XAPP1064 (v1.1) June 3, 2010

Source-Synchronous Serialization and Deserialization (up to 1050 Mb/s)

Author: NIck Sawyer

Summary

Spartan®-6 devices contain input SerDes (ISERDES) and output SerDes (OSERDES) blocks. These primitives simplify the design of serializing and deserializing circuits, while allowing higher operational speeds. This application note discusses how to efficiently use these primitives in conjunction with the input delay blocks and phase detector circuitry.

ISERDES and OSERDES Guidelines

Each Spartan-6 FPGA input/output block (IOB) contains a 4-bit input SerDes and a 4-bit output SerDes. The SerDes from two adjacent blocks (master and slave) can be cascaded to make an 8-bit block. This gives the possibility of SerDes ratios from 2:1 to 8:1 on both output and input for both single and double data rate I/O clocks.

Cascading the ISERDES blocks is not an issue when a differential signaling standard is being used because these standards use the two IOBs (master and slave) associated with the two sets of SerDes registers. Thus, using two ISERDES effectively reduces design cost. However, when using a single-ended signaling standard, some care needs to be taken when the design requires either a SerDes ratio of five or more or the phase detector mode. Specifically, two data lines cannot enter the device in adjacent master and slave IOBs when using cascaded SerDes. This limitation is not necessary when the SerDes ratio is four or less and the phase detector mode is not being used because the SerDes is not cascaded. However, by not using the phase detector mode, data loss will occur during calibration and the application will need to account for this loss.

Introduction to Deserialization and Data Reception

A deserializer design and its associated clocking primitives are dependent on the format of the incoming receive data stream. This data tends to fall into three categories.

Case 1

The data stream is a multiple of the rate of the incoming clock, and the clock signal is used as a framing signal for the received data. Multiple changes in the state of the data lines occur during one clock period. A widely used example is the 7:1 interface used in cameras and flat panel TVs and monitors. Other ratios are obviously possible, and the Spartan-6 FPGA ISERDES can support ratios of 2, 3, and 4:1, and also 5, 6, 7, and 8:1 when cascaded. In this example, the received clock is multiplied in a PLL, and the resultant high-speed capture clock is passed to the input logic through the BUFPLL primitive. The BUFPLL capture clock is designed to always be used in single data rate (SDR) mode with respect to the input data. For example, a 150 MHz input clock with accompanying 7:1 data requires the PLL and BUFPLL to operate at 1050 MHz (equals 150 x 7). This high-speed capture clock is used to clock the receive data into the input deserializers and is capable of driving one whole edge of a device. Parallel data is then presented to the FPGA logic at the speed of the original incoming clock. Figure 1 shows this 7:1 data formatting example.

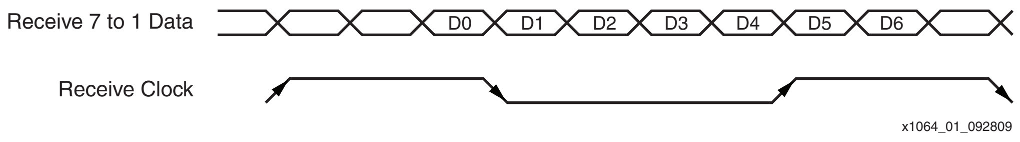

Figure 1: **Data Stream Using a Low-Speed Clock with a 7:1 SerDes Ratio**

Case 2

The data stream is a multiply by two of the incoming clock, commonly called Double Data Rate (DDR) reception. A DDR data stream is shown in Figure 2. Each transition of the clock indicates a change in the state of a data line. There are two ways of receiving this kind of data. The first is to use a PLL and a BUFPLL (see Case 1), where the PLL is being used to multiply the incoming clock by two and the BUFPLL allows use of the whole edge of a device. The other method is to use the BUFIO2 primitive, where two BUFIO2s are required to receive DDR data, and the BUFIO2s are only able to drive the same half edge of a device where the clock input is located. The deserialization factor (ratio) can be chosen by the designer (values of 2, 4, 6, and 8:1). The necessary divided clock for the parallel data is generated through one of the BUFIO2 primitives. Two BUFIO2s must be used to multiply the incoming DDR clock by two to generate an SDR capture clock.

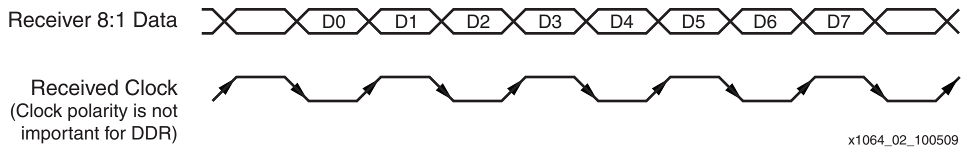

Figure 2: **8:1 Data Stream Using DDR**

Case 3

The data stream is at the same rate as the receiver clock (SDR). A drawing of an SDR data stream is shown in Figure 3. Each data bit changes every two clock transitions, normally on the rising edge of the clock. There are two ways of receiving this kind of data stream. The received clock is multiplied by one in a PLL and the BUFPLL is used to receive data on a whole edge of a device, or a single BUFIO2 or PLL is used to drive the inputs in the half edge of the device where the clock input is situated. The BUFIO2 is also used to divide down the received clock to be used with the deserialized parallel data. Using SDR, ratios of 2, 3, 4, 5, 6, 7, and 8:1 are possible.

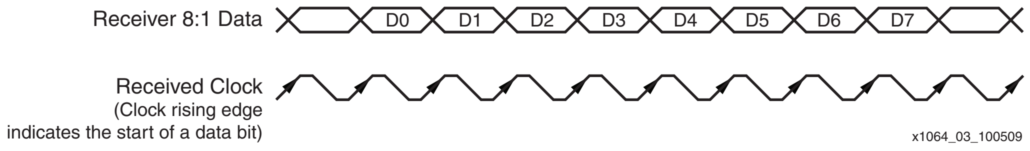

Figure 3: **8:1 Data Stream Using SDR**

Higher Deserialization Factors

Deserialization using factors greater than 8:1 is possible when receiving data. The PLL can be used to generate a third clock, which is intermediate to the high-speed I/O capture clock and the low-speed parallel data clock. Examples of designs using SerDes ratios of 10, 12, 14, and 16:1 are included in the Reference Design Files. Essentially, the input SerDes primitives are still used in 5, 6, 7, and 8:1 modes, receiving data through a high-speed capture clock from the PLL through the BUFPLL. The received parallel data is transferred to the FPGA logic in the intermediate clock domain and then further transferred to the main clock domain using a 2:1 gearbox, also in the FPGA logic. A drawing of the mechanism is shown in Figure 4. In any of these examples of higher deserialization factors where the PLL is used, the receiver clock can be SDR, DDR, or a divided clock.

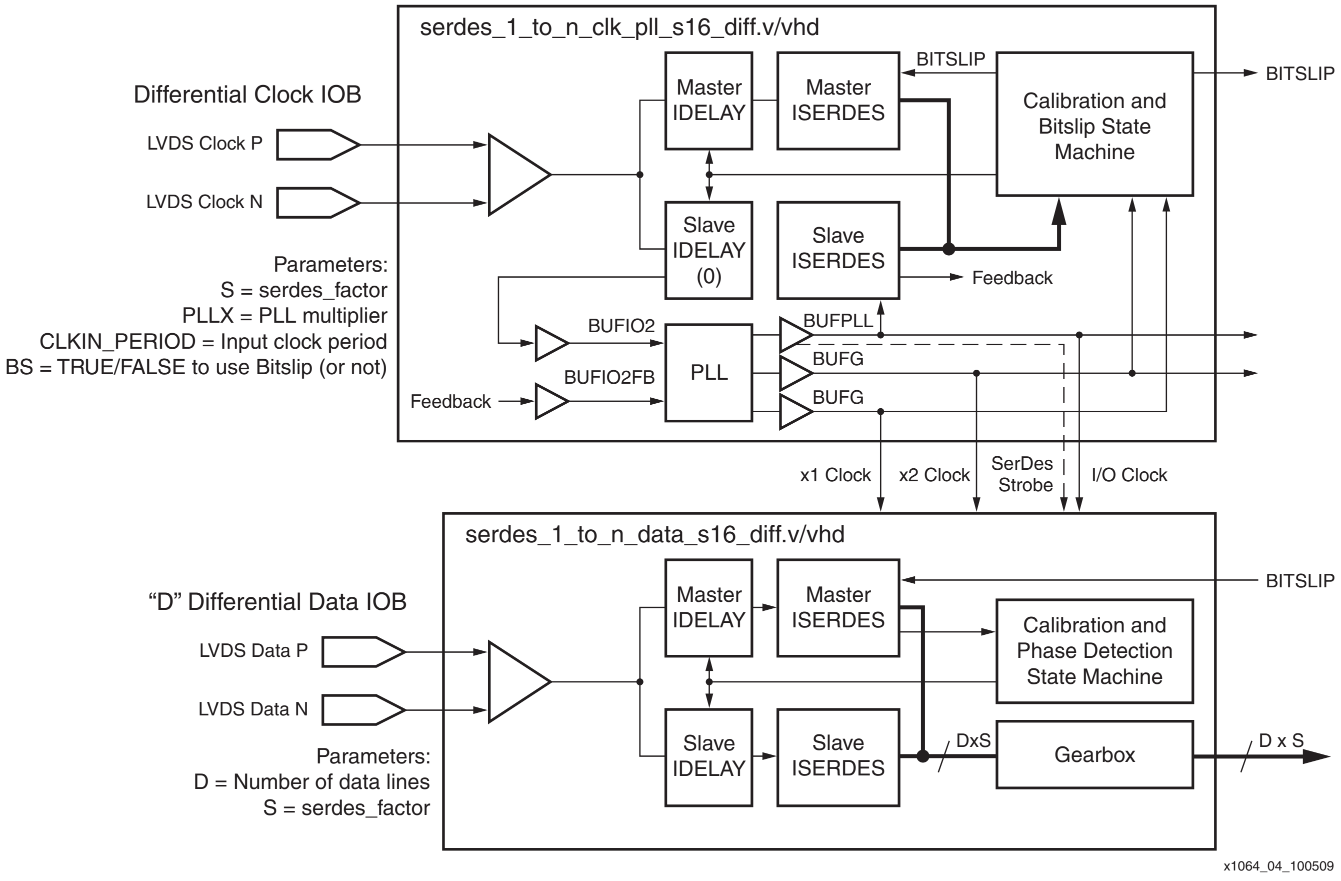

Figure 4: **Receiving Data at Higher SerDes Factors**

Data Reception Using PLL and BUFPLL

The topology for data reception using PLL and BUFPLL is uncomplicated. The receiver clock is multiplied as required in the PLL to generate an internal single data-rate capture clock. The incoming clock signal needs to pass through an input delay block (to balance datapath delays) and a BUFIO2 to reach the PLL. In the 7:1 video example, the received pixel clock must be multiplied by seven. The clock signal is routed from the PLL to a BUFPLL primitive to drive one whole edge of the device. CLKOUT0 and CLKOUT1 are the only outputs of the PLL that are capable of driving high-speed capture clocks to the BUFPLL. The BUFPLL also requires a global clock signal equal to the original non-multiplied source clock, which can be driven from any of the PLL outputs through a global buffer (BUFG), and the LOCKED signal from the PLL, which is required for synchronization inside the BUFPLL.

The three input signals to the BUFPLL allow the BUFPLL to distribute the high-speed receiver clock to the input delay and SerDes primitives in the same edge of the device, along with the required SerDes strobe signal (appropriately aligned) that allows safe transfer of low-speed parallel data to the FPGA logic from the input SerDes.

When using the PLL for data reception, PLL deskew is required. The feedback clock signal is routed from an I/O clock destination at the input SerDes primitive of the clock input pin back to the PLL using a BUFIO2FB primitive. This mechanism forces the multiplied clock to be in the same phase as the original received clock.

The mechanism for centering and capturing data reliably is based on the IODELAY2 primitive, used in input delay mode only. This mechanism is discussed in the Delaying Input Data and Clocks section. The block diagram of the receiver is shown in Figure 5.

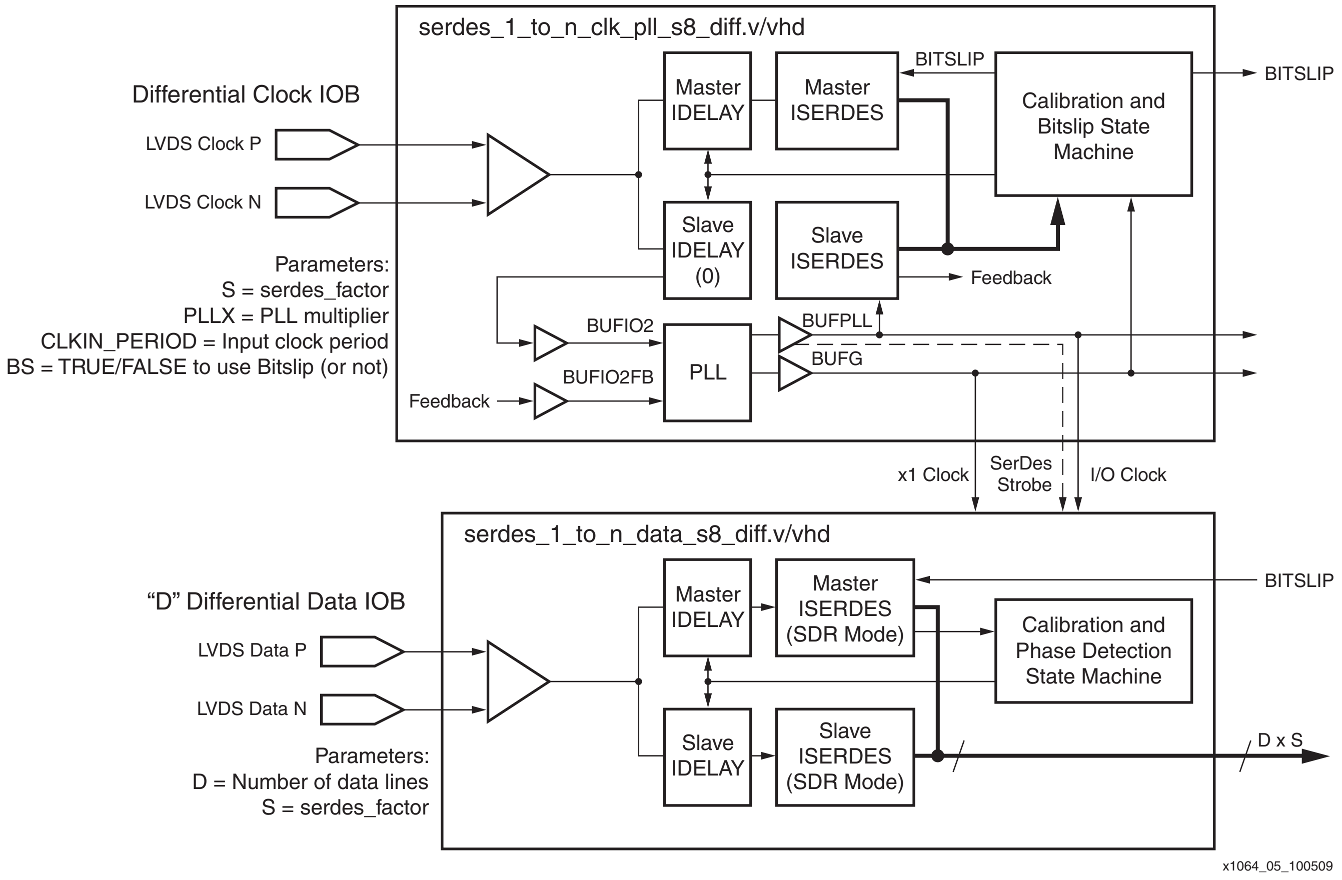

Figure 5: **Data Reception Using PLL and BUFPLL**

DDR Data Reception Using Two BUFIO2s

The topology for DDR data reception using two BUFIO2s uses the incoming clock to directly capture data without the use of a PLL. The incoming clock signal is fed through a delay block to balance the data and clock delays. In the case of a differential signal, as shown in Figure 6, the true and complement signals are fed through master and slave input delays (both set to zero) and then to a pair of BUFIO2 primitives. The first BUFIO2 accepts both true and complement input clocks and uses these to generate the appropriate divided clock and SerDes strobe for the input SerDes primitives. For example, if the receiver clock is 311 MHz (622 Mb/s data) and the design requires an 8:1 SerDes reduction, the BUFIO2 being driven by true and complement receiver clocks with its divide parameter set to eight actually divides the input clock by 4 to 77.75 MHz. The resultant I/O clock is routed to the input SerDes primitives along with the other (inverted) I/O clock generated from the other BUFIO2. These two clocks are doubled in the input SerDes to give a 622 MHz sampling clock for the 622 Mb/s data.

The BUFIO2s need to be located in the same half edge as the clock input, and when using input delays, it is not possible to simultaneously use the alternate BUFIO2s in the other half edge. Reception of data buses when using input delays is therefore limited to the half edge where the clock input is located.

The mechanism for centering and capturing data reliably is based on the IODELAY2 primitive, used in input delay mode only. This mechanism is discussed in the Delaying Input Data and Clocks section. The block diagram of the receiver is shown in Figure 6.

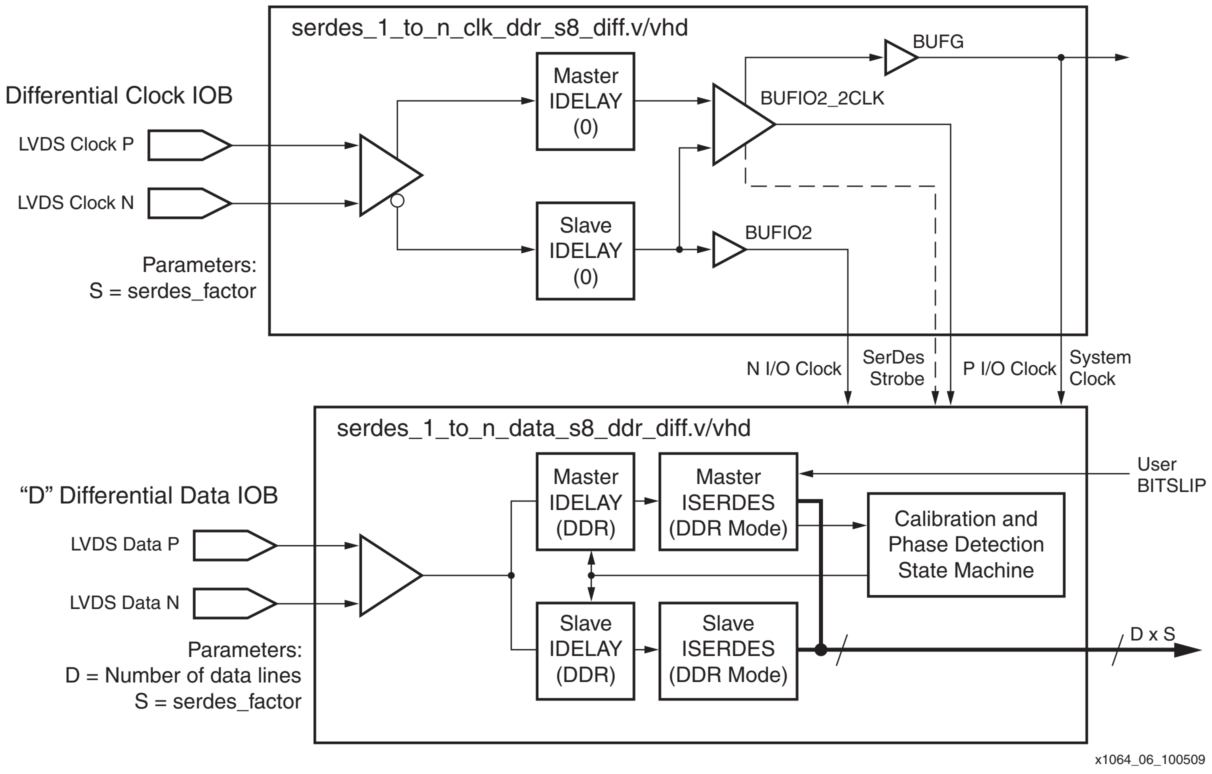

Figure 6: **DDR Data Reception Using Two BUFIO2s**

SDR Data Reception Using BUFIO2

The topology for SDR data reception using BUFIO2 uses the incoming clock directly to capture data. The clock signal is fed through a delay block (set to 0) to a BUFIO2. The BUFIO2 uses this input clock to generate the appropriate divided clock and SerDes strobe for the input SerDes primitives. For example, if the receiver clock is 525 MHz (525 Mb/s data) and the design requires an 8:1 SerDes reduction, then the BUFIO2 with its divide parameter set to eight divides the input clock by 8 to 65.625 MHz. The resultant I/O clock is routed to the input SerDes primitives for data capture. The limiting factor in this case is the maximum clock frequency allowed through the clock input pin.

The BUFIO2 needs to be located in the same half edge as the clock input. When using input delays, it is not possible to simultaneously use the alternate BUFIO2 in the other half edge. Reception of data buses is therefore limited to the half edge, the location of the clock input when using input delays.

The mechanism for centering and capturing data reliably is based on the IODELAY2 primitive, used in input delay mode only. This mechanism is described in the Delaying Input Data and Clocks section. The block diagram of the receiver is shown in Figure 7.