Designing Using SelectIO Interface Component Primitives

Author: Jim Tatsukawa

Summary

Based on performance requirements, UltraScale™ and UltraScale+™ devices provide various options for I/O interface designs. For high-performance designs, Xilinx® recommends using the high-speed SelectIO™ Wizard in native mode (RX_BITSLICE, TX_BITSLICE, and BITSLICE_CONTROL). Legacy I/O interfaces can be designed using SelectIO interface component mode primitives (IDDRE1, ODDRE1, ISERDESE3, and OSERDESE3). For low-performance designs, IOB registers (FDCE, FDPE, FDRE, or FDSE attached to the I/Os) can be used.

Each of these three methodologies have special timing and design considerations for reliable operation. Native mode methodology is handled through the SelectIO wizard and data sheet timing parameters. For native mode timing budgets, refer to *UltraScale – High Speed Select IO – Example timing budget for Native mode* (Xilinx Answer 68618) [Ref 1]. For interfaces using IOB registers, timing methodology is handled by the Vivado® tools.

This application note describes the recommended usage models and timing analysis methodologies that can be used for I/O interfaces using component mode primitives (ODDRE1, IDDRE1, ISERDESE3, and OSERDESE3).

Input Use Models For Component Mode Applications

This section describes the recommended topologies and various clocking considerations for receiver interfaces when using component mode primitives in UltraScale and UltraScale+ devices. Input logic must deal with the fact that I/O logic has direct connections to the I/O pads, but the clock routing must use dedicated clock buffers to minimize the clock skews. To compensate for the longer clock routing delays, a mixed-mode clock manager (MMCM) can be used, as shown in Figure 1.
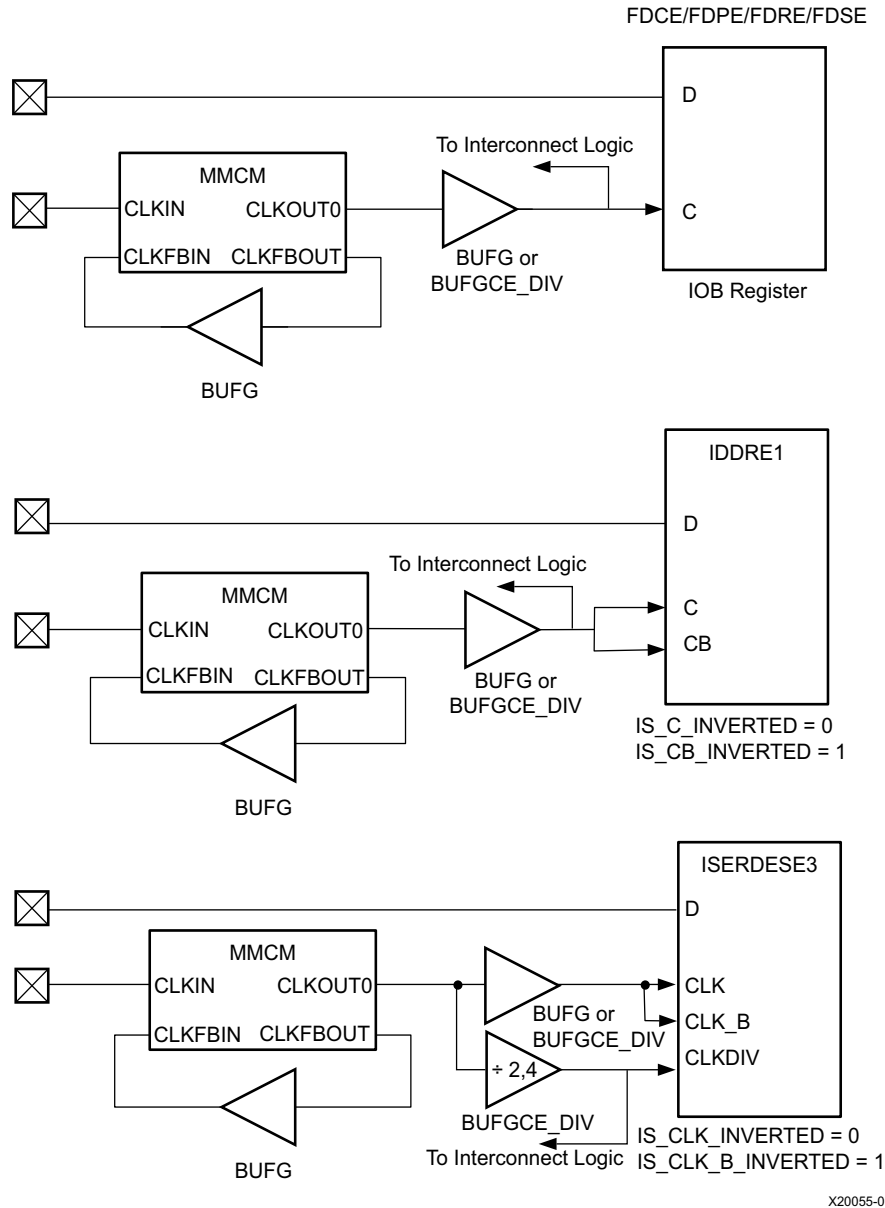
Figure 1: **Using MMCM for Input Logic**

The MMCM has different types of compensation settings. For example, ZHOLD deskews the clock delays to provide zero hold times. In timing analysis, the ZHOLD mode has a large negative delay. The following two mechanisms are at work in the ZHOLD mode.

1. The BUFG routes for CLKOUT0 have a delay similar to the feedback path for CLKFBOUT (see Figure 2).

2. ZHOLD adds additional deskew so that the I/Os within the bank are negatively compensated to ensure zero hold times.

IMPORTANT: *The MMCM is not correctly aligned if the feedback path does not match the routing for CLKOUT0.*
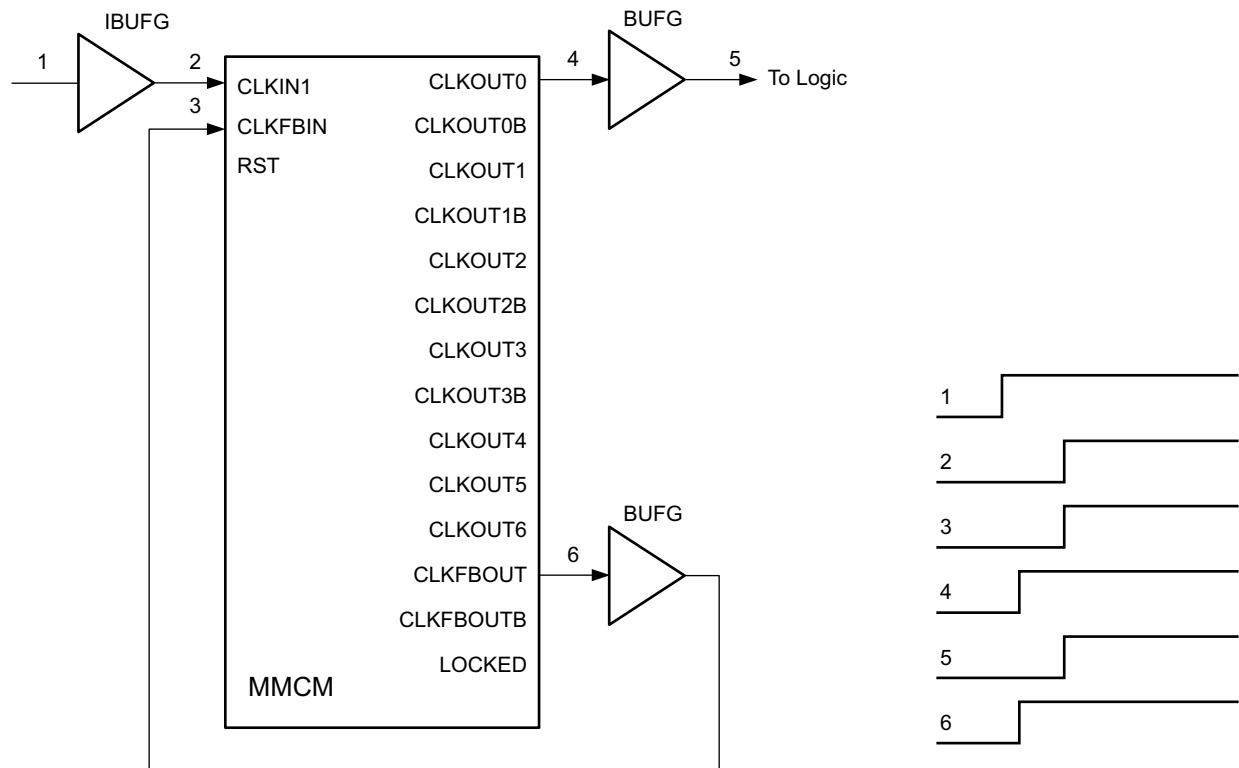

Figure 2:   **Global Clock Network Deskew Using Two BUFGs**

Another compensation setting the MMCM can use is BUF_IN. BUF_IN does not include the additional deskew delay that ensures zero hold times. Consequently BUF_IN has a smaller negative delay for the MMCM in timing anlaysis that only compensates for the clock buffer in the feedback path.

RECOMMENDED: *For many applications such as input interfaces, phase alignment for the clock is critical. The performance degrades for these applications if the MMCM bandwidth attribute is not set to OPTIMIZED or HIGH.*

The MMCM can also use internal feedback. However, internal feedback is not recommended for input logic because the clock network is not part of the feedback path (Figure 3). In timing analysis, the INTERNAL feedback has the smallest compensation because neither the clock routing nor the zero hold delays are being compensated. Internal feedback is primarily used when phase alignment is not needed such as source synchronous output interfaces.
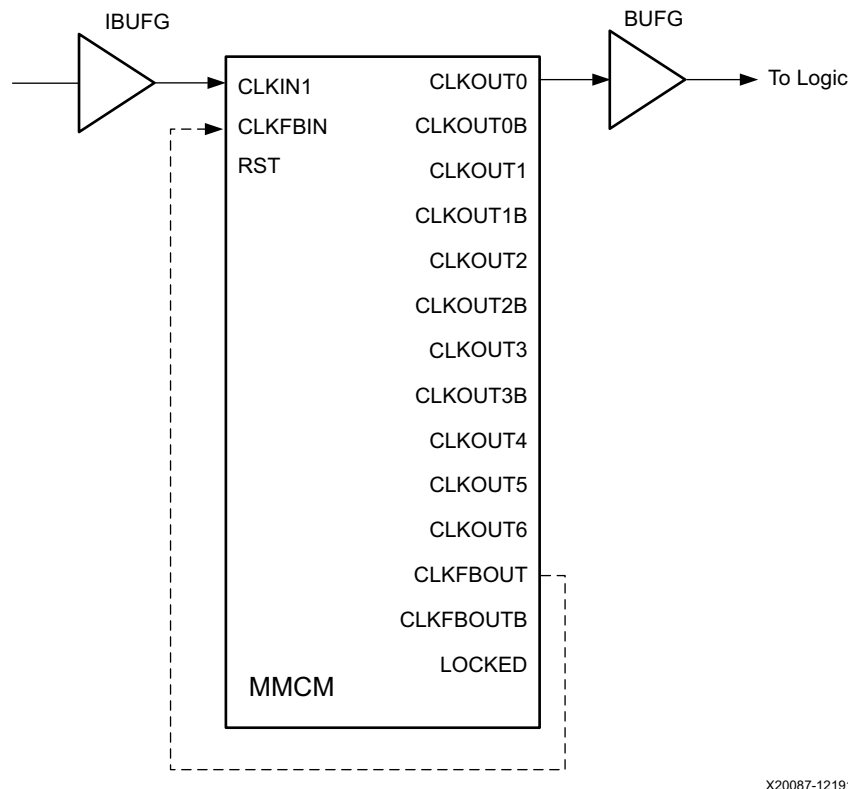
X20087-121917

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

In addition to the different compensations, the MMCM can be used for phase shifting the clocks using either the static or dynamic phase shift. For applications where the dynamic alignment is required, the dynamic phase shifting feature allows the MMCM to be continuously incremented or decremented for an individual clock output using the PSINCDEC DRP port.

In UltraScale and UltraScale+ devices, the phase-locked loop (PLL) primarily supports internal feedback with limited phase shifting. As a result, PLLs are less flexible for receiver applications where clock alignment is crucial. PLLs are better suited for source-synchronous transmit interfaces and native-mode applications where relative delays are more critical.

When the MMCM cannot be used, the clock routing must be balanced using IDELAY as shown in Figure 4. When IDELAY is used in TIME mode, a fixed delay can be used to balance the clock routing delays. To use IDELAY in TIME mode, instantiate the IDELAYCTRL block and follow the component mode reset sequence. Refer to the *UltraScale Architecture SelectIO Resources User Guide* (UG571) [Ref 2] for more information.

Note: When IDELAYCTRL is used, an additional delay factor (ALIGN_DELAY) is added to the IDELAY besides the programmed initial delay value by built-in self-calibration (BISC) routine. Refer to the *UltraScale Architecture SelectIO Resources User Guide* (UG571) [Ref 2] for more information on ALIGN_DELAY.

Note: In TIME mode, delay elements are calibrated for process, voltage, and temperature (PVT) variations by BISC, whereas in COUNT mode they remain uncalibrated and are therefore prone to wider delay tap variations. Refer to the *UltraScale Architecture SelectIO Resources User Guide* (UG571) [Ref 2] for more information.
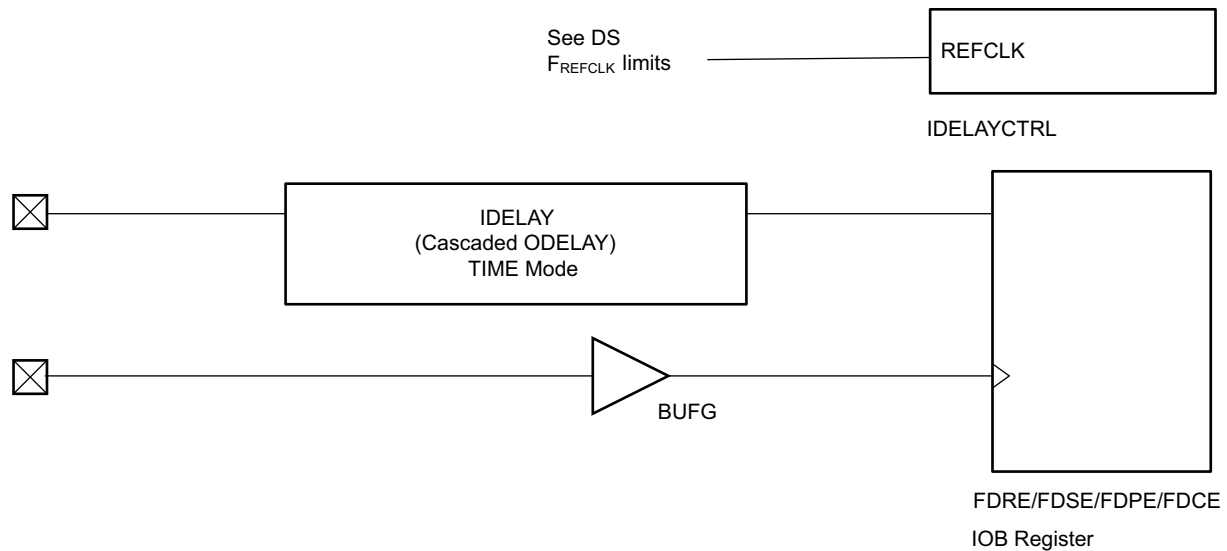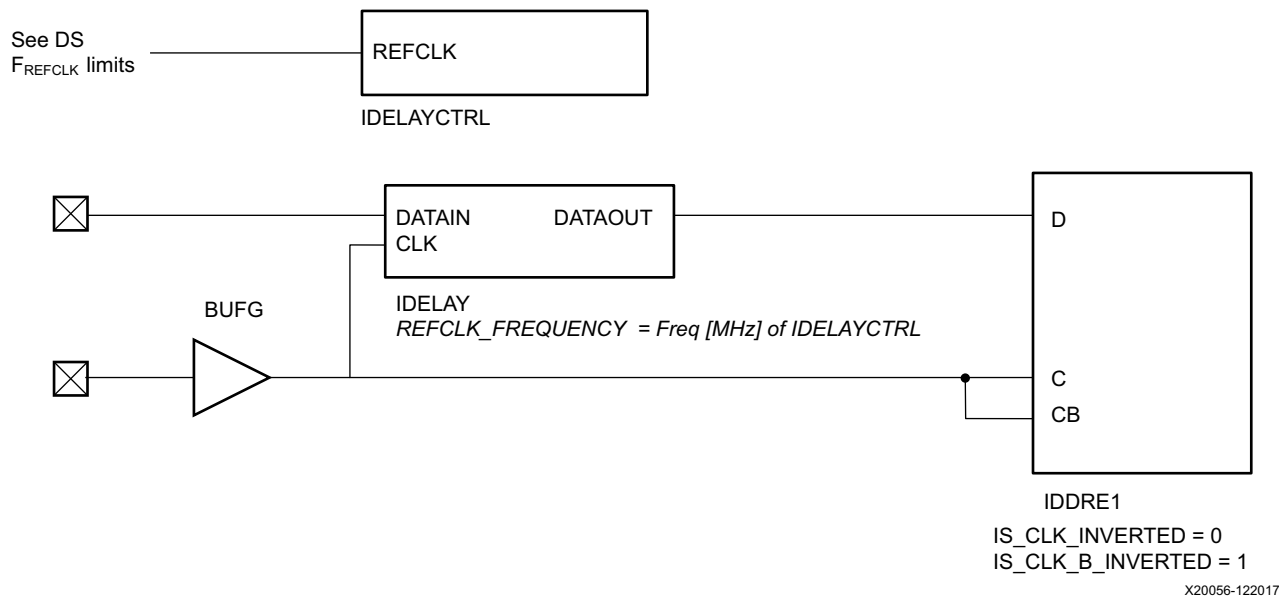
Figure 4: **IDELAY with IOB Flip-Flop**

Note: For F_{REFCLK} limits, see the *UltraScale device data sheets* [Ref 5] and *UltraScale+ device data sheets* [Ref 6].

Using the IDELAY with IDDRE1 is almost the same as using IDELAY with IOB flip-flop except that the IDELAY and IDDRE1 must share the same clock, as shown in Figure 5.
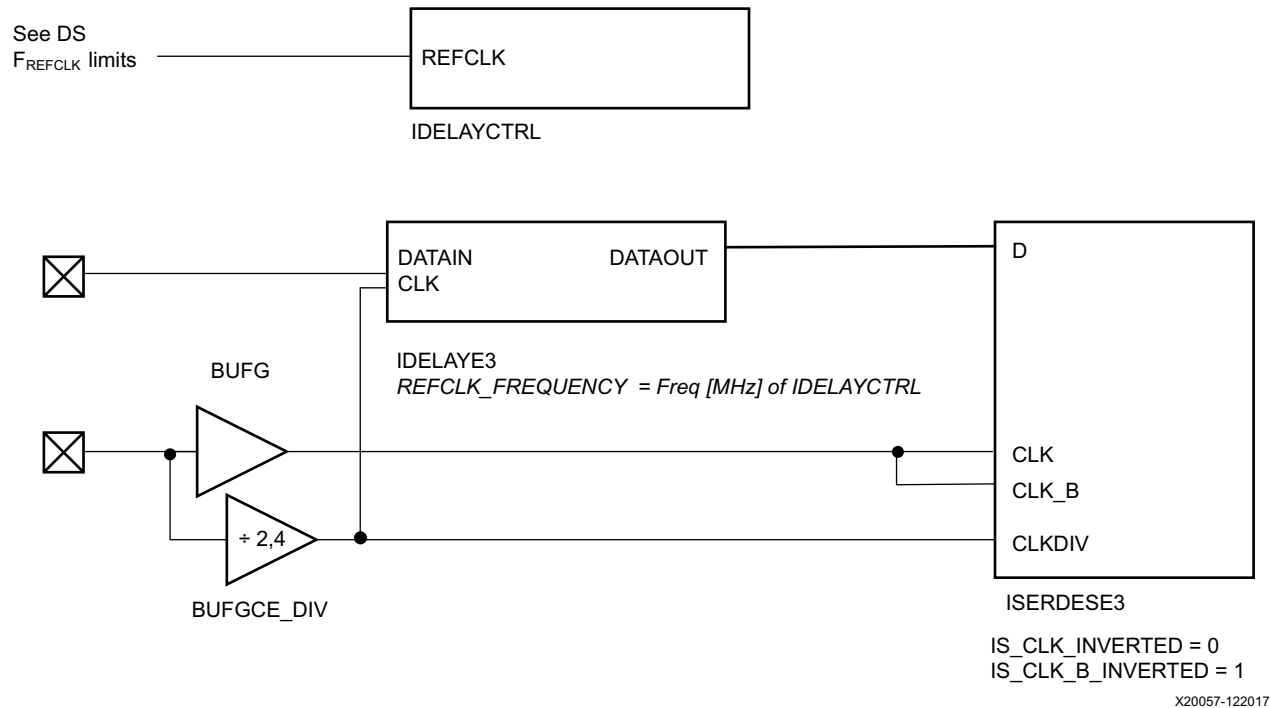

Figure 5: **IDELAY with IDDR**

Note: For F_{REFCLK} limits, see the *UltraScale device data sheets* [Ref 5] and *UltraScale+ device data sheets* [Ref 6].

When using the ISERDESE3, the CLK (IDELAYE3) and CLKDIV (ISERDES3) must share the same clock as shown in Figure 6.


Figure 6: **IDELAY with ISERDES**

Note: For F_{REFCLK} limits, see the *UltraScale device data sheets* [Ref 5] and *UltraScale+ device data sheets* [Ref 6].

For applications where the delays elements are used to dynamically align the data, the IDELAY can be used in COUNT mode, for example, using DELAY_TYPE = VARIABLE and incrementing and decrementing the delays to determine the alignment. COUNT mode is primarily used for dynamic alignment where the fine resolution of the delay line allows small delay adjustments.

Output Use Models for Component Mode Applications

This section describes the recommended topologies and various clocking considerations for transmitter interfaces when using component mode primitives in UltraScale and UltraScale+ devices. Output registers (as shown in Figure 7) and ODDRE1 do not have any special requirements.
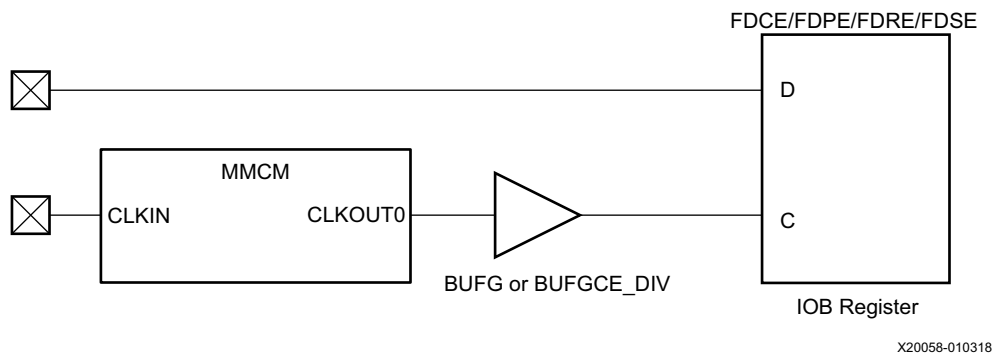
Figure 7:   **Output IOB Register**

For applications that require a fixed alignment with respect to the input clock pad, the MMCM clock outputs support two types of phase shifting. Clock outputs can use either a static phase shift configured by the MMCM settings, or dynamically-configured phase shifts that can be incremented or decremented using the MMCM port.

Note: The feedback, compensation, and clock routing of the MMCM affect the clock delays from the input clock pad.

Many systems use source-synchronous interfaces where the transmitter device forwards an output clock along with the data because it can be difficult to manage routing delays relative to the input clock through the device. Source-synchronous outputs can achieve a high performance because the only delays that require management are the relative delays between the forwarded clock and data.

Source-synchronous interfaces are commonly classified as either edge-aligned or center-aligned. For an edge-aligned interface, the forwarded clock and data are aligned at the near end (see Figure 8). The receiver must take the zero hold time into account for the alignment because the clock and data arrive at the same time. Center-aligned interfaces simplify the receiver by having the transmitter center the clock edges to maximize the setup/hold at the receiver.

www.xilinx.com
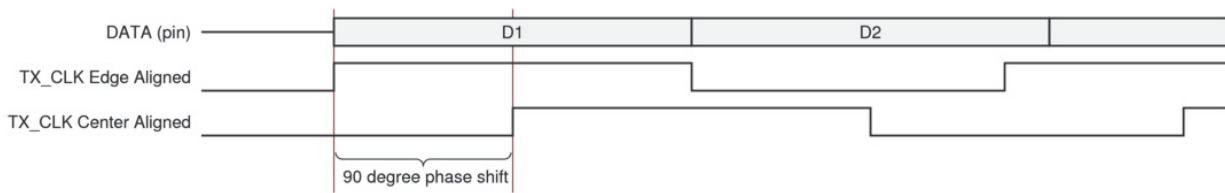
X20059-121917

Figure 8: **Edge-Aligned versus Center-Aligned Source Synchronous Clocks**

The forwarded clock and data must use the same primitives (for example, ODDRE1 or OSERDESE3) and share the same clock source. As shown in Figure 9, the forwarded clock uses an ODDRE1 with D1 = 1 and D2 = 0 that matches the ODDRE1 for the data pins. Similarly, if an OSERDESE3 is used for data, then an OSERDESE3 must be used for the clock path. To match output delays, the clock buffer is NOT directly connected to the output.
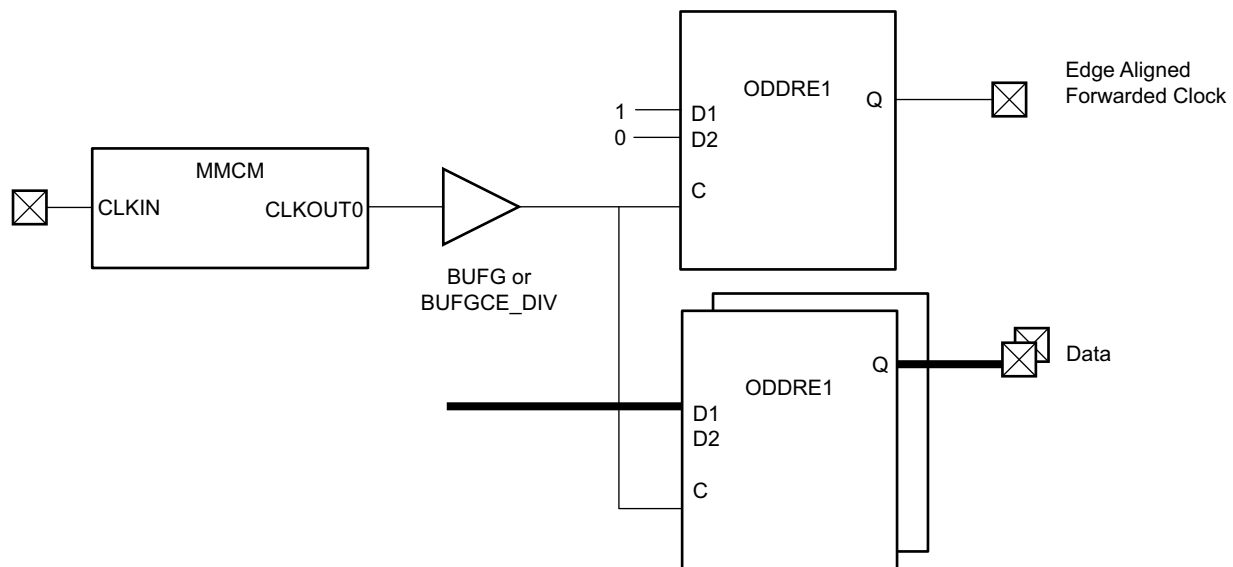

X20062-121917

Figure 9: **Edge-Aligned Source Synchronous DDR Outputs**

A second MMCM output must be used to center align the clock to the data. As shown in Figure 10, CLKOUT1 is added for the forwarded clock. For DDR applications a 90° phase shift is used. The phase alignment between CLKOUT0 and CLKOUT1 is critical. Refer to the Clock Buffers and Clock Routing section in *UltraScale Architecture Clocking Resources User Guide* (UG572) [Ref 3] for additional details on maintaining consistent clock routing.

Note: In the example CLKOUT0/CLKOUT1 are used for illustrative purposes. Any valid MMCM configuration can be used with a 90 degree difference in the output phases. When using multiple clock outputs, see Clocking Requirements for details on CLOCK_DELAY_GROUP.

The output delays for the ODDRE1 and OSERDESE3 are different. Therefore, the same primitives must be used to ensure that the clock and data delays match.
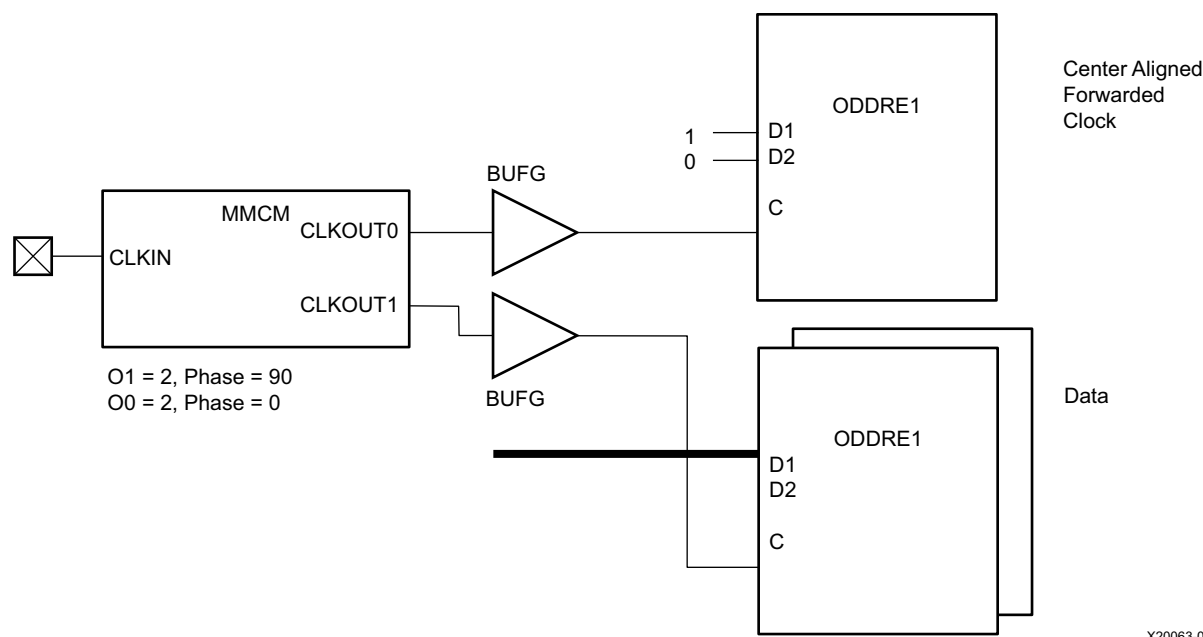
Figure 10: **Center-Aligned Source Synchronous DDR Outputs**

For applications requiring the OSERDESE3 (DATA_WIDTH = 4 or 8), Figure 11 shows how the clocks are connected. In this example the BUFG and BUFGCE_DIV are used to generate the required clocks for the OSERDESE3. As shown in Figure 11, using a single CLKOUT of the MMCM to drive both CLK and CLKDIV minimizes phase error (PE).
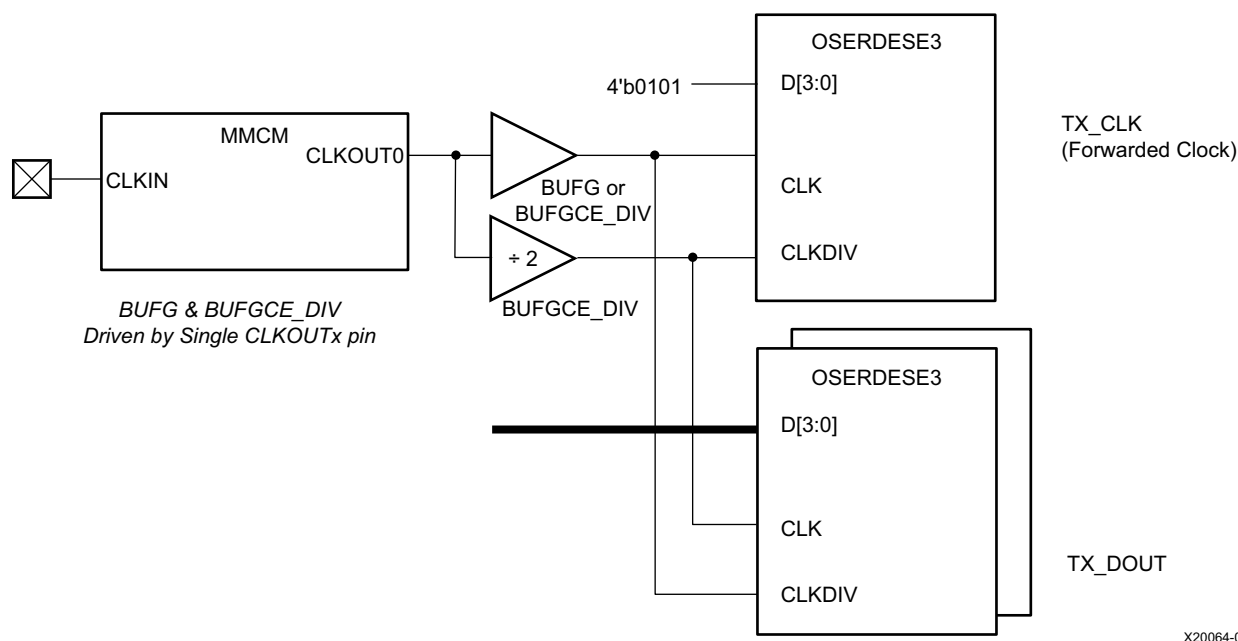
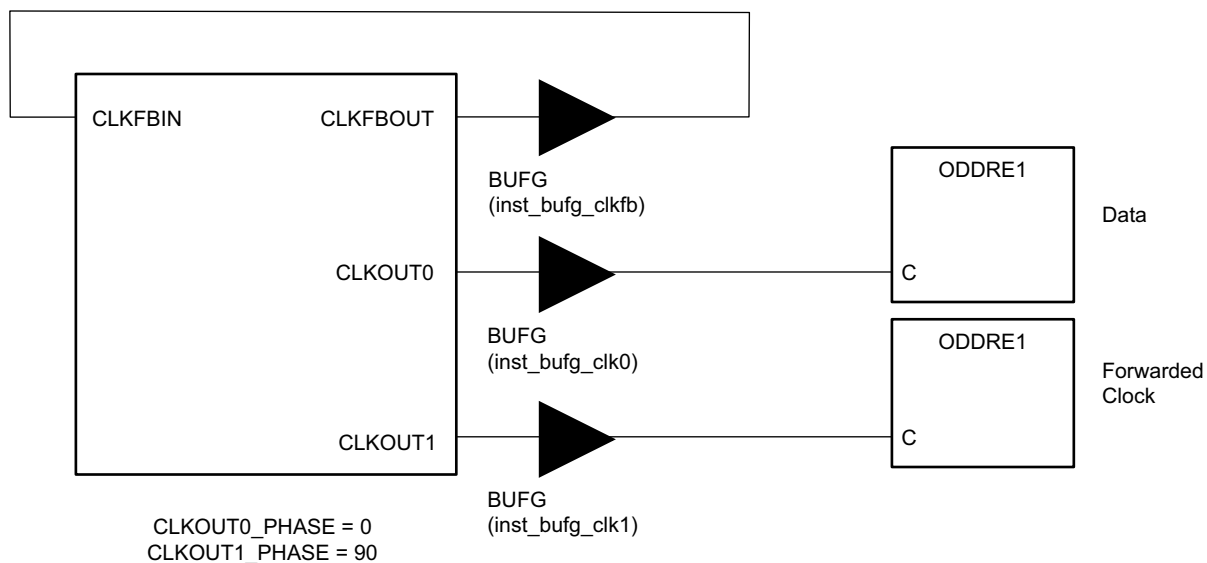*Figure 11:* **Source-Synchronous OSERDESE3 Outputs with Edge-Aligned Clock (DATA_WIDTH = 4)**

Clocking Requirements

Clocking directly affects the performance of the I/O interface. For more information on clock routing and clock buffers in UltraScale and UltraScale+ devices, refer to the Clocking Architecture Overview section in the *UltraScale Architecture Clocking Resources User Guide* (UG572) [Ref 3].

Keeping the input clock, MMCM, global clock buffer, CLOCK_ROOT, and I/Os in the same bank provides the best performance because the user-accessible clocking structures are limited to global clock buffers. Performance can degrade if routing spreads across banks and I/O columns. Vivado tools can be used to analyze the clock skews. Crossing super logic regions (SLRs) is not recommended for performance reasons.

The clock routing varies depending upon the requirements of the clock router. Consistent clock routing can be achieved by constraining the clock routing. As shown in Figure 12, for the MMCM feedback path, CLKOUT0 and CLKOUT1 must be matched to ensure that the phase shift settings have the desired affect.

Note: LOCKED indicates that the clocks are aligned and can be used. The clock outputs are active while acquiring the lock.

CLKFBIN CLKFBOUT

BUFG
(inst_bufg_clkfb)

ODDRE1

Data

CLKOUT0

C

BUFG
(inst_bufg_clk0)

ODDRE1

Forwarded
Clock

CLKOUT1

C

BUFG
(inst_bufg_clk1)

CLKOUT0_PHASE = 0
CLKOUT1_PHASE = 90

X20066-121917

Figure 12: **Using MMCM with Multiple Outputs**

To correctly control clock skews, define a CLOCK_DELAY_GROUP on the net segment directly driven by the output of the clock buffer:

```
set_property CLOCK_DELAY_GROUP <Clock Delay Group Name> [get_nets-of_objects
[get_pins inst_bufg_clkfb/O] ]
set_property CLOCK_DELAY_GROUP <Clock Delay Group Name> [get_nets-of_objects
[get_pins inst_bufg_clk0/O] ]
set_property CLOCK_DELAY_GROUP <Clock Delay Group Name> [get_nets-of_objects
[get_pins inst_bufg_clk1/O] ]
```

The CLOCK_ROOT for the clock buffers changes depending on the loading requirements. Although the MMCM and BUFG can be placed in CLOCK_REGION X0Y0, the clock root is determined by the fanout of the clock and could be placed in a different clock region. As a result the USER_CLOCK_ROOT is also needed:

```
set_property USER_CLOCK_ROOT {X0Y0} [get_nets inst_bufg_clkfb]
set_property USER_CLOCK_ROOT {X0Y0} [get_nets inst_bufg_clk0]
set_property USER_CLOCK_ROOT {X0Y0} [get_nets inst_bufg_clk1]
```

It is essential to understand the alignment of a divided clock for the clocking solutions that require the use of BUFGCE_DIV. For example, in Figure 13 the BUFGCE_DIV is used to create the divided-down clock for CLKDIV to save an output.

X20067-121917
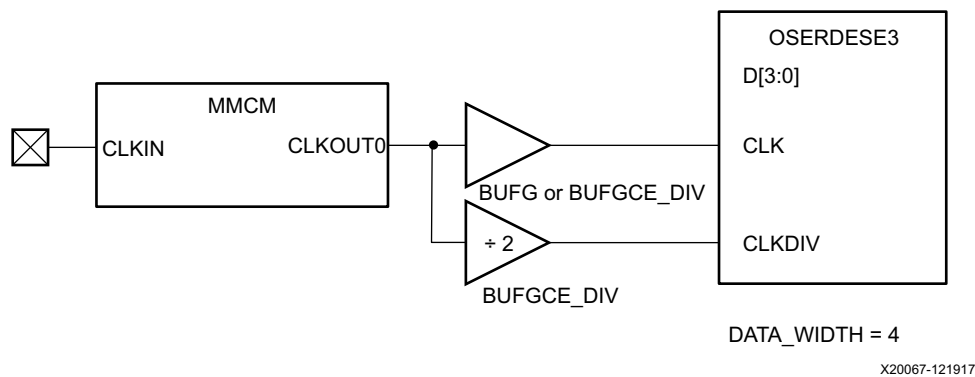
Figure 13: **OSERDESE3 Using BUFG and BUFGCE_DIV Driven by a Single MMCM Output**

The BUFGCE_DIV output can be aligned to any of the input clocks when used as a clock divider. Figure 14 shows the possible alignments. CLK_DIV2 A and CLK_DIV2 B show the two alignments when the divide value is 2. CLK_DIV4 A/B/C/D show the four alignments that are possible when the divide value is 4. Using the clear (CLR) and clock enable (CE) inputs, the counters can be aligned to a given clock edge.

X20060-121917
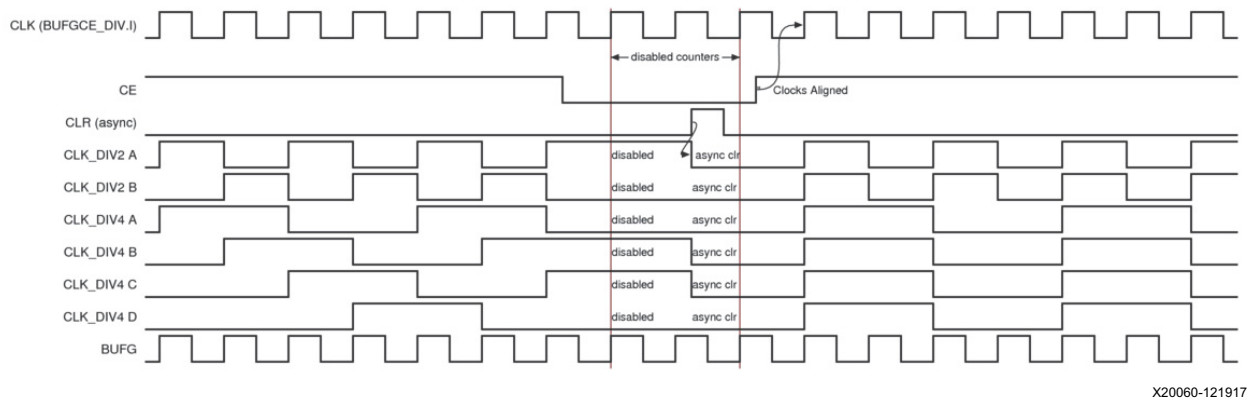
Figure 14: **BUFGCE_DIV Alignment with BUFGCE_DIVIDE = 2 or 4**

Performance Expectations

The implementation of the I/O interfaces becomes more critical with increasing design performance. The common interfaces can be roughly categorized as being either system-synchronous for low performance communication or source-synchronous for high performance.

System Synchronous Using Component Primitives

System synchronous means that the interfaces between the transmitting and receiving devices are clocked by the single common system clock (see Figure 15). This is the most commonly used interface for simplicity of the setup. However, this setup sacrifices performance and adds complexity. Setup and hold timing analysis is performed across fast as well as slow processes by the Vivado tools, and all delay variations from the input pin of the system clock are taken into account.
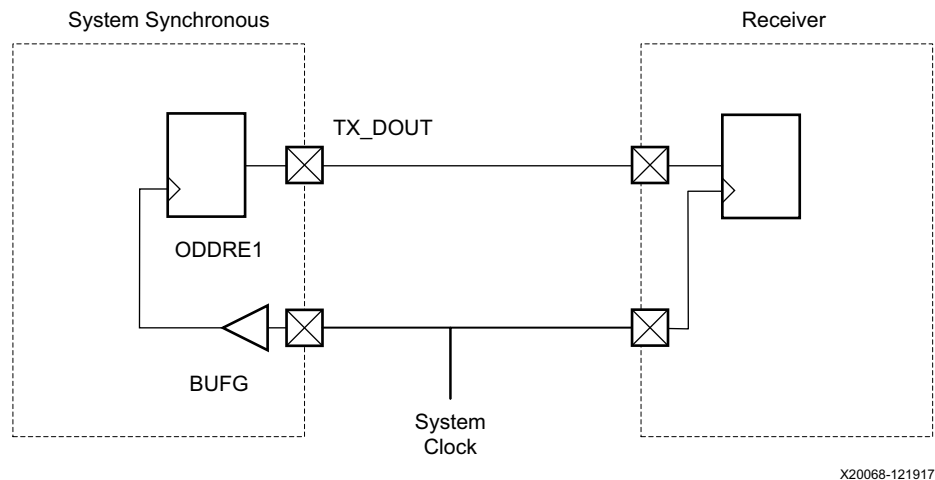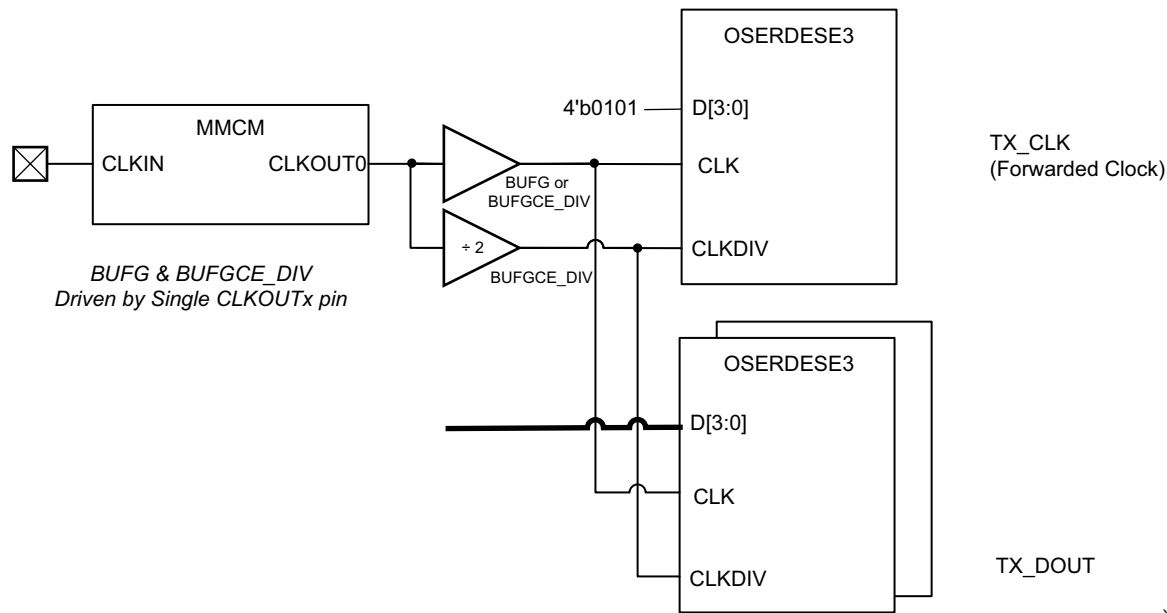

Figure 15:   **System Synchronous**

Output—Source Synchronous Using Component Primitives

Source-synchronous outputs must be analyzed differently when the clock is edge-aligned, as opposed to center-aligned.

Output—Edge-Aligned Source Synchronous Output Bus Timing

Figure 16: **Edge-Aligned Source Synchronous Output Bus**

The Vivado tools analyze timing across process corners to determine the output delays for the output bus TX_DOUT (Figure 16). For source-synchronous designs, an accurate estimate for the amount of error for an output bus requires information from the Vivado tools as well as characterization data. This can be broken down into the following requirements.

Source-Synchronous Output Bus Skew

The source-synchronous output bus skew captures the variation in delays associated with routing and logic differences. For example, a design that spans multiple banks cannot achieve the same performance as a design in which the clock and data reside within a single bank. As shown in Equation 1, the source-synchronous output bus skew comprises the datasheet_bus_skew and source_sync_transmit_delay_variation.
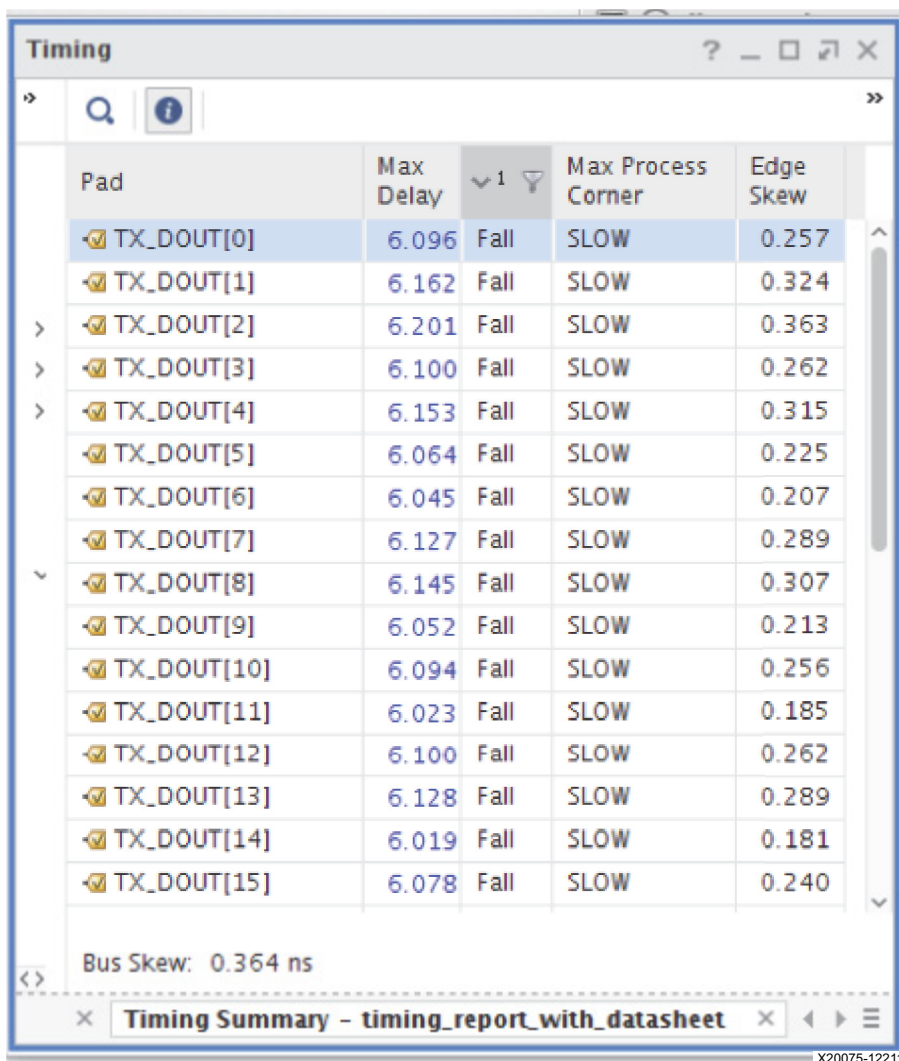
$$\text{source\_sync\_out\_bus\_skew} = \text{datasheet\_bus\_skew} + \text{source\_sync\_transmit\_delay\_variation} \qquad \textit{Equation 1}$$

Datasheet_bus_skew

The data sheet bus skew uses the timing information from the Vivado tools (maximum delays and slow process) to analyze routing variations for a design based on the pins, logic, and clocking components used. For example, the following timing summary uses the `-datasheet` TCL option to create the data sheet report:

```
report_timing_summary -delay_type max -check_timing_verbose -max_paths 100 -input_pins
-datasheet -name timing_report_with_datasheet
```

When reviewing the data sheet report, use either the rising (r) or falling (f) data to determine the maximum and minimum delays. Effects of duty cycle distortion (DCD or $T_{OUTDUTY}$) are managed separately when reviewing the clocking errors. For the data sheet report shown in Figure 17, the bus skew is 182 ps (6.201 ns for TX_DOUT[2] – 6.019 ns for TX_DOUT[14]).

Timing				
Pad	Max Delay		Max Process Corner	Edge Skew
☑ TX_DOUT[0]	6.096	Fall	SLOW	0.257
☑ TX_DOUT[1]	6.162	Fall	SLOW	0.324
☑ TX_DOUT[2]	6.201	Fall	SLOW	0.363
☑ TX_DOUT[3]	6.100	Fall	SLOW	0.262
☑ TX_DOUT[4]	6.153	Fall	SLOW	0.315
☑ TX_DOUT[5]	6.064	Fall	SLOW	0.225
☑ TX_DOUT[6]	6.045	Fall	SLOW	0.207
☑ TX_DOUT[7]	6.127	Fall	SLOW	0.289
☑ TX_DOUT[8]	6.145	Fall	SLOW	0.307
☑ TX_DOUT[9]	6.052	Fall	SLOW	0.213
☑ TX_DOUT[10]	6.094	Fall	SLOW	0.256
☑ TX_DOUT[11]	6.023	Fall	SLOW	0.185
☑ TX_DOUT[12]	6.100	Fall	SLOW	0.262
☑ TX_DOUT[13]	6.128	Fall	SLOW	0.289
☑ TX_DOUT[14]	6.019	Fall	SLOW	0.181
☑ TX_DOUT[15]	6.078	Fall	SLOW	0.240

Bus Skew: 0.364 ns

Timing Summary – timing_report_with_datasheet

X20075-122117

*Figure 17:* **Example Data Sheet Report**

Note: The datasheet report may be run with or without the package skew. If the package skews are compensated by PCB routing, disable the package skews using the `disable_flight_delays` option: `config_timing_analysis -disable_flight_delays true`.

Analyzing only the p-side (master) of a differential pair simplifies the analysis because differential outputs are covered by the data sheet specification for buses with differential outputs.

The maximum delays reported in the data sheet report do not directly appear in the properties because the maximum delay value must be calculated as shown in Equation 2. After running the data sheet report, the timing path can be reviewed by clicking one of the delay values (see Figure 18). When selecting the 6.096 ns delay (TX_DOUT[0] Fall), the timing path is shown in the Path Properties window, but the value of 6.096 does not appear in the path properties. Instead, the arrival time is 6.843 ns which includes the (f) 0.800 ns because the falling edge is being analyzed. As a result, the maximum delay from the data sheet report includes clock uncertainty and excludes the initial falling edge delay of 0.800 ns (see Equation 3).

$$\text{Max Delay} = \text{Arrival Time} - \text{CLKOUT0 fall at 0.800 ns} + \text{Clock Uncertainty} \qquad \textit{Equation 2}$$

$$6.096 \text{ ns (Max Delay)} = 6.843 - 0.800 + 0.053 \qquad \textit{Equation 3}$$
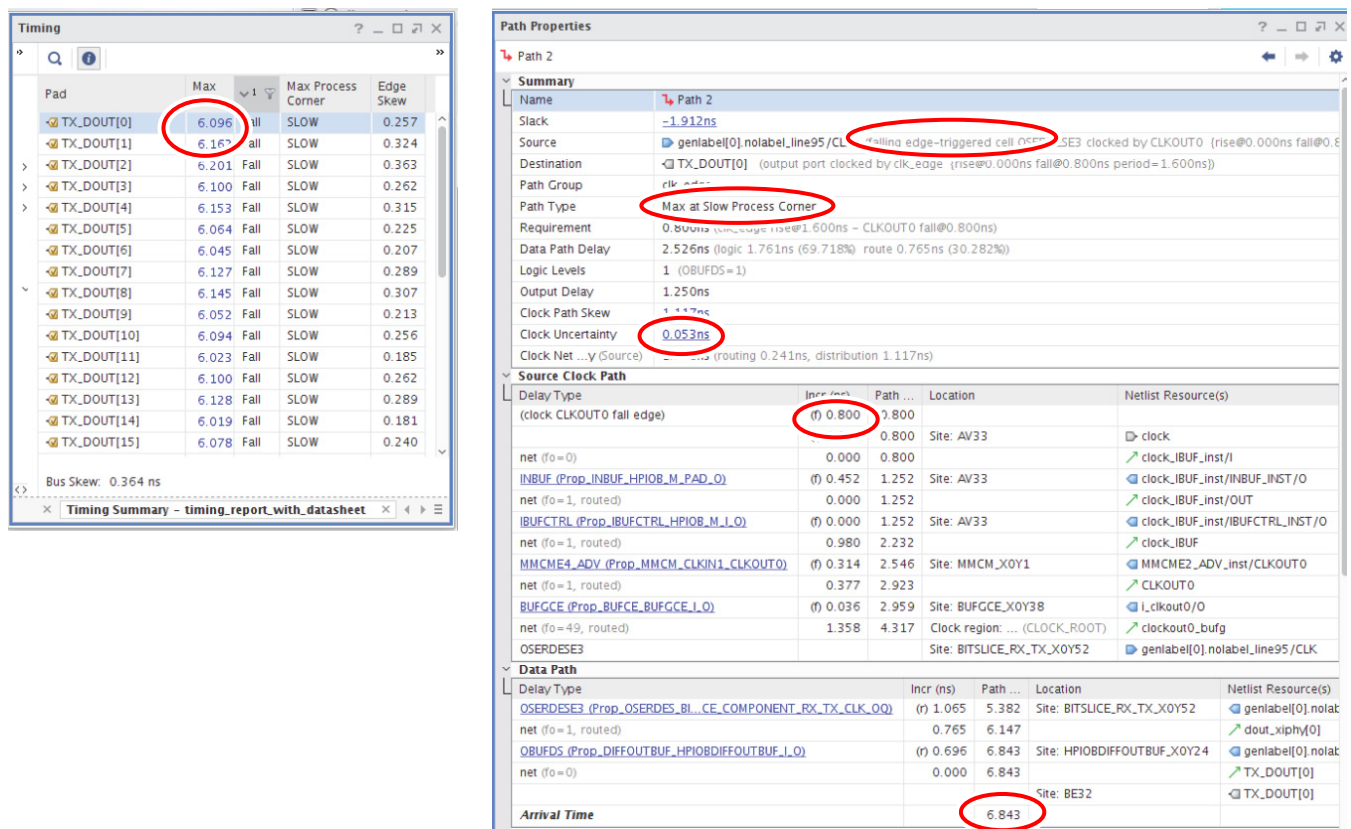
Figure 18: **Calculating Maximum Delays from Path Properties**

Source_sync_transmit_delay_variation

Bus skew calculation from Datasheet_bus_skew accounts for the routing differences. To account for variation of the switching delays, an additional 80 ps must be added based on the silicon characterization for all speed grades of UltraScale and UltraScale+ devices.

Source-Synchronous Clocking Error

Source-synchronous clocking errors can reduce the ideal bit period or unit interval. The source-synchronous clock error is the sum of the duty cycle distortion and uncertainty, as shown in Equation 4.

$$\text{source\_sync\_clock\_err} = \text{Duty Cycle Distortion} (T_{OUTDUTY}) + \text{Uncertainty}$$
Equation 4

Duty Cycle Distortion

The duty cycle of the clocking buffers must be taken into account because ODDRE1 and OSERDESE3 transmit data on the rising and falling edges of the clock. As shown in Figure 19, DCD erodes the bit period. When using the MMCM or PLL, the output counters determine the duty cycle which is typically programmed at 50%. But system noise or power fluctuations can impact the duty cycle after the MMCM or PLL because the clock signal is routed from the MMCM or PLL block using clock buffers, and ultimately to the leaf clock that drives the I/O bank.

The duty cycle for the MMCM (MMCM_$T_{OUTDUTY}$) or PLL (PLL_$T_{OUTDUTY}$) is specified in the data sheet and includes the output clock buffers to ODDRE1/OSERDESE3.

DCD does not affect the outputs in designs that do not transmit data on both edges of the clock. In Figure 19, if D1 and D2 are the same value, switching is not seen at the falling edge of the clock, and the DCD is not seen in operation.
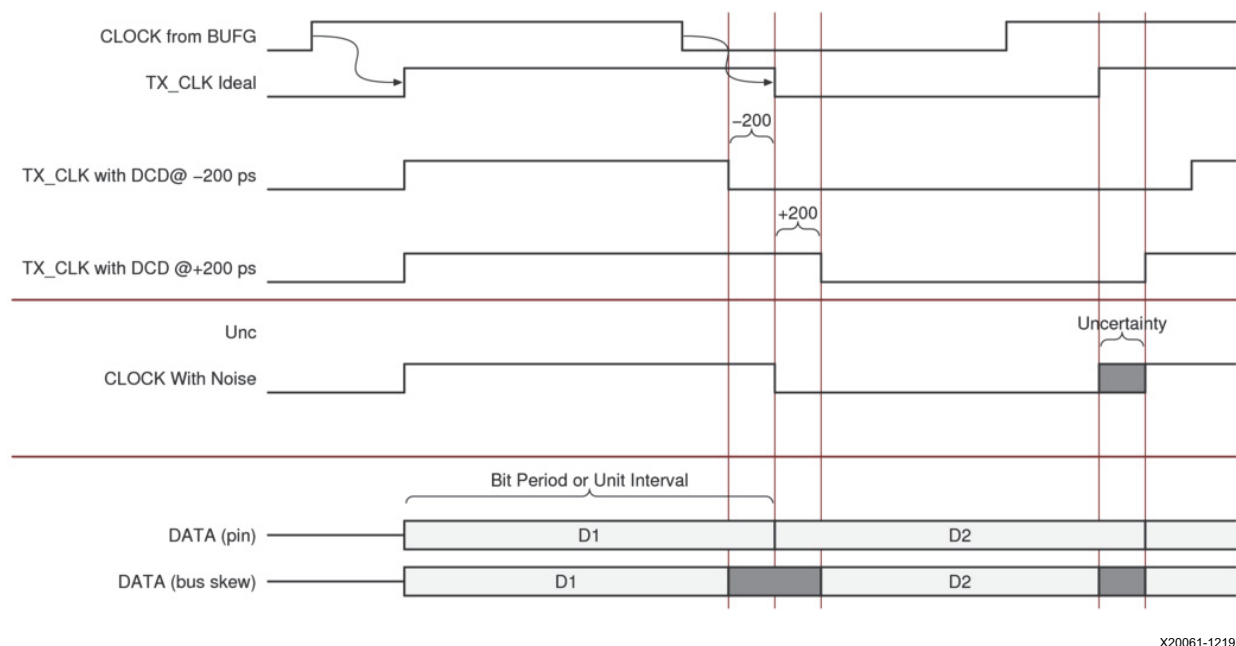

X20061-121917

Figure 19: **DCD and Uncertainty for Source-Synchronous Outputs (ODDRE1 Shown)**

Clock Uncertainty (Clock_unc_ss_edge_aligned_output)

Clock uncertainty is the variability of a clock edge from the ideal location or the result of system jitter, discrete jitter, and phase error attributable to the clocking components (CMT, clock buffers, and power noise). For a source-synchronous bus, phase error can be ignored because TX_CLK and TX_DOUT are driven by the same clock buffer.

The values required for clock uncertainty are listed in the path properties. After running the data sheet report, the timing path can be reviewed by clicking one of the delay values (Figure 20). When selecting the 5.903 delay (dout[15]), the timing path is shown in the Path Properties window. In this example, the clock uncertainty is reported as 0.053 ns, which is context sensitive. In the Path Properties window (Figure 20), for a clock uncertainty value of 0.053 ns, the clock uncertainty equation (Equation 5) shows that the total system jitter (TSJ) is 0.071 ns and discrete jitter (DJ) is 0.78 ns.
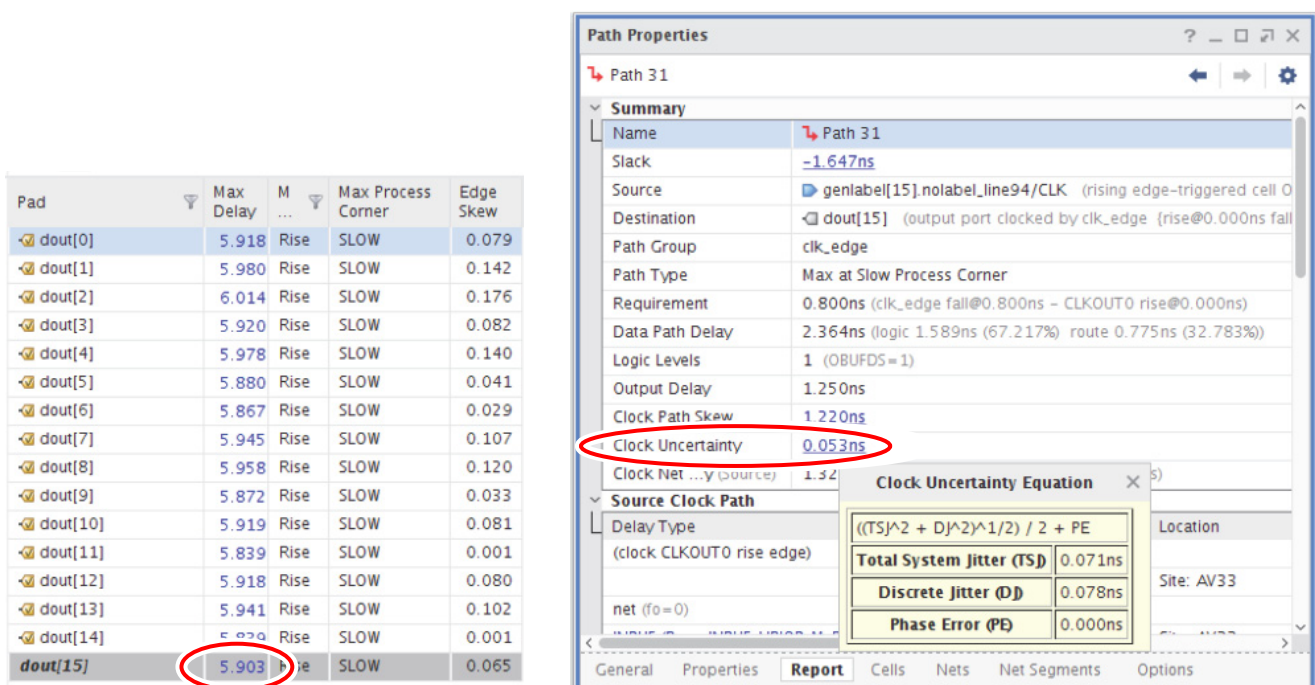
Figure 20: **Analyzing Data Sheet Path Properties for Output Bus**

For the edge-aligned source-synchronous outputs, phase error does not affect the transmitter error and can be removed even when the value is not zero. The clock uncertainty for this example MMCM configured for 1250 Mb/s, taking into account TSJ and DJ, is as shown in Equation 5 and Table 1.

$$Clock\_unc\_ss\_edge\_aligned\_output = \frac{(TSJ^2 + DJ^2)^{1/2}}{2}$$

Equation 5

Table 1: **Clock_unc_ss_edge_aligned_output**

Parameter	Value (ps)
TSJ	71
DJ	78
clock_unc_ss_edge_aligned_output	53

In source-synchronous designs there is no clock uncertainty for the first edge because the forwarded clock (TX_CLK) and data (TX_DOUT) are launched off the same clock edge. While clock and data continue to be transmitted off the same clock edge, the uncertainty of the clock can compress the bit period and must be taken into account.

Figure 21 illustrates various clock error factors that affect the transmit bit period. DCD appears in an eye diagram as two distinct edges of the bit period. Figure 21 has been annotated to show how a duty cycle of 48%/52% should be interpreted.
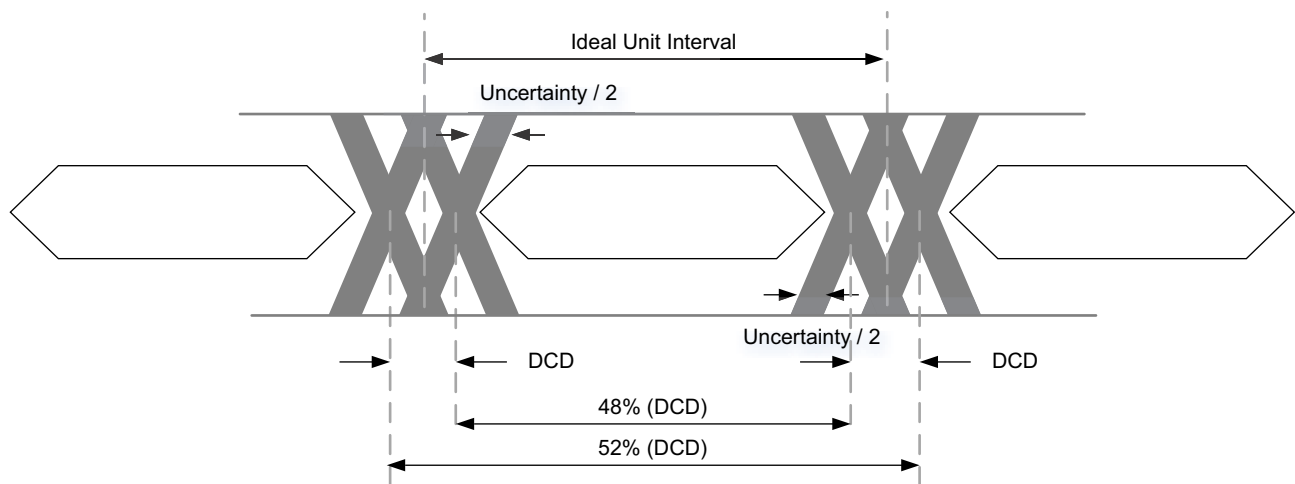
X20074-121917

Figure 21: **Normalized Eye Diagram with DCD and Clock Uncertainty**

Combined Source-Synchronous Edge-Aligned Transmitter Timing Budget

The total transmitter timing budget of a source-synchronous edge-aligned transmitter interface is a combination of the output bus skew and the clock error. The output bus skew is a combination of the bus skew from the data sheet report and the transmit delay variation (Equation 6). Clock error accounts for the duty cycle distortion ($T_{OUTDUTY}$) and clock uncertainty of the clock network (Equation 7). The equation for the total transmit timing budget is shown in Equation 8.

$$\text{source\_sync\_out\_bus\_skew} = \text{datasheet\_bus\_skew} + T_{OUTPUT\_LOGIC\_DELAY\_VARIATION}$$ *Equation 6*

$$\text{source\_sync\_clock\_error} = \text{Duty Cycle Distortion} (T_{OUTDUTY}) + \text{Clock Uncertainty}$$ *Equation 7*

$$\text{Transmit Error} = \text{source\_sync\_out\_bus\_skew} + \text{source\_sync\_clock\_err}$$ *Equation 8*

For the edge-aligned example being analyzed, the transmit error is 508 ps, as shown in Table 2.

Table 2: **Source-Synchronous Edge-Aligned Transmitter Timing Budget**

Parameter	Value (ps)
datasheet_bus_skew	182
$T_{OUTPUT\_LOGIC\_DELAY\_VARIATION}$	100 (UltraScale Devices)
	80 (UltraScale+ Devices)
$T_{OUTDUTY}$ (Duty Cycle Distortion)	200
Clock_unc_ss_edge_aligned_output	53
Total	535 (UltraScale Devices)
	515 (UltraScale+ Devices)

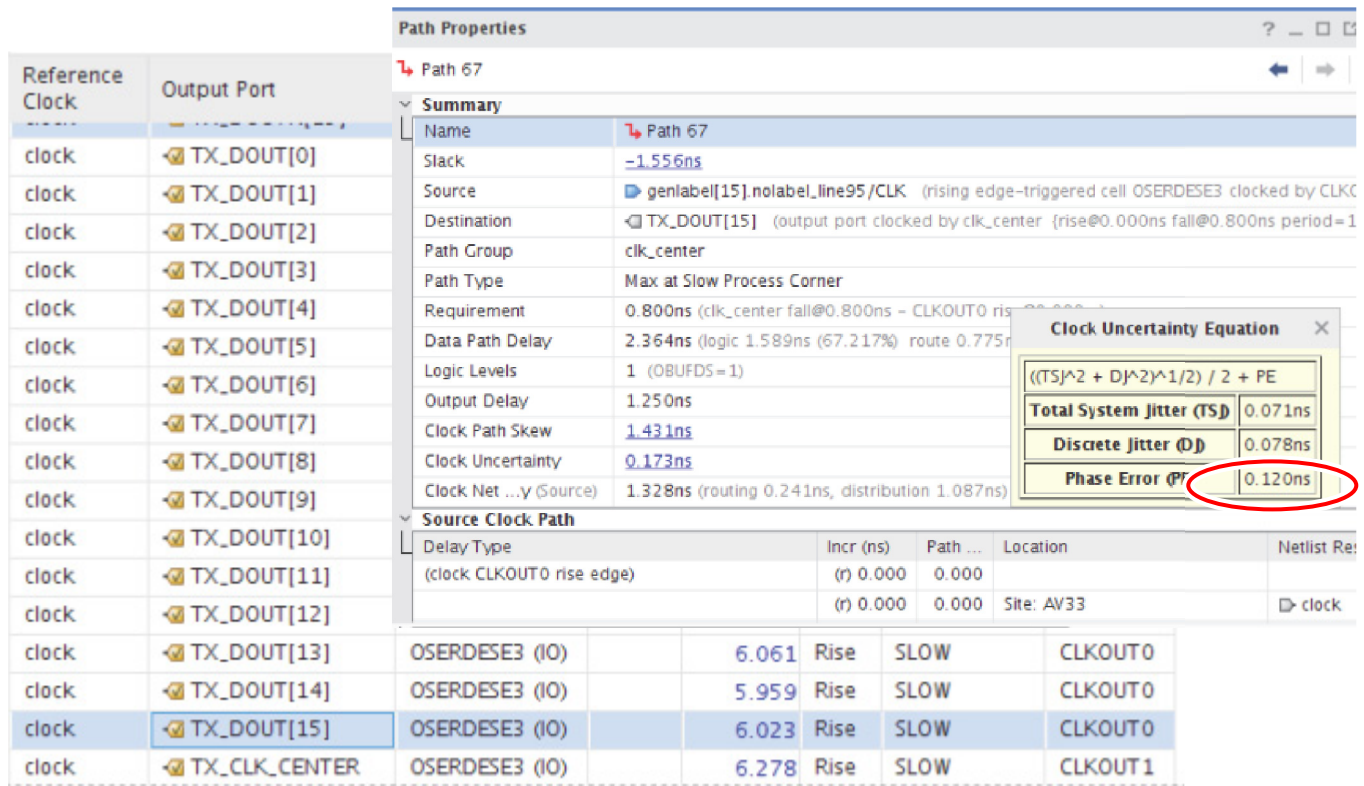## *Output—Center-Aligned Source-Synchronous Output Bus Timing*
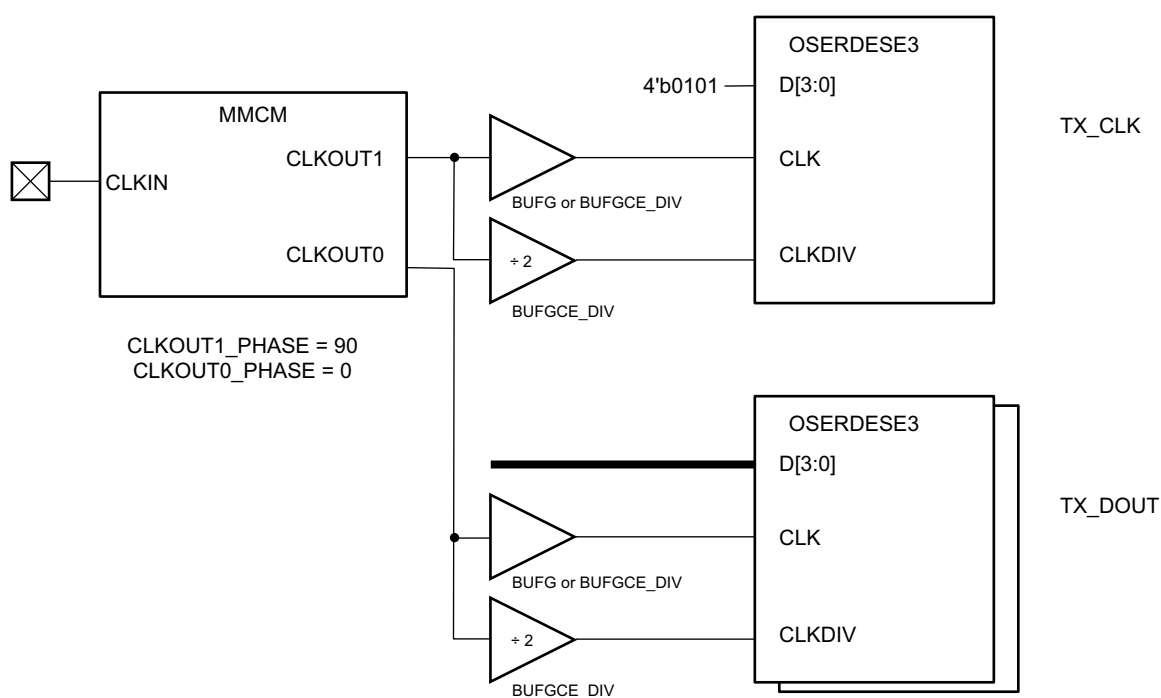
Figure 22: **Source-Synchronous Output Bus (OSERDESE3) Center-Aligned**

The center-aligned source-synchronous output bus timing budget is very similar to the edge-aligned source-synchronous output bus timing with the exception that the forwarded clock (TX_CLK) must be centered at the middle of the bit period for TX_DOUT. The datasheet_bus_skew, source_sync_transmit, and DCD are the same as described in Output—Edge-Aligned Source Synchronous Output Bus Timing.

Clock Uncertainty (Clock_unc_ss_center_aligned_output)

The clock uncertainty in Figure 23 shows a phase error of 120 ps in this example. For center-aligned clocks, the static phase offset between any two MMCM clock outputs must be taken into account. However, for consistency this phase error is handled independently of the clock uncertainty.

Figure 23: **Example Data Sheet Report Center-Aligned for TX_DOUT**

The clock uncertainty for this example MMCM configured for 1250 Mb/s is 53 ps taking into account TSJ and DJ as shown in Equation 9.

$$\text{Clock\_unc\_ss\_center\_aligned\_output} = \frac{(TSJ^2 + DJ^2)^{1/2}}{2}$$

Equation 9

Static Phase Offset for MMCM/PLL Outputs

The timing budget for the center-aligned source-synchronous output bus timing budget only needs to add an allowance for the phase difference between the different clock outputs. For the MMCM this can be taken from the data sheet as MMCM_T$_{STATPHAOFFSET}$ or PLL_T$_{STATPHAOFFSET}$.

The static phase offset is measured from the outputs of the MMCM to the clock buffers within the same clock region and does not include routing variation from the output of the clock buffers. Thus, it is critical to follow the Clocking Requirements to ensure that the clock routing from the clock buffer is properly aligned.

Combined Source-Synchronous Center-Aligned Transmitter Timing Budget

The total transmitter timing budget of a source-synchronous center-aligned transmitter interface is a combination of the output bus skew and the clock error. The output bus skew is a combination of the bus skew from the Vivado tools data sheet report and the transmit delay variation (Equation 10). Clock error accounts for the duty cycle distortion, clock uncertainty, as well as the static phase offset of the clock network (Equation 11). The equation for the total transmit timing budget is shown in Equation 12.

$$\text{source\_sync\_out\_bus\_skew} = \text{datasheet\_bus\_skew} + T_{\text{OUTPUT\_LOGIC\_DELAY\_VARIATION}} \qquad \textit{Equation 10}$$

$$\text{source\_sync\_clock\_err} = T_{\text{OUTDUTY}} + \text{Clock Uncertainty} + T_{\text{STATPHAOFFSET}} \qquad \textit{Equation 11}$$

$$\text{Transmit Error} = \text{source\_sync\_out\_bus\_skew} + \text{source\_sync\_clock\_err} \qquad \textit{Equation 12}$$

For the center-aligned example being analyzed, the transmit error is 628 ps, as shown in Table 3.

Table 3: **Synchronous-Source Center-Aligned Transmitter Timing Budget**

Parameter	Value (ps)
datasheet_bus_skew	182
$T_{\text{OUTPUT\_LOGIC\_DELAY\_VARIATION}}$	80
T_{OUTDUTY} (Duty Cycle Distortion)	200
Clock_unc_ss_center_aligned_output	53
$T_{\text{STATPHAOFFSET}}$ (Static Phase Offset)	120
Total	635

Input—No Calibration

For the typical input bus without any additional alignment, the timing should be analyzed using Vivado timing analysis.

Note: For designs using IDELAY/ODELAY in COUNT mode, the programmed delays can vary depending on the data sheet parameters $T_{\text{IDELAY\_RESOLUTION}}$ and $T_{\text{ODELAY\_RESOLUTION}}$. The entire range of the delay resolution must be taken into account for COUNT mode. Higher performances can be achieved using delays in TIME mode where the delays are calibrated using a reference clock or using COUNT mode with a calibration technique as described in Input—with Bus Alignment.

Input—with Bus Alignment

For input buses that align the clock such as the 7:1 interface as explained in *LVDS Source Synchronous 7:1 Serialization and Deserialization Using Clock Multiplication* (XAPP1315) [Ref 4], the input clock delays can be calibrated at startup using differential inputs with input delays.

The timings from the Vivado tools do not correctly reflect the alignment from the calibration process because the calibration removes clock delays. The error for the receiver can be calculated using the following parameters.

- T_{SAMP_BUFG} (See the *UltraScale device data sheets* [Ref 5] and *UltraScale+ device data sheets* [Ref 6].)

- Datasheet_input_bus_skew (Vivado tools data sheet report. See Figure 24 for an example report.)

T_{SAMP_BUFG}

The sampling window for I/O logic including IDDRE1 or ISERDESE3 is defined in the data sheet as T_{SAMP_BUFG}. T_{SAMP_BUFG} accounts for the noise associated with the clock such as an MMCM. The clock skews are accounted for separately.

T_{SAMP_BUFG} additionally includes the variation in the delays after the initial calibration as long as the device is within the recommended operating conditions for V_{CCINT} and temperature.

Datasheet_input_bus_skew

The data sheet report must be used for analyzing the clock path skews. For the slow, maximum delays (setup), check the variation in the setup delays. As shown in Figure 24, the setup range is 109 ps.

Source	Setup	Setu...	Setup Process Corner
din[0]	-2.127	Rise	SLOW
din[1]	-2.089	Rise	SLOW
din[2]	-2.199	Rise	SLOW
din[3]	-2.179	Rise	SLOW
din[4]	-2.203	Rise	SLOW
din[5]	-2.109	Rise	SLOW
din[6]	-2.078	Rise	SLOW
din[7]	-2.102	Rise	SLOW
din[8]	-2.084	Rise	SLOW
din[9]	-2.118	Rise	SLOW
din[10]	-2.127	Rise	SLOW
din[11]	-2.102	Rise	SLOW
din[12]	-2.103	Rise	SLOW
din[13]	-2.106	Rise	SLOW
din[14]	-2.102	Rise	SLOW
din[15]	-2.175	Rise	SLOW

X20081-121917

Figure 24: **Input Setup Bus from the Data Sheet Report**

Note: The datasheet report may be run with or without the package skew. If the package skews are compensated by PCB routing, disable the package skews using the `disable_flight_delays` option: `config_timing_analysis -disable_flight_delays true`.

Combined Input Bus with Calibrated Clocks at Startup

Receiver timing budget for an input bus with a calibrated clock is the sum of T_{SAMP_BUFG} and input bus skew. For the input bus example being analyzed, where the clock delays are calibrated at startup, the input bus error is 735 ps as shown in Table 4.

Table 4: **Receiver with Bus Alignment Timing Budget**

Parameter	Value (ps)
T_{SAMP_BUFG}	610
datasheet_input_bus_skew	109
Total	719

Input—with Dynamic Phase Alignment

Some receiver interfaces use dynamic phase alignment to align the data and clock independently at each receiver. The timing margin gained from this calibration can be calculated using the following parameters.

$T_{INPUT\_LOGIC\_UNCERTAINTY}$

$T_{INPUT\_LOGIC\_UNCERTAINTY}$ accounts for the setup/hold and any pattern dependent jitter for the input logic (input register, IDDRE1, or ISERDESE3). This characterized value excludes the clock routing that must be taken into consideration separately.

Table 5: **Input Logic Uncertainty**

Parameter (ps)	UltraScale Device	UltraScale+ Device
$T_{INPUT\_LOGIC\_UNCERTAINTY}$	40	40

Duty Cycle Distortion (DCD or $T_{OUTDUTY}$)

The duty cycle for the MMCM (MMCM_$T_{OUTDUTY}$) or PLL (PLL_$T_{OUTDUTY}$) is specified in the data sheet and includes the output clock buffers to the ODDRE1/OSERDESE3.

DCD does not affect the inputs in designs that do not transmit data on both edges of the clock. In Figure 19, if D1 and D2 are the same value, switching is not seen at the falling edge of the clock and the DCD is not seen in operation. Consequently, designs that exclusively use either the rising or the falling edge of the clock do not need to account for the effects of DCD.

Calibration Error (T_{CAL_ERROR})

To account for the error associated with the calibration circuitry, the resolution of IDELAY must be taken into account. The IDELAY resolution is defined in the data sheet for the device as $T_{IDELAY\_RESOLUTION}$. As shown in Equation 13, because of quantization effects, the exact value chosen can vary up to twice the maximum resolution of IDELAY (15 ps for UltraScale devices and 12 ps for UltraScale+ devices).

$$T_{CAL\_ERROR} = 2 \times T_{IDELAY\_RESOLUTION}$$

Equation 13

Clock Uncertainty

When using dynamic phase alignment and training each data input, the clock uncertainty must account for TSJ and DJ, as shown in Equation 14 and Table 6.

$$Clock\_unc\_dpa = \frac{(TSJ^2 + DJ^2)^{1/2}}{2}$$
Equation 14

Clock uncertainty is the variability of a clock edge from the ideal location or the result of system jitter, discrete jitter, and phase error attributable to the clocking components (CMT, clock buffers, and power noise).

Clock uncertainty is reported in the timing path reports. For input buses, the phase error impacts the clock uncertainty. When working with an MMCM, set up the MMCM for phase alignment by setting the bandwidth to either Optimized or High.

Table 6: **Clock_unc_dpa**

Parameter	Value (ps)
TSJ	71
DJ	78
clock_unc_dpa	53

Combined Input With Calibrated Data

For an input bus with dynamic phase alignment, receiver timing budget is the sum of input logic uncertainty, calibration error, duty cycle distortion ($T_{OUTDUTY}$), and clock uncertainty as shown in Equation 15.

$$Comb\_Input\_Cal\_Data = T_{INPUT\_LOGIC\_UNCERTAINTY} + T_{CAL\_ERROR} + T_{OUTDUTY} + clock\_unc\_dpa$$
Equation 15

For the input bus example being analyzed, where the data delays are continuously calibrated, the input bus error is 317 ps as shown in Table 7.

Table 7: **Receiver with Bus Alignment Timing Budget**

Parameter	Value (ps)
$T_{INPUT\_LOGIC\_UNCERTAINTY}$	40
T_{CAL_ERROR}	24
$T_{OUTDUTY}$	200
clock_unc_dpa	53
Total	317

Conclusion

Source-synchronous I/O interfaces have special timing requirements and hence require diligent clocking topologies and design considerations. This application note explains the recommended usage models and timing analysis methodologies for I/O interfaces that use SelectIO component mode primitives (ODDRE1, IDDRE1, ISERDESE3, and OSERDESE3) in UltraScale and UltraScale+ devices.

Documentation Navigator and Design Hubs

Xilinx Documentation Navigator provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open the Xilinx Documentation Navigator (DocNav):

- From the Vivado® IDE, select **Help > Documentation and Tutorials**.

- On Windows, select **Start > All Programs > Xilinx Design Tools > DocNav**.

- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In the Xilinx Documentation Navigator, click the **Design Hubs View** tab.

- On the Xilinx website, see the Design Hubs page.

Note: For more information on Documentation Navigator, see the Documentation Navigator page on the Xilinx website.

References

1. UltraScale – High Speed Select IO – Example timing budget for Native mode (Xilinx Answer 68618)

2. *UltraScale Architecture SelectIO Resources User Guide* (UG571)

3. *UltraScale Architecture Clocking Resources User Guide* (UG572)

4. *LVDS Source Synchronous 7:1 Serialization and Deserialization Using Clock Multiplication* (XAPP1315)

5. UltraScale device data sheets:

   ◦ *UltraScale Architecture and Products Overview* (DS890)

   ◦ *Kintex UltraScale Architecture Data Sheet: DC and AC Switching Characteristics* (DS892)

   ◦ *Virtex UltraScale Architecture Data Sheet: DC and AC Switching Characteristics* (DS893)

6. UltraScale+ device data sheets:

  ◦ *Kintex UltraScale+ FPGAs Data Sheet: DC and AC Switching Characteristics* (DS922)

  ◦ *Virtex UltraScale+ FPGA Data Sheet: DC and AC Switching Characteristics* (DS923)

  ◦ *Zynq UltraScale+ MPSoC Data Sheet: DC and AC Switching Characteristics* (DS925)

7. *Vivado Design Suite User Guide Design Analysis and Closure Techniques* (UG906)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
01/18/2018	1.0	Initial Xilinx release.
08/23/2018	1.1	Added a note about CLKOUT0/CLKOUT1 in the Output Use Models for Component Mode Applications section. Updated the values in Table 2, Table 3, and Table 4. Updated throughout: ◦ Duty Cycle Distortion to $T_{OUTDUTY}$ ◦ Output_logic_delay_variation to $T_{OUTPUT\_LOGIC\_DELAY\_VARIATION}$ ◦ Statphaseoffset to $T_{STATPHAOFFSET}$ ◦ Input_logic_uncertainty to $T_{INPUT\_LOGIC\_UNCERTAINTY}$ ◦ Cal_err to T_{CAL_ERROR}

Please Read: Important Legal Notices