficiently routing and processing the data brought on chip. UltraScale architecture-based devices address a vast spectrum of high-bandwidth, high-utilization system requirements through industry-leading technical innovations. The devices share many building blocks to provide optimized scalability across the product range, as well as numerous new power reduction features for low total power consumption.

Kintex® UltraScale FPGAs provide high performance with a focus on optimized performance per watt for applications including wireless, wired, and signal or image processing. High digital signal processing and block RAM-to-logic ratios, and next generation transceivers are combined with low-cost packaging to enable an optimum blend of capability for these applications.

Kintex UltraScale+™ FPGAs deliver increased performance over the Kintex UltraScale family with on-chip UltraRAM memory to reduce BOM cost, providing the ideal mix of high-performance peripherals and cost-effective system implementation. In addition, Kintex UltraScale+ FPGAs have numerous power options that deliver the optimal balance between the required system performance and the smallest power envelope.

Virtex® UltraScale FPGAs provide the highest system capacity, bandwidth, and performance. Delivering unprecedented logic capacity, serial I/O bandwidth, and on-chip memory, the Virtex UltraScale family pushes the performance envelope ever higher.

Virtex UltraScale+ FPGAs have the highest transceiver bandwidth, highest DSP count, and highest on-chip UltraRAM memory available for the ultimate in system performance. In addition, Virtex UltraScale+ FPGAs also provide numerous power options that deliver the optimal balance between the required system performance and the smallest power envelope.

Zynq® UltraScale+ MPSoCs combine the Arm® v8-based Cortex®-A53 high-performance energy-efficient 64-bit application processor with the Arm Cortex-R5 real-time processor and the UltraScale architecture to create the industry's first All Programmable MPSoCs. With next-generation programmable engines, security, safety, reliability, and scalability from 32 to 64 bits, the Zynq UltraScale+ MPSoCs provide unprecedented power savings,

processing, programmable acceleration, I/O, and memory bandwidth ideal for applications that require heterogeneous processing.

This user guide describes the UltraScale architecture clocking resources and is part of the UltraScale architecture documentation suite available at: www.xilinx.com/documentation.

Clocking Overview

This chapter provides an overview of clocking and a comparison between clocking in the UltraScale architecture and previous FPGA generations. For detailed information on usage of clocking resources, see Chapter 2, Clocking Resources and Chapter 3, Clock Management Tile. For more information refer to the Clocking Guidelines section in the *UltraFast Design Methodology Guide for the Vivado Design Suite* (UG949) [Ref 1].

Clocking Architecture Overview

The UltraScale architecture clocking resources manage complex and simple clocking requirements with dedicated global clocks distributed on clock routing and clock distribution resources. The clock management tiles (CMTs) provide clock frequency synthesis, deskew, and jitter filtering functionality. Non-clock resources such as local routing are not recommended when designing for clock functions.

- The device is subdivided into columns and rows of segmented clock regions (CRs). CRs differ from previous families because they are arranged in tiles and do not span half the width of a device. A CR contains configurable logic blocks (CLBs), DSP slices, block RAMs, interconnect, and associated clocking. The height of a CR is 60 CLBs, 24 DSP slices, and 12 block RAMs with a horizontal clock spine (HCS) at its center. The HCS contains the horizontal routing and distribution resources, leaf clock buffers, clock network interconnections, and the root of the clock network. Clock buffers drive directly into the HCS. There are 52 I/Os per bank and four gigabit transceivers (GTs) that are pitch matched to the CRs. A core column contains configuration, System Monitor (SYSMON), and PCIe® blocks to complete a basic device.
- Adjacent to the input/output block columns are the physical layer (PHY) blocks with CMTs, global clock buffers, global clock multiplexing structures, and I/O logic management functions. The clocking drives vertical and horizontal connectivity through separate clock routing and clock distribution resources via HCS into the CRs and I/Os.
- Horizontal clock routing and distribution tracks drive horizontally into the CRs. Vertical routing and distribution tracks drive vertically adjacent CRs. The tracks are segmentable at the CR boundaries in both the horizontal and vertical directions. This allows for the creation of device-wide global clocks or local clocks of variable size.

- The distribution tracks drive the clocking of synchronous elements across the device. Distribution tracks are driven by routing tracks or directly by the clocking structures in the PHY.
- I/Os are directly driven from the PHY clocking and/or an adjacent PHY via routing tracks.
- A CMT contains one mixed-mode clock manager (MMCM) and two phase-locked loops (PLLs).

Clock Routing Resources Overview

Each I/O bank contains global clock input pins to bring user clocks onto the device clock management and routing resources. The global clock inputs bring user clocks onto:

- Clock buffers in the PHY adjacent to the same bank
- CMTs in the PHY adjacent to the same bank

Each device has three global clock buffers: BUFGCTRL, BUFGCE, and BUFGCE_DIV. In addition, there is a local BUFCE_LEAF clock buffer for driving leaf clocks from horizontal distribution to various blocks in the device. BUFGCTRL has derivative software representations of types BUFGMUX, BUFGMUX1, BUFGMUX_CTRL, and BUFGCE_1. BUFGCE is for glitchless clock gating and has software derivative BUFG (BUFGCE with clock enable tied High). The global clock buffers drive routing and distribution tracks into the device logic via HCS rows. There are 24 routing and 24 distribution tracks in each HCS row. There is also a BUFG_GT that generates divided clocks for GT clocking. The clock buffers:

- Can be used as a clock enable circuit to enable or disable clocks either globally, locally, or within a CR for fine-grained power control.
- Can be used as a glitch-free multiplexer to:
  - select between two clock sources.
  - switch away from a failed clock source.
- Are often driven by a CMT to:
  - eliminate the clock distribution delay.
  - adjust clock delay relative to another clock.

Chapter 2, Clocking Resources, has further details on global clocks, I/O, and GT clocking. It also describes which clock routing resources to utilize for various applications.

CMT Overview

Each device has a CMT as part of the PHY next to each of the I/O banks. A CMT consists of one MMCM and two PLLs. The MMCM is the primary block for frequency synthesis for a wide range of frequencies, and serves as a jitter filter for either external or internal clocks, and deskew clocks among a wide range of other functions. The PLL's primary purpose is to provide clocking to the PHY I/Os, but can also be used for clocking other resources in the device in a limited fashion. The device clock input connectivity allows multiple resources to provide the reference clock(s) to the MMCM and PLL.

MMCMs have infinite fine phase shift capability in either direction and can be used in dynamic phase shift mode. MMCMs also have a fractional counter in either the feedback path or in one output path, enabling further granularity of frequency synthesis capabilities.

The LogiCORE™ IP clocking wizard is available to assist in utilizing MMCMs and PLLs to create clock networks in UltraScale architecture designs. The GUI interface is used to collect clock network parameters. The clocking wizard chooses the appropriate CMT resource and optimally configures the CMT resource and associated clock routing resources.

Chapter 3, Clock Management Tile, has further details on the CMT block features and connectivity.

Clocking Differences from Previous FPGA Generations

UltraScale architecture-based devices have significant innovations in the clocking architecture. In general, there is a minimal difference between global and local clock buffers. Thus, the 7 series regional clock buffers have been replaced by new clock buffers with more global reach while automatically utilizing local clock buffers for local distribution of the clocks. The CMT block consists of one MMCM and two PLLs. The MMCM is very similar to the 7 series family while the PLL has new features for the I/O PHY clocking, but a reduced set of functionality and connectivity with respect to clocking the rest of the device.

Key Differences from 7 Series FPGAs

- BUFMRs, BUFRs, and BUFIOs, and the associated routing resources have been removed from this architecture and are replaced by new clock buffers, clock routing, and a completely new I/O clocking architecture.
- The BUFGCTRL and its derivatives are still available. Two new global clock buffer resources BUFGCE and BUFGCE_DIV have been introduced in the new architecture. At the local clocking level, a new BUFCE_LEAF clock buffer provides local, vertical clocking with additional features.
- A BUFG_GT buffer for clock division of GT clocks has been added.
- A new and improved clock routing architecture is available. There are now two types of global routing tracks called routing and distribution. Both types of routing provide a segmentable clock network at the CR level. Both types can be driven by the global clock buffers. The distribution tracks can be driven by routing tracks or directly by clock buffer resources. The distribution tracks provide connectivity to all clocking points in UltraScale devices.
- The CMTs now have two PLLs instead of one.
- MMCMs are similar to the MMCM in the 7 series devices. PLLs have new features related to I/O PHY clocking. However, other clocking related functionality and connectivity has been reduced as compared to the 7 series FPGAs. For example, the PLLs do not support phase compensation or external feedback, have fewer outputs, share a voltage-controlled oscillator (VCO) with the PHY clocking, and have other features removed as compared to the 7 series devices. For this reason, most customers should use the MMCM for general clocking. However, leftover PLLs are also available for use.
- The MMCM output clock frequencies can be dynamically changed without resetting the MMCM.
- The definition of clock region has changed. A clock region no longer spans half a device width in the horizontal direction. UltraScale architecture clock regions have a rectangular shape with a fixed width and height and are organized in tiles. Horizontal and vertical clock tracks are segmented at the clock region boundaries.
- The clock capable pins (CC) have been replaced by global clock pins (GC). In addition, the UltraScale+ architecture has high-density (HD) I/O banks. These banks contain four global clock pins called HDGC which can connect to the BUFGCEs.

Clocking Resources

Overview

UltraScale™ architecture-based devices have several clock routing resources to support various clocking schemes and requirements, including high fanout, short propagation delay, and extremely low skew. To best utilize the clock routing resources, the designer must understand how to get user clocks from the PCB to the UltraScale devices, decide which clock routing resources are optimal, and then access those clock routing resources by utilizing the appropriate I/O and clock buffers.

Global Clock Inputs

External global user clocks must be brought into the UltraScale device on differential clock pin pairs called global clock (GC) inputs. There are four GC pin pairs in each bank that have direct access to the global clock buffers, MMCMs, and PLLs that are in the CMT adjacent to the same I/O bank. The UltraScale+ architecture has four HDGC pins per HD I/O bank. HD I/O banks are only part of the UltraScale+ family. Since HD I/O banks do not have a XIPHY and CMT next to them, the HDGC pins can only directly drive BUFGCEs (BUFGs) and not MMCMs/PLLs. Therefore, clocks that are connected to an HDGC pin can only connect to MMCMs/PLLs through the BUFGCEs. To avoid a design rule check (DRC) error, set the property CLOCK_DEDICATED_ROUTE = FALSE. GC inputs provide dedicated, high-speed access to the internal global and regional clock resources. GC inputs use dedicated routing and must be used for clock inputs where the timing of various clocking features is imperative. General-purpose I/O with local interconnects should not be used for clock signals.

Each I/O bank is located in a single clock region and includes 52 I/O pins. Of the 52 I/O pins in each I/O bank in every I/O column, there are four global clock input pin pairs (a total of eight pins). Each global clock input:

- Can be connected to a differential or single-ended clock on the PCB.
- Can be configured for any I/O standard, including differential I/O standards.
- Has a P-side (master), and an N-side (slave).

Single-ended clock inputs must be assigned to the P (master) side of the GC input pin pair. If a single-ended clock is connected to the P-side of a differential clock pin pair, the N-side cannot be used as another single-ended clock pin—it can only be used as a user I/O. For pin naming conventions, refer to the *UltraScale Architecture Packaging and Pinout User Guide* (UG575) [Ref 2].

GC inputs can be used as regular I/O if not used as clocks. When used as regular I/O, global clock input pins can be configured as any single-ended or differential I/O standard. GC inputs can connect to the PHY adjacent to the banks they reside in.

Byte Clock Inputs

Byte-lane clock (DBC and QBC) input pin pairs are dedicated clock inputs directly driving source synchronous clocks to the bit slices in the I/O banks. In memory applications, these are also known as DQS. When not used for I/O byte clocking these pin have other functions such as general purpose I/Os. For more information, consult the *UltraScale Architecture SelectIO Resources User Guide* (UG571) [Ref 3].

Clock Buffers and Clock Routing

Global clocks are a dedicated network of interconnects specifically designed to reach all clock inputs to the various resources in a device. These networks are designed to have low skew and low duty cycle distortion, low power, and improved jitter tolerance. They are also designed to support very high-frequency signals.

Understanding the signal path for a global clock expands the understanding of the various global clocking resources. The global clocking resources and network consist of these paths and components:

Clock Structure

The basic device architecture is composed of blocks of CRs. CRs are organized into tiles and thus build columns and rows. Each CR contains slices (CLBs), DSPs, and 36K block RAM blocks. The mix of slice, DSP, and block RAM columns in each CR can be different, but are always identical when stacked in the vertical direction, thus building columns of those resources for the entire device. I/O and GT columns are then inserted with columns of CRs. In addition, there is a single column that contains the configuration logic, SYSMON, and PCIe blocks. An HCS runs horizontally through the device in the center of each row of CRs, I/Os, and GTs. The HCS contains the horizontal routing and distribution tracks as well as leaf clock buffers and clock network interconnects between horizontal/vertical routing and distribution. Vertical tracks of routing and distribution connect all CRs in a column, while vertical routing spans an entire I/O column. There are 24 horizontal routing and 24 distribution tracks (Figure 2-1), and 24 vertical routing and 24 distribution tracks (Figure 2-2). The purpose of the clock routing resources is to route a clock from the global clock buffers to a central point from where it is connected to the loads via the distribution resources. This central point of the clock network is called a clock root in the UltraScale architecture. The root can be in any CR in a device from where it is routed to the loads via the clock distribution resources. This architecture optimized clock skew. Routing and distribution resources can either connect to adjacent CRs or disconnect (isolated) at the border of the CR as needed. This concept extends to SSI devices as well.

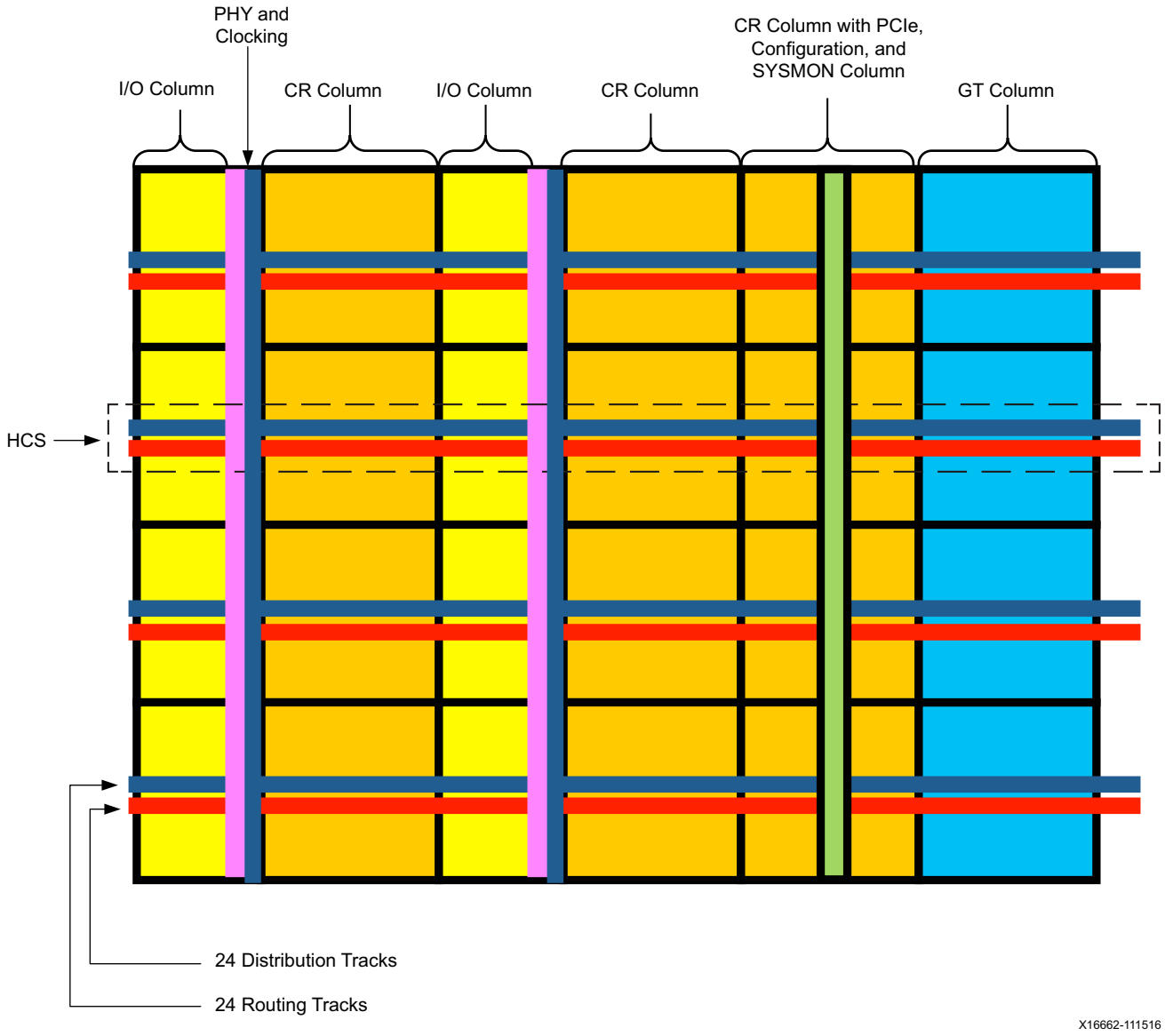

Figure 2-1: **Horizontal Clocking**

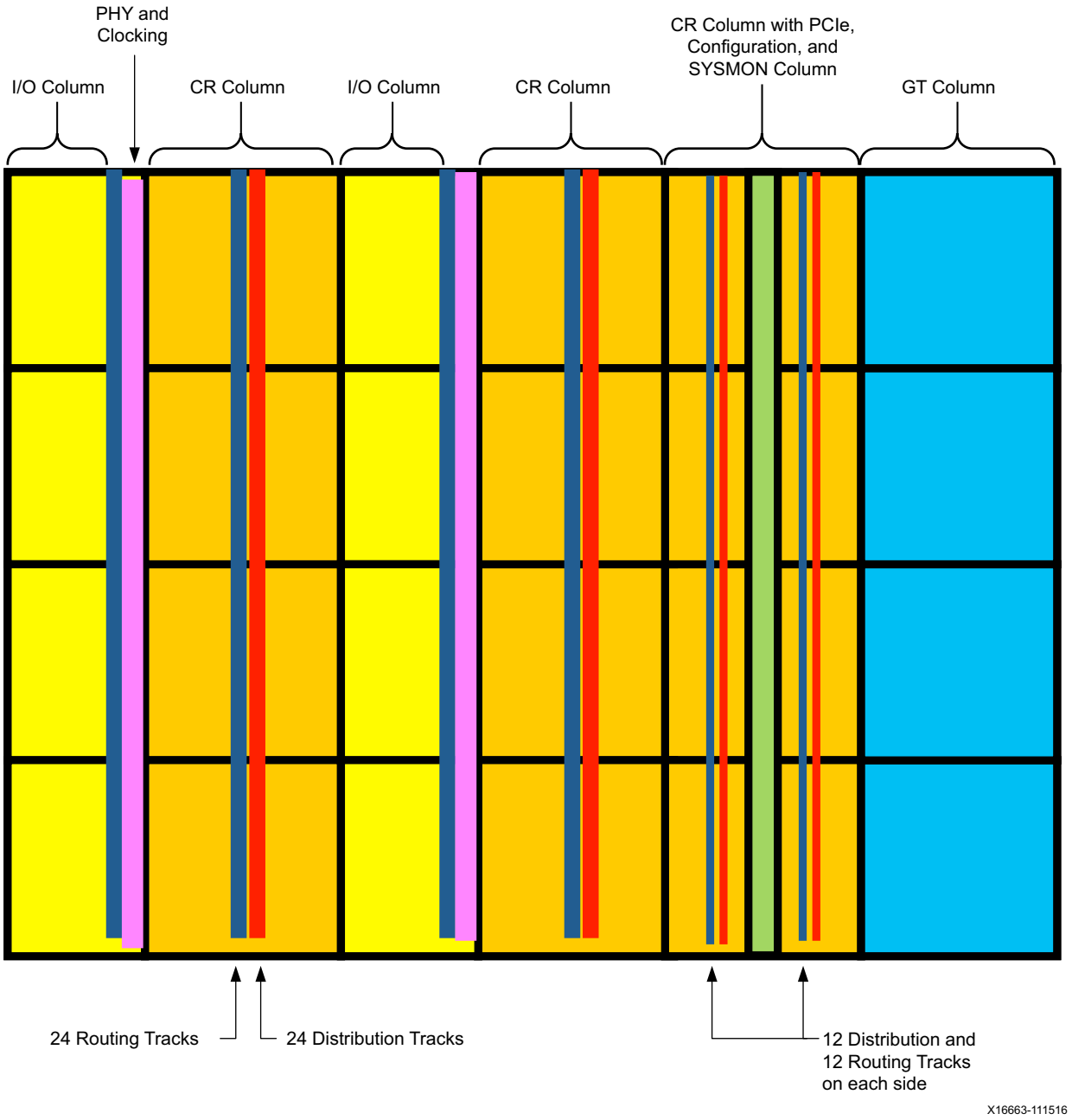

Figure 2-2: **Vertical Clocking**

The clocks can be distributed from their sources in one of two ways (Figure 2-3):

- The clocks can go onto routing tracks that take the clocks to a central point in a CR without going to any loads. The clocks can then drive the distribution tracks unidirectionally from which the clock networks fan out. In this way, the clock buffers can drive to a specific point in the CRs from which the clock buffers travel vertically and then horizontally on the distribution tracks to drive the clocking points. The clocking

points are driven via leaf clocks with clock enable (CE) in that CR and adjacent CRs, if needed. Distribution tracks cannot drive routing tracks.

This distribution scheme is used to move the root for all the loads to be at a specific location for improved, localized skew. Furthermore, both routing and distribution tracks can drive into horizontally or vertically adjacent CRs in a segmented fashion. Routing tracks can drive both routing and distribution tracks in the adjacent CRs while the distribution tracks can drive other horizontal distribution tracks in adjacent CRs. The CR boundary segmentation allows construction of either truly global, device-wide clock networks or more local clock networks of variable sizes by reusing clocking tracks.

- Alternatively, clock buffers can drive straight onto the distribution tracks and distribute the clock in that manner. This reduces the clock insertion delay.

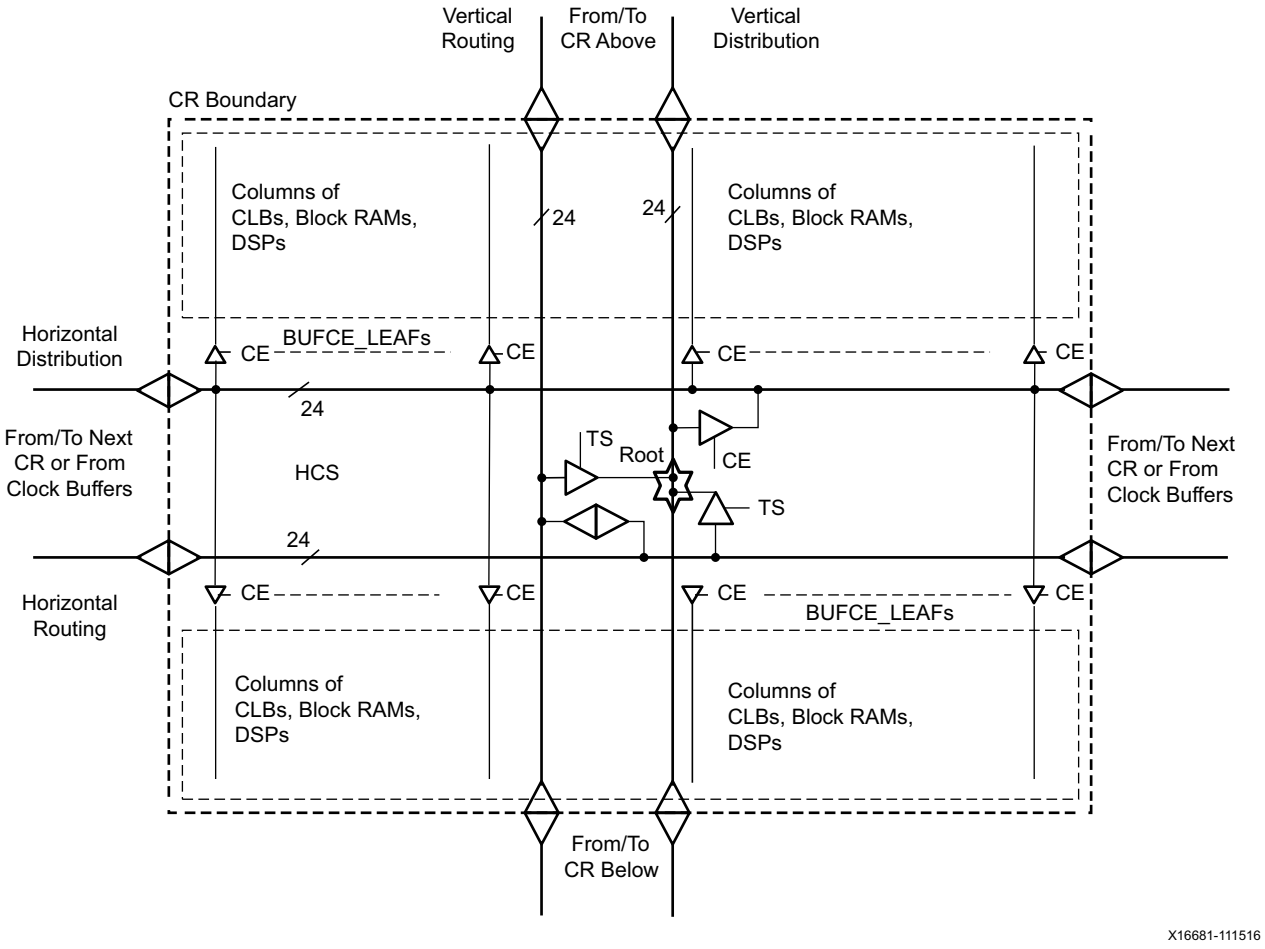

Figure 2-3: **Clock Region Clocking**

- Each of the four bytes in the XIPHY BITSLICE have six connections from the HCS to their global clocking pins. Therefore, only six BUFGs can drive the BITSLICE clocking pins in either half of an I/O bank (a maximum of 6 clocks can drive any half of an I/O bank).

Clock Buffers

The PHY global clocking contains several sets of BUFGCTRLs, BUFGCEs, and BUFGCE_DIVs. Each set can be driven by four GC pins from the adjacent bank, MMCMs, PLLs in the same PHY, and interconnect. The clock buffers then drive the routing and distribution resources across the entire device. Each PHY contains 24 BUFGCEs, 8 BUFGCTRLs, and 4 BUFGCE_DIVs but only 24 of them can be used at the same time.

IMPORTANT: *It is recommended to only allow the Vivado® Placer to assign all global clock buffers to specific locations. Each CR contains 24 BUFGCEs, 8 BUFGCTRLs and 4 BUFGCE_DIVs. These clock buffers share the 24 routing tracks and therefore collisions may occur resulting in unroutable designs. If the design requires a number of global clock buffers to be in a certain CR then it is recommended to attach the CLOCK_REGION property to these buffers instead of a specific LOCATION property.*

In the clocking architecture, BUFGCTRL multiplexers and all derivatives can be cascaded to adjacent clock buffers, effectively creating a ring of eight BUFGMUXes (BUFGCTRL multiplexers). Figure 2-4 shows a simplified diagram of cascading BUFGCTRLs.

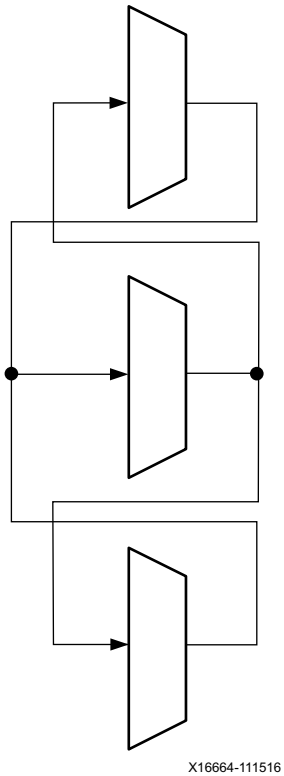

Figure 2-4: **Cascading BUFGCTRLs**

The following subsections detail the various configurations, primitives, and use models of the clock buffers.

BUFGCTRL Clock Buffer Primitives

The primitives in Table 2-1 are different configurations of the clock BUFGCTRL buffers. The Vivado tools manage the configuration of all these primitives, and the *Vivado Design Suite User Guide: Using Constraints* (UG903) [Ref 4] describes the LOC constraint.

Table 2-1: **BUFGCTRL Clock Buffer Primitives**

Primitive	Input	Output	Control
BUFGCTRL	I0, I1	O	CE0, CE1, IGNORE0, IGNORE1, S0, S1
BUFGCE_1	I	O	CE
BUFGMUX	I0, I1	O	S
BUFGMUX_1	I0, I1	O	S
BUFGMUX_CTRL	I0, I1	O	S

BUFGCTRL

The BUFGCTRL primitive shown in Figure 2-5 can switch between two asynchronous clocks. All other global clock buffer primitives are derived from certain configurations of BUFGCTRL.

BUFGCTRL has four select lines, S0, S1, CE0, and CE1. It also has two additional control lines, IGNORE0 and IGNORE1. These six control lines are used to control the inputs I0 and I1.

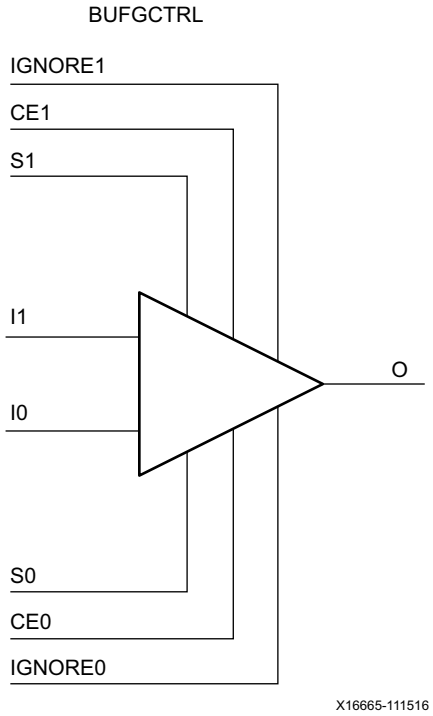

Figure 2-5: **BUFGCTRL Primitive**

BUFGCTRL is designed to switch between two clock inputs without the possibility of a glitch. When the presently selected clock transitions from High to Low after S0 and S1 change, the output is kept Low until the other (to-be-selected) clock transitions from High to Low. Then, the new clock starts driving the output.The default configuration for BUFGCTRL is falling-edge sensitive and held at Low prior to the input switching. BUFGCTRL can also be rising-edge sensitive and held at High prior to the input switching by using the INIT_OUT attribute.

In some applications, the conditions previously described are not desirable. Asserting the IGNORE pins bypasses the BUFGCTRL from detecting the conditions for switching between two clock inputs. In other words, asserting IGNORE causes the MUX to switch the inputs at the instant the select pin changes. IGNORE0 causes the output to switch away from the I0 input immediately when the select pin changes, while IGNORE1 causes the output to switch away from the I1 input immediately when the select pin changes.

Selection of an input clock requires a "select" pair (S0 and CE0, or S1 and CE1) to be asserted High. If either S or clock enable (CE) is not asserted High, the desired input is not selected. In normal operation, both S and CE pairs (all four select lines) are not expected to be asserted High simultaneously. Typically, only one pin of a "select" pair is used as a select line, while the other pin is tied High. The truth table is shown in Table 2-2.

Table 2-2: **Truth Table for Clocking Resources**

CE0	S0	CE1	S1	O
1	1	0	X	I0
1	1	X	0	I0
0	X	1	1	I1
X	0	1	1	I1
1	1	1	1	Old Input (1)

Notes:

1. Old input refers to the valid input clock before this state is achieved.
2. For all other states, the output becomes the value of INIT_OUT and does not toggle.

Although both S and CE are used to select a desired output, only S is suggested for glitch-free switching. This is because when using CE to switch clocks, the change in clock selection can be faster than when using S. A violation in the setup/hold time of the CE pins causes a glitch at the clock output. On the other hand, using the S pins allows the user to switch between the two clock inputs without regard to setup/hold times. As a result, using S to switch clocks does not result in a glitch. See BUFGMUX_CTRL, page 24.

The timing diagram in Figure 2-6 illustrates various clock switching conditions using the BUFGCTRL primitives. Exact timing numbers are best found using the speed specification.

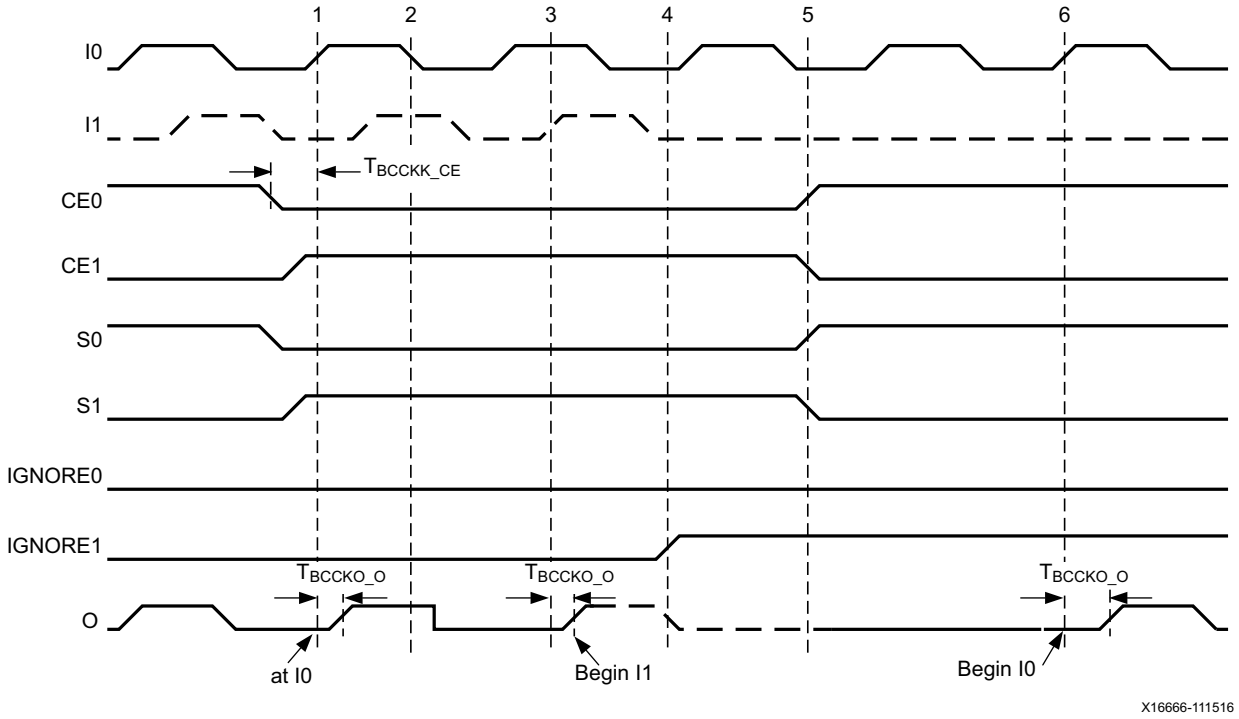

Figure 2-6: **BUFGCTRL Timing Diagram**

- Before time event 1, output O uses input I0.
- At time TBCCCK_CE, before the rising edge at time event 1, both CE0 and S0 are deasserted Low. At about the same time, both CE1 and S1 are asserted High.
- At time TBCCKO_O, after time event 3, output O uses input I1. This occurs after a High-to-Low transition of I0 (event 2) followed by a High-to-Low transition of I1.
- At time event 4, IGNORE1 is asserted.
- At time event 5, CE0 and S0 are asserted High while CE1 and S1 are deasserted Low. At TBCCKO_O, after time event 6, output O has switched from I1 to I0 without requiring a High-to-Low transition of I1.

Other capabilities of BUFGCTRL are:

- Pre-selection of the I0 and I1 inputs are made after configuration but before device operation.
- The initial output after configuration can be selected as either High or Low.
- Clock selection using CE0 and CE1 only (S0 and S1 tied High) can change the clock selection without waiting for a High-to-Low transition on the previously selected clock.

Table 2-3 summarizes the attributes for the BUFGCTRL primitive.

Table 2-3: **BUFGCTRL Attributes**

Attribute Name	Description	Possible Values
INIT_OUT	Initializes the BUFGCTRL output to the specified value after configuration. Sets the positive or negative edge behavior. Sets the output level when changing clock selection.	0 (default), 1
PRESELECT_I0	If TRUE, BUFGCTRL output uses the I0 input after configuration.(1)	FALSE (default), TRUE
PRESELECT_I1	If TRUE, BUFGCTRL output will use the I1 input after configuration.(1)	FALSE (default), TRUE

Notes:
1. Both PRESELECT attributes cannot be TRUE at the same time.

BUFGCE_1

BUFGCE_1 is a clock buffer with one clock input, one clock output, and a clock enable line. This primitive is based on BUFGCTRL with some pins connected to logic High or Low. Figure 2-7 illustrates the relationship of BUFGCE_1 and BUFGCTRL. The LOC constraint is available for manually placing the BUFGCE_1 location. See the *Vivado Design Suite User Guide: Using Constraints* (UG903) [Ref 4] for more information.

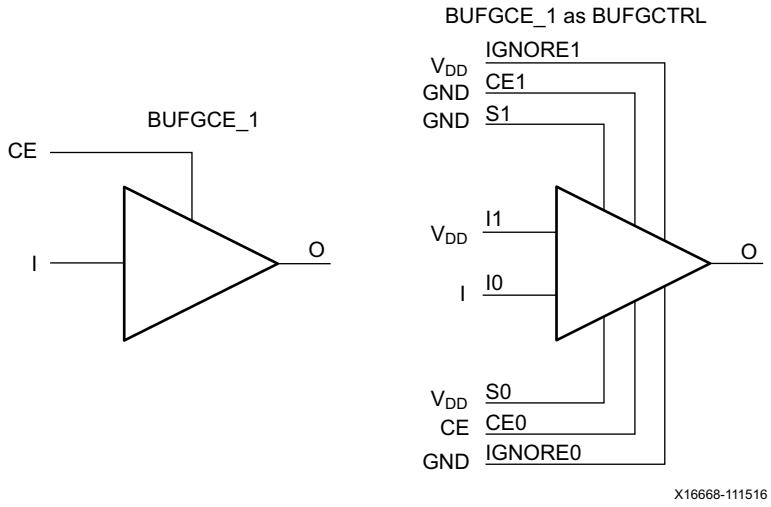

Figure 2-7: **BUFGCE_1 as BUFGCTRL**

The switching condition for BUFGCE_1 is similar to BUFGCTRL with INIT_OUT set to 1. If the CE input is Low prior to the incoming falling clock edge, the following clock pulse does not pass through the clock buffer, and the output stays High. Any level change of CE during the incoming clock Low pulse has no effect until the clock transitions High. The output stays High when the clock is disabled. However, when the clock is being disabled, it completes the clock Low pulse.

IMPORTANT: *Because the clock enable line uses the CE pin of the BUFGCTRL, the select signal must meet the setup time requirement. Violating this setup time can result in a glitch.*

Figure 2-8 illustrates the timing diagram for BUFGCE_1.

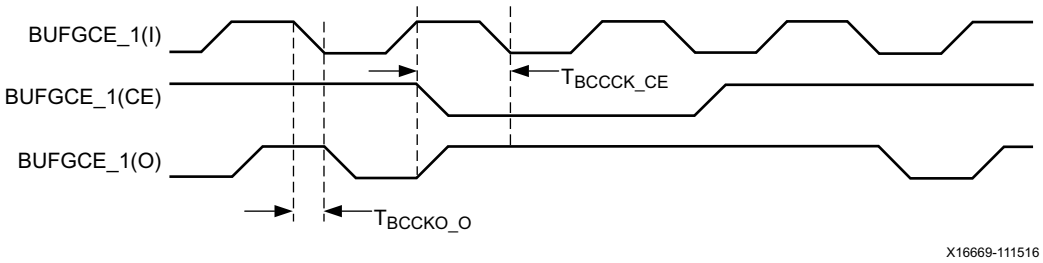

Figure 2-8: **BUFGCE_1 Timing Diagram**

BUFGMUX and BUFGMUX_1

BUFGMUX is a clock buffer with two clock inputs, one clock output, and a select line. This primitive is based on BUFGCTRL with some pins connected to logic High or Low.

Figure 2-9 illustrates the relationship of BUFGMUX and BUFGCTRL. The LOC constraint is available for manually placing the BUFGMUX and BUFGCTRL locations. See the *Vivado Design Suite User Guide: Using Constraints* (UG903) [Ref 4] for more information.

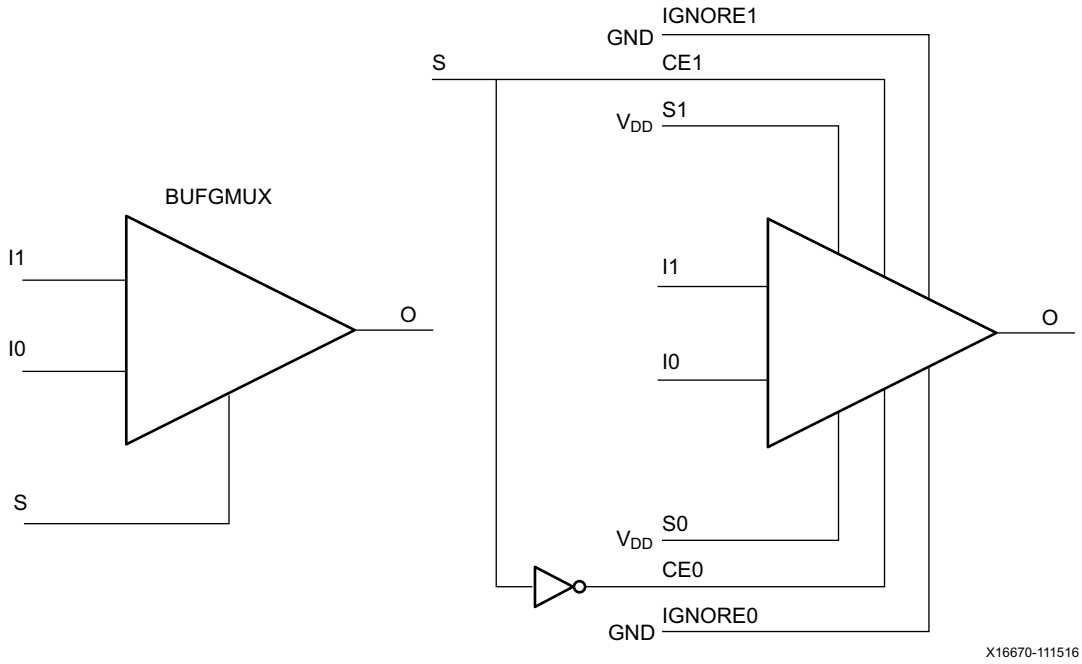

Figure 2-9: **BUFGMUX as BUFGCTRL**

IMPORTANT: *Because BUFGMUX uses the CE pins as select pins, when using the select, the setup time requirement must be met. Violating this setup time can result in a glitch.*

Switching conditions for BUFGMUX are the same as the CE pins on BUFGCTRL. Figure 2-10 illustrates the timing diagram for BUFGMUX.

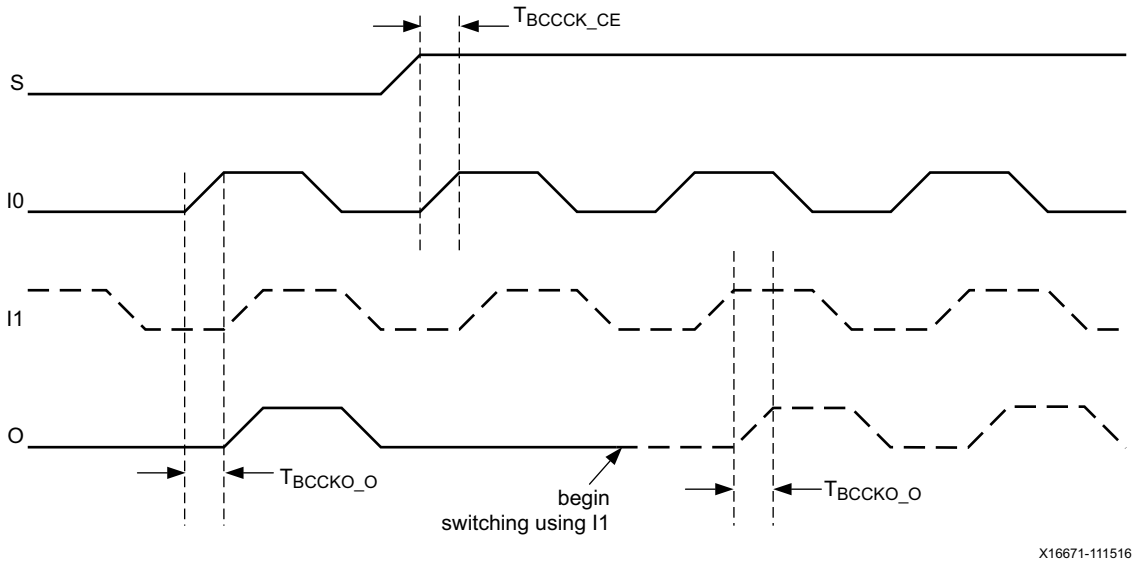

Figure 2-10: **BUFGMUX Timing Diagram**

In Figure 2-10:

- The current clock is I0.
- S is activated High.
- If I0 is currently High, the multiplexer waits for I0 to deassert Low.
- After I0 is Low, the multiplexer output stays Low until I1 transitions from High to Low.
- When I1 transitions from High to Low, the output switches to I1.
- If setup/hold times are met, no glitches or short pulses can appear on the output.

BUFGMUX_1 is rising-edge sensitive and held at High prior to input switch. Figure 2-11 illustrates the timing diagram for BUFGMUX_1. The LOC constraint is available for manually placing the BUFGMUX and BUFGMUX_1 locations. See the *Vivado Design Suite User Guide: Using Constraints* (UG903) [Ref 4] for more information.

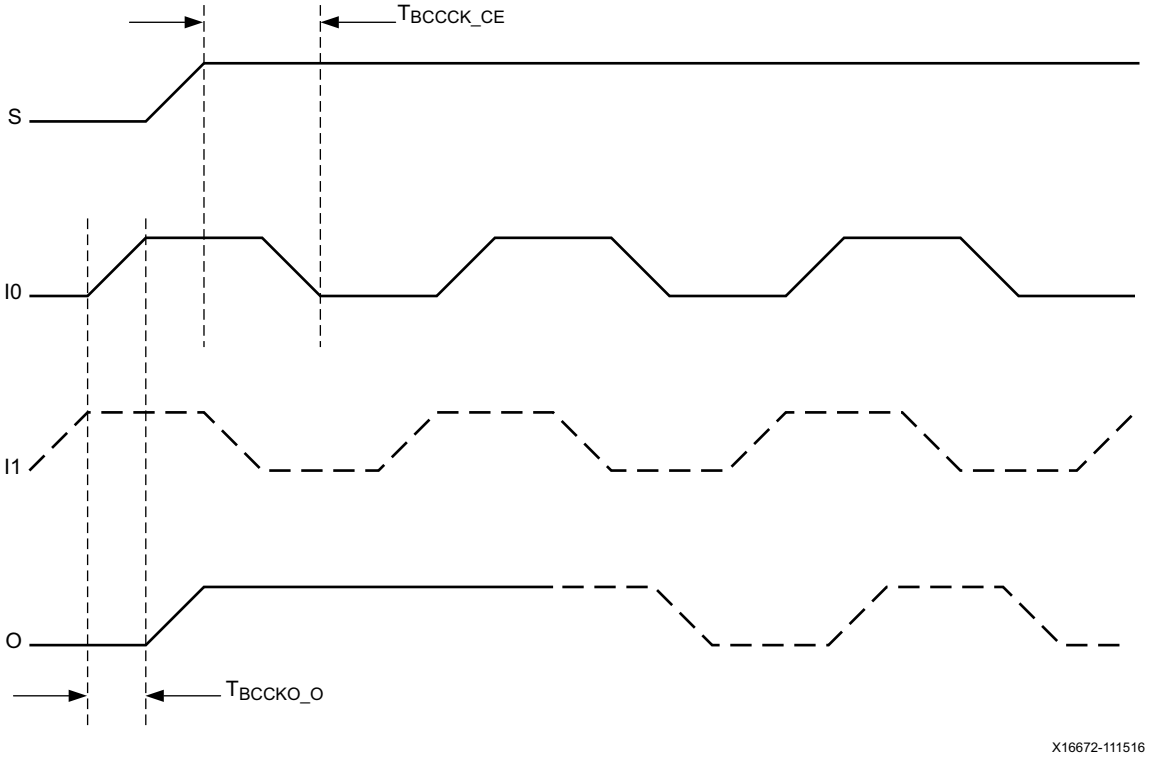

Figure 2-11: **BUFGMUX_1 Timing Diagram**

In Figure 2-11:

- The current clock is I0.
- S is activated High.
- If I0 is currently Low, the multiplexer waits for I0 to be asserted High.
- After I0 is High, the multiplexer output stays High until I1 transitions from Low to High.
- When I1 transitions from Low to High, the output switches to I1.
- If setup/hold times are met, no glitches or short pulses can appear on the output.

Table 2-4 summarizes the attributes for the BUFGMUX primitive.

Table 2-4: **BUFGMUX Attributes**

Attribute Name	Description	Possible Values
CLK_SEL_TYPE	Specifies synchronous or asynchronous clock switching.	SYNC (default), ASYNC

BUFGMUX_CTRL

BUFGMUX_CTRL is a clock buffer with two clock inputs, one clock output, and a select line. This primitive is based on BUFGCTRL with some pins connected to logic High or Low. Figure 2-12 illustrates the relationship of BUFGMUX_CTRL and BUFGCTRL.

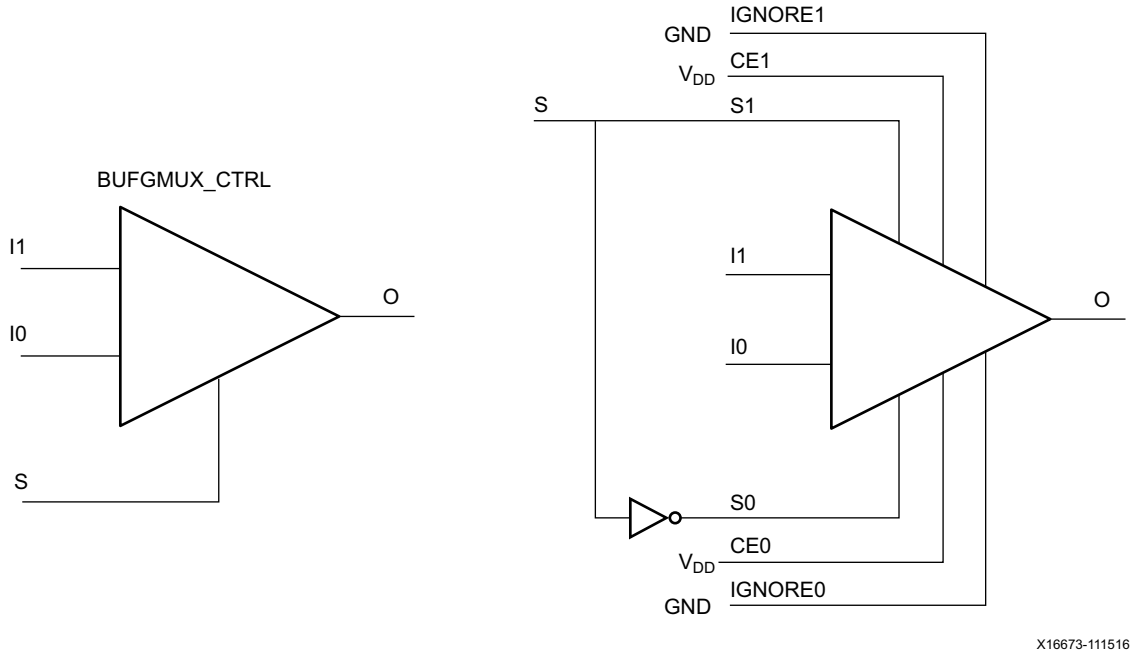

Figure 2-12: **BUFGMUX_CTRL as BUFGCTRL**

BUFGMUX_CTRL uses the S pins as select pins. S can switch anytime without causing a glitch. The setup/hold time on S is for determining whether the output passes an extra pulse of the previously selected clock before switching to the new clock. If S changes as shown in Figure 2-13 prior to the setup time TBCCCK_S and before I0 transitions from High to Low, the output does not pass an extra pulse of I0. If S changes following the hold time for S, the output passes an extra pulse. If S violates the setup/hold requirements, the output might pass the extra pulse but it will not glitch. In any case, the output changes to the new clock within three clock cycles of the slower clock.

The setup/hold requirements for S0 and S1 are with respect to the falling clock edge, not the rising edge as for CE0 and CE1.

Switching conditions for BUFGMUX_CTRL are the same as the S pin of BUFGCTRL. Figure 2-13 illustrates the timing diagram for BUFGMUX_CTRL.

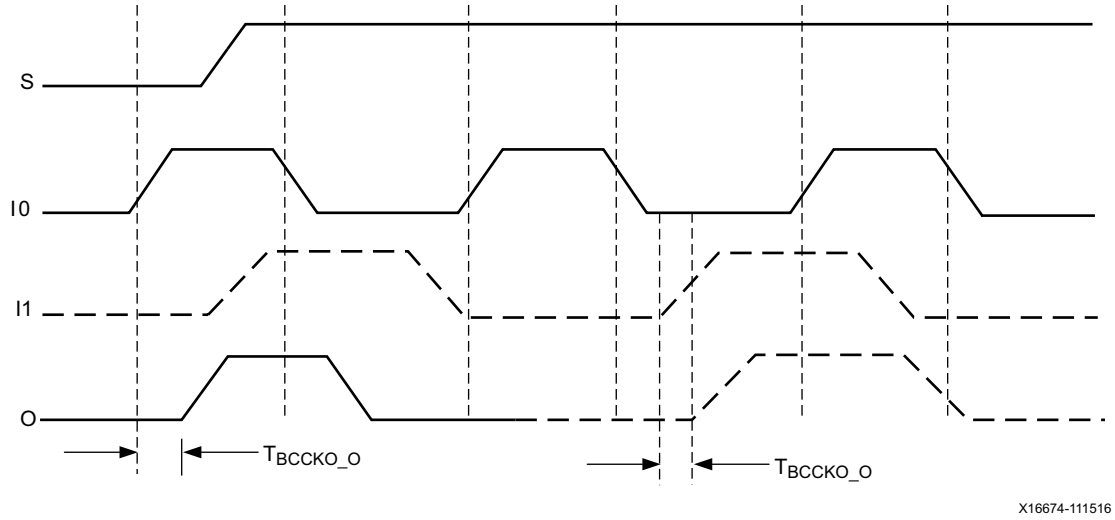

Figure 2-13: **BUFGMUX_CTRL Timing Diagram**

Other capabilities of the BUFGMUX_CTRL primitive are:

- I0 and I1 inputs can be preselected after configuration.
- Initial output can be selected as High or Low after configuration.

Additional BUFGCTRL Use Models

Asynchronous MUX Using BUFGCTRL

In some cases an application requires immediate switching between clock inputs or bypassing the edge sensitivity of BUFGCTRL. An example is when one of the clock inputs is no longer switching. If this happens, the clock output would not have the proper switching conditions because the BUFGCTRL never detected a clock edge. This case uses the asynchronous MUX. Figure 2-14 illustrates an asynchronous MUX with BUFGCTRL design example.

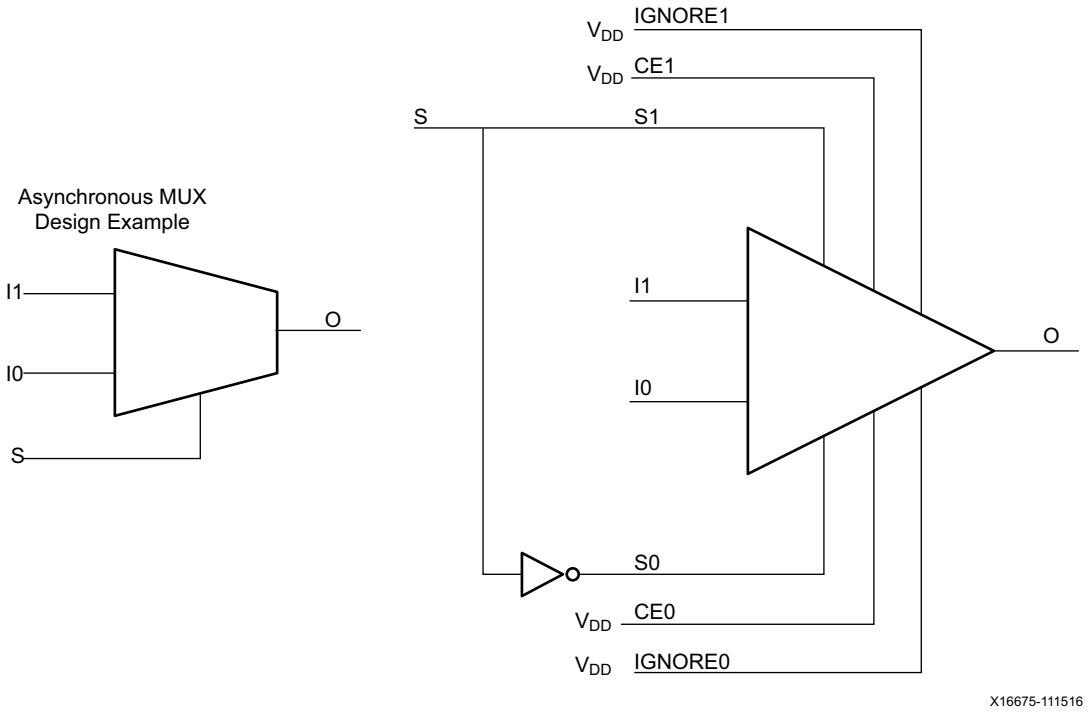

Figure 2-14: **Asynchronous MUX with BUFGCTRL Design Example**

Figure 2-15 shows the asynchronous MUX timing diagram.

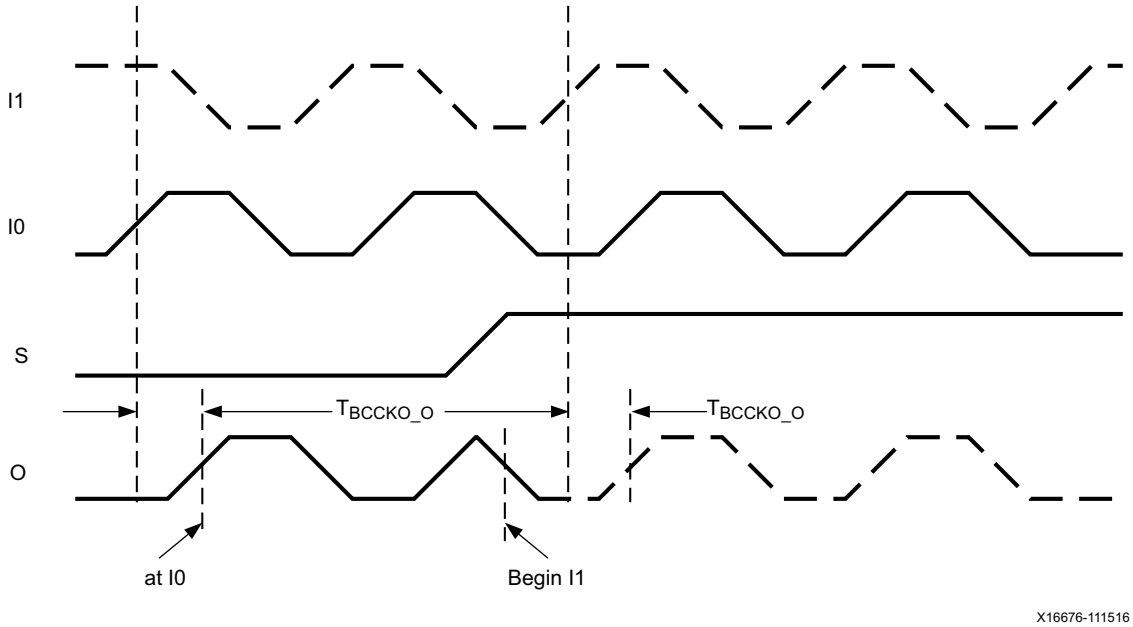

Figure 2-15: **Asynchronous MUX Timing Diagram**

In Figure 2-15:

- The current clock is from I0.
- S is activated High.
- The clock output immediately switches to I1.
- When ignore signals are asserted High, glitch protection is disabled.

BUFGMUX_CTRL with a Clock Enable

A BUFGMUX_CTRL with a clock enable BUFGCTRL configuration allows you to choose between the incoming clock inputs. If needed, the clock enable is used to disable the output. Figure 2-16 illustrates the BUFGCTRL usage design example and Figure 2-17 shows the timing diagram.

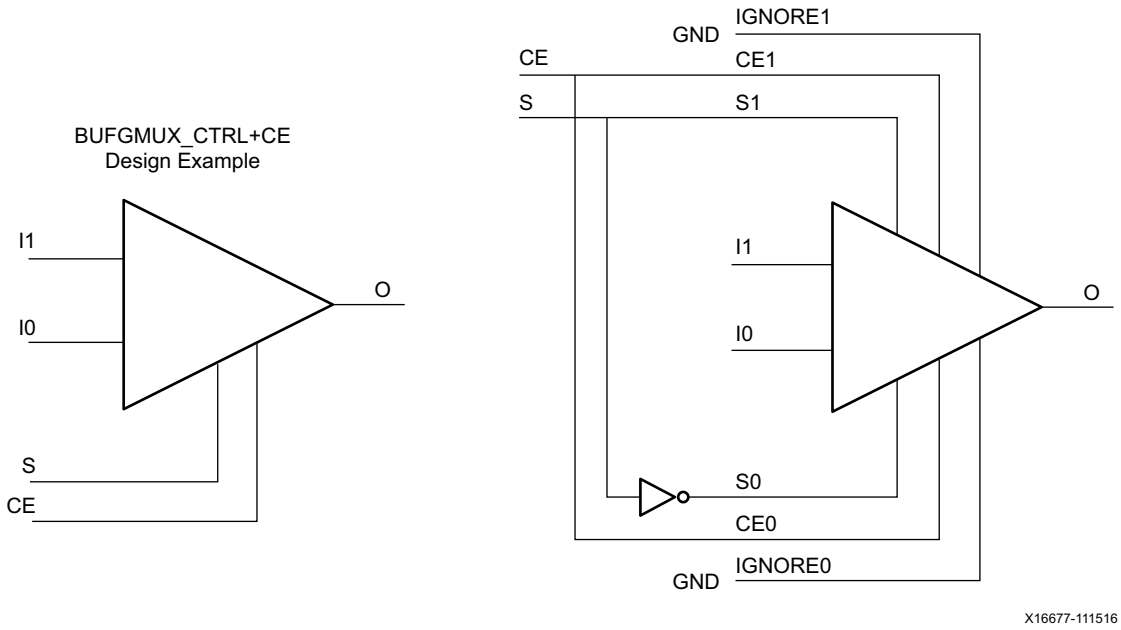

Figure 2-16: **BUFGMUX_CTRL with a CE and BUFGCTRL**

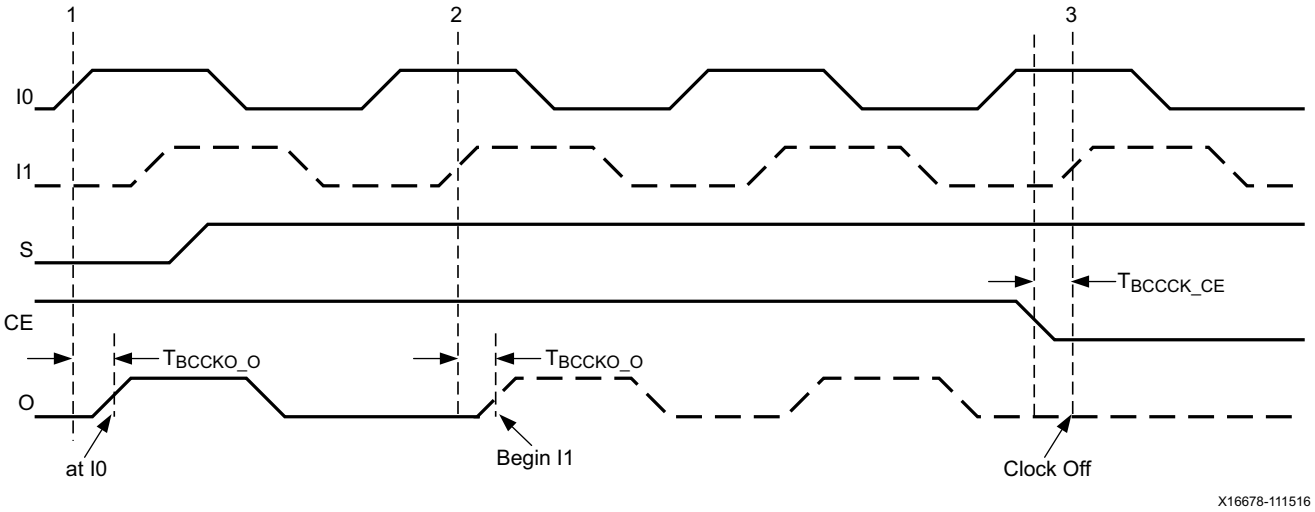

Figure 2-17: **BUFGMUX_CTRL with a CE Timing Diagram**

In Figure 2-17:

- At time event 1, output O uses input I0.
- Before time event 2, S is asserted High.
- At time TBCCKO_O, after time event 2, output O uses input I1. This occurs after a High-to-Low transition of I0 followed by a High-to-Low transition of I1 is completed.
- At time TBCCCK_CE, before time event 3, CE is asserted Low. To avoid any output clock glitches, the clock output is switched Low and kept at Low until after a High-to-Low transition of I1 is completed.

BUFGCE Clock Buffers

BUFGCE is a clock buffer with one clock input, one clock output, and a clock enable line (Figure 2-18). This buffer provides glitchless clock gating. BUFGCE can directly drive the routing resources and is a clock buffer with a single gated input. Its O output is 0 when CE is Low (inactive). When CE is High, the I input is transferred to the O output.

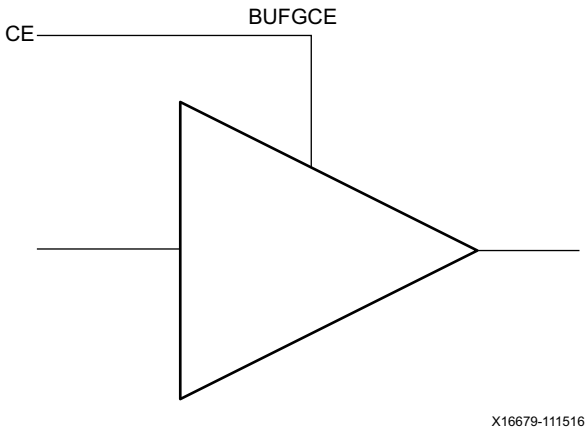

Figure 2-18: **BUFGCE Buffer**

Table 2-5 lists the BUFGCE pins.

Table 2-5: **BUFGCE Pins**

Pin Name	Type	Invertible	Description
CE	Input	TRUE	Clock enable
I	Input	FALSE	Clock buffer
O	Output	FALSE	Clock buffer

Table 2-6 shows the BUFGCE attributes.

Table 2-6: **BUFGCE Attributes**

Attribute Name	Values	Default	Type	Description
CE_TYPE	SYNC, ASYNC	SYNC	STRING	Sets the clock enable behavior where SYNC allows for glitchless transition while ASYNC allows immediate transition.

Figure 2-19 shows the BUFGCE timing diagram.

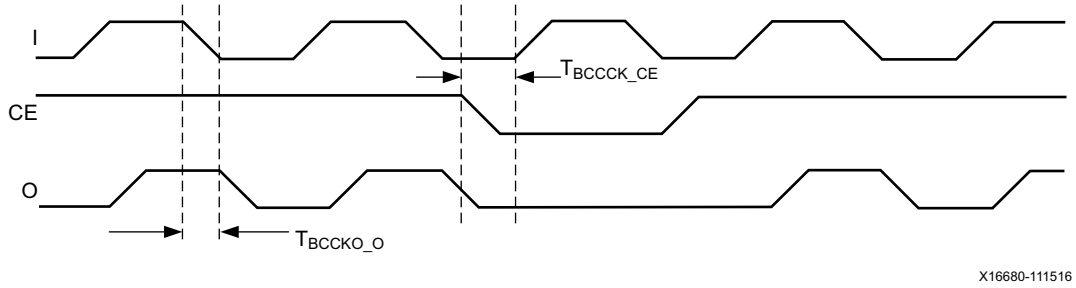

Figure 2-19: **BUFGCE Timing Diagram**

BUFG Clock Buffer

BUFG is a clock buffer with one clock input and one clock output. This primitive is based on BUFGCE with the CE pin connected to High, as shown in Figure 2-20.

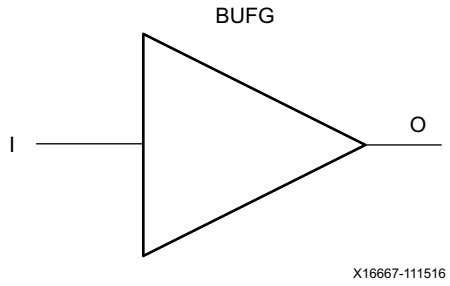

Figure 2-20: **BUFG Buffer**

BUFCE_LEAF Clock Buffer

BUFCE_LEAF is a clock buffer with CE for leaf driving off horizontal HCS row. This buffer is an interconnect leaf clock buffer driving the clocking point of the various blocks with a single gated input. Its O output is 0 when CE is Low (inactive). When CE is High, the I input is transferred to the O output. Table 2-7 shows the BUFCE_LEAF attributes.

Table 2-7: **BUFCE_LEAF Attributes**

Attribute Name	Values	Default	Type	Description
CE_TYPE	SYNC, ASYNC	SYNC	STRING	Sets the clock enable behavior where SYNC allows for glitchless transition while ASYNC allows immediate transition.

The BUFGCE_LEAF is documented for information purpose only and is not user accessible in Vivado design suite (e.g. for instantiation, placement, etc.)

BUFGCE_DIV

BUFGCE_DIV is a clock buffer with one clock input (I), one clock output (O), one clear input (CLR) and a clock enable (CE) input. BUFGCE_DIV can directly drive the routing and distribution resources and is a clock buffer with a single gated input and a reset. Its O output is 0 when CLR is High (active). When CE is High, the I input is transferred to the O output. CE is synchronous to the clock for glitch-free operation. CLR is an asynchronous reset assertion and synchronous reset deassertion to this buffer. BUFGCE_DIV can also divide the input clock by 1 to 8.

When CLR (reset) is deasserted, the output clock transitions from Low to High on the first edge after the CLR is deasserted, regardless of the divide value. Therefore, BUFGCE_DIV output clocks are always aligned, regardless of the divide value. The output clock then toggles at the divided frequency. When CLR is asserted, the clock stops toggling after some clock-to-out time. For an odd divide, the duty cycle is not 50% because the clock is High one cycle less than it is Low. For example, for a divide value of 7, the clock is High for 3 cycles and Low for 4 cycles.

When CE is deasserted, the output stops at its current state, High or Low. When CE is reasserted, the internal counter restarts from where it stopped. For example, if the divide value is 8 and CE is deasserted two input clock cycles after the last output High transition, the output stays High. Then when CE is reasserted, the output transitions Low after two input clock cycles. If the reset input is used, upon assertion the output transitions Low immediately if the current output is High, otherwise it stays Low.

Since reset is synchronously deasserted, when reset is deasserted in the previous example, the output transitions High at the next input clock edge and transitions Low four input clock cycles later.

Table 2-8 shows the BUFGCE_DIV pins.

Table 2-8: **BUFGCE_DIV Pins**

Pin Name	Type	Invertible	Description
I	Input	FALSE	Clock input
CLR	Input	TRUE	Reset
CE	Input	TRUE	Clock enable
O	Output	FALSE	Clock output

Table 2-9 shows the BUFGCE_DIV attributes.

Table 2-9: **BUFGCE_DIV Attributes**

Attribute Name	Values	Default	Type	Description
BUFGCE_DIVIDE	1, 2, 3, 4, 5, 6, 7, 8	1	STRING	Defines whether the output clock is a divided version of the input clock.

BUFG_GT and BUFG_GT_SYNC

The BUFG_GTs are driven by the gigabit transceivers (GTs) and the ADC/DAC blocks in the RFSoC devices. BUFG_GTs provide the only means for those blocks to drive the clock routing resources. Only GTs, ADCs, and DACs can drive BUFG_GTs. BUFG_GT (Figure 2-21) is a clock buffer with one clock input (I), one clock output (O), one clear input (CLR) with CLR mask input (CLRMASK), a clock enable (CE) input with a CE mask input (CEMASK) and a 3-bit divide (DIV[2:0]) input. BUFG_GT_SYNC is the synchronizer circuit for the BUFG_GTs and is shown here explicitly. The BUFG_GT_SYNC primitive is automatically inserted by the Vivado tools, if not present in the design. This buffer can directly drive the routing and distribution resources and is a clock buffer with a single gated input and a reset. When CE is deasserted (Low) the output stops at its current state, High or Low. When CE is High, the I input is transferred to the O output. Both edges of CE and the deassertion of CLR are automatically synchronized to the clock for glitch-free operation. The Vivado tools do not support timing for the CE pin, therefore, a deterministic latency cannot be achieved. CLR is an asynchronous reset assertion and synchronous reset deassertion to the BUFG_GTs. The synchronizers have two stages, but the CLR pin does not have a setup/hold timing arc assigned. Therefore, the latency is not deterministic. BUFG_GTs can also divide the input clock by 1 to 8. The DIV[2:0] value is the actual divide minus 1 (i.e., `3'b000` corresponds to 1 while `3'b111` corresponds to 8). The divide value (DIV inputs), CEMASK, and CLRMASK must be changed while the buffer is held in reset. The input clock is allowed to change while CE is deasserted or reset is asserted. However, there is a minimum deassertion/assertion time for those control signals.

IMPORTANT: *In RFSoC devices, the ADC and DAC tiles replace the GTH transceivers that are present in the MPSoC devices. Therefore, ADC and DAC utilize the existing BUFG_GT clock buffers to drive the global clock trees in the device and then back into the ADC/DAC tiles from the fabric. However, the DIV function cannot be used when connecting to the ADC/DAC clocks. Hence the BUFG_GT functions more like a simple global clock buffer with CE and CLR.*

IMPORTANT: *For devices in Zynq Ultrascale+ and selected devices in Kintex Ultrascale+ families (XCKU9P and above), assigning the clock root in the same region as the BUFG_GT driver (X0 column) can cause an unroutable situation and prevent the output clocks from reaching loads that are placed in the clock regions to the right of the Zynq UltraScale+ device PS or Kintex UltraScale+ empty PL regions in the Y0, Y1, and Y2 rows. To avoid the issue, users need to assign clock root one clock region to the right, in this case the X1 column.*

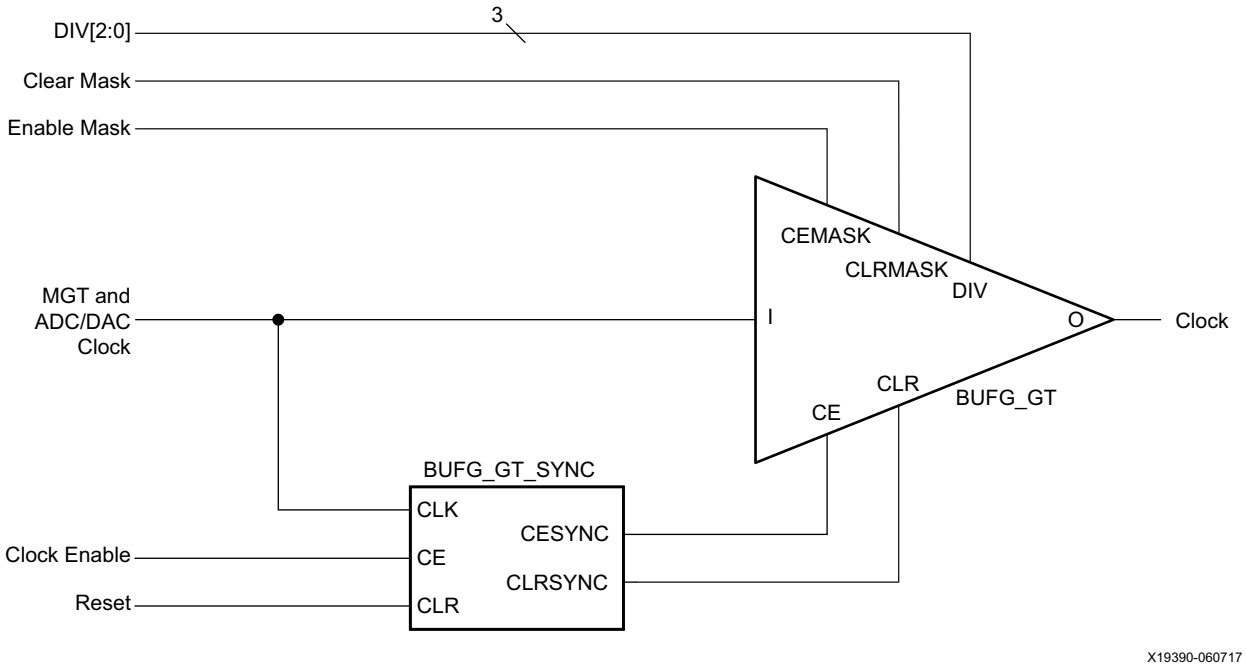

Figure 2-21: **BUFG_GT Primitive**

UltraScale FPGAs have 24 BUFG_GTs and 10 BUFG_GT_SYNCs per GT Quad. UltraScale+ devices also have 24 BUFG_GTs but they have 14 BUFG_GT_SYNCs per GT Quad. Any of the GT output clocks in a Quad can be multiplexed to any of the BUFG_GTs. In UltraScale devices, there are 10 CE and CLR pins which correspond to the 10 BUFG_GT_SYNCs and that can drive the 24 BUFG_GTs. In UltraScale+ devices, there are 14 CE and CLR pins which correspond to the 14 BUFG_GT_SYNCs and that can drive the 24 BUFG_GTs. Each of the BUFG_GT buffers have an individual mask for both CE and CLR (24). All BUFG_GTs driven by the same clock source must also have a common CE and CLR signal. Tying off CE and CLR to a constant signal in this case is not allowed, but a mask can be set to provide the same functionality. The output clocks of the BUFG_GTs connected to the same input clock are synchronized (phase aligned) to each other when coming out of reset (CLR) or on CE assertion. Individual mask pins can be used to control which BUFG_GT(s) out of the group of 24 respond to CE and CLR and therefore are synchronized to each other or retain their previous phase and divide value. These clock buffers are located in the HCS and are directly driven by the GT output clocks. Their purpose is to directly drive hard blocks and logic in the CRs via routing and distribution resources. GTs have no other direct, dedicated connections to other clock resources. However, they can connect to the CMT via the BUFG_GT and the clock routing resources.

When CLR (reset) is deasserted, the output transitions High at the next input clock edge and transitions Low divide_value/2 input clock cycles later. Because reset is synchronously deasserted, two clock cycles of synchronization latency need to be added to the output to transition it to High. The next transition to Low then occurs four input clock cycles after that (divide by 8). The output transitions to High a number of clock cycles later, determined by the divide value specified, after which the output clock toggles at the divided frequency.

When CLR is asserted, the clock stops toggling at Low after some clock-to-out time. For an odd divide, the duty cycle is not 50% because the clock is High one cycle less than it is Low. For example, for a divide value of 7, the clock is High for 3 cycles and Low for 4 cycles.

When CE is deasserted, the output stops at its current state, High or Low. When CE is reasserted, the internal counter restarts from where it stopped. For example, if the divide value is 8 and CE is deasserted two input clock cycles after the last output High transition, the output stays High. Then, when CE is reasserted, the output transitions Low four input clock cycles later (two for synchronization and two to complete the High time period of the output clock because of being a divide by 8). If the reset input is used, upon assertion the output transitions Low immediately if the current output is High, otherwise it stays Low. Because reset is synchronously deasserted, when reset is deasserted in the previous example, the output transitions High two input clock cycles later due to synchronization and transitions Low four input clock cycles after that (divide by 8).

The mask pins (CEMASK and CLRMASK) control how a specific, single BUFG_GT responds to the CE/CLR control inputs. When a mask pin is deasserted, its respective control pin has their normal function. When a mask pin is asserted, the respective control pin is ignored, in effect allowing the clock to propagate through (i.e., CE is effectively High and reset is effectively Low). The internal synchronizers phase align the clock outputs of the BUFG_GTs that are not masked. Both edges of CE are synchronized while only the deassertion of reset is synchronized. Assertion of reset immediately causes the output of the BUFG_GT to go Low if it was previously High. This can cause a potential glitch or runt pulse. If this is not acceptable, CE should be used to stop the output. A reset should then be asserted after two input clock cycles plus half the "divide value." This ensures that the output clock High time (if the output clock happened to be disabled High) is no less than normal.

IMPORTANT: *While the synchronizers ensure that all BUFG_GTs driven by the same clock come out of reset in phase, they might not be in phase with BUFG_GTs that have not been reset (i.e., that have their reset mask asserted).*

BUFG_PS

The BUFG_PS is a simple clock buffer with one clock input (I), one clock output (O). This clock buffer is a resource for the Zynq UltraScale+ MPSoC processor system (PS) and provides access to the programmable logic (PL) clock routing resources for clocks from the processor into the PL. Up to 18 PS clocks can drive the BUFG_PS. This clock buffer resides next to the PS.

Clock Management Tile

Overview

In UltraScale™ architecture-based devices, the clock management tile (CMT) includes a mixed-mode clock manager (MMCM) and two phase-locked loops (PLLs). The main purpose of the PLL is to generate clocking for the I/Os. But it also contains a limited subset of the MMCM functions that can be used for general clocking purposes.

The clock input connectivity allows multiple resources to provide the reference clock(s) to the MMCM. The number of output counters (dividers) is eight, with some of them capable of driving out an inverted clock signal (180° phase shift). MMCMs have infinite fine phase shift capability in either direction and can be used in dynamic phase shift mode. The resolution of the fine phase shift depends on the voltage-controlled oscillator (VCO) frequency. Fractional divide functionality in increments of 1/8th (0.125) for CLKFBOUT and CLKOUT0 are available to support greater clock frequency synthesis capability. UltraScale architecture-based devices have a spread spectrum (SS) capability. If the MMCM spread-spectrum feature is not used, a spread spectrum on an external input clock will not be filtered and thus passed on to the output clock.

MMCMs

UltraScale architecture-based devices contain one CMT per I/O bank. The MMCMs serve as frequency synthesizers for a wide range of frequencies, and as jitter filters for either external or internal clocks, and deskew clocks.

Input multiplexers select the reference and feedback clocks from either the global clock I/Os or the clock routing or distribution resources. Each clock input has a programmable counter divider (D). The phase-frequency detector (PFD) compares both phase and frequency of the rising edges of both the input (reference) clock and the feedback clock. If a minimum High/Low pulse is maintained, the duty cycle is ancillary. The PFD is used to generate a signal proportional to the phase and frequency between the two clocks. This signal drives the charge pump (CP) and loop filter (LF) to generate a reference voltage to the VCO. The PFD produces an up or down signal to the charge pump and loop filter to determine whether the VCO should operate at a higher or lower frequency. When VCO operates at a frequency that is too high, the PFD activates a down signal causing the control

voltage to be reduced, thus decreasing the VCO operating frequency. When the VCO operates at a frequency that is too low, an up signal increases voltage. The VCO produces eight output phases and one variable phase for fine-phase shifting. Each output phase can be selected as the reference clock to the output counters (Figure 3-1). Each counter can be independently programmed for a given customer design. A special counter M is also provided. This counter controls the feedback clock of the MMCM, allowing a wide range of frequency synthesis.

In addition to integer divide output counters, MMCMs add a fractional counter for CLKOUT0 and CLKFBOUT.

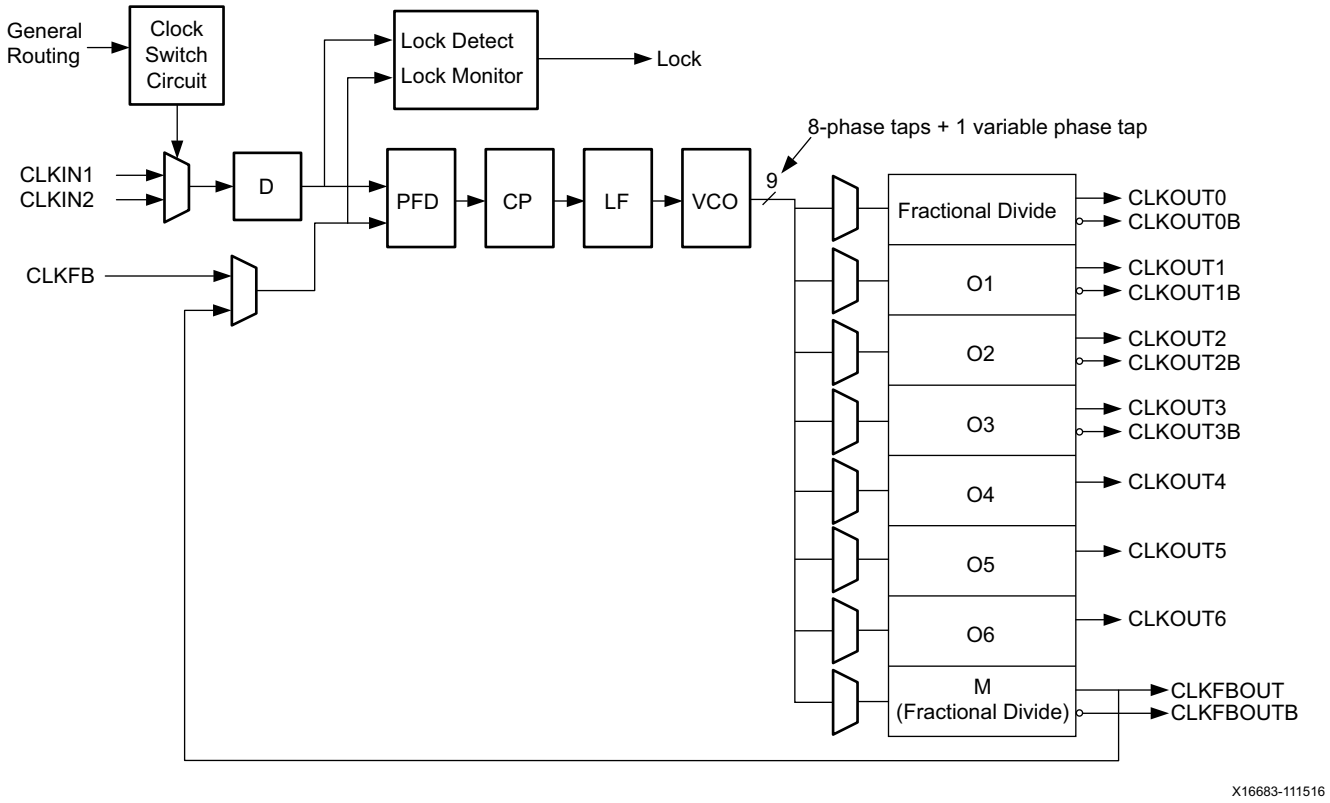

Figure 3-1: **Detailed MMCM Block Diagram**

General Usage Description

MMCM Primitives

The UltraScale device MMCM primitives, MMCME3_BASE and MMCME3_ADV, are shown in Figure 3-2. The UltraScale+ devices have the same primitives with an E4 instead of an E3. In this user guide, MMCME4_ADV is the same as the MMCME3_ADV, and MMCME4_BASE is the same as MMCME3_BASE.

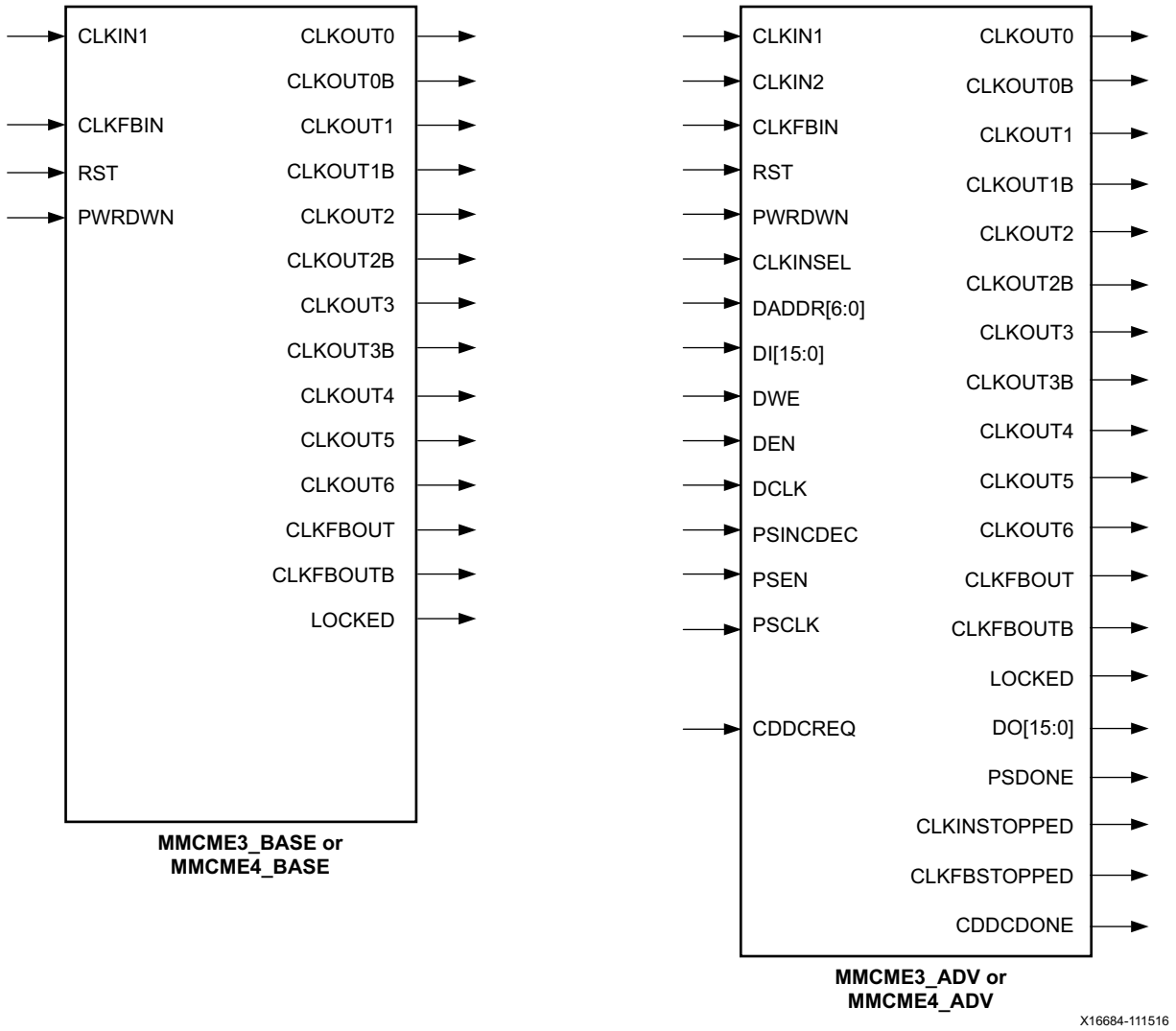

Figure 3-2: **MMCM Primitives**

MMCME3_BASE or MMCME4_BASE Primitives

The MMCME#_BASE primitives provide access to the most frequently used features of a stand-alone MMCM. Clock deskew, frequency synthesis, coarse phase shifting, and duty

cycle programming are available to use with the MMCME#_BASE. The ports are listed in Table 3-1.

Table 3-1: **MMCME#_BASE Ports**

Description	Ports
Clock input	CLKIN1, CLKFBIN
Control inputs	RST
Clock output	CLKOUT0 to CLKOUT6, CLKOUT0B to CLKOUT3B, CLKFBOUT, and CLKFBOUTB
Status and data outputs	LOCKED
Power control	PWRDWN

MMCME3_ADV and MMCME4_ADV Primitives

The MMCME#_ADV primitive provides access to all MMCME#_BASE features plus additional ports for clock switching, access to the dynamic reconfiguration port (DRP), and dynamic fine-phase shifting. The MMCME#_ADV ports are listed in Table 3-2.

Table 3-2: **MMCME#_ADV Ports**

Description	Ports
Clock input	CLKIN1, CLKIN2, CLKFBIN, DCLK, PSCLK
Control and data input	RST, CLKINSEL, DWE, DEN, DADDR, DI, PSINCDEC, PSEN, CDDCREQ
Clock output	CLKOUT0 to CLKOUT6, CLKOUT0B to CLKOUT3B, CLKFBOUT, and CLKFBOUTB
Status and data output	LOCKED, DO, DRDY, PSDONE, CLKINSTOPPED, CLKFBSTOPPED, CDDCDONE
Power control	PWRDWN

The MMCM is a mixed-signal block designed to support clock network deskew, frequency synthesis, and jitter reduction. These three modes of operation are discussed in more detail in this section. The VCO operating frequency can be determined by using the following relationship:

$$F_{VCO} = F_{CLKIN} \times \frac{M}{D} \quad \textit{Equation 3-1}$$

$$F_{OUT} = F_{CLKIN} \times \frac{M}{D \times O} \quad \textit{Equation 3-2}$$

where the M, D, and O counters are shown in Figure 3-2, page 37. The value of M corresponds to the CLKFBOUT_MULT_F setting, the value of D to the DIVCLK_DIVIDE, and O to the CLKOUT_DIVIDE.

The seven "O" counters can be independently programmed. For example, O0 can be programmed to do a divide-by-two while O1 is programmed for a divide-by-three. The only constraint is that the VCO operating frequency must be the same for all the output counters because a single VCO drives all the counters.

Clock Network Deskew

In many cases, designers do not want to incur the delay on a clock network in their I/O timing budget. Therefore, an MMCM is used to compensate for the clock network delay. This feature is supported in UltraScale architecture-based devices. A clock output matching the reference clock CLKIN frequency (always CLKFBOUT) is connected to a clock buffer of the same type driving the logic and fed back to the CLKFBIN feedback pin of the MMCM. The remaining outputs can still be used to divide the clock down for additionally synthesized frequencies. In this case, all output clocks have a defined phase relationship to the input reference clock.

Frequency Synthesis Only Using Integer Divide

The MMCMs can also be used for stand-alone frequency synthesis. In this application, the MMCM is not used to deskew a clock network. Rather, it generates an output clock frequency for other blocks. In this mode, the MMCM feedback paths are internal, which keeps all the routing local, minimizing the jitter. Figure 3-3 shows the MMCM configured as a frequency synthesizer. In this example, an external 33 MHz reference clock is available. The reference clock can be a crystal oscillator or the output of another MMCM. Setting the M counter to 32 makes the VCO oscillate at 1056 MHz (33 MHz x 32). The MMCM outputs are programmed to provide (for example) a 528 MHz processor clock, a 264 MHz gasket clock, a 176 MHz clock, a 132 MHz memory interface clock, a 66 MHz interface, and a 33 MHz interface. In this example, there are no required phase relationships between the reference clock and the output clocks, but there are required relationships between the output clocks.

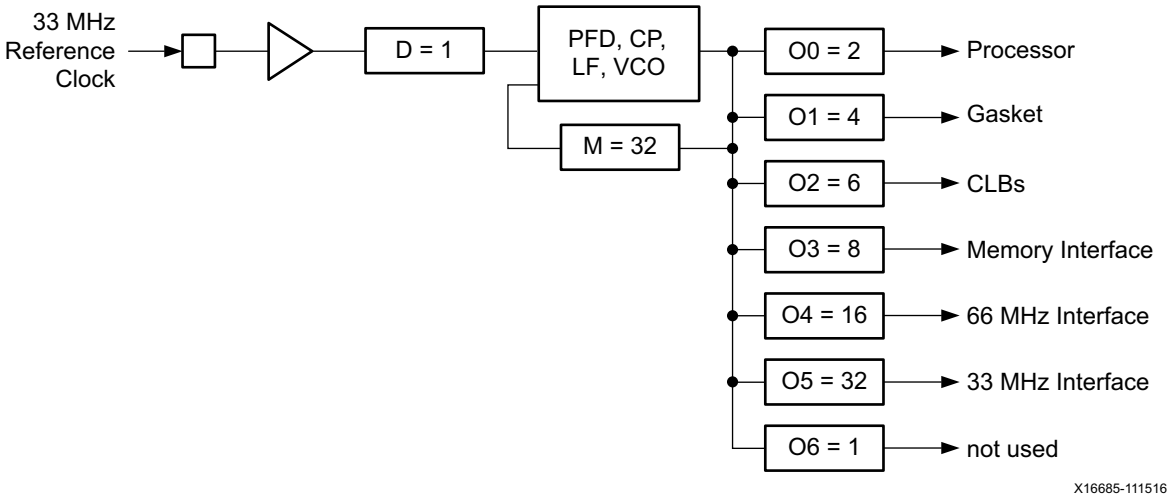

Figure 3-3: **MMCM as a Frequency Synthesizer**

Frequency Synthesis Using Fractional Divide in the MMCM

Devices support fractional (non-integer) divides in the CLKOUT0 output path. The resolution of the fractional divide is 1/8 or 0.125, effectively increasing the number of synthesizeable frequencies by a factor of eight. For example, if the CLKIN frequency is 100 MHz and the M divide value is set to 8, the VCO frequency is 800 MHz. CLKOUT0 can be used to further fractionally divide the 800 MHz VCO frequency (e.g., CLKOUT0_DIVIDE = 2.5, resulting in a 320 MHz output frequency).

When using the fractional divider, the duty cycle is not programmable for outputs used in the fractional mode.

Jitter Filter

MMCMs reduce the jitter inherent on a reference clock. The MMCM can be instantiated as a stand-alone function to only support filtering jitter from an external clock before it is driven into another block. As a jitter filter, it is usually assumed that the MMCM act as a buffer and regenerate the input frequency on the output (e.g., F_{IN} = 100 MHz, F_{OUT} = 100 MHz). In general, greater jitter filtering is possible by using the MMCM attribute BANDWIDTH set to Low. Setting the BANDWIDTH to Low can incur an increase in the static offset of the MMCM.

Limitations

The MMCM has some restrictions that must be adhered to. These are summarized in the MMCM electrical specifications in the UltraScale device data sheets [Ref 5]. In general, the major limitations are VCO operation range, input frequency, duty cycle programmability, and phase shift. In addition, there are connectivity limitations to other clocking elements (pins, GTs, and clock buffers). Cascading MMCMs can only occur through the clock routing network.

VCO Operating Range

The minimum and maximum VCO operating frequencies are defined in the electrical specification of the UltraScale device data sheets [Ref 5]. These values can also be extracted from the speed specification.

Minimum and Maximum Input Frequencies

The minimum and maximum CLKIN input frequencies are defined in the electrical specification of the UltraScale device data sheets [Ref 5].

Duty Cycle Programmability

Only discrete duty cycles are possible given a VCO operating frequency. Depending on the CLKOUT_DIVIDE value, a minimum and maximum range is possible with a step size that is also dependent on the CLKOUT_DIVIDE value. The Clocking Wizard tool gives the possible values for a given CLKOUT_DIVIDE.

Phase Shift

In many cases, there needs to be a phase shift between clocks. The MMCM has multiple options to implement phase shifting. Static phase shifting can be achieved by selecting one of the eight VCO output phases with additional fine phase shifting available in the CLKOUT output counters depending on the CLKOUT divide value. There is also an interpolated phase shifting capability in either fixed or dynamic mode. The MMCM phase shifting capabilities are very powerful, which can lead to complex scenarios. By using the Clocking Wizard, the allowable phase shift values are determined based on the MMCM configuration settings.

Static Phase Shift Mode (MMCM and PLL)

The static phase shift (SPS) resolution in time units is defined as:

$$SPS = \frac{1}{8F_{VCO}} period \text{ or } \frac{D}{8MF_{IN}} period \qquad \text{Equation 3-3}$$

Because the VCO can provide eight phase-shifted clocks at 45° each; always providing possible settings for 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315° of phase shift. The higher the VCO frequency is, the smaller the phase shift resolution. Because the VCO has a distinct operating range, it is possible to bound the phase shift resolution using from

$\frac{1}{8F_{VCOMIN}}$ *to* $\frac{1}{8F_{VCOMAX}}$ period.

Each CLKOUT output counter is individually programmable allowing each to have an additional phase shift resolution in degrees based on the phase of the VCO selected and the CLKOUT counter divide value. The granularity of the CLKOUT phase shift value can be calculated as 45°/CLKOUT_DIVIDE value. The maximum phase shift range is also determined by the CLKOUT_DIVIDE value. The maximum phase shift is 360° when CLKOUT_DIVIDE ≤ 64. When CLKOUT_DIVIDE is > 64, the maximum phase shift is:

$$Maximum\ Phase\ Shift = \left(\frac{63}{CLKOUT\_DIVIDE} \times 360\right) + (7 \times Phase\ Shift\ Value) \qquad \text{Equation 3-4}$$

It is possible to phase shift the CLKFBOUT feedback clock. In that case, all CLKOUT output clocks are negatively phase shifted with respect to CLKIN.

The two fractional counters (CLKFBOUT and CLKOUT0) also have static phase shift capability. A phase shift step is defined as:

$$SPS(frac) = \frac{360}{8 \times fractional\_divide\_value} or \frac{45}{fractional\_divide\_value} \quad \text{Equation 3-5}$$

For example, if the fractional divide value is 2.125, a static phase shift step is 360/(2.125 x 8) = 21.176 degrees.

Interpolated Fine Phase Shift in Fixed or Dynamic Mode in the MMCM

Interpolated fine phase shift (IFPS) mode in the MMCM has linear shift behavior independent of the CLKOUT_DIVIDE value, and the phase shift resolution only depends on the VCO frequency. In this mode, the output clocks can be rotated 360° round robin in linear increments of $\frac{1}{56F_{VCO}}$.

If the VCO runs at 600 MHz, the phase resolution is approximately (rounded) 30 ps, and at 1.6 GHz is approximately (rounded) 11 ps.

No initial phase shift value can be programmed during configuration. When using fine phase shift, no initial phase shift amount can be set. The phase always starts at zero and can then be dynamically incremented or decremented. The dynamic phase shift is controlled by the PS interface of the MMCME#_ADV. This phase shift mode equally affects all CLKOUT output clocks that are selected for this mode by setting the USE_FINE_PS attribute to TRUE. In interpolated fine phase shift mode, a clock must always be connected to the PSCLK pin of the MMCM. Regardless of the interpolated fine phase shift mode (fixed or dynamic) a clock is in, the clock must always be connected to the PSCLK pin of the MMCM. Each individual CLKOUT counter can independently either select the interpolated phase shift, the previously described static phase shift mode, or none. Fractional divide is not allowed in either fixed or dynamic interpolated fine phase shift mode. Fixed or dynamic phase shifting of the feedback path results in a negative phase shift of all output clocks with respect to CLKIN. The dynamic phase shift interface cannot be used when the phase shift mode is set to fixed.

Dynamic Phase Shift Interface in the MMCM

The MMCME#_ADV primitive provides three inputs and one output for dynamic fine phase shifting. Each CLKOUT and the CLKFBOUT divider can be individually selected for phase shifting. The attributes CLKOUT[0:6]_USE_FINE_PS and CLKFBOUT_USE_FINE_PS select the output clocks to be dynamically phase shifted. The dynamic phase shift amount is common to all the output clocks selected.

The variable phase shift is controlled by the PSEN, PSINCDEC, PSCLK, and PSDONE ports (Figure 3-4). The phase of the MMCM output clock(s) increments/decrements according to the interaction of PSEN, PSINCDEC, PSCLK, and PSDONE from the initial or previously performed dynamic phase shift. PSEN, PSINCDEC, and PSDONE are synchronous to PSCLK. When PSEN is asserted for one PSCLK clock period, a phase shift increment/decrement is initiated. When PSINCDEC is High, an increment is initiated and when PSINCDEC is Low, a decrement is initiated. Each increment adds to the phase shift of the MMCM clock outputs by 1/56th of the VCO period. Similarly, each decrement decreases the phase shift by 1/56th of the VCO period. PSEN must be active for one PSCLK period. PSDONE is High for exactly one clock period when the phase shift is complete. The number of PSCLK cycles is deterministic (12 PSCLK cycles). After initiating the phase shift by asserting PSEN and the completion of the phase shift signaled by PSDONE, the MMCM output clocks gradually drift from their original phase shift to an increment/decrement phase shift in a linear fashion. The completion of the increment or decrement is signaled when PSDONE asserts High. After PSDONE has pulsed High, another increment/decrement can be initiated. There is no maximum phase shift or phase shift overflow. An entire clock period (360°) can always be phase shifted regardless of frequency. When the end of the period is reached, the phase shift wraps around round-robin style.

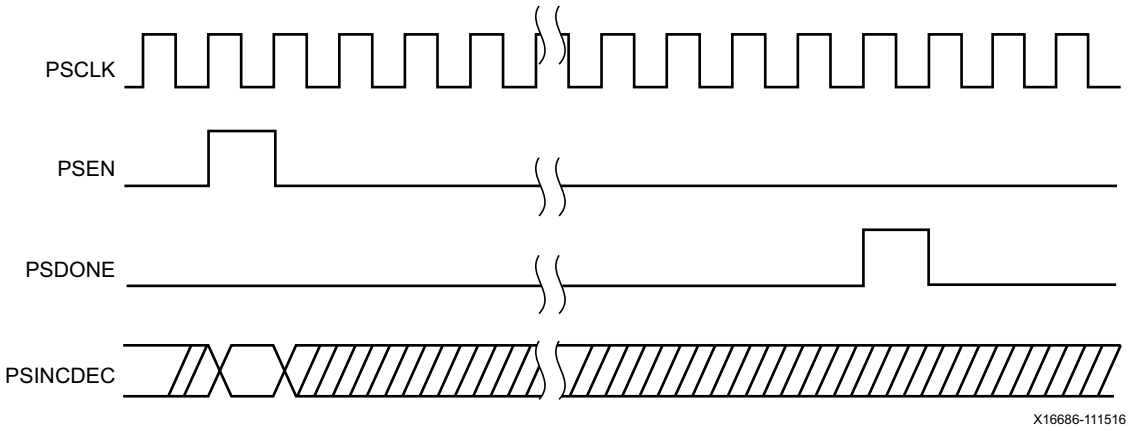

Figure 3-4: **Phase Shift Timing Diagram**

MMCM Clock Divide Dynamic Change

The Clock Divide Dynamic Change (CDDC) feature supports the dynamic change of the clock output dividers (CLKOUT[6:0]_DIVIDE) in conjunction with the DRP interface without the need for resetting the MMCM. Effectively, one or more of the MMCM output clock frequencies can be changed while leaving other output clocks untouched and running continuously. Two pins (CDDCREQ and CDDCDONE) control the handshaking. The application requests a change of output counter values (the CLKOUT_DIVIDE value) by asserting the CDDCREQ signal. New values are written through the standard DRP interface one port at a time and governed by standard DRP protocol (DEN, DWE, and DRDY). The DRP address of the CLKOUT[6:0] counter written to determines which output clocks are affected. After a CLKOUTx counter is written to, the associated clock output stops toggling. This can be followed by more changes (DRP writes) to other CLKOUT counters in an identical fashion. When all DRP writes have been completed (after the last DRDY), the CDDCREQ input must be deasserted, after which the affected output counter(s) are synchronously restarted. The MMCM acknowledges that the changes have taken place and the new output clocks (frequencies) are available for use by asserting CDDCDONE.

CLKOUT ports not affected by the CDDC change continue to function uninterrupted during this operation and maintain their phase relationship to each other (Figure 3-5). However, the output clocks (ports) that were changed via the CDDC procedure are not phase aligned (synchronized) to the other output clocks not affected by CDDCREQ. Clocks affected by CDDCREQ should not be used after the signal has been asserted because the output might glitch and the clocks stop toggling. This feature is not available in fractional mode.

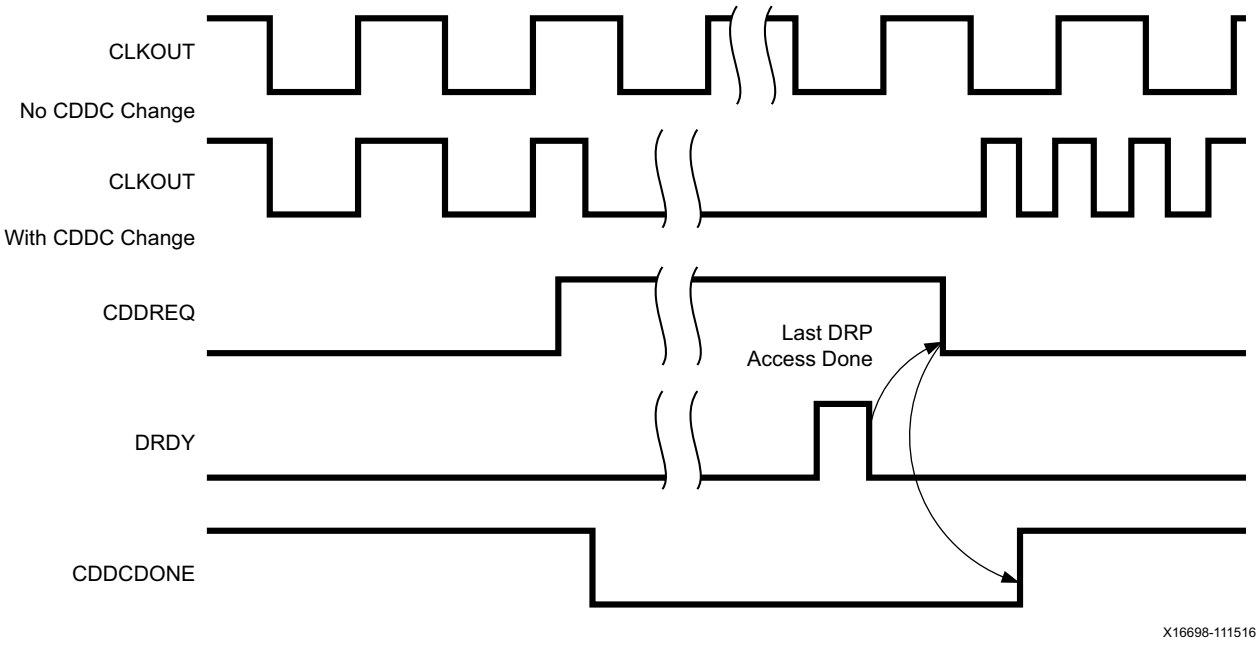

Figure 3-5: **CDDC Timing Diagram**

MMCM Counter Cascading

The CLKOUT6 divider (counter) can be cascaded with the CLKOUT4 divider. This provides the capability of an output divider that is larger than 128. CLKOUT6 feeds the input of the CLKOUT4 divider. There is a static phase offset between the output of the cascaded divider and all other output dividers.

MMCM Programming

Programming of the MMCM must follow a set flow to ensure configuration that guarantees stability and performance. This section describes how to program the MMCM based on certain design requirements. A design can be implemented in two ways, directly through the GUI interface (the Clocking Wizard) or implementing the MMCM through instantiation. Regardless of the method selected, the following information is necessary to program the MMCM:

- Reference clock period
- Output clock frequencies (up to seven maximum)
- Output clock duty cycle (default is 50%)
- Output clock phase shift in number of degrees relative to the original 0 phase of the clock
- Desired bandwidth of the MMCM (default is OPTIMIZED and the bandwidth is chosen in software)
- Compensation mode (automatically determined by the software)
- Reference clock jitter in UI (i.e., a percentage of the reference clock period)

Determine the Input Frequency

The first step is to determine the input frequency. This allows all possible output frequencies to be determined by using the minimum and maximum input frequencies to define the D counter range, the VCO operating range to determine the M counter range, and the output counter range. There can be a very large number of frequencies. When using integer divides, in the worst case there are 106 x 64 x 136 = 868,363 possible combinations. In reality, the total number of different frequencies is less because the entire range of the M and D counters cannot be realized, and there is overlap between the various settings.

As an example, consider F_{IN} = 100 MHz, F_{VCO} = between 600 MHz and 1600 MHz, and F_{PFD} = between 10 MHz and 550 MHz.

For a F_{PFDMIN} of 10 MHz, the value of D can only be between 1 and 10.

D_{MIN} (see Equation 3-6) = 1

D_{MAX} (see Equation 3-7) = 10

- For D = 1, M can only have a value between 6 and 16.
- For D = 2, M can only have a value between 12 and 32.
- For D = 3, M can only have a value between 24 and 64.

In addition, D = 1 M = 4 is a subset of D = 2 M = 8, D = 4 M = 16, and D = 8 M = 32 allowing these cases to be dropped. For this case, only D = 1, 3, 5, 6, 7, and 9 are considered because all other D values are subsets of these cases. This drastically reduces the number of possible output frequencies. The output frequencies are sequentially selected. The desired output frequency should be checked against the possible output frequencies generated. After the first output frequency is determined, an additional constraint can be imposed on the values of M and D. This can further limit the possible output frequencies for the second output frequency. This process is continued until all the output frequencies are selected.

The constraints used to determine the allowed M and D values are shown in these equations:

$$D_{MIN} = roundup\frac{f_{IN}}{f_{PFD\ MAX}} \quad \textit{Equation 3-6}$$

$$D_{MAX} = rounddown\frac{f_{IN}}{f_{PFD\ MIN}} \quad \textit{Equation 3-7}$$

$$M_{MIN} = roundup\left(\frac{f_{VCOMIN}}{f_{IN}} \times D_{MIN}\right) \quad \textit{Equation 3-8}$$

$$M_{MAX} = rounddown\left(\frac{f_{VCOMAX}}{f_{IN}} \times D_{MAX}\right) \quad \textit{Equation 3-9}$$

Determine the M and D Values

Determining the input frequency can result in several possible M and D values. The next step is to determine the optimum M and D values. The starting M value is first determined. This is based off the VCO target frequency, the ideal operating frequency of the VCO.

$$M_{IDEAL} = \frac{D_{MIN} \times f_{VCOMAX}}{f_{IN}} \quad \textit{Equation 3-10}$$

The goal is to find the M value closest to the ideal operating point of the VCO. The minimum D value is used to start the process. The goal is to make D and M values as small as possible while keeping f_{VCO} as high as possible.

MMCM Ports

Table 3-3 summarizes the MMCM ports.

Table 3-3: **MMCM Ports(1)**

Pin Name	I/O	Pin Description
CLKIN1	Input	General clock input. See CLKIN1 – Primary Reference Clock Input.
CLKIN2	Input	Secondary clock input for the MMCM reference clock. See CLKIN2 – Secondary Clock Input.
CLKFBIN	Input	Feedback clock input. See CLKFBIN – Feedback Clock Input.
CLKINSEL	Input	This signal controls the state of the clock input MUX, High = CLKIN1, Low = CLKIN2. CLKINSEL dynamically switches the MMCM reference clock. See CLKINSEL – Clock Input Select.
RST	Input	Asynchronous reset signal. The RST signal is an asynchronous reset for the MMCM. The MMCM synchronously re-enables itself when this signal is released (i.e., MMCM re-enabled). A reset is required when the input clock conditions change (e.g., frequency). See RST – Asynchronous Reset Signal.
PWRDWN	Input	Powers down instantiated but unused MMCMs. See PWRDWN – Power Down.
DADDR[6:0]	Input	The dynamic reconfiguration address (DADDR) input bus provides a reconfiguration address for the dynamic reconfiguration. When not used, all bits must be assigned zeros. See DADDR[6:0] – Dynamic Reconfiguration Address.
DI[15:0]	Input	The dynamic reconfiguration data input (DI) bus provides reconfiguration data. When not used, all bits must be set to zero. See DI[15:0] – Dynamic Reconfiguration Data Input.
DWE	Input	The dynamic reconfiguration write enable (DWE) input pin provides the write enable control signal to write the DI data into the DADDR address. When not used, it must be tied Low. See DWE – Dynamic Reconfiguration Write Enable.
DEN	Input	The dynamic reconfiguration enable (DEN) provides the enable control signal to access the dynamic reconfiguration feature. When the dynamic reconfiguration feature is not used, DEN must be tied Low. See DEN – Dynamic Reconfiguration Enable Strobe.
DCLK	Input	The DCLK signal is the reference clock for the dynamic reconfiguration port. See DCLK – Dynamic Reconfiguration Reference Clock.
PSCLK	Input	Phase shift clock. See PSCLK – Phase Shift Clock.
PSEN	Input	Phase shift enable. See PSEN – Phase Shift Enable.
PSINCDEC	Input	Phase shift increment/decrement control. See PSINCDEC – Phase Shift Increment/Decrement Control.
CLKOUT[0:6]	Output	User configurable clock outputs (0 through 6) that can be divided versions of the VCO phase outputs (user controllable) from 1 (bypassed) to 128. The output clocks are phase aligned to each other (unless phase shifted) and aligned to the input clock with a proper feedback configuration.
CLKOUT[0:3]B	Output	Inverted CLKOUT[0:3]. See CLKOUT[0:3]B – Inverted Output Clocks.
CLKFBOUT	Output	Dedicated MMCM feedback output. See CLKFBOUT – Dedicated MMCM Feedback Output.
CLKFBOUTB	Output	Inverted CLKFBOUT. See CLKFBOUTB – Inverted CLKFBOUT.

Table 3-3: **MMCM Ports[1] *(Cont'd)***

Pin Name	I/O	Pin Description
CLKINSTOPPED	Output	Status pin indicating that the input clock has stopped. See CLKINSTOPPED – Input Clock Status.
CLKFBSTOPPED	Output	Status pin indicating that the feedback clock has stopped. See CLKFBSTOPPED – Feedback Clock Status.
LOCKED	Output	An output from the MMCM that indicates when the MMCM has achieved phase alignment within a predefined window and frequency matching within a predefined PPM range. The MMCM automatically locks after power on. No extra reset is required. LOCKED is deasserted if the input clock stops or the phase alignment is violated (e.g., input clock phase shift). The MMCM must be reset after LOCKED is deasserted.
DO[15:0]	Output	The dynamic reconfiguration output bus provides MMCM data output when using dynamic reconfiguration. See DO[15:0] – Dynamic Reconfiguration Output Bus.
DRDY	Output	The dynamic reconfiguration ready output (DRDY) provides the response to the DEN signal for the MMCM's dynamic reconfiguration feature. See DRDY – Dynamic Reconfiguration Ready.
PSDONE	Output	Phase shift done. See PSDONE – Phase Shift Done.
CDDCREQ	Input	Requests a dynamic frequency change for selected clock outputs. See MMCM Clock Divide Dynamic Change, page 44.
CDDCDONE	Output	Signals that the dynamic frequency change is completed. See MMCM Clock Divide Dynamic Change, page 44.

Notes:

1. All control and status signals except PSINCDEC are active High.

TIP: *The port names generated by the clocking wizard can differ from the port names used on the primitive.*

MMCM Port Descriptions

CLKIN1 – Primary Reference Clock Input

CLKIN1 can be driven by a global clock I/O directly when in the same bank adjacent to the PHY tile.

CLKIN2 – Secondary Clock Input

CLKIN2 can be driven by a global clock I/O directly when in the same bank adjacent to the PHY tile.

CLKFBIN – Feedback Clock Input

CLKFBIN must be connected either directly to the CLKFBOUT for internal feedback, or to the CLKFBOUT via a BUFG for clock buffer feedback matching, or IBUFG (through a global clock pin for external deskew) or interconnect (not recommended). For clock alignment, the feedback path clock buffer type should match the forward clock buffer type.

IMPORTANT: *The internal compensation mode setting is determined by a direct connection (wire) from the CLKFBOUT to the CLKFBIN port in the source. However, synthesis optimizes this connection away such that the CLKFBOUT to CLKFBIN connection is removed from all subsequent representations in Vivado design suite. However, the INTERNAL compensation attribute attached to the MMCM/PLL indicates that the compensation is still internal to the MMCM/PLL.*

CLKFBOUT – Dedicated MMCM Feedback Output

For possible configuration of CLKFBOUT, see MMCM Use Models, page 58. CLKFBOUT can also drive logic if the feedback path contains a clock buffer.

CLKFBOUTB – Inverted CLKFBOUT

This signal should not be used for feedback. It provides an additional, inverted CLKFBOUT output clock. CLKFBOUTB can drive logic if the feedback path contains a clock buffer.

CLKINSEL – Clock Input Select

The CLKINSEL signal controls the state of the clock input multiplexers. High = CLKIN1, Low = CLKIN2 (see Reference Clock Switching, page 58). The MMCM must be held in RESET during clock switchover.

RST – Asynchronous Reset Signal

The RST signal is an asynchronous reset for the MMCM. The MMCM is synchronously re-enabled when this signal is deasserted.

PWRDWN – Power Down

This signal powers down instantiated but currently unused MMCMs. This mode can be used to save power for temporarily inactive portions of the design and/or MMCMs that are not active in certain system configurations. No MMCM power is consumed in this mode.

DADDR[6:0] – Dynamic Reconfiguration Address

The dynamic reconfiguration address (DADDR) input bus provides a reconfiguration address for dynamic reconfiguration. The address value on this bus specifies the 16 configuration bits that are written or read with the next DCLK cycle. When not used, all bits must be assigned zeros.

DI[15:0] – Dynamic Reconfiguration Data Input

The dynamic reconfiguration data input (DI) bus provides reconfiguration data. The value of this bus is written to the configuration cells. The data is presented in the cycle that DEN and DWE are active. The data is captured in a shadow register and written at a later time. DRDY indicates when the DRP port is ready to accept another write. When not used, all bits must be set to zero.

DWE – Dynamic Reconfiguration Write Enable

The dynamic reconfiguration write enable (DWE) input pin provides the write/read enable control signal to write the DI data into or read the DO data from the DADDR address. When not used, it must be tied Low.

DEN – Dynamic Reconfiguration Enable Strobe

The dynamic reconfiguration enable strobe (DEN) provides the enable control signal to access the dynamic reconfiguration feature and enables all DRP port operations. When the dynamic reconfiguration feature is not used, DEN must be tied Low.

DCLK – Dynamic Reconfiguration Reference Clock

The DCLK signal is the reference clock for the dynamic reconfiguration port. The rising edge of this signal is the timing reference for all other port signals. The setup time is specified in the UltraScale device data sheets [Ref 5]. There is no hold time requirement for the other input signals relative to the rising edge of the DCLK. The pin can be driven by an IBUF, IBUFG, BUFGCE, or BUFGCTRL. There are no dedicated connections to this clock input.

PSCLK – Phase Shift Clock

This input pin provides the source clock for the dynamic phase shift interface. All other inputs are synchronous to the positive edge of this clock. The pin can be driven by an IBUF, IBUFG, BUFG, or BUFGCE. There are no dedicated connections to this clock input.

PSEN – Phase Shift Enable

A dynamic (variable) phase shift operation is initiated by synchronously asserting this signal. PSEN must be activated for one cycle of PSCLK. After initiating a phase shift, the phase is gradually shifted until a High pulse on PSDONE indicates that the operation is complete. There are no glitches or sporadic changes during the operation. From the start to the end of the operation, the phase is shifted in a continuous analog manner.

PSINCDEC – Phase Shift Increment/Decrement Control

This input signal synchronously indicates if the dynamic phase shift is an increment or decrement operation (positive or negative phase shift). PSENCDEC is asserted High for increment and Low for decrement. There is no phase shift overflow associated with the dynamic phase shift operation. If 360° or more are shifted, the phase wraps around, starting at the original phase.

CLKOUT[0:6] – Output Clocks

These user-configurable clock outputs (CLKOUT0 through CLKOUT6) can be divided versions of the VCO phase outputs (user controllable) from 1 (bypassed) to 128. The input clock and output clocks can be phase aligned.

RECOMMENDED: *CLKOUT0 is first used to place the root clock point. In ZHOLD mode, it is used to set the compensation. Therefore, Xilinx recommends using CLKOUT0 as the main clock.*

For possible configurations, see MMCM Use Models, page 58. In the MMCM, CLKOUT0 and CLKFBOUT can be used in fractional divide mode. All CLKOUT outputs can be used in non-fractional mode to provide a static or dynamic phase shift. In fractional mode, only fixed phase shift is allowed. See Static Phase Shift Mode (MMCM and PLL), page 41 for more information.

CLKOUT[0:3]B – Inverted Output Clocks

This is the inverted (180° phase shift) CLKOUT[0:3].

CLKINSTOPPED – Input Clock Status

This is a status pin indicating that the input clock has stopped. This signal is asserted within two CLKFBOUT clock cycles of clock stoppage. The signal is deasserted after the clock has restarted and LOCKED is achieved, or the clock is switched to the alternate clock input and the MMCM has re-locked.

CLKFBSTOPPED – Feedback Clock Status

This is a status pin indicating that the feedback clock has stopped. CLKFBSTOPPED is asserted within one clock cycle of clock stoppage. The signal is deasserted after the feedback clock has restarted and the MMCM has re-locked.

LOCKED

This is an output from the MMCM used to indicate when the MMCM has achieved phase and frequency alignment of the reference clock and the feedback clock at the input pins. Phase alignment is within a predefined window and frequency matching within a predefined PPM range. The MMCM automatically locks after power on; no extra reset is required. LOCKED is deasserted within one PFD clock cycle if the input clock stops, the phase alignment is violated (e.g., input clock phase shift), or the frequency has changed. The MMCM must be reset when LOCKED is deasserted. The clock outputs should not be used prior to the assertion of LOCKED.

DO[15:0] – Dynamic Reconfiguration Output Bus

The dynamic reconfiguration output bus provides MMCM data output when using dynamic reconfiguration. If DWE is inactive while DEN is active at the rising edge of DCLK, this bus holds the content of the configuration cells addressed by DADDR. The DO bus must be captured on the rising edge of DCLK when DRDY is active. The DO bus value is held until the next DRP operation.

DRDY – Dynamic Reconfiguration Ready

The dynamic reconfiguration ready output (DRDY) provides the response to the DEN signal for the MMCM's dynamic reconfiguration feature. This signal indicates that a DEN/ DCLK operation has completed.

PSDONE – Phase Shift Done

The phase shift done output signal is synchronous to the PSCLK. When the current phase shift operation is completed, the PSDONE signal is asserted for one clock cycle indicating that a new phase shift cycle can be initiated.

CDDCREQ – Request a Clock Output Divide Change

This is a request signal for dynamically changing the output clock divide value and therefore the frequency. When asserted High, a request is sent to all affected counters and must stay asserted until the last change via the DRP has been completed.

CDDCDONE – Clock Output Change Done

This is an acknowledge signal from the MMCM that the output clock divide change is complete and the output is valid.

MMCM Attributes

Table 3-4 lists the attributes for the MMCME#_BASE and MMCME#_ADV primitives.

Table 3-4: **MMCM Attributes**

Attribute	Type	Allowed Values	Default	Description
BANDWIDTH	String	HIGH LOW OPTIMIZED(6)	OPTIMIZED	Specifies the MMCM programming algorithm affecting the jitter, phase margin, and other characteristics of the MMCM.
CLKOUT[1:6]_DIVIDE	Integer	1 to 128	1	Specifies the amount to divide the associated CLKOUT clock output if a different frequency is desired. This number, in combination with the CLKFBOUT_MULT_F and DIVCLK_DIVIDE values, determines the output frequency.
CLKOUT[0]_DIVIDE_F(1)	Integer or Real	1 to 128 or 2.000 to 128.000 in increments of 0.125	1	
CLKOUT[0:6]_PHASE	Real	–360.000 to 360.000 See equations in the Static Phase Shift Mode (MMCM and PLL) section.	0.0	Specifies the output phase relationship of the associated CLKOUT clock output in number of degrees offset (i.e., 90 indicates a 90° or ¼ cycle offset phase offset while 180 indicates a 180° offset or ½ cycle phase offset).
CLKOUT[0:6]_DUTY_CYCLE	Real	0.01 to 0.99	0.50	Specifies the duty cycle of the associated CLKOUT clock output as a percentage (i.e., 0.50 generates a 50% duty cycle).
CLKFBOUT_MULT_F(1)	Integer or Real	2 to 64 or 2.000 to 64.000 in increments of 0.125	5	This CLKFBOUT_MULT_F range applies to UltraScale devices. Specifies the amount to multiply all CLKOUT clock outputs if a different frequency is desired. This number, in combination with the associated CLKOUT#_DIVIDE value and DIVCLK_DIVIDE value, determines the output frequency.
CLKFBOUT_MULT_F(1)	Integer or Real	2 to 128 or 2.000 to 128.000 in increments of 0.125	5	This CLKFBOUT_MULT_F range applies to UltraScale+ devices. Specifies the amount to multiply all CLKOUT clock outputs if a different frequency is desired. This number, in combination with the associated CLKOUT#_DIVIDE value and DIVCLK_DIVIDE value, determines the output frequency.
DIVCLK_DIVIDE	Integer	1 to 106	1	Specifies the division ratio for all output clocks with respect to the input clock. Effectively divides the CLKIN going into the PFD.

Table 3-4: **MMCM Attributes *(Cont'd)***

Attribute	Type	Allowed Values	Default	Description
CLKFBOUT_PHASE	Real	–360.000 to 360.000	0.0	Specifies the phase offset in degrees of the clock feedback output. Shifting the feedback clock results in a negative phase shift of all output clocks to the MMCM.
REF_JITTER1 REF_JITTER2	Real	0.000 to 0.999	0.010	Allows specification of the expected jitter on the reference clock to better optimize MMCM performance. A bandwidth setting of OPTIMIZED attempts to choose the best parameter for input clocking when unknown. If known, the value provided should be specified in terms of the unit interval (UI) (the maximum peak-to-peak value) of the expected jitter on the input clock.
CLKIN1_PERIOD	Real	0.968 to 100.000	0.000	Specifies the input period in ns to the MMCM CLKIN1 input. Resolution is down to the ps. This information is mandatory and must be supplied.
CLKIN2_PERIOD	Real	0.968 to 100.000	0.000	Specifies the input period in ns to the MMCM CLKIN2 input. Resolution is down to the ps. This information is mandatory and must be supplied.
CLKFBOUT_USE_FINE_PS (2)	String	FALSE, TRUE	FALSE	CLKFBOUT counter variable fine phase shift enable.
CLKOUT0_USE_FINE_PS (2)	String	FALSE, TRUE	FALSE	CLKOUT0 counter variable fine phase shift enable. CLKOUT0_DIVIDE must be an integer. Therefore, fractional divide is not allowed.
CLKOUT[1:6]_USE_FINE_PS (2)	String	FALSE, TRUE	FALSE	CLKOUT[1:6] variable fine phase shift enable.
STARTUP_WAIT	String	FALSE, TRUE	FALSE	Wait during the configuration start-up cycle for the MMCM to lock.
CLKOUT4_CASCADE	String	FALSE, TRUE	FALSE	Cascades the output divider (counter) CLKOUT6 into the input of the CLKOUT4 divider for an output clock divider that is greater than 128, effectively providing a total divide value of 16,384.

Table 3-4: **MMCM Attributes *(Cont'd)***

Attribute	Type	Allowed Values	Default	Description
COMPENSATION	String	AUTO[3], ZHOLD, EXTERNAL, INTERNAL[5], BUF_IN	AUTO	Clock input compensation. Must be set to AUTO. Defines how the MMCM feedback is configured. ZHOLD: Indicates the MMCM is configured to provide a negative hold time at the I/O registers. EXTERNAL: Indicates a network external to the device is being compensated. INTERNAL: Indicates the MMCM is using its own internal feedback path so no delay is being compensated. BUF_IN: Indicates that the configuration does not match with the other compensation modes. The CLKIN and CLKFBIN pins are aligned in a way that a delay in the feedback path is compensated with respect to CLKIN.
SS_EN	String	FALSE, TRUE	FALSE	Enables spread spectrum generation.
SS_MODE	String	DOWN_LOW, DOWN_HIGH, CENTER_LOW, CENTER_HIGH	CENTER_HIGH	Controls the spread spectrum frequency deviation and the spread type.
SS_MOD_PERIOD	Integer	4000–40000	10000	Specifies the spread spectrum modulation period (ns).

Notes:

1. The Vivado tools round up or down to the nearest multiple of 0.125 if the value is not specified as an exact 1/8th fraction.
2. When using the variable fine phase shift, the initial phase shift value is always zero and cannot be preset to a static, initial phase.
3. The COMPENSATION attribute values are documented for informational purpose only. The Vivado tools automatically select the appropriate compensation based on circuit topology. Do not manually select a compensation value, leave the attribute at the default value.
4. The specifications for the VCO frequencies MMCM_FVCOMIN/MMCM_FVCOMAX and minimum out frequency MMCM_FOUTMIN are different for the UltraScale and UltraScale+ families. Consult the appropriate data sheets.
5. The direct source code connection (wire) from CLKFBOUT to CLKFBIN is optimized away during synthesis.
6. When using an SEM-IP in Ultrascale devices only, additional noise is coupled into VCO of MMCM and PLL. This results in higher TIE jitter value as described in AR:71314. Refer to the AR for guidance and mitigation techniques. To resolve the issue of TIE jitter for SEM-IP, there are two new configurable properties: BITSTREAM.MMCM.BANDWIDTH and BITSTREAM.PLL.BANDWIDTH. If the properties are set to POSTCRC, each MMCM instance that has the BANDWIDTH attribute set to OPTIMIZED, or to a PLL instance with an implicit BANDWIDTH=OPTIMIZED attribute, will get configured to POSTCTRC bandwidth settings. Refer to the *Vivado Design Suite User Guide* [Ref 13]for BITSTREAM property information.

MMCM Clock Input Signals

The possible clock sources for the MMCM include:

- IBUF – Global clock input buffer. The MMCM compensates the delay of this path. IBUF represents a global clock pin in the same region. The IBUF must be located at a global clock pin location.
- BUFGCTRL – Internal global clock buffer. The MMCM does not compensate the delay of this path.
- BUFGCE – Global clock buffer. The MMCM does not compensate the delay of this path.

Counter Control

The MMCM output counters provide a wide variety of synthesized clocks using a combination of DIVIDE, DUTY_CYCLE, and PHASE. Figure 3-6 illustrates how the counter settings impact the counter output.

The top waveform represents the output from the VCO.

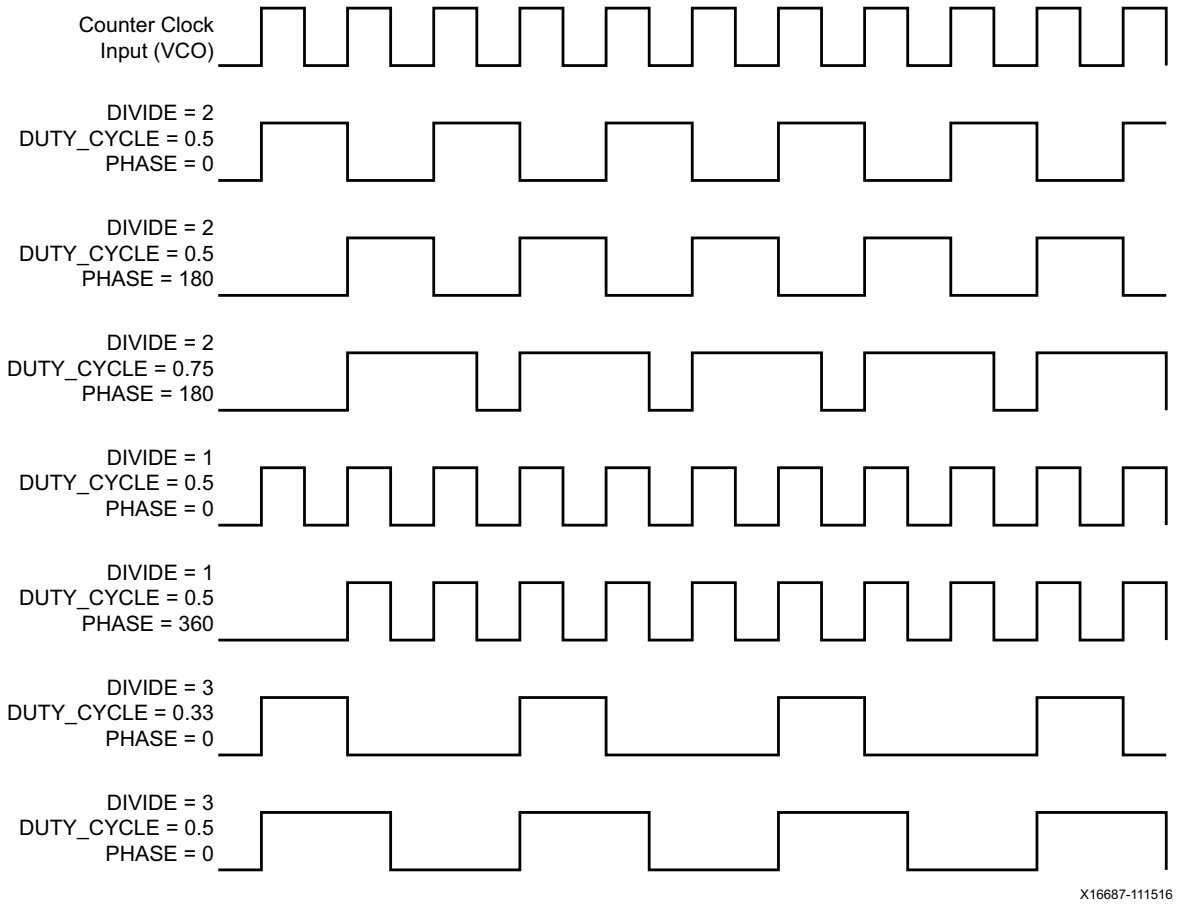

Figure 3-6: **Output Counter Clock Synthesis Examples**

Detailed VCO and Output Counter Waveforms

Figure 3-7 shows the eight VCO phase outputs and four different counter outputs. Each VCO phase is shown with the appropriate start-up sequence. The phase relationship and start-up sequence are guaranteed to ensure the correct phase is maintained. This means the rising edge of the 0° phase happens before the rising edge of the 45° phase. The O0 counter is programmed to do a simple divide-by-two with the 0° phase tap as the reference clock. The O1 counter is programmed to do a simple divide-by-two but uses the 180° phase tap from the VCO. This counter setting can be used to generate a clock for a DDR interface where the reference clock is edge aligned to the data transition. The O2 counter is programmed to do a divide-by-three. The O3 output has the same programming as the O2 output except the phase is set for a one cycle delay. Phase shifts greater than one VCO period are possible.

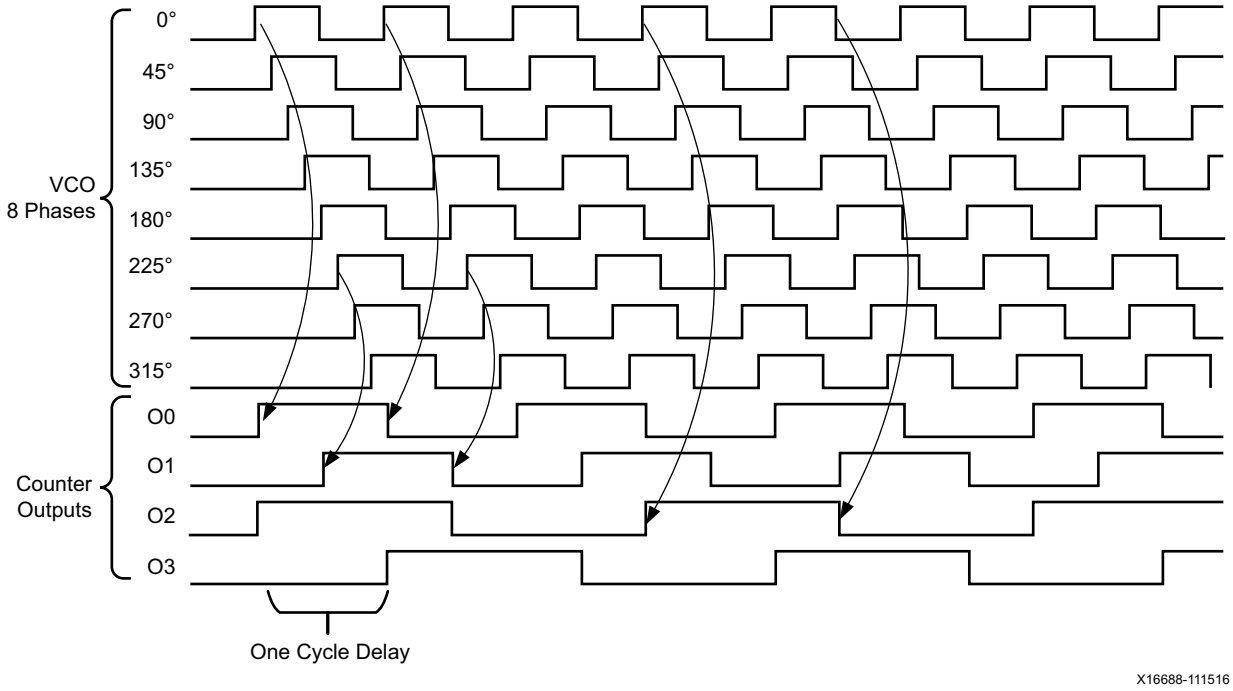

Figure 3-7: **Selecting VCO Phases**

If the MMCM is configured to provide a certain phase relationship and the input frequency is changed, this phase relationship is also changed because the VCO frequency changes and therefore the absolute shift in picoseconds changes. This aspect must be considered when designing with the MMCM. When an important aspect of the design is to maintain a certain phase relationship among various clock outputs, (e.g., CLK and CLK90), this relationship is maintained regardless of the input frequency.

All O counters can be equivalent; anything O0 can do, O1 can do. The O0 counter has the additional capability to be used in fractional divide mode. The MMCM outputs are flexible when connecting to the global clock network because they are identical. In most cases, this

level of detail is imperceptible because the software and Clocking Wizard determine the proper settings through the MMCM attributes and Wizard inputs.

Reference Clock Switching

The MMCM reference clock can be dynamically switched by using the CLKINSEL pin. The switching is done asynchronously. After the clock switches, the MMCM is likely to lose LOCKED and automatically lock onto the new clock. Therefore, after the clock switches, the MMCM must be reset. The MMCM clock MUX switching is shown in Figure 3-8. The CLKINSEL signal directly controls the MUX. No synchronization logic is present.

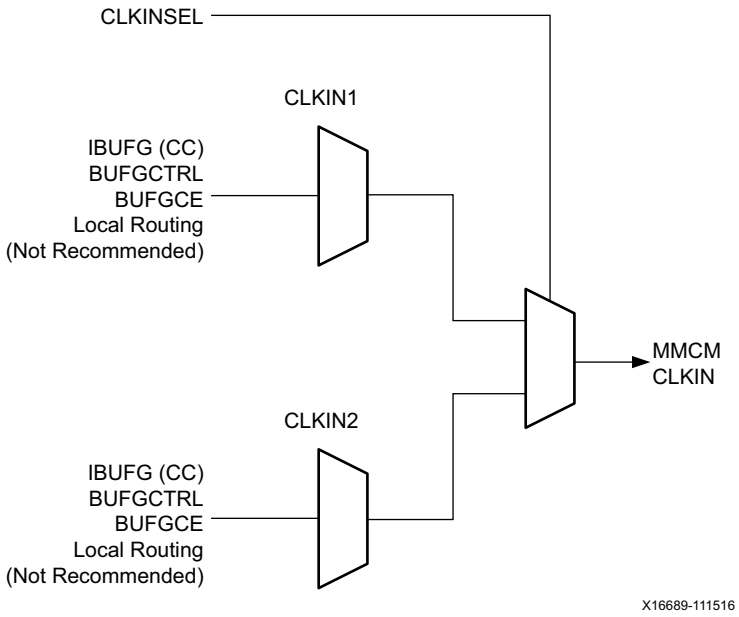

Figure 3-8: **Input Clock Switching**

Missing Input Clock or Feedback Clock

When the input clock or feedback clock is lost, the CLKINSTOPPED or CLKFBSTOPPED status signal is asserted. The MMCM deasserts the LOCKED signal. After the clock returns, the CLKINSTOPPED signal is deasserted and a RESET must be applied.

MMCM Use Models

The examples in this section show the MMCM. There are several methods to design with the MMCM. The Clocking Wizard in the Vivado tools can assist with generating the various MMCM parameters. Additionally, the MMCM can be manually instantiated as a component. It is also possible for the MMCM to be merged with an IP core. The IP core would contain and manage the MMCM.

Clock Network Deskew

One of the predominant uses of the MMCM is for clock network deskew. Figure 3-9 shows the MMCM in this mode. The clock output from one of the CLKOUT counters is used to drive logic within the device and/or the I/Os. The feedback counter is used to control the exact phase relationship between the input clock and the output clock (if, for example a 90° phase shift is required). The associated clock waveforms are shown to the right for the case where the input clock and output clock need to be phase aligned. The configuration in Figure 3-9 is the most flexible, but it does require two global clock networks.

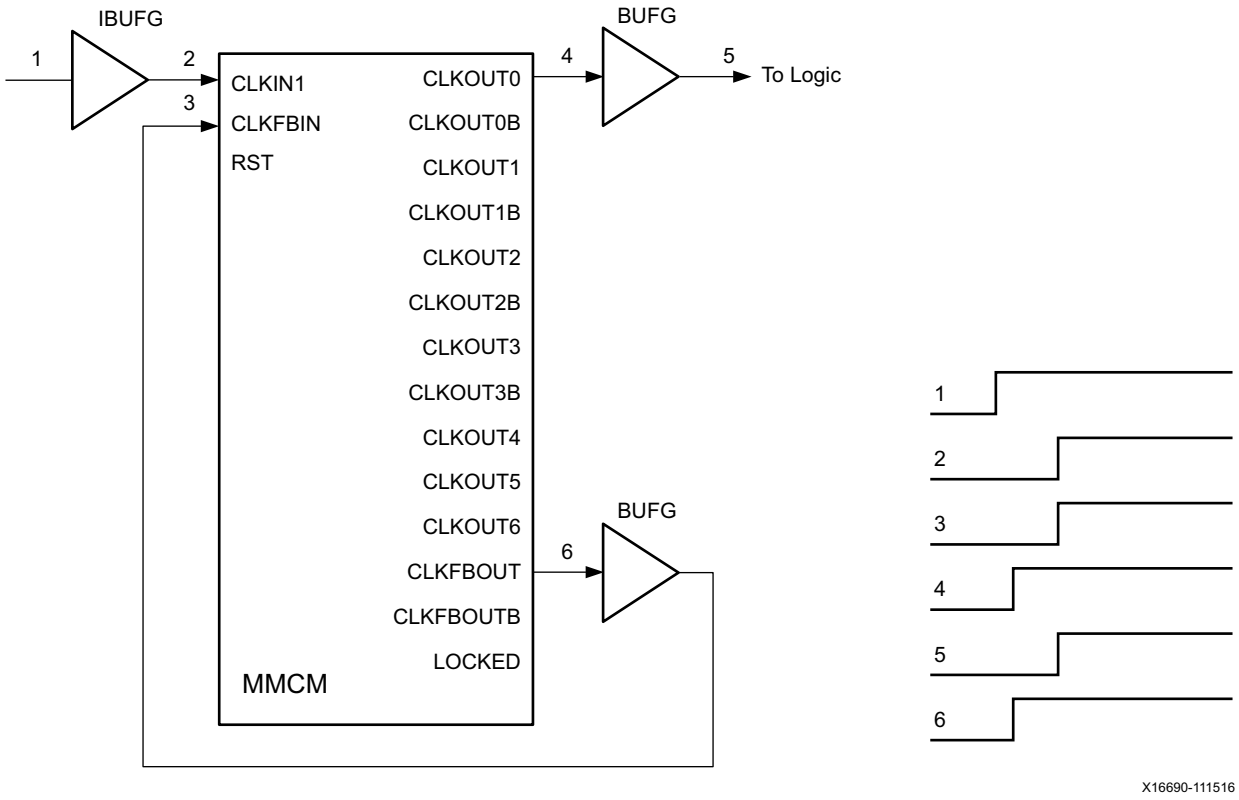

Figure 3-9: **Global Clock Network Deskew Using Two BUFGs**

There are certain restrictions on implementing the feedback. The CLKFBOUT output can be used to provide the feedback clock signal. When an MMCM is driving both BUFGs and BUFGCTRL, only one of the clock buffers that is also used in the feedback path is deskewed. The fundamental restriction is that both input frequencies to the PFD must be identical. Therefore, this relationship must be met:

$$\frac{f_{IN}}{D} = f_{FB} = \frac{f_{VCO}}{M} \qquad \textit{Equation 3-11}$$

As an example, if f_{IN} is 166 MHz, D = 1, M = 6, and O = 2, then VCO is 996 MHz and the clock output frequency is 498 MHz. Because the M value in the feedback path is 6, both input frequencies at the PFD are 166 MHz.

Another more complex scenario has an input frequency of 66.66 MHz and D = 2, M = 30, and O = 4. The VCO frequency in this case is 1000 MHz and the CLKOUT output frequency is 250 MHz. Therefore, the feedback frequency at the PFD is 1000/30 or 33.33 MHz, matching the 66.66 MHz/2 input clock frequency at the PFD.

MMCM with Internal Feedback

The MMCM feedback can be internal to the MMCM when the MMCM is used as a synthesizer or jitter filter, and there is no required phase relationship between the MMCM input clock and the MMCM output clock. The MMCM performance increases because the feedback clock is not subjected to noise on the core supply since it never passes through a block powered by this supply. However, noise introduced on the CLKIN signal and the BUFG are still present (Figure 3-10).

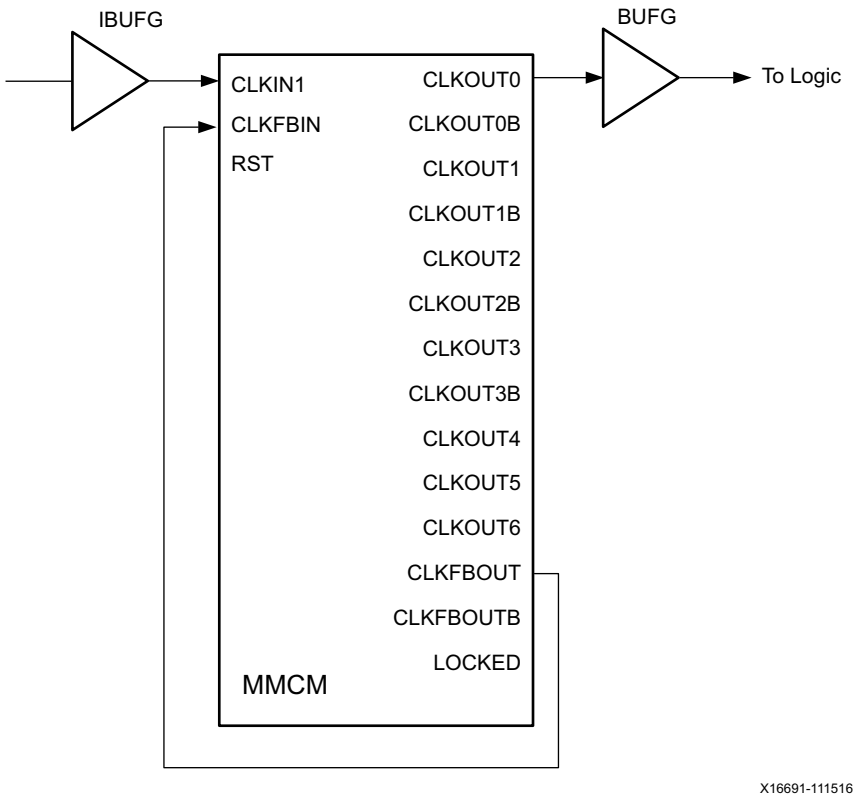

Figure 3-10: **MMCM with Internal Feedback**

Zero Delay Buffer

The MMCM can also be used to generate a zero delay buffer clock. A zero delay buffer can be useful for applications where there is a single clock signal fanout to multiple destinations with a low skew between them. This configuration is shown in Figure 3-11. Here, the feedback signal drives off chip and the board trace feedback is designed to match the trace to the external components. In this configuration, it is assumed that the clock edges are aligned at the input of the UltraScale device and the input of the external component. The input clock buffers for CLKIN and CLKFBIN must be in the same bank.

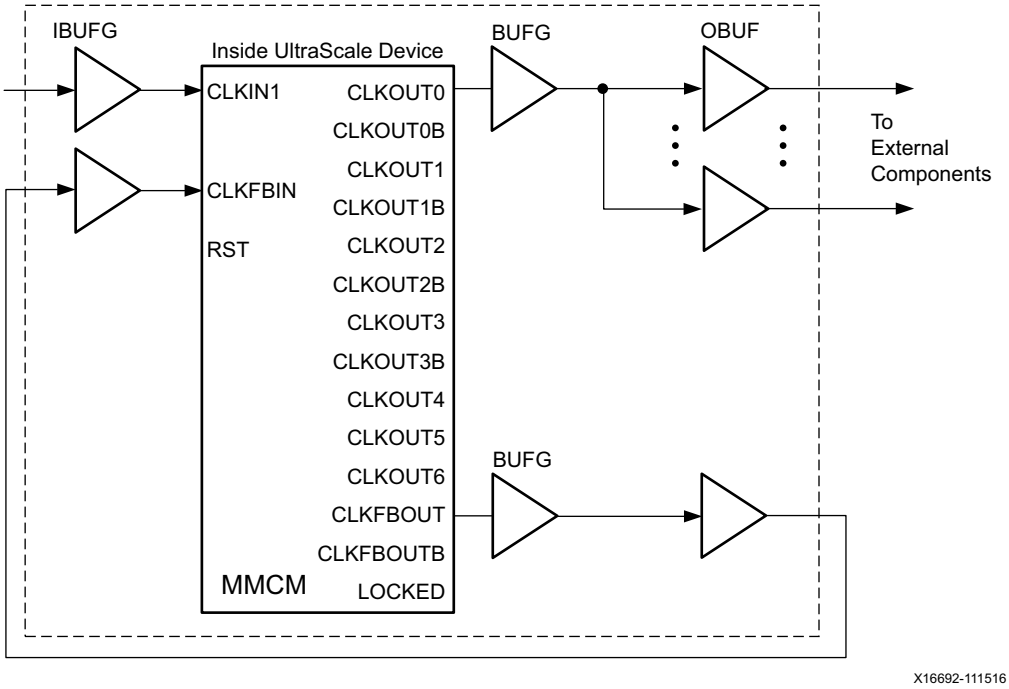

Figure 3-11: **Zero Delay Buffer**

In some cases, precise alignment cannot occur because of the difference in loading between the input capacitance of the external component and the feedback path capacitance of the UltraScale device. For example, the external components can have an input capacitance of 1 pF to 4 pF while the part has an input capacitance as specified in the UltraScale device data sheets [Ref 5]. There is a difference in the signal slope, which is basically skew. Designers should be aware of this effect to ensure timing.

CMT to CMT Connection

The MMCM can be cascaded through the routing resources only. There is no compensation for routing delays.

Spread-Spectrum Clock Generation

Spread-spectrum clock generation (SSCG) is widely used by manufacturers of electronic devices to reduce the spectral density of the electromagnetic interference (EMI) generated by these devices. Manufacturers must ensure that levels of electromagnetic energy emitted do not interfere with the operation of other nearby electronic devices. For example, the clarity of a phone call should not degrade when the phone is next to a video display. In the same way, the display should not be affected when the phone is used.

Electromagnetic compatibility (EMC) regulations are used to control the noise or EMI that causes these disturbances. Typical solutions for meeting EMC requirements involve adding expensive shielding, ferrite beads, or chokes. These solutions can adversely impact the cost of the final product by complicating PCB routing and forcing longer product development cycles.

SSCG spreads the electromagnetic energy over a large frequency band to effectively reduce the electrical and magnetic field strengths measured within a narrow window of frequencies. The peak electromagnetic energy at any one frequency is reduced by modulating the SSCG output.

The MMCME# can generate a spread-spectrum clock from a standard fixed frequency oscillator when SS_EN is set to TRUE (see Figure 3-12). Within the MMCME#, the VCO frequency is modulated along with CLKFBOUT and CLKOUT[6:4,1,0]. Clock outputs CLKOUT[3:2] are used to control the modulation period and are not available for general use. As long as the clock frequency is adjusted slowly, the spread-spectrum does not affect the period jitter of the MMCME#.

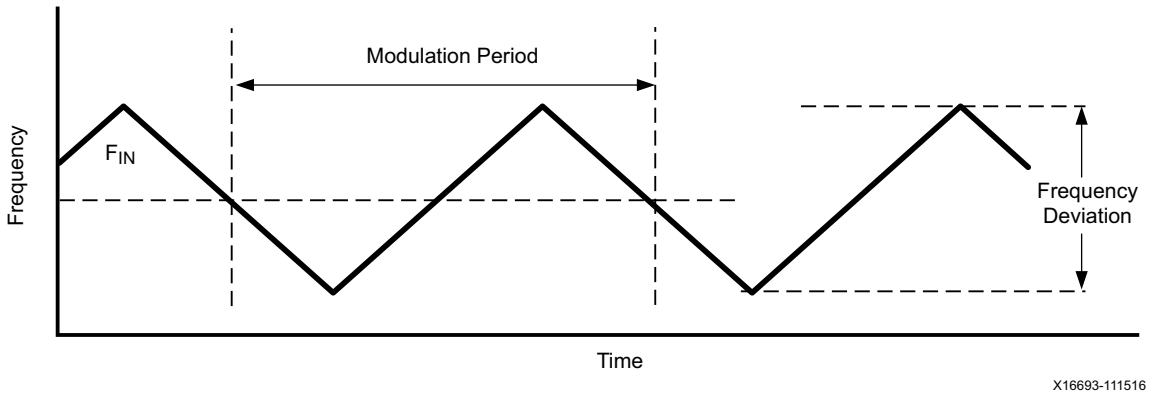

Figure 3-12: **Center-Spread Modulation**

Adjusting the modulation period SS_MOD_PERIOD allows you to direct the tools to select the closest modulation period based on the MMCME# settings. The spread-spectrum modulation reduces EMI as long as the modulation frequencies are higher than the audible frequency range of 30 KHz. Typically, lower modulation frequencies are preferred to minimize the impact of the introduction of spread-spectrum.

Increasing the frequency deviation with SS_MODE (CENTER_HIGH or DOWN_HIGH) increases the overall EMI reduction, but care must be taken to ensure that the increased range of frequencies does not affect the overall system operation (see Figure 3-13). Because the spread-spectrum clock and the input clock are operating at different frequencies, any data being transferred between the clock domains should use an asynchronous FIFO to ensure that data is not lost. Increasing the frequency deviation requires a larger FIFO.

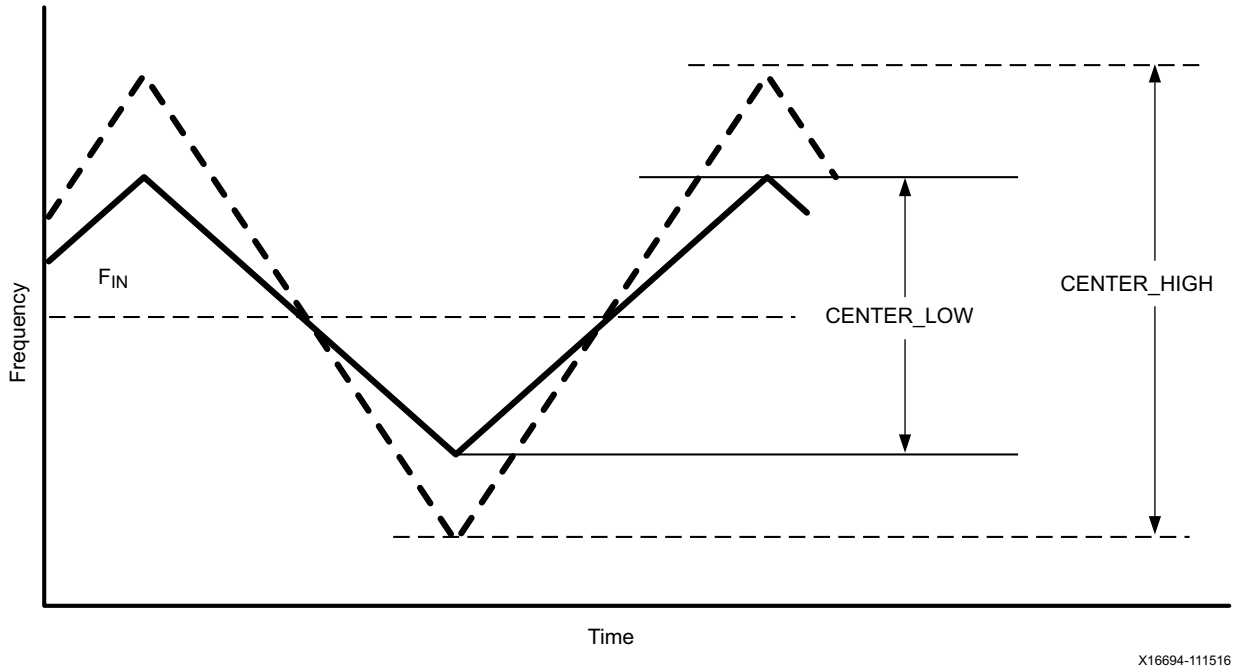

Figure 3-13: **Center-Spread Modulation (CENTER_LOW vs. CENTER_HIGH)**

Another design trade-off is the decision to use a center spread or down spread. Selecting SS_MODE (DOWN_HIGH, DOWN_LOW) spreads the frequencies to lower frequencies as shown in Figure 3-14. DOWN_HIGH has similar frequency deviation to CENTER_LOW.

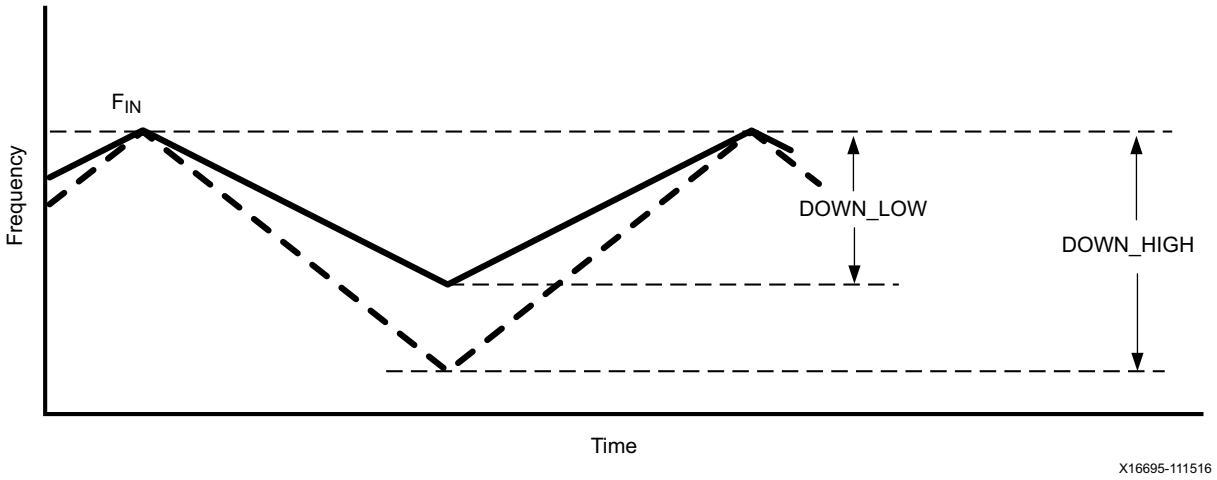

Figure 3-14: **Down-Spread Modulation**

The decision to use down spread is often the result of considering the timing analysis impact of spread-spectrum. When using a spread-spectrum clock, the design must meet timing at the highest frequency in the frequency deviation. Therefore, if a 100 MHz clock with SS_MODE (CENTER_LOW) produces a 3% (±1.5%) center spread, the 100 MHz clock with 3% center spread must pass timing analysis as a 101.5 MHz clock. However, if SS_MODE (DOWN_HIGH) produces a 3% down spread, the input frequency is the highest frequency within the frequency deviation. Consequently, for a 100 MHz clock with 3% down spread, the down-spread clock would continue to be analyzed by timing analysis as a 100 MHz clock.

Table 3-5: **Manual SS Timing Adjustment Using Input Frequency for UltraScale Devices**

Parameter	Input Frequency (MHz)	M	Input Frequency Adjustment (FIN_SS)
SS_MODE(CENTER_HIGH)	$25 < F_{IN} < 35$	M = 28	FIN_SS = F_{IN} x 56/55
	$35 < F_{IN} < 50$	M = 21	FIN_SS = F_{IN} x 42/41
		M = 22	FIN_SS = FIN x 44/43
	$50 < F_{IN} < 75$	M = 28	FIN_SS = F_{IN} x 56/55
	$75 < F_{IN} < 150$	M = 21	FIN_SS = F_{IN} x 42/41
		M = 22	FIN_SS = F_{IN} x 44/43
SS_MODE (CENTER_LOW)	$25 < F_{IN} < 35$	M = 56	FIN_SS = F_{IN} x 112/111
	$35 < F_{IN} < 50$	M = 42	FIN_SS = F_{IN} x 84/83
		M= 44	FIN_SS = F_{IN} x 88/87
	$50 < F_{IN} < 75$	M = 56	FIN_SS = F_{IN} x 112/111
	$75 < F_{IN} < 150$	M = 42	FIN_SS = F_{IN} x 84/83
		M = 44	FIN_SS = F_{IN} x 88/87

Table 3-5: **Manual SS Timing Adjustment Using Input Frequency for UltraScale Devices *(Cont'd)***

Parameter	Input Frequency (MHz)	M	Input Frequency Adjustment (FIN_SS)
SS_MODE (DOWN_HIGH)	25 < F_{IN} < 35	M=28	FIN_SS = F_{IN}
	35 < F_{IN} < 50	M = 21, 22	FIN_SS = F_{IN}
	50 < F_{IN} < 75	M = 28	FIN_SS = F_{IN}
	75 < F_{IN} < 100	M = 21, 22	FIN_SS = F_{IN}
	100 < F_{IN} < 150	M = 21, 22	FIN_SS = F_{IN}
SS_MODE (DOWN_LOW)	25 < F_{IN} < 35	M = 56	FIN_SS = F_{IN}
	35 < F_{IN} < 50	M = 42, 44	FIN_SS = F_{IN}
	50 < F_{IN} < 75	M = 56	FIN_SS = F_{IN}
	75 < F_{IN} < 100	M = 42, 44	FIN_SS = F_{IN}
	100 < F_{IN} < 150	M = 42, 44	FIN_SS = F_{IN}

Table 3-6: **Manual SS Timing Adjustment Using Input Frequency for UltraScale+ Devices**

Parameter	Input Frequency (MHz)	M	D	Input Frequency Adjustment (FIN_SS)
SS_MODE (CENTER_HIGH)	30 < F_{IN} < 40	M = 28	D = 1	FIN_SS = F_{IN} x 56/55
	40 < F_{IN} < 60	M = 21	D = 1	FIN_SS = F_{IN} x 42/41
		M = 22	D = 1	FIN_SS = FIN x 44/43
	60 < F_{IN} < 80	M = 28	D = 2	FIN_SS = F_{IN} x 56/55
	80 < F_{IN} < 120	M = 21	D = 2	FIN_SS = F_{IN} x 42/41
		M = 22	D = 2	FIN_SS = F_{IN} x 44/43
	120 < F_{IN} < 150	M = 21	D = 3	FIN_SS = F_{IN} x 42/41
		M = 22	D = 3	FIN_SS = F_{IN} x 44/43
SS_MODE (CENTER_LOW)	30 < F_{IN} < 40	M = 56	D = 2	FIN_SS = F_{IN} x 112/111
	40 < F_{IN} < 60	M = 42	D = 2	FIN_SS = F_{IN} x 84/83
		M = 44	D = 2	FIN_SS = F_{IN} x 88/87
	60 < F_{IN} < 80	M = 56	D = 4	FIN_SS = F_{IN} x 112/111
	80 < F_{IN} < 120	M = 42	D = 4	FIN_SS = F_{IN} x 84/83
		M = 44	D = 4	FIN_SS = F_{IN} x 88/87
	120 < F_{IN} < 150	M = 42	D = 6	FIN_SS = F_{IN} x 84/83
		M = 44	D = 6	FIN_SS = F_{IN} x 88/87
SS_MODE (DOWN_HIGH)	35 < F_{IN} < 40	M=28	D = 1	FIN_SS = F_{IN}
	40 < F_{IN} < 60	M = 21, 22	D = 1	FIN_SS = F_{IN}
	60 < F_{IN} < 80	M = 28	D = 2	FIN_SS = F_{IN}
	80 < F_{IN} < 120	M = 21, 22	D = 2	FIN_SS = F_{IN}
	120 < F_{IN} < 150	M = 21, 22	D = 3	FIN_SS = F_{IN}

Table 3-6: **Manual SS Timing Adjustment Using Input Frequency for UltraScale+ Devices *(Cont'd)***

Parameter	Input Frequency (MHz)	M	D	Input Frequency Adjustment (FIN_SS)
SS_MODE (DOWN_LOW)	$35 < F_{IN} < 40$	M = 56	D = 2	FIN_SS = F_{IN}
	$40 < F_{IN} < 60$	M = 42, 44	D = 2	FIN_SS = F_{IN}
	$60 < F_{IN} < 80$	M = 56	D = 4	FIN_SS = F_{IN}
	$80 < F_{IN} < 120$	M = 42, 44	D = 4	FIN_SS = F_{IN}
	$120 < F_{IN} < 150$	M = 42, 44	D = 6	FIN_SS = F_{IN}

For a 25 MHz input clock, the new timing constraints would be:

- SS_MODE(CENTER_HIGH) = 25 x 56/55 = 25.45 MHz
- SS_MODE (CENTER_LOW) = 25 x 112/111 = 25.23 MHz
- SS_MODE (DOWN_HIGH) = 25 MHz
- SS_MODE (DOWN_LOW) = 25 MHz

For an 80 MHz input clock, the new timing constraints would be:

- SS_MODE(CENTER_HIGH) = 80 x 44/43 = 81.86 MHz
- SS_MODE (CENTER_LOW) = 80 x 88/87 = 80.92 MHz
- SS_MODE (DOWN_HIGH) = 80 MHz
- SS_MODE (DOWN_LOW) = 80 MHz

Table 3-5 and Table 3-6 provide information which allows the manual adjustment of timing constraints to the frequency range of the spread-spectrum enabled clock. This is for the generation of timing constraints in an XDC file used by Vivado tools.

Table 3-5 and Table 3-6 show that timing constraints should be modified when spread-spectrum clocking parameter SS_MODE is set to CENTER_LOW or CENTER_HIGH. When SS_MODE attribute is set to DOWN_LOW or DOWN_HIGH timing constraint adjustment is not necessary.

Also note that manually adjusting timing constraints is not needed because the Vivado tools detect when spread-spectrum clocking in a design. Vivado tools (static timing analysis) automatically account for any timing spread caused by the spread-spectrum enabled clocks. When spread-spectrum clocks are used, Vivado static timing analysis adds a spread-spectrum (SS) uncertainty value of the total uncertainty calculation formula. The formula used by the static analysis tools is as follows:

Equation 3-12

$$\frac{((TS_J^2 - D_J^2)^{\frac{1}{2}})}{2} + PE + SS$$

where:

- TS_J^2 = Total system jitter
- D_J^2 = Discrete jitter
- PE = Phase error
- SS = Spread-spectrum

CAUTION! *When using spread-spectrum clocking in a design, it is necessary to use appropriated clock domain crossing (CDC) circuitry for all signals, data and non-data, crossing clock and spread-spectrum clock domains, and vice versa.*

Asynchronous FIFOs should be used to transfer data between two clock domains. The depth of the FIFO depends on the modulation frequency in the clock. The slower the modulation, the deeper the FIFO needs to be:

$$\text{FIFO depth is proportional to} = \frac{frequencyDeviation}{ModulationFrequency} \qquad \text{Equation 3-13}$$

When spread-spectrum clocking is used with SS_MODE set as DOWN_LOW or DOWN_HIGH the calculated F~IN_SS~ frequency (using data from Table 3-5 and/or Table 3-6) is lower than the original clock frequency (Refer to the examples after Table 3-6). If no precautions are taken, the used FIFO can fill up and over-run. Prevent this by using a FIFO with throttle control.

Table 3-7: **Spread-Spectrum Generation Restrictions for UltraScale Devices**

Parameter	Value	
$F_{MODULATION}$	Minimum	25 [KHz]
	Maximum	250 [KHz]
Input Clock Frequency	Minimum	25 [MHz]
	Maximum	150 [MHz]
SS_MODE (CENTER_HIGH)	25 MHz < F_{IN} < 35 MHz	M = 28 D = 1
	35 MHz < F_{IN} < 50 MHz	M = 21, 22 D = 1
	50 MHz < F_{IN} < 75 MHz	M = 28 D = 2
	75 MHz < F_{IN} < 100 MHz	M = 21, 22 D = 2
	100 MHz < F_{IN} < 150 MHz	M = 21, 22 D = 3

Table 3-7: **Spread-Spectrum Generation Restrictions for UltraScale Devices *(Cont'd)***

Parameter	Value	
SS_MODE (CENTER_LOW)	25 MHz < F_{IN} < 35 MHz	M = 56 D = 2
	35 MHz < F_{IN} < 50 MHz	M = 42, 44 D = 2
	50 MHz < F_{IN} < 75 MHz	M = 56 D = 4
	75 MHz < F_{IN} < 100 MHz	M = 42, 44 D = 4
	100 MHz < F_{IN} < 150 MHz	M = 42, 44 D = 6
SS_MODE (DOWN_HIGH)	25 MHz < F_{IN} < 35 MHz	M = 28 D = 1
	35 < F_{IN} < 50 MHz	M = 21, 22 D = 1
	50 MHz < F_{IN} < 75 MHz	M = 28 D = 2
	75 MHz < F_{IN} < 100 MHz	M = 21, 22 D = 2
	100 MHz < F_{IN} < 150 MHz	M = 21, 22 D = 3
SS_MODE (DOWN_LOW)	25 MHz < F_{IN} < 35 MHz	M = 56 D = 2
	35 MHz < F_{IN} < 50 MHz	M = 42, 44 D = 2
	50 MHz < F_{IN} < 75 MHz	M = 56 D = 4
	75 MHz < F_{IN} < 100 MHz	M = 42, 44 D = 4
	100 MHz < F_{IN} < 150 MHz	M = 42, 44 D = 6
CLKOUT[3:2]_DIVIDE	N/A	
CLKOUT[6:4,1,0]_DIVIDE	1 to 128	
Bandwidth	Low	

Table 3-8: **Spread-Spectrum Generation Restrictions for UltraScale+ Devices**

Parameter	Value	
$F_{MODULATION}$	Minimum	30 [KHz]
	Maximum	250 [KHz]
Input Clock Frequency	Minimum	25 [MHz]
	Maximum	150 [MHz]

Table 3-8: **Spread-Spectrum Generation Restrictions for UltraScale+ Devices *(Cont'd)***

Parameter	Value	
SS_MODE (CENTER_HIGH)	30 MHz < F_{IN} < 40 MHz	M = 28 D = 1
	40 MHz < F_{IN} < 60 MHz	M = 21, 22 D = 1
	60 MHz < F_{IN} < 80 MHz	M = 28 D = 2
	80 MHz < F_{IN} < 120 MHz	M = 21, 22 D = 2
	120 MHz < F_{IN} < 150 MHz	M = 21, 22 D = 3
SS_MODE (CENTER_LOW)	30 MHz < F_{IN} < 40 MHz	M = 56 D = 2
	40 MHz < F_{IN} < 60 MHz	M = 42, 44 D = 2
	60 MHz < F_{IN} < 80 MHz	M = 56 D = 4
	80 MHz < F_{IN} < 120 MHz	M = 42, 44 D = 4
	120 MHz < F_{IN} < 150 MHz	M = 42, 44 D = 6
SS_MODE (DOWN_HIGH)	30 MHz < F_{IN} < 40 MHz	M = 28 D = 1
	40 MHz < F_{IN} < 60 MHz	M = 21, 22 D = 1
	60 MHz < F_{IN} < 80 MHz	M = 28 D = 2
	80 MHz < F_{IN} < 120 MHz	M = 21, 22 D = 2
	120 MHz < F_{IN} < 150 MHz	M = 21, 22 D = 3
SS_MODE (DOWN_LOW)	30 MHz < F_{IN} < 40 MHz	M = 56 D = 2
	40 MHz < F_{IN} < 60 MHz	M = 42, 44 D = 2
	60 MHz < F_{IN} < 80 MHz	M = 56 D = 4
	80 MHz < F_{IN} < 120 MHz	M = 42, 44 D = 4
	120 MHz < F_{IN} < 150 MHz	M = 42, 44 D = 6
CLKOUT[3:2]_DIVIDE	N/A	

Table 3-8: **Spread-Spectrum Generation Restrictions for UltraScale+ Devices *(Cont'd)***

Parameter	Value
CLKOUT[6:4,1,0]_DIVIDE	1 to 128
Bandwidth	Low

When using spread-spectrum generation, the VCO frequency is set by the clocking wizard based on the input frequency and SS_MODE. As a result, the clocking wizard is recommended to set the output frequencies for CLKOUT[6:4,1,0].

Based on the VCO frequency and SS_MOD_PERIOD, the clocking wizard also determines the correct modulation settings to set the modulation frequency within 10% of SS_MOD_PERIOD. Because the modulation frequency is dependent on the VCO frequency, the modulation frequency scales as the input frequency changes for a given compilation.

MMCM Application Example

These MMCM attribute settings result in a wide variety of synthesized clocks:

```
CLKOUT0_PHASE = 0;
CLKOUT0_DUTY_CYCLE = 0.5;
CLKOUT0_DIVIDE = 2;
CLKOUT1_PHASE = 90;
CLKOUT1_DUTY_CYCLE = 0.5;
CLKOUT1_DIVIDE = 2;
CLKOUT2_PHASE = 0;
CLKOUT2_DUTY_CYCLE = 0.25;
CLKOUT2_DIVIDE = 4;
CLKOUT3_PHASE = 90;
CLKOUT3_DUTY_CYCLE = 0.5;
CLKOUT3_DIVIDE = 8;
CLKOUT4_PHASE = 0;
CLKOUT4_DUTY_CYCLE = 0.5;
CLKOUT4_DIVIDE = 8;
CLKOUT5_PHASE = 135;
CLKOUT5_DUTY_CYCLE = 0.5;
CLKOUT5_DIVIDE = 8;
CLKFBOUT_PHASE = 0;
CLKFBOUT_MULT_F = 8;
DIVCLK_DIVIDE = 1;
CLKIN1_PERIOD = 10.0;
```

Figure 3-15 displays the resulting waveforms.

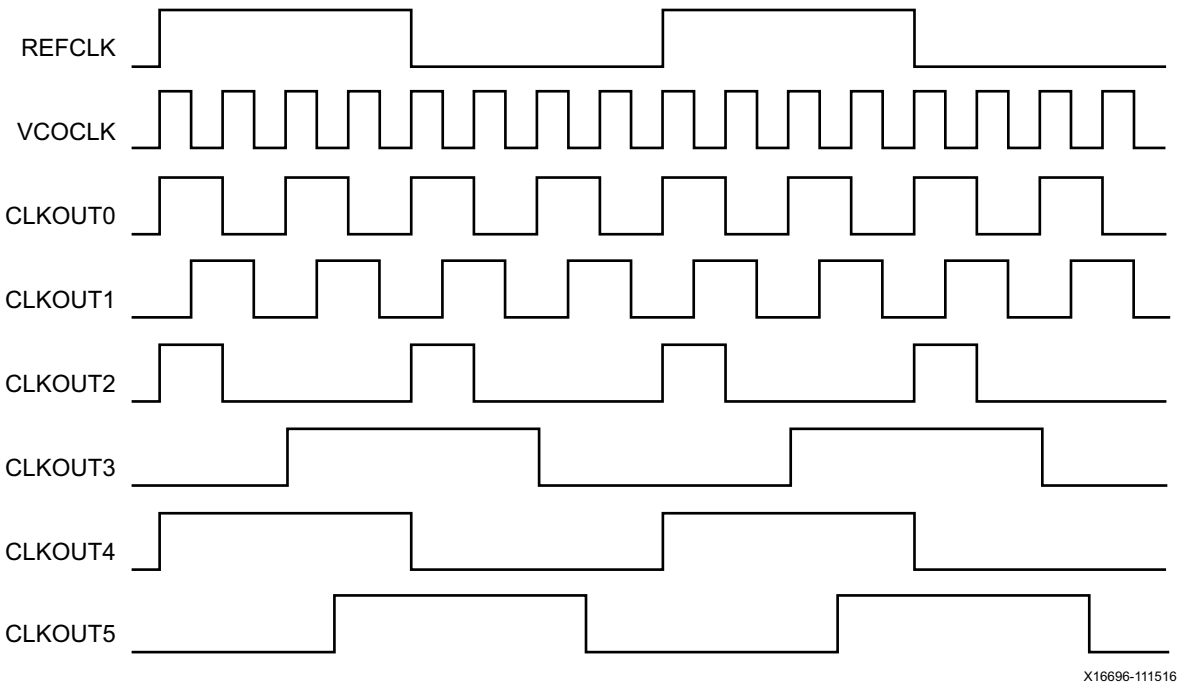

Figure 3-15: **Example Waveform**

PLLs

There are two PLLs per CMT that provide clocking to the PHY logic and I/Os. In addition, they can be used as frequency synthesizers for a wide range of frequencies, serve as jitter filters, and provide basic phase shift capabilities and duty cycle programming. The PLLs differ from the MMCM in number of outputs, cannot deskew clock nets, and do not have advanced phase shift capabilities, Multipliers and input dividers have a smaller value range and do not have many of the other advanced features of the MMCM.

General Usage Description

PLL Primitives

The UltraScale device PLL primitives, PLLE3_BASE and PLLE3_ADV, are shown in Figure 3-16. For UltraScale+ devices have the same primitives with an E4 instead of an E3. In this user guide, PLLE4_ADV is the same as the PLLE3_ADV, and PLLE4_BASE is the same as PLLE3_BASE.

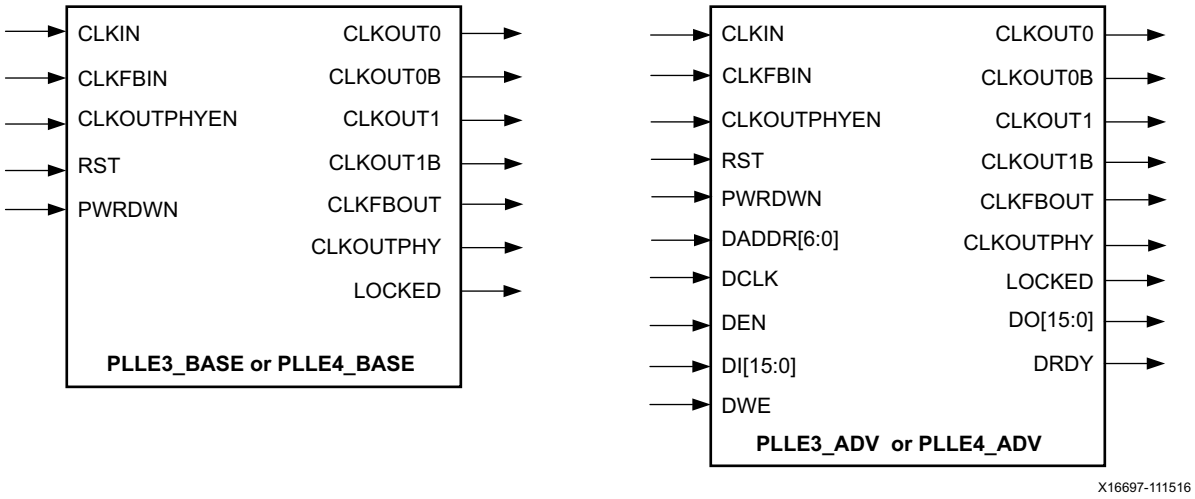

Figure 3-16: **PLL Primitives**

PLLE3_BASE and PLLE4_BASE Primitive

The PLLE#_BASE primitive provides access to the most frequently used features of a stand-alone PLL. Clock deskew, frequency synthesis, and duty cycle programming are available to use with the PLLE#_BASE. The ports are listed in Table 3-9.

Table 3-9: **PLLE#_BASE Ports**

Description	Ports
Clock input	CLKIN, CLKFBIN
Control inputs	RST, CLKOUTPHYEN
Clock output	CLKOUT0, CLKOUT1, CLKOUT0B, CLKOUT1B, CLKOUTPHY, CLKFBOUT
Status and data outputs	LOCKED
Power control	PWRDWN

PLLE3_ADV and PLLE4_ADV Primitive

The PLLE#_ADV primitive provides access to all PLLE#_BASE features plus additional ports for access to the DRP. The ports are listed in Table 3-10.

Table 3-10: **PLLE#_ADV Ports**

Description	Ports
Clock input	CLKIN, DCLK, CLKFBIN
Control and data input	RST, CLKOUTPHYEN, DWE, DEN, DADDR, DI
Clock output	CLKOUT0, CLKOUT1, CLKOUT0B, CLKOUT1B, CLKOUTPHY, CLKFBOUT
Status and data output	LOCKED, DO, DRDY
Power control	PWRDWN

PLL Ports

Table 3-11 summarizes the PLL ports.

Table 3-11: **PLL Ports**

Pin Name	I/O	Pin Description
CLKIN	Input	General clock input.
RST	Input	Asynchronous reset signal. The RST signal is an asynchronous reset for the PLL. The PLL synchronously re-enables itself when this signal is released (i.e., PLL re-enabled). A reset is required when the input clock conditions change (e.g., frequency).
PWRDWN	Input	Powers down instantiated but unused PLLs. See PWRDWN – Power Down, page 74.
CLKOUT[0:1] CLKOUT[0:1]B	Output	User configurable clock outputs 0 and 1 and their inverted versions. The CLKOUT can be divided versions of the VCO phase outputs (user controllable) from 1 (bypassed) to 128.

Table 3-11: **PLL Ports *(Cont'd)***

Pin Name	I/O	Pin Description
CLKFBOUT	Output	Dedicated PLLE3 or PLLE4 feedback output.
CLKFBIN	Input	Feedback clock input.
CLKOUTPHYEN	Input	Enable PHY clocking.
CLKOUTPHY	Output	Dedicated PHY clock.
LOCKED	Output	An output from the PLL that indicates when the PLL has achieved phase alignment within a predefined window and frequency matching within a predefined PPM range. The PLL automatically locks after power on; no extra reset is required. LOCKED is deasserted if the input clock stops or the phase alignment is violated (e.g., input clock phase shift). The PLL must be reset after LOCKED is deasserted.
DADDR[6:0]	Input	The dynamic reconfiguration address (DADDR) input bus provides a reconfiguration address for the dynamic reconfiguration. When not used, all bits must be assigned zeros.
DI[15:0]	Input	The dynamic reconfiguration data input (DI) bus provides reconfiguration data. When not used, all bits must be set to zero.
DWE	Input	The dynamic reconfiguration write enable (DWE) input pin provides the write enable control signal to write the DI data into the DADDR address. When not used, DWE must be tied Low.
DEN	Input	The dynamic reconfiguration enable (DEN) provides the enable control signal to access the dynamic reconfiguration feature. When the dynamic reconfiguration feature is not used, DEN must be tied Low.
DCLK	Input	The DCLK signal is the reference clock for the dynamic reconfiguration port.
DO[15:0]	Output	The dynamic reconfiguration output bus provides PLL data output when using dynamic reconfiguration.
DRDY	Output	The dynamic reconfiguration ready output (DRDY) provides the response to the DEN signal for the PLL's dynamic reconfiguration feature.

PLL Port Descriptions

CLKIN – Reference Clock Input

This is a general clock input to the PLL.

RST – Asynchronous Reset Signal

The RST signal is an asynchronous reset for the PLL. The PLL is synchronously re-enabled when this signal is deasserted.

PWRDWN – Power Down

This signal powers down instantiated but currently unused PLLs. This mode can be used to save power for temporarily inactive portions of the design and/or PLLs that are not active in certain system configurations. No PLL power is consumed in this mode.

CLKOUT[0:1] – Output Clocks

These are user-configurable clock outputs and can be divided versions of the VCO phase outputs (user controllable) from 1 (bypassed) to 128. The input clock and output clocks can be phase aligned.

CLKFBOUT – Dedicated PLL Feedback Output

For the possible configurations of CLKFBOUT, see Figure 3-17 and Figure 3-18. Unlike the MMCM, the CLKFBOUT cannot drive logic.

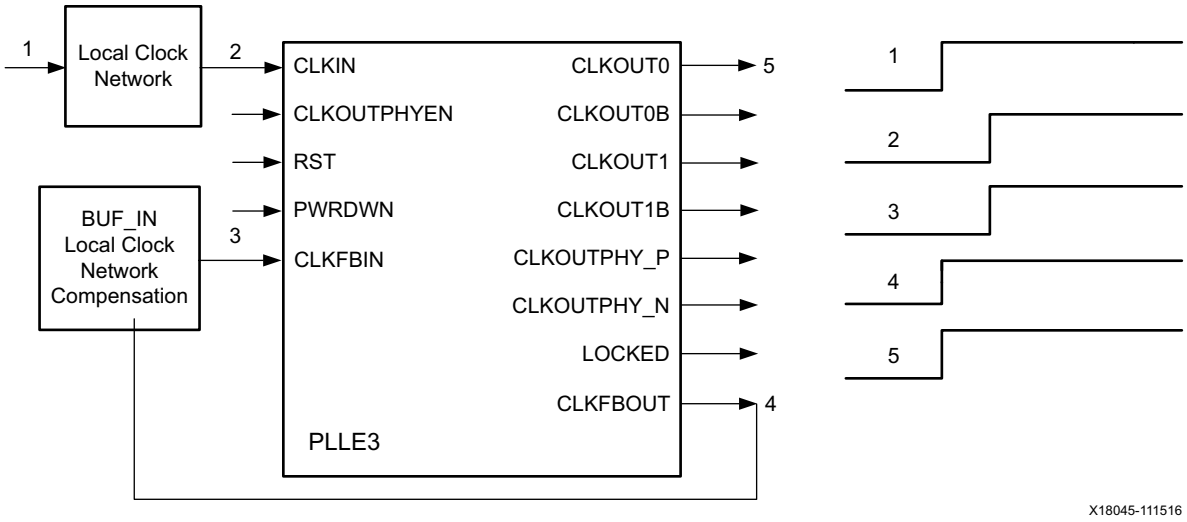

Figure 3-17: **Clock Deskew Using BUF_IN Compensation Mode**

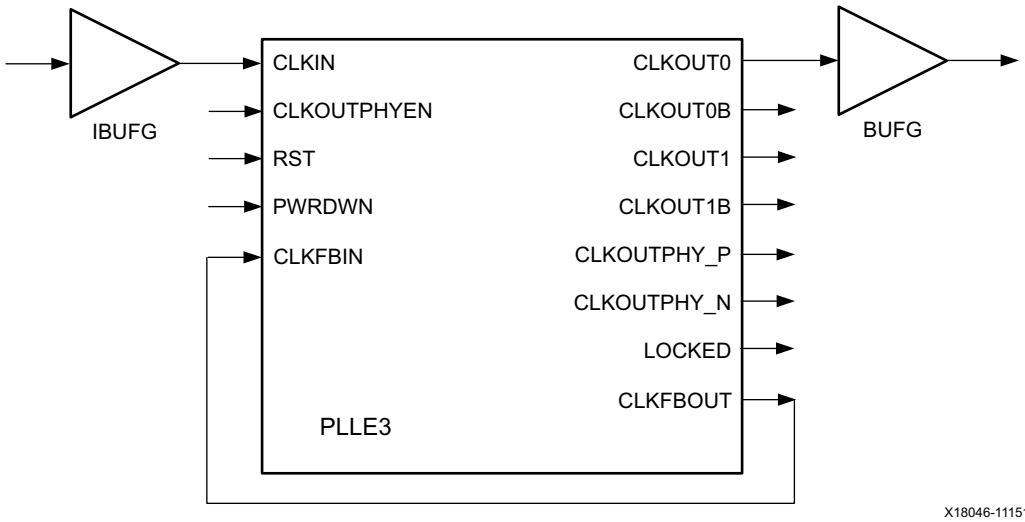

Figure 3-18: **PLL Internal Feedback**

CLKFBIN – Feedback Clock Input

CLKFBIN must be connected either directly to the CLKFBOUT for internal feedback, or to the CLKFBOUT through a BUF_IN. Using BUF_IN in the feedback path compensates for the clock network delay in the same XIPHY bank as shown in Figure 3-17 where the nodes 1 and 5 are phase aligned.

CLKOUTPHY – PHY Clock Output

This is a dedicated clock output for use by the PHY byte logic and I/O. It can be 2X, 1X, or 0.5X of the VCO frequency.

CLKOUTPHYEN – PHY Clock Enable

CLKOUTPHYEN enables the CLKOUTPHY clock outputs. The PLL employs enable logic to synchronize the asynchronous CLKOUTPHYEN signal from your design and controls when the CLKOUTPHY clocks are released. After the CLKOUTPHY clock is released, the rising edge is aligned to the rising edge of the input clock CLKIN. Glitch-free enabling and disabling of the CLKOUTPHY output clock is assured for all configurations.

However, phase alignment between multiple PLL CLKOUTPHY clocks is only assured when both the CLKFBOUT_MULT and CLKOUT[0:1]_DIVIDE values are set to 1, 2, 4, or 8. Rising edges do not align for CLKFBOUT = 3, 5, 6, 7, 9,...

CLKOUT[0:1]B – Inverted Output Clocks

This is the inverted (180° phase shift) CLKOUT[0:1].

LOCKED

This output from the PLL is used to indicate when the PLLs have achieved frequency alignment of the reference clock and the internal feedback. Frequency alignment is within a predefined window of frequency matching within a predefined PPM range. The PLL automatically locks after power on; no extra reset is required. LOCKED is deasserted within one PFD clock cycle if the input clock stops or the frequency has changed. The PLL must be reset when LOCKED is deasserted. The clock outputs should not be used prior to the assertion of LOCKED.

DADDR[6:0] – Dynamic Reconfiguration Address

The dynamic reconfiguration address (DADDR) input bus provides a reconfiguration address for the dynamic reconfiguration. The address value on this bus specifies the 16 configuration bits that are written or read with the next DCLK cycle. When not used, all bits must be assigned zeros.

DI[15:0] – Dynamic Reconfiguration Data Input

The dynamic reconfiguration data input (DI) bus provides reconfiguration data. The value of this bus is written to the configuration cells. The data is presented in the cycle that DEN and DWE are active. The data is captured in a shadow register and written at a later time. DRDY indicates when the DRP port is ready to accept another write. When not used, all bits must be set to zero.

DWE – Dynamic Reconfiguration Write Enable

The dynamic reconfiguration write enable (DWE) input pin provides the write/read enable control signal to write the DI data into or read the DO data from the DADDR address. When not used, DWE must be tied Low.

DEN – Dynamic Reconfiguration Enable Strobe

The dynamic reconfiguration enable strobe (DEN) provides the enable control signal to access the dynamic reconfiguration feature and enable all DRP port operations. When the dynamic reconfiguration feature is not used, DEN must be tied Low.

DCLK – Dynamic Reconfiguration Reference Clock

DCLK is the reference clock for the dynamic reconfiguration port. The rising edge of this signal is the timing reference for all other port signals. The setup time is specified in the UltraScale device data sheets [Ref 5]. There is no hold time requirement for the other input signals relative to the rising edge of DCLK. This signal can be driven by an IBUF, IBUFG, BUFGCE, or BUFGCTRL. There are no dedicated connections to this clock input.

DO[15:0] – Dynamic Reconfiguration Output Bus

The dynamic reconfiguration output bus provides PLL data output when using dynamic reconfiguration. If DWE is inactive while DEN is active at the rising edge of DCLK, this bus holds the content of the configuration cells addressed by DADDR. The DO bus must be captured on the rising edge of DCLK when DRDY is active. The DO bus value is held until the next DRP operation.

DRDY – Dynamic Reconfiguration Ready

The dynamic reconfiguration ready output (DRDY) provides the response to the DEN signal for the PLL's dynamic reconfiguration feature. This signal indicates that a DEN/ DCLK operation has completed.

PLL Attributes

Table 3-12 lists the attributes for the PLLE#_BASE and PLLE#_ADV primitives.

Table 3-12: **PLL Attributes**

Attribute	Type	Allowed Values	Default	Description
CLKOUT[0:1]_DIVIDE	Integer	1 to 128	1	Specifies the amount to divide the associated CLKOUT clock output if a different frequency is desired. This number, in combination with the CLKFBOUT_MULT values, determines the output frequency.
CLKOUT[0:1]_ DUTY_CYCLE	Real	0.01 to 0.99	0.50	Specifies the duty cycle of the associated CLKOUT clock output in percentages (i.e., 0.50 generates a 50% duty cycle).
CLKFBOUT_MULT	Boolean	1 to 19	5	This CLKFBOUT_MULT range applies to UltraScale devices. Specifies the amount to multiply all CLKOUT clock outputs if a different frequency is desired. This number, in combination with the associated CLKOUT#_DIVIDE value and DIVCLK_DIVIDE value, determines the output frequency.
CLKFBOUT_MULT	Integer	2 to 21	5	This CLKFBOUT_MULT range applies to UltraScale+ devices. Specifies the amount to multiply all CLKOUT clock outputs if a different frequency is desired. This number, in combination with the associated CLKOUT#_DIVIDE value and DIVCLK_DIVIDE value, determines the output frequency.
DIVCLK_DIVIDE	Integer	1 to 15	1	Specifies the division ratio for all output clocks with respect to the input clock.

Table 3-12: **PLL Attributes *(Cont'd)***

Attribute	Type	Allowed Values	Default	Description
REF_JITTER	Real	0.000 to 0.999	0.010	Allows specification of the expected jitter on the reference clock to better optimize PLL performance. A bandwidth setting of OPTIMIZED attempts to choose the best parameter for input clocking when unknown. If known, the value provided should be specified in terms of the unit interval (UI) (the maximum peak-to-peak value) of the expected jitter on the input clock.
CLKIN_PERIOD	Real	0.938 to 14.286	0.000	Specifies the input period in ns to the PLL CLKIN input. Resolution is down to the ps. This information is mandatory and must be supplied.
STARTUP_WAIT	String	FALSE, TRUE	FALSE	Wait during the configuration start-up cycle for the PLL to lock.
CLKOUT[0:1]_PHASE	Real	–360.000 to 360.000	0.000	Allows specification of the output phase relationship of the associated CLKOUT clock output in number of degrees offset (i.e., 90 indicates a 90° offset or ¼ cycle phase offset while 180 indicates a 180° offset or ½ cycle phase offset). Valid phase shifts are in 360÷CLKOUT[0:1]_DIVIDE degree increments.
CLKFBOUT_PHASE	Real	–360.000 to 360.000		Specifies the phase offset in degrees of the clock feedback output. Shifting the feedback clock results in a negative phase shift of all output clocks to the PLL. Valid phase shifts are in 360÷CLKFBOUT_MULT degree increments.

Table 3-12: **PLL Attributes *(Cont'd)***

Attribute	Type	Allowed Values	Default	Description
COMPENSATION	String	AUTO(1), PHY_ALIGN, BUF_IN, INTERNAL	AUTO	Clock input compensation. Must be set to AUTO. Defines how the PLL feedback is configured. INTERNAL: Indicates that the PLL is using its own internal feedback path so no delay is being compensated. BUF_IN: Indicates that the clock network delay within the same XIPHY bank is compensated. Both CLKIN and CLKFB must be the same frequency at the PFD ($F_{IN}/D = F_{VCO}/M$). The feedback must be limited to within the same XIPHY and cannot be routed to adjacent banks or outside the device. PHY_ALIGN (for UltraScale+ devices only): allows fine- grained adjustments of the PLL output phase for the alignment of the XIPHY and internal logic flip-flops using the same source clock. Primarily for memory controller use.
CLKOUTPHY_MODE	String	VCO_2X, VCO, VCO_HALF	VCO_2X	Determines the clock output frequency based on the VCO frequency for the BITSLICE_CONTROL block.

Notes:

1. The specifications for the VCO frequencies PLL_FVCOMIN/PLL_FVCOMAX and minimum out frequency PLL_FOUTMIN are different for the UltraScale and UltraScale+ families. Consult the appropriate data sheets.

Dynamic Reconfiguration Port

In most circumstances, the MMCM and/or PLL used in a design are configured using static calculated values to setup the used outputs. A wizard can be used to calculate all the values and generate an instantiable wrapper containing a configured MMCM or PLL. The MMCM and/or PLL primitive can also be instantiated as primitives and the values to make the primitive function correctly can be calculated using the equations provided in the MMCM Programming section.

The DRP port provides the ability to use a MMCM and/or PLL as a dynamic element in a design. The DRP port setup is that of a common microcontroller peripheral and gives the user access to a set of registers in the MMCM or PLL. These registers allow the user to fully control the MMCM or PLL. Inputs pins and the values to define output clocks are turned into register bits making it possible to use the primitives as active elements in a design.

Using the DRP port means reading and writing of registers of a peripheral. When using the Clocking Wizard the DRP port can be enabled through an AXI-Lite controller to a hard or soft microcontroller in the FPGA. Nevertheless, it might be necessary by design and other requirements to use the DRP port in a bare metal configuration (also selectable in the Clocking Wizard). The DRP port can then be used as such through a state machine based design. To help with this the provided description of the functioning of the DRP port can be used.

For additional DRP usage information, see *MMCM and PLL Dynamic Reconfiguration* (XAPP888) [Ref 6] and the associated reference.

The DRP port connections are shown in the following figure.

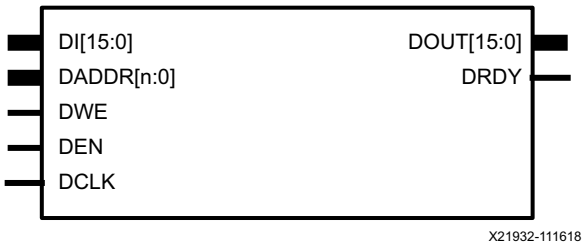

Figure 3-19: **DRP Port Connections**

Table 3-13: **DRP Port Signals**

Port	Size	I/O	Description
DCLK	1	Input	The DCLK signal is the reference clock for the dynamic reconfiguration port. This clock is normally about 100 MHz to 200 MHz. The newer the technology of the FPGA family used, the faster this clock can be.
DEN	1	Input	The dynamic reconfiguration enable (DEN) provides the enable control signal to access the dynamic reconfiguration feature. When the dynamic reconfiguration feature is not used, DEN must be tied Low.
DWE	1	Input	The dynamic reconfiguration write enable (DWE) input pin provides the write enable control signal to write the data on the DI port into the register selected by the DADDR address. When not used, DWE must be tied Low.
DADDR	n(1)	Input	The dynamic reconfiguration address (DADDR) input bus provides a reconfiguration address to access a specific register in the primitive for dynamic reconfiguration. When not used, all bits must be assigned zeros.
DI	16	Input	The dynamic reconfiguration data input (DI) bus provides reconfiguration data that is written into a specified address (DADDR) of the register set. When not used, all bits must be set to zero.
DO	16	Output	The dynamic reconfiguration output bus provides data output of the register selected by the DADDR bus. This port can be used to control DRP register contents.
DRDY	1	Output	The dynamic reconfiguration ready output (DRDY) provides the response to the DEN signal for the PLLs dynamic reconfiguration feature. This signal is pulsed high when a write or read operation is successful.

Notes:

1. The width of the DADDR bus depends on the primitive that the DRP port is a part of. For a MMCM, the address bus is 7-bit wide and for a PLL the DRP address bus is 7-bit wide (DADDR(6:0)). The DADDR port of an ADC/DAC in a RFSoC device is 12-bit wide while the DADDR port of a GTP is 10-bit wide.

Writing to the DRP Port

1. Put the address to write to and the data that needs to be written on the buses, DADDR[n:0] and DI[15:0] respectively.
2. Make the DWE (Data Write Enable) signal High.
3. Pulse the DEN signal High for one clock cycle. The DEN signal is the trigger that makes the DRP port function. When this signal is captured on the rising edge of the clock the internals of the DRP port capture address and data and fill the correct register in the DRP map.
4. When the DEN signal goes Low for one pulse, make the DWE signal Low.
5. The DRP ports pulse the DRDY High for a clock cycle to confirm that the provided data is written into the provided address space. This also signals that a new write or read operation can start.

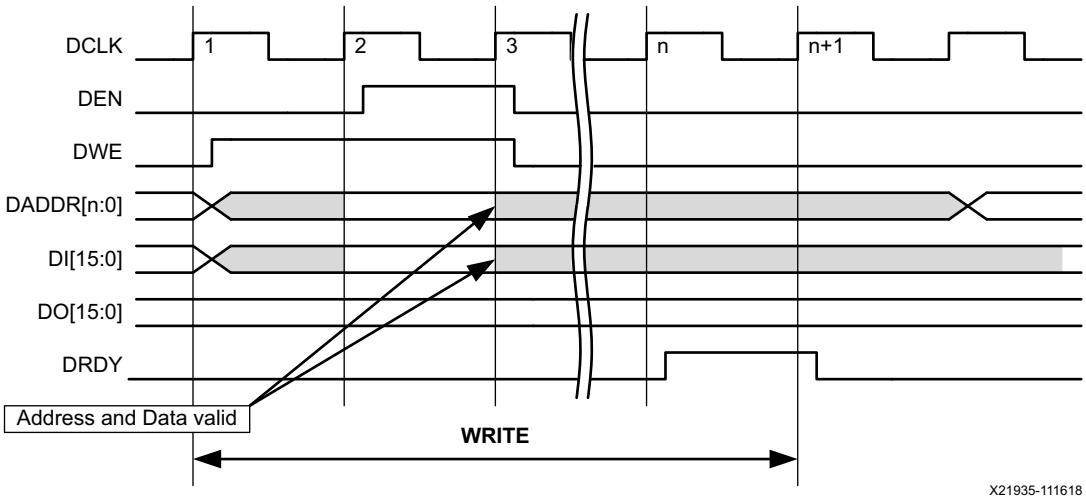

Figure 3-20: **Writing to the DRP Port**

Reading from the DRP Port:

1. Put the address of the register to read from on the DADDR[n:0] bus.
2. Leave the DWE signal Low at all times when reading.
3. The value ON/OFF of the DI[15:0] bus does not matter.
4. Pulse the DEN signal High for one clock cycle. The DEN signal is the trigger that makes the DRP port function. When this signal is captured on the rising edge of the clock the internals of the DRP port capture the address to make sure that the contents of the correct register in the DRP map are reflected on the DO[15:0] output.
5. The DRP ports pulse the DRDY High for a clock cycle to confirm that the provided data is written into the provided address space. This also signals that a new write or read operation can start.

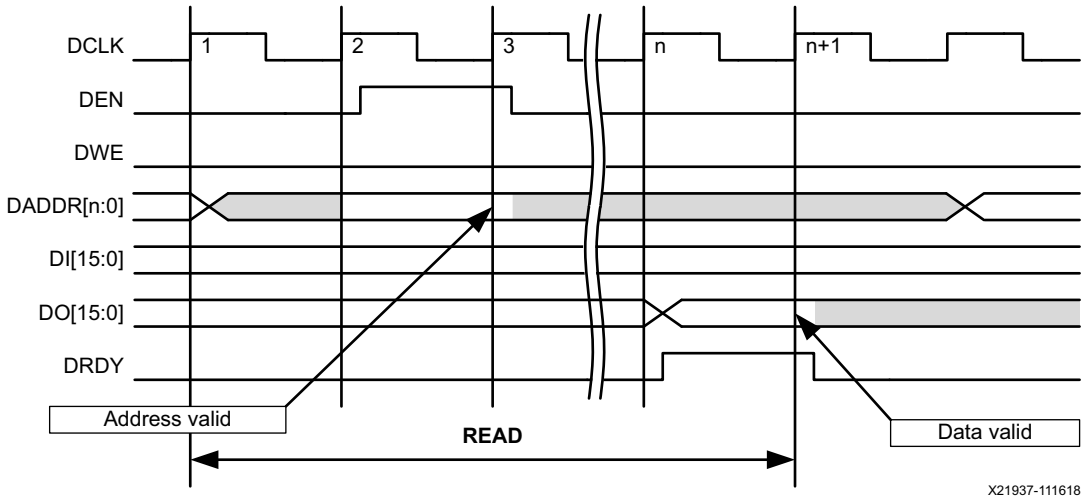

Figure 3-21: **Reading from the DRP Port**

Read—Write Operation

A read-write operation must always be executed with respect to the DRDY signal. Only when the DRDY signal pulses High, a new read or write operation can be initiated. If the DRDY signal is not controlled after a read or write operation, from or to the DRP port, it is not certain that the written bits are set or the obtained bits are representing the value of the register.

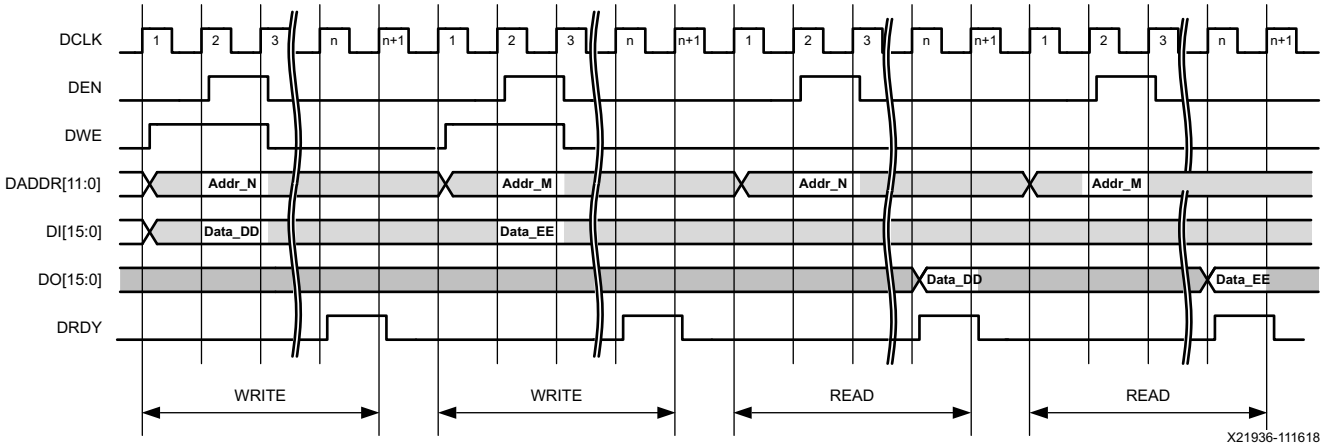

Figure 3-22: **Read—Write Operation**

DRP Register Set

The user accessible DRP register set is described in this section. The DRP register map spans from address `0x00` to address `0x7F` (7-bit address bus). The figure below shows the layout of the register map of user accessible registers. Be aware that values of different counters overlap register boundaries.

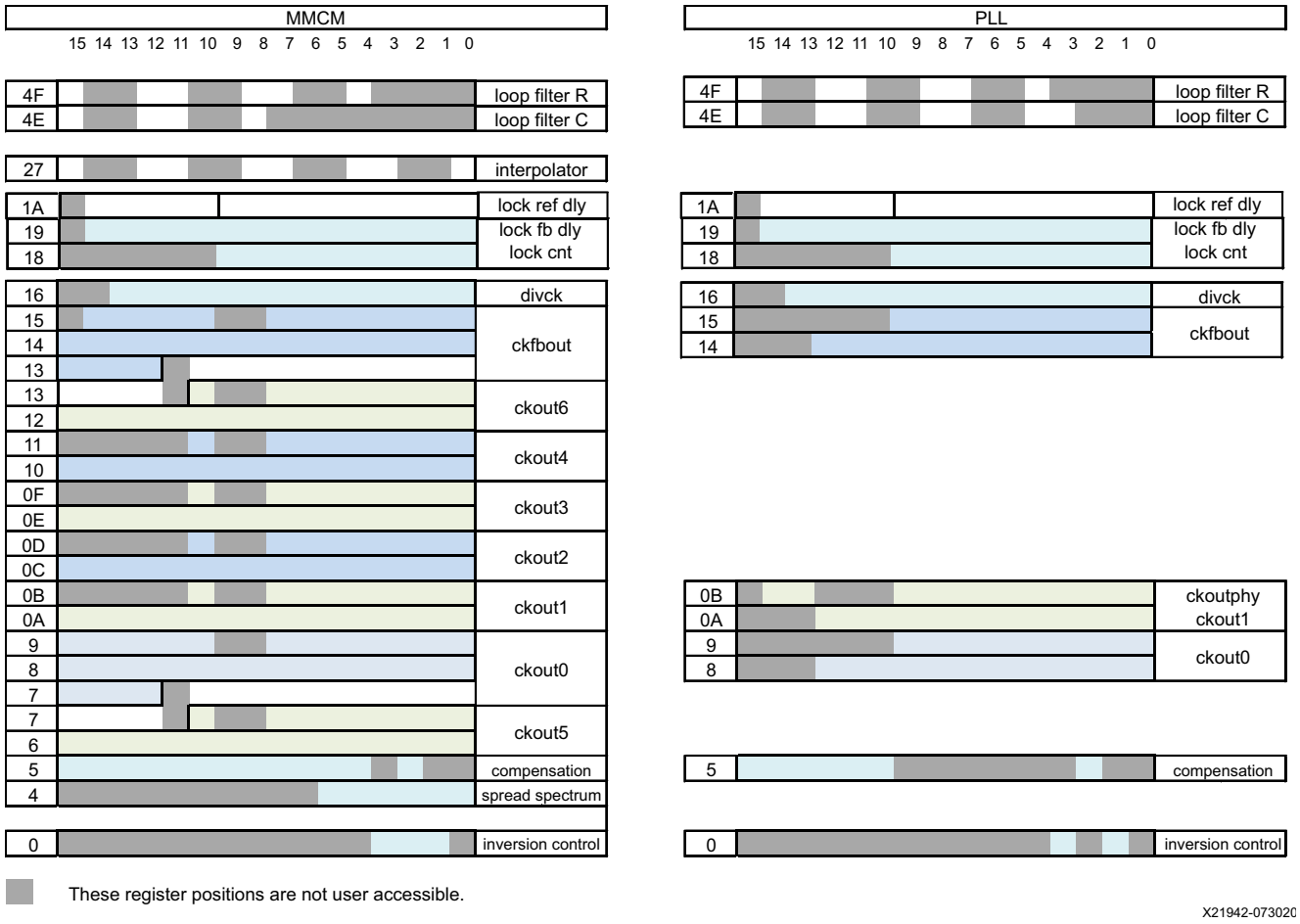

Figure 3-23: **MMCM DRP Register Set**

IMPORTANT: *When operating a DRP port, it is recommended that the existing contents of the register that is going to be changed are first read. Write back the register contents where only the required bits are modified. Modify only the colored bits and always maintain the state of the gray bits.*

MMCM Registers

Reg	**4F**												**ADDR: 0x4F**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0			0	0			0	0			0				
Access	R/W			R/W	R/W			R/W	R/W			R/W				
15	mc_res(3)					Loop filter resistor setting.										
12	mc_res(2)															
11	mc_res(1)															
8	mc_res(0)															
7	mc_lfhf(1)					Loop filter high frequency capacitor setting.										
4	mc_lfhf(0)															

Reg	**4E**												**ADDR: 0x4E**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0			0	0			0								
Access	R/W			R/W	R/W			R/W								
15	mc_cp(3)					Charge pump settings.										
12	mc_cp(2)															
11	mc_cp(1)															
8	mc_cp(0)															

Registers 0x4F and 0x4E define the values for the loop filters. Pick the appropriate values for these filters from the *MMCM and PLL Dynamic Reconfiguration* (XAPP888) [Ref 6].

Reg	**27**												**ADDR: 0x27**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0			0	0			1	0			0	0			0
Access	R/W			R/W	R/W			R/W	R/W			R/W	R/W			R/W
15	mc_interp_en(7)					Interpolator selection. Default value is 00010000.										
12	mc_interp_en(6)															
11	mc_interp_en(5)															
8	mc_interp_en(4)															
7	mc_interp_en(3)															
4	mc_interp_en(2)															
3	mc_interp_en(1)															
0	mc_interp_en(0)															

Notes:

1. If any of the output counters is using fine phase shift then mc_interp_en[3:0] must be set to 1111 otherwise mc_interp_en[3:0] must be set to 0000.
2. mc_interp_en(4) is always set to 1.
3. If any of the output counters is using a phase of VCO other than 0 or 180, uses fractional division for a counter, or uses spread-spectrum mode then mc_interp_en[7:5] must be set to 111 otherwise mc_interp_en[7:5] must be set to 000.

Reg	**1A**												**ADDR: 0x1A**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default		0	0	0	1	1	1	1	1	1	1	0	1	0	0	1
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
14:10	mc_lock_ref_dly[4:0]					Window setting for the lock circuit of the reference clock.										
9:0	mc_lock_sat_high[9:0]					Maximum value of the lock counter. Default value is d1001.										

Reg	**19**												**ADDR: 0x19**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default		0	0	0	1	1	0	0	0	0	0	0	0	0	0	1
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
14:10	mc_lock_fb_dly[4:0]					Window setting for the lock circuit of the feedback clock.										
9:0	mc_lock_sat_high[9:0]					Counter setting the number of clock cycles the MMCM needs to have CLKREF and CLKFB misaligned within a certain window before deasserting the LOCKED output. Default value is 1.										

Reg	**18**												**ADDR: 0x18**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default							1	1	1	1	1	0	1	0	0	0
Access							R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
9:0	mc_lock_cnt[9:0]						Counter setting the number of clock cycles the MMCM needs to have CLKREF and CLKFB aligned within a certain window before the LOCKED output is asserted. Default value is 1000.									

Refer to *MMCM and PLL Dynamic Reconfiguration* (XAPP888) [Ref 6] to determine the values for registers 0x1A, 0x19, and 0x18.

Reg	**16**												**ADDR: 0x16**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default			0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
13	mc_divck_edge					High to low clock edge transition control.										
12	mc_divck_nocount					Bypass counter.										
11:6	mc_divck_ht[5:0]					Counter high time.										
5:0	mc_divck_lt[5:0]					Counter low time.										

Register 0x16 controls the divider (D counter) shown in Figure 3-1.

Reg	**15**												**ADDR: 0x15**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default		0	0	1	0	0			0	1	0	0	0	0	0	1
Access		R/W	R/W	R/W	R/W	R/W			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
14:12	mc_ckfbout_frac[2:0]					Fractional phase.										
11	mc_ckfbout_frac_en					Enable fractional counter.										
10	mc_ckfbout_frac_wf_r					Fractional mode. Rising edge wait.										
7	mc_ckfbout_edge					High to low clock edge transition control.										
6	mc_ckfbout_nocount					Bypass counter.										
5:0	mc_ckfbout_frac[5:0]					Counter delay.										

Reg	**14**												**ADDR: 0x14**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15:13	mc_ckfbout_pm_r[2:0]					VCO phase selection mux and rising edge control.										
12	mc_ckfbout_en					Counter enable										
11:6	mc_ckfbout_ht[5:0]					Counter high time										
5:0	mc_ckfbout_lt[5:0]					Counter low time										

Reg	**13**												**ADDR: 0x13**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	0		0			0	1	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W		R/W			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15:13	mc_ckfbout_pm_f[2:0]					VCO phase selection mux and falling edge control.										
12	mc_ckfbout_frac_wf_f					Fractional mode. Falling edge wait.										
10	mc_ckout6_cddc_en					Clock divide dynamic change enable (DRP only)										
7	mc_ckout6_edge					High to low clock edge control										
6	mc_ckout6_nocount					Counter bypass										
5:0	mc_ckout6_dt[5:0]					Counter delay										

Registers `0x15`, `0x14`, and bits [15:12] of `0x13` control the fractional feedback (M counter) shown in Figure 3-1.

Reg	**12**												**ADDR: 0x12**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15:13	mc_ckout6_pm[2:0]					VCO phase selection multiplexer										
12	mc_ckout6_en					Counter enable										
11:6	mc_ckout6_ht[5:0]					Counter high time										
5:0	mc_ckout6_lt[5:0]					Counter low time										

Bits [10:0] of register `0x13` and register `0x12` control the CLKOUT6 counter.

Reg	**11**												**ADDR: 0x11**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default						0			0	1	0	0	0	0	0	0
Access						R/W			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
10	mc_ckout4_cddc_en					Clock divide dynamic change control enable (DRP only).										
7	mc_ckout4_edge					High to low clock edge control.										
6	mc_ckout4_nocount					Counter bypass.										
5:0	mc_ckout4_dt[5:0]					Counter delay.										

Reg	**10**												**ADDR: 0x10**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15:13	mc_ckout4_pm[2:0]					VCO phase selection multiplexer.										
12	mc_ckout4_en					Counter enable.										
11:6	mc_ckout4_ht[5:0]					Counter high time.										
5:0	mc_ckout4_lt[5:0]					Counter low time.										

Registers `0x11` and `0x10` control the output counter for CLKOUT4.

Reg	**0F**												**ADDR: 0x0F**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default						0			0	1	0	0	0	0	0	0
Access						R/W			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
10	mc_ckout3_cddc_en					Clock divide dynamic change control enable (DRP only).										
7	mc_ckout3_edge					High to low clock edge control.										
6	mc_ckout3_nocount					Counter bypass.										
5:0	mc_ckout3_dt[5:0]					Counter delay.										

Reg	**0E**												**ADDR: 0x0E**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15:13	mc_ckout3_pm[2:0]					VCO phase selection multiplexer.										
12	mc_ckout3_en					Counter enable.										
11:6	mc_ckout3_ht[5:0]					Counter high time.										
5:0	mc_ckout3_lt[5:0]					Counter low time.										

Registers 0x0F and 0x0E control the output counter for CLKOUT3.

Reg	**0D**												**ADDR: 0x0D**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default						0			0	1	0	0	0	0	0	0
Access						R/W			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
10	mc_ckout2_cddc_en					Clock divide dynamic change control enable (DRP only).										
7	mc_ckout2_edge					High to low clock edge control.										
6	mc_ckout2_nocount					Counter bypass.										
5:0	mc_ckout2_dt[5:0]					Counter delay.										

Reg	**0C**												**ADDR: 0x0C**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15:13	mc_ckout2_pm[2:0]					VCO phase selection multiplexer.										
12	mc_ckout2_en					Counter enable.										
11:6	mc_ckout2_ht[5:0]					Counter high time.										
5:0	mc_ckout2_lt[5:0]					Counter low time.										

Registers 0x0D and 0x0C control the output counter for CLKOUT2.

Reg	**0B**												**ADDR: 0x0B**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default						0			0	1	0	0	0	0	0	0
Access						R/W			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
10	mc_ckout1_cddc_en					Clock divide dynamic change control enable (DRP only).										
7	mc_ckout1_edge					High to low clock edge control.										
6	mc_ckout1_nocount					Counter bypass.										
5:0	mc_ckout1_dt[5:0]					Counter delay.										

Reg	**0A**												**ADDR: 0x0A**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15:13	mc_ckout1_pm[2:0]					VCO phase selection multiplexer.										
12	mc_ckout1_en					Counter enable.										
11:6	mc_ckout1_ht[5:0]					Counter high time.										
5:0	mc_ckout1_lt[5:0]					Counter low time.										

Registers 0x0B and 0x0A control the output counter for CLKOUT1.

Reg	**09**												**ADDR: 0x09**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	0		0			0	1	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W		R/W			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	mc_ckout0_cddc_en					Clock divide dynamic change control enable (DRP only).										
14:12	mc_ckout0_frac[2:0]					Fractional phase.										
11	mc_ckout0_frac_en					Enable fractional phase counter.										
10	mc_ckout0_frac_wf_r					Falling edge wait.										
7	mc_ckout0_edge					High to low clock edge control.										
6	mc_ckout0_nocount					Counter bypass.										
5:0	mc_ckout0_dt[5:0]					Counter delay.										

Reg	**08**												**ADDR: 0x8**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15:13	mc_ckout0_pm_r[2:0]					Rising edge control.										
12	mc_ckout0_en					Counter enable.										
11:6	mc_ckout0_ht[5:0]					Counter high time.										
5:0	mc_ckout0_lt[5:0]					Counter low time.										

Reg	**07**												**ADDR: 0x07**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	0		0			0	1	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W		R/W			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15:13	mc_ckout0_pm_f[2:0]	Falling edge control.
12	mc_ckout0_frac_wf_f	Rising edge wait.
10	mc_ckout0_cddc_en	Clock divide dynamic change control enable (DRP only).
7	mc_ckout5_edge	High to low clock edge control.
6	mc_ckout5_nocount	Counter bypass.
5:0	mc_ckout5_dt[5:0]	Counter delay.

The fractional output counter for CLKOUT0 is controlled by registers 0x09, 0x08, and bits [15:12] of register 0x07.

Reg	**06**												**ADDR: 0x06**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15:13	mc_ckout5_pm[2:0]	VCO phase selection multiplexer.
12	mc_ckout5_en	Counter enable.
11:6	mc_ckout5_ht[5:0]	Counter high time.
5:0	mc_ckout5_lt[5:0]	Counter low time.

Bits [10:0] of register 0x07 and register 0x06 control the CLKOUT5 counter.

The MMCM clock outputs are all defined by a configurable counter. The parameters defining the clock outputs CLKOUT6 to CLKOUT1 are explained in the following table. Refer to *MMCM and PLL Dynamic Reconfiguration* (XAPP888) [Ref 6] for calculation instructions and methods.

Table 3-14: **Clock Output Parameters**

Type	Description
CDDC	Clock divide dynamic change. The possibility to change the DRP registers without the need for a MMCM reset. When this option is enabled it functions with the CDDCREQ and CDDCDONE handshake pins. For more information read MMCM Clock Divide Dynamic Change.
MX	Clock input multiplexer control.
EDGE	Clock edge identification. Identify the clock edge used for a high to low transition of the counter.
NOCOUNT	Counter bypass.

Table 3-14: **Clock Output Parameters *(Cont'd)***

Type	Description
DT	Delay time. Counter delay or coarse phase shift setting.
PM	VCO phase selection. Used to select one of the eight possible VCO outputs.
EN	Counter enable.
HT	Counter high time. Set the delay the counter needs to output a high value.
LT	Counter low time. Set the delay the counter needs to output a low value.

Two of the counters, CLKFBOUT and CLKOUT0, are fractional counters. A fractional counter uses two non-fractional counters, an extra state, and adder logic. This is the reason a fractional counter has two enables (one for each counter to allow non-fractional use) and two VCO phase selection settings. For the adder and state logic, the VCO phase selection is extra split in rising and falling settings. Additional register configuration options defining the fractional counters are listed in the following table.

Table 3-15: **Register Configuration Options**

Type	Description
FRAC	Select a VCO phase to operate the fractional counter.
FRAC_EN	Enable the fractional counter.
FRAC_WF_R	Fractional counter wait for rising edge.
FRAC_WF_L	Fractional counter wait for falling edge.
PM_R	Select one of the eight VCO phased outputs as rising edge counter clock.
PM_L	Select one of the eight VCO phased outputs as falling edge counter clock.

Fractional counter mode is enabled when both mc_ckout_en and mc_ckout_frac_en are set. Both counters take different phases from the VCO outputs.

Example 1

Assume that a division of 2.5 is required, then:

- Counter A would take phase 0 (0 degrees) of the VCO
- Counter B would take phase 4 (180 degree offset)

Output starts High with rising edge on counter A and goes Low with second rising edge counter B. It goes High again with second rising edge of counter A after that and so on.

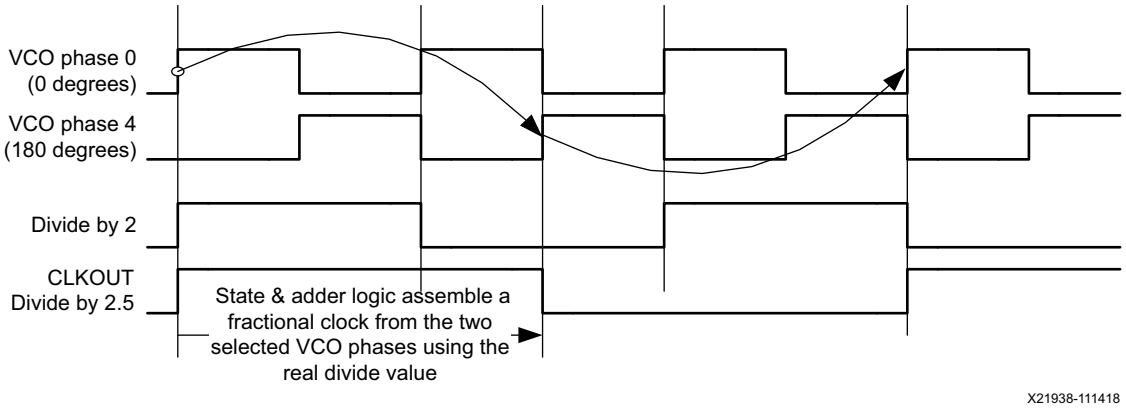

Figure 3-24: **Fractional Counter Mode Example 1**

Example 2

Assume a division by 2.125 is required, then:

- Counter A would take phase 0 (0 degrees) of the VCO
- Counter B would take as start phase 2 (45 degree offset)

Output starts High with rising edge on counter A and goes Low with second rising edge counter B. It goes High again with next rising edge of counter B, counter B switches to VCO phase 90 and the output goes Low again by the second rising edge of that phase and so on.

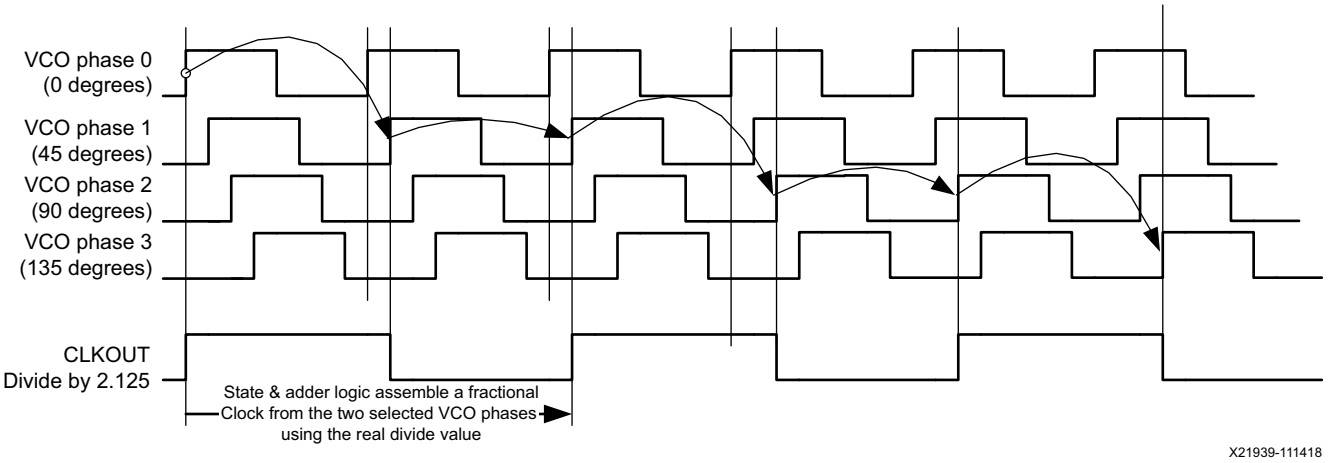

Figure 3-25: **Fractional Counter Mode Example 2**

Reg	**04**												**ADDR: 0x04**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default											1	0	0	1	1	1
Access											R/W	R/W	R/W	R/W	R/W	R/W
5:3	mc_ss_steps_init[2:0]					Start the correct spread based on the SS_MODE attribute. Default value is 100.										
2:0	mc_ss_steps[2:0]					Control the spread of the spread-spectrum clocking based on the SS_MODE attribute. Default value is 111.										

The settings for register 4 are controlled by the SS_MODE attribute of the MMCM. Refer to the Spread-Spectrum Clock Generation section for detailed information on spread-spectrum clocking set up and behavior.

	ss_steps_init	**ss_steps**
DOWN_LOW	100	011
DOWN_HIGH	100	011
CENTER_LOW	100	111
CENTER_HIGH	100	111

Reg	**00**												**ADDR: 0x00**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default													0	0	0	
Access													R/W	R/W	R/W	
3	mc_inv_clkfbin					Enable inversion on the CLKFBIN input. This is the same as setting the attribute IS_CLKFBIN_INVERTED to 1.										
2	mc_inv_clkin2					Enable inversion on the CLKIN2 input. This is the same as setting the attribute IS_CLKIN2_INVERTED to 1.										
1	mc_inv_clkin1					Enable inversion on the CLKIN1 input. This is the same as setting the attribute IS_CLKIN1_INVERTED to 1.										

Register 0 represents the bits that are also available as attributes of the MMCM primitive. For the functional explanation of these bits, refer to the MMCM Attributes section.

PLL Registers

The PLL DRP register set is similar and runs parallel with that of the MMCM. The number of possible changeable registers in the PLL DRP resister set is smaller than that of the MMCM because the PLL has only two clock outputs and doe not use a selectable VCO output multiplexer and interpolator.

Reg	**73**												**ADDR: 0x73**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0		1												
Access	R/W	R/W		R/W												

15	mc_gts_wait	Wait for the GTS_CFG_B signal before starting the LOCKED process.
14	mc_startup_wait	Wait during the configuration start-up cycle for the MMCM to lock.
12	mc_mmcm_en	Enable the PLL.

Reg	**4F**												**ADDR: 0x4F**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0			0	0			0	0			0				
Access	R/W			R/W	R/W			R/W	R/W			R/W				

15	mc_res(3)	Loop filter resistor setting.
12	mc_res(2)	
11	mc_res(1)	
8	mc_res(0)	Loop filter high frequency capacitor setting.
7	mc_lfhf(1)	
4	mc_lfhf(0)	

Reg	**4E**												**ADDR: 0x4E**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0			0	0			0	0			0	1			
Access	R/W			R/W	R/W			R/W	R/W			R/W	R/W			

15	mc_cp(3)	Charge pump setttings.
12	mc_cp(2)	
11	mc_cp(1)	
8	mc_cp(0)	
7	mc_cp_bias_trip_set	Control of the low trip point sense circuit.
4	mc_cp_res(1)	Charge pump reference current control.
3	mc_cp_res(0)	

Reg	**19**												**ADDR: 0x19**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default		0	0	0	1	1	0	0	0	0	0	0	0	0	0	1
Access		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
14:10	mc_lock_fb_dly[4:0]					Window setting for the lock circuit on the feedback clock.										
9:0	mc_unlock_cnt[9:0]					Counter setting the number of clock cycles the PLL needs to have CLKREF and CLKFB misaligned within a certain window before deasserting the LOCKED output. Default value is 1.										

Reg	**18**												**ADDR: 0x18**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default							0	0	0	0	0	0	0	0	0	1
Access							R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
9:0	mc_lock_cnt[9:0]					Counter setting the number of clock cycles the PLL needs to have CLKREF and CLKFB aligned within a certain window before the LOCKED output is asserted. Default value is 1000.										

Reg	**16**												**ADDR: 0x16**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default			0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
13	mc_divck_edge					High to low clock edge transition control.										
12	mc_divck_nocount					Bypass counter.										
11:6	mc_divck_ht[5:0]					Counter high time.										
5:0	mc_divck_lt[5:0]					Counter low time.										

Reg	**15**												**ADDR: 0x15**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default									0	1	0	0	0	0	0	0
Access									R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	mc_ckfbout_edge					High to low clock edge transition control.										
6	mc_ckfbout_nocount					Bypass counter.										
5:0	mc_ckfbout_dt[5:0]					Counter delay.										

Reg	**14**												**ADDR: 0x14**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default			0	1	0	0	0	0	0	1	0	0	0	0	0	1
Access			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
12	mc_ckfbout_en					Counter enable.										
11:6	mc_ckfbout_ht[5:0]					Counter high time.										
5:0	mc_ckfbout_lt[5:0]					Counter low time.										

Reg	**0B**												**ADDR: 0x0B**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default		0	0						0	1	0	0	0	0	0	0
Access		R/W	R/W						R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
14:13	mc_ckoutphy_mode[1:0]					CLKOUTPHY mode (VCO_2X, VCO, VCO_HALF).										
7	mc_ckout1_edge					High to low clock edge control.										
6	mc_ckout1_nocount					Counter bypass.										
5:0	mc_ckout1_dt[5:0]					Counter delay.										

Reg	**0A**												**ADDR: 0x0A**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default				1	0	0	0	0	0	1	0	0	0	0	0	0
Access				R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
12	mc_ckout1_en					Counter enable.										
11:6	mc_ckout1_ht[5:0]					Counter high time.										
5:0	mc_ckout1_lt[5:0]					Counter low time.										

Reg	**09**												**ADDR: 0x09**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default									0	1	0	0	0	0	0	0
Access									R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	mc_ckout0_edge					High to low clock edge control.										
6	mc_ckout0_nocount					Counter bypass.										
5:0	mc_ckout0_dt[5:0]					Counter delay.										

Reg	**08**												**ADDR: 0x08**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default				1	0	0	0	0	0	1	0	0	0	0	0	1
Access				R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
12	mc_ckout0_en					Counter enable.										
11:6	mc_ckout0_ht[5:0]					Counter high time.										
5:0	mc_ckout0_lt[5:0]					Counter low time.										

Reg	**05**												**ADDR: 0x05**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default	0	0	0	0	0	0								0		
Access	R/W	R/W	R/W	R/W	R/W	R/W								R/W		
15:10	mc_in_dly_set[5:0]					Counter delay setting. Control how much delay is inserted in the path.										
9:4	mc_in_dly_mx_dvdd[5:0]															
3	mc_direct_path_cntrl					Reserved.										
2	mc_in_dly_en					Compensation delay enable.										

Reg	**00**												**ADDR: 0x00**			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Default													0		0	
Access													R/W		R/W	
3	mc_inv_clkfbin					Enable inversion on the CLKFBIN input. This is the same as setting the attribute IS_CLKFBIN_INVERTED to 1.										
1	mc_inv_clkin					Enable inversion on the CLKIN1 input. This is the same as setting the attribute IS_CLKIN1_INVERTED to 1.										

VHDL and Verilog Templates and the Clocking Wizard

The VHDL and Verilog code for all clocking resource primitives and Vivado tools language templates are available in the *UltraScale Architecture Libraries Guide* (UG974) [Ref 7].

The Clocking Wizard helps to correctly set up the MMCM and PLL resources. Additionally, the Clocking Wizard reports the jitter and supports phase and frequency synthesis. See *LogiCORE IP Clocking Wizard User Guide* (PG065) [Ref 8] for more information.

Clocking Guidelines

Clocking in a design is not just applying clock buffers, instantiating MMCM and/or PLL, and applying one or a couple of constraints in a XDC file. Clocking and the setup of a clocking network needs attention. To create a design, that is implemented (synthesize, place, and route) using all of Vivado Design Suite features, and when downloaded makes the FPGA function at optimal conditions, follow the guidelines provided in the chapters Clocking Guidelines and Clock Domain Crossing of the *UltraFast Design Methodology Guide for the Vivado Design Suite* (UG949) [Ref 1].

The *UltraFast Design Methodology Guide for the Vivado Design Suite* (UG949) [Ref 1] offers a set of best practices intended to help streamline the design process for new devices. The size and complexity of these designs require specific steps and design tasks to ensure success at each stage of the design. Following these steps and adhering to the best practices will help you achieve your desired design goals as quickly and efficiently as possible. Two other documents that can be useful for designing are:

- *UltraFast Design Methodology Quick Reference Guide* (UG1231) [Ref 9]
- *UltraFast Design Methodology Checklist* (XTP301) [Ref 10]

XILINX

Appendix A

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

Solution Centers

See the Xilinx Solution Centers for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

References

1. *UltraFast Design Methodology Guide for the Vivado Design Suite* (UG949)
2. *UltraScale Architecture Packaging and Pinout User Guide* (UG575)
3. *UltraScale Architecture SelectIO Resources User Guide* (UG571)
4. *Vivado Design Suite User Guide: Using Constraints* (UG903)
5. UltraScale and UltraScale+ device data sheets:
   - *UltraScale Architecture and Products Overview* (DS890)
   - *Zynq UltraScale+ MPSoC Overview* (DS891)
   - *Kintex UltraScale Architecture Data Sheet: DC and AC Switching Characteristics* (DS892)
   - *Virtex UltraScale Architecture Data Sheet: DC and AC Switching Characteristics* (DS893)
   - *Kintex UltraScale+ FPGAs Data Sheet: DC and AC Switching Characteristics* (DS922)
   - *Zynq UltraScale+ MPSoC Data Sheet: DC and AC Switching Characteristics* (DS925)

- *Virtex UltraScale+ FPGAs Data Sheet: DC and AC Switching Characteristics* (DS923)

6. *MMCM and PLL Dynamic Reconfiguration* (XAPP888)
7. *UltraScale Architecture Libraries Guide* (UG974)
8. *LogiCORE IP Clocking Wizard User Guide* (PG065)
9. *UltraFast Design Methodology Quick Reference Guide* (UG1231)
10. *UltraFast Design Methodology Checklist* (XTP301)
11. *Zynq UltraScale+ MPSoC Packaging and Pinout User Guide* (UG1075)
12. *Zynq UltraScale+ MPSoC Technical Reference Manual* (UG1085)
13. *Vivado Design Suite User Guide* (UG908)

Please Read: Important Legal Notices