XAPP704 (v1.2) February 8, 2005

Virtex-4 High-Speed Single Data Rate LVDS Transceiver

Author: Markus Adhiwiyogo

Summary

This application note describes single data rate (SDR) transmitter (Tx) and receiver (Rx) interfaces in an Virtex-4™ FPGA using 17 low-voltage differential signaling (LVDS) pairs (one clock and 16 data channels), suitable for SFI-4 or XSBI related applications. This design is implemented using the ChipSync™ features. The accompanying reference design files include an example targeting a Virtex-4 XC4VLX25-FF668 device. A UCF file is provided for implementation of this design on the Xilinx ML450 development board. Please see design characteristics/recommendation summary for further information on design requirements.

Introduction

An SDR interface is defined as having only one single positive and negative transition of the clock with respect to the data bit (shown in Figure 1). Thus, if the data rate is 500 Mb/s, the clock frequency would be 500 MHz. SDR LVDS interfaces are described in a variety of standards, among others, SFI-4 and XSBI.

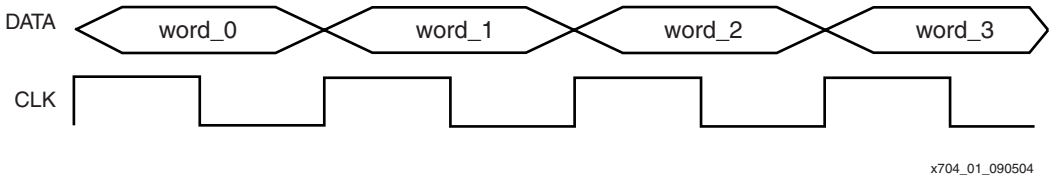

Figure 1: **SDR Clock and Data Interface**

This application note describes a method of implementing an SDR transceiver at frequencies greater than the maximum operating frequency of the Digital Clock Manager (DCM). It also uses components/features unique to the Virtex-4 architecture to facilitate creating designs at 700 MHz without exceeding the Virtex-4 AC timing specifications.

Figure 2 illustrates the overall system configuration, showing a full-duplex SDR link between a Virtex-4 device and another device with an SDR transceiver. The Virtex-4 device requires a reference clock with either LVDS or LVPECL differential outputs operating at the SDR clock frequency to generate the transmit clock from the Virtex-4 device. Figure 2 shows a discrete clock source operating at the SDR frequency.

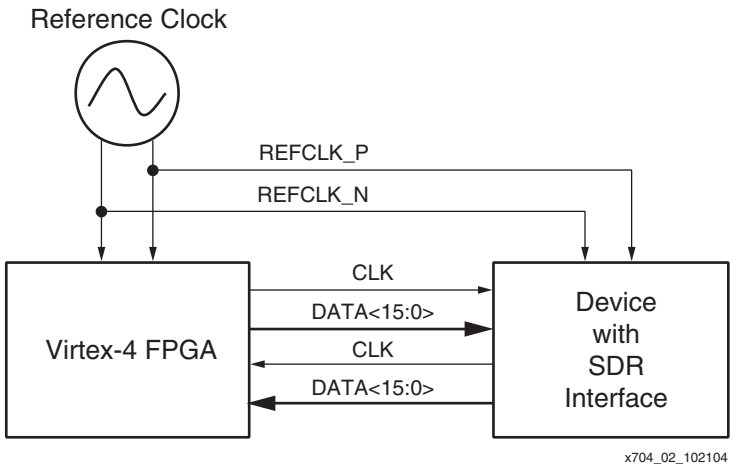

Figure 2: **Typical SDR Link System**

Virtex-4 Implementation

Figure 3 shows a simplified Virtex-4 SDR transceiver block diagram as found in the reference design, SDR_LVDS_TX_RX. This module contains IDELAYCTRL, TX_CLOCKS, TX_CLK_AND_DAT, RX_CLK_AND_DAT, and RST_MACHINE. Details on each module are described in the following sections.

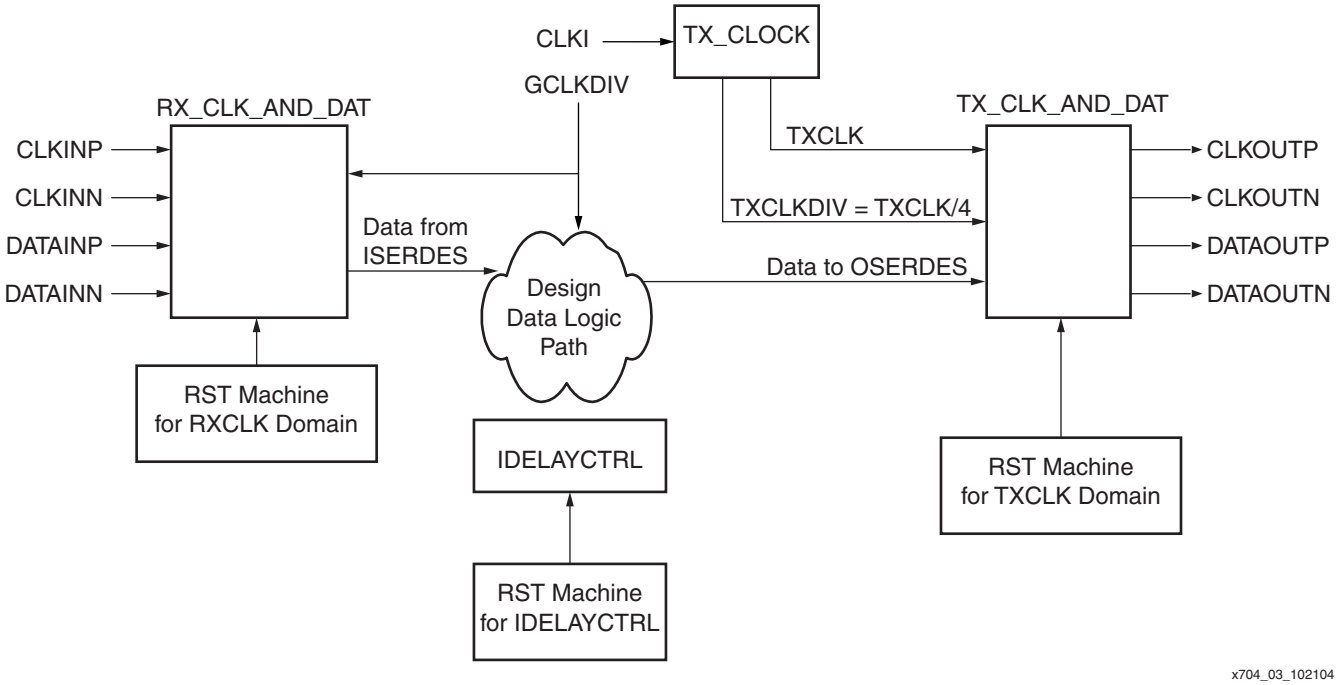

Figure 3: **Simplified Virtex-4 SDR Transceiver Block Diagram**

Multiple transmitters and receivers can be implemented in the same Virtex-4 FPGA. When multiple instances are needed, only TX_CLK_AND_DAT and RX_CLK_AND_DAT modules are replicated, saving valuable global clock resources by not replicating the TX_CLOCKS module.

Sample code and design for the ML450 board are also provided in the reference design file SDR_LVDS_AND_LOGIC_TOP.

TX_CLOCKS Module

The TX_CLOCKS module is designed to provide/generate all the clock frequencies necessary to perform the transmit operations using OSERDES. There are two clocks generated by this module: TXCLK and TXCLKDIV.

The reference design uses the SDR clock input (CLKI) to generate TXCLK and TXCLKDIV. The CLKI input must already be in the global clock network. In this example, the frequency of TXCLK is four times faster than TXCLKDIV. Connect these two clocks to the CLK and CLKDIV inputs of the desired OSERDES.

Figure 4 illustrates the generated clocks.

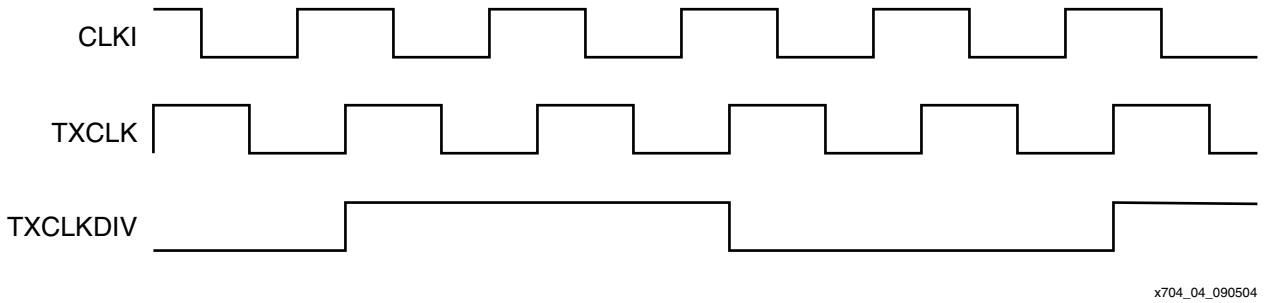

Figure 4: **TX_CLOCKS Module Clock Waveforms**

To use this reference design, the skew of both TXCLK and TXCLKDIV must be minimized, because the DCM is used to generate both TXCLK and TXCLKDIV. The TXCLKDIV is generated at the CLKDV output of the DCM. Since the input clock frequency to the DCM is greater than the DCM input frequency specification, the CLKIN_DIVIDE_BY_2 of the DCM must be set to TRUE. Figure 5 shows a block diagram of the TX_CLOCK module using the DCM.

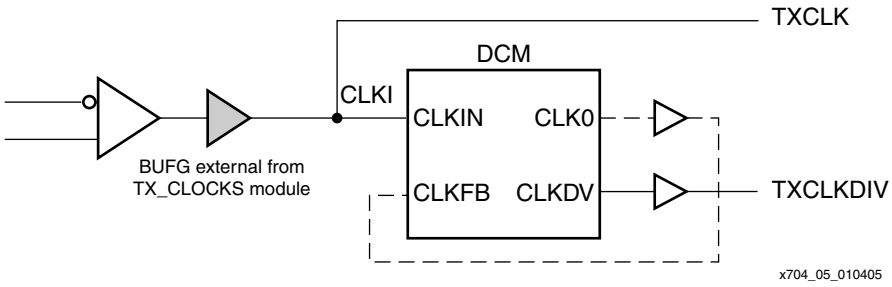

Figure 5: **TX_CLOCKS Module Using a DCM**

Table 1 contains the module pin descriptions.

Table 1: **TX_CLOCKS Module Pin Definitions**

I/O Type	Module Pin Name	Definition
Input	CLKI	SDR Clock Input
	RST	Active High Reset
Output	TXCLK[1]	Generated TXCLK Domain Output
	TXCLKDIV[1]	Generated TXCLK divided-by-four domain output
	TXDCMLOCKED	DCM Locked Pin

Notes:

1. Both TXCLK and TXCLKDIV must be phase aligned.

TX_CLK_AND_DAT Module

The transmitter (TX_CLK_AND_DAT) uses two different types of output modules, OSERDES for the data channels and ODDR for the clock output. The data channels have instantiation names with the prefix TX_DAT_OUT_ followed by a two digit number to denote the bit number. The clock channel has an instantiation name with the prefix TX_CLK_OUT_ followed by a two digit number. More of these blocks can be instantiated. Table 2 contains the module pin description.

Table 2: **TX_CLK_AND_DAT Module Pin Definitions**

I/O Type	Module Pin Name	Definition
Input	ORST	Active High Reset
	OCE	Active High Output Enable
	TXCLK[1]	SDR Clock
	TXCLKDIV[1]	SDR Clock divided by 4
	DATA_IN<63:0>	64-bit Parallel Data Input
Output	CLKOUTP CLKOUTN	Differential Transmit Clock Output
	DATAOUTP<15:0> DATAOUTN<15:0>	16-bit Differential Data Clock Output

Notes:

1. Both TXCLK and TXCLKDIV must be phase aligned for proper transmitter operation. Xilinx recommends using the TX_CLOCKS module to generate these two clocks.

There are sixteen OSERDES blocks in this module to accommodate 64-bit of parallel data input. Each OSERDES is set for 4:1 serialization. Table 3 summarizes the settings applied to all OSERDES data channels.

Table 3: **OSERDES Data Channel Settings**

Parameter Name	Parameter Value
DATA_RATE_OQ	SDR
DATA_WIDTH	4
SERDES_MODE	MASTER

When using OSERDES, the order of the data transmitted at every positive TXCLK edge is from D1 to D4 (LSB to MSB). For cases larger than 4:1 serialization, the order of the data transmitted is from D1 to D6 of MASTER OSERDES followed by D3 to D6 of SLAVE OSERDES. Since 3-state is not used, all 3-state pins (TCE and T1 through T4) are tied to a logic Low. The 3-state attributes are left as "Don't Care".

The ODDR is used to forward the SDR transmit clock from the Virtex-4 FPGA. This is implemented by connecting TXCLK into the ODDR clock (C) input pin and connect D1 and D2 pins to a logic High and a logic Low respectively. Using ODDR is the only way to forward clocks from Virtex-4 FPGA to external devices.

Figure 6 shows the timing waveform of the transmitted data with respect to TXCLK and TXCLKDIV.

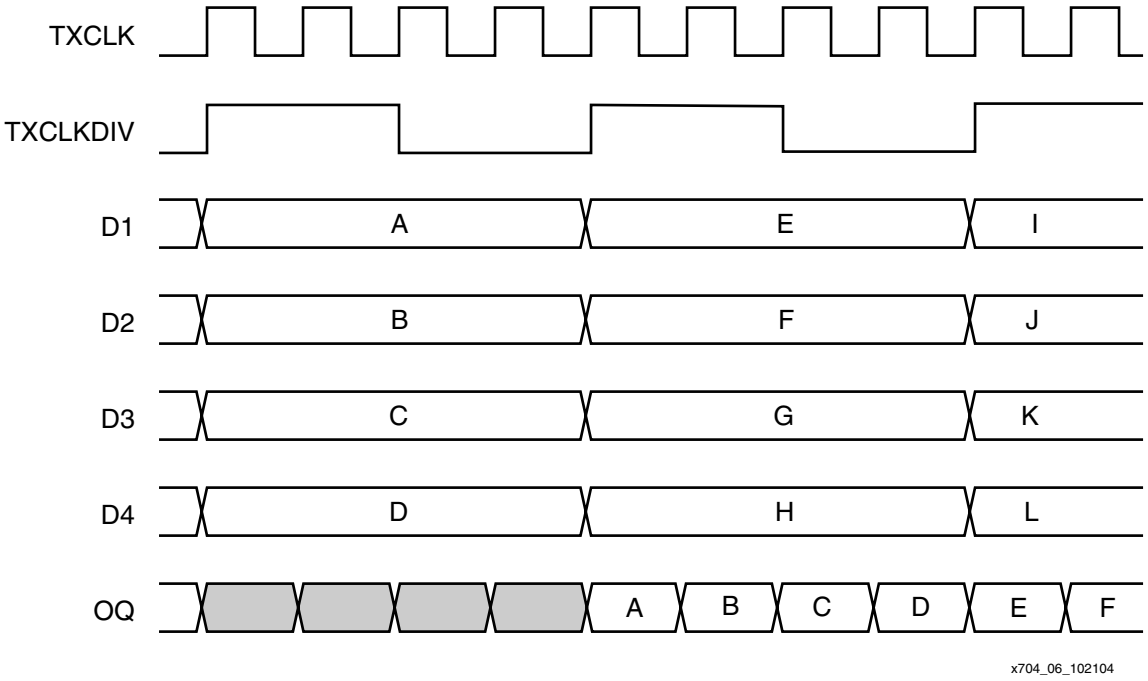

Figure 6: **TX_CLK_AND_DAT Output Waveforms**

All output pins from this module are connected to LVDS_25 output buffers. Figure 7 shows the block diagram for TX_CLK_AND_DAT module.

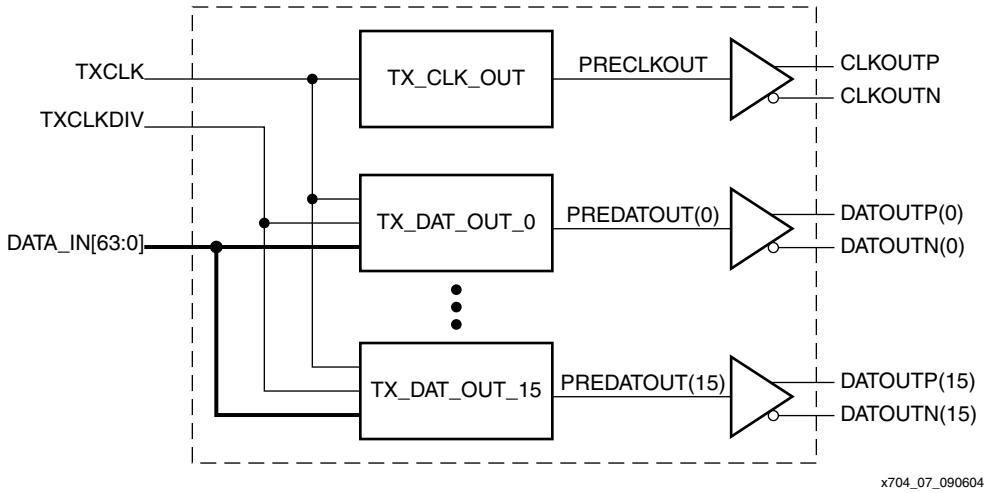

Figure 7: **TX_CLK_AND_DAT Module Block Diagram**

RX_CLK_AND_DAT Module

The receiver (RX_CLK_AND_DAT) module has both clock recovery and data recovery blocks.

The clock recovery blocks include:

- BUFIO - to access IOCLK network
- BUFR - to access Regional Clock network

The data recovery blocks include:

- ISERDES - SERDES used to deserialize data
- ISERDES_ALIGNMENT_MACHINE - Logic to control data recovery in one channel using IDELAY and BITSLIP
- FIFO16 - A FIFO to move data from the Regional Clock network into the Global Clock network

Figure 8 shows a simplified block diagram for RX_CLK_AND_DAT module.

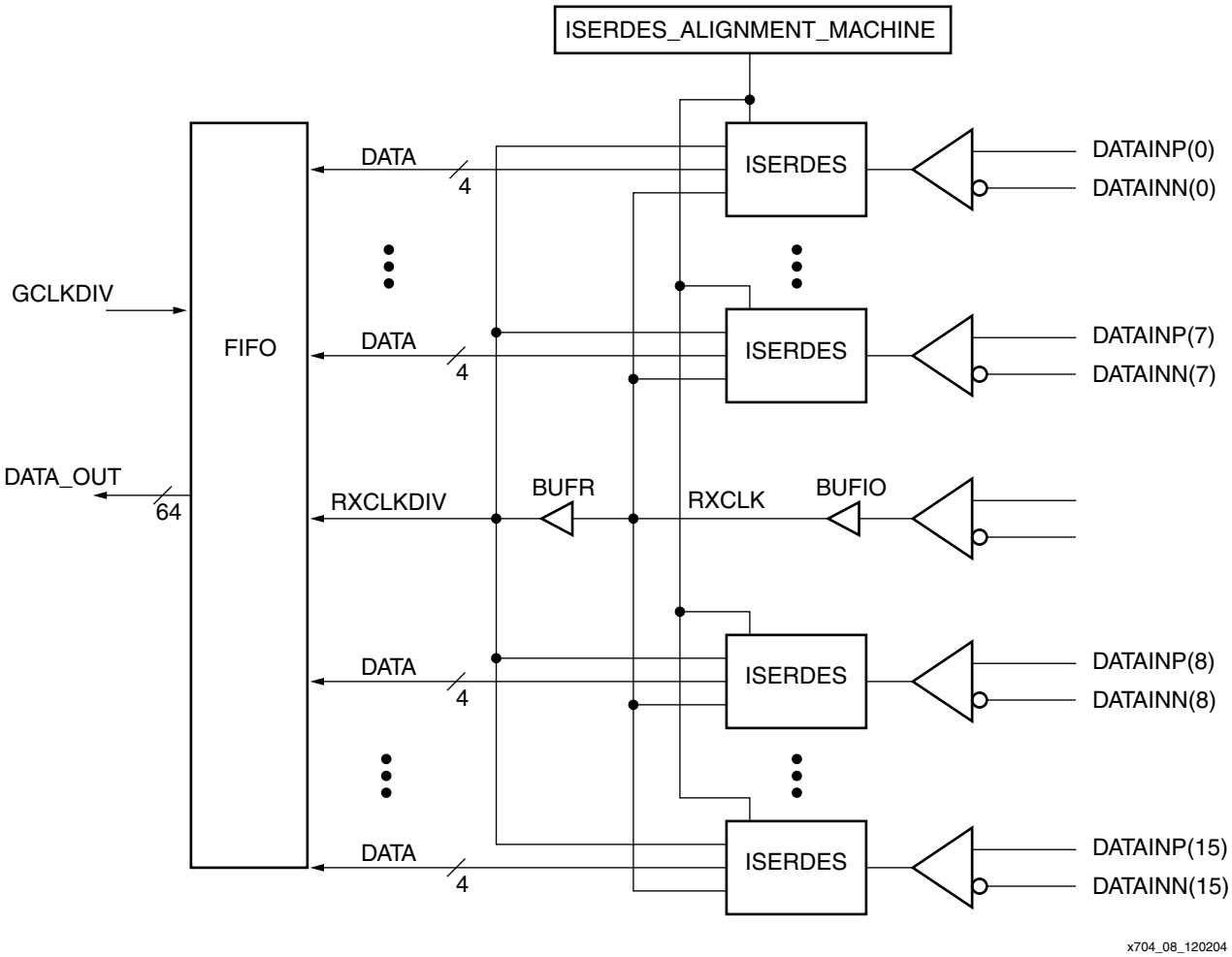

Figure 8: **RX_CLK_AND_DAT Module Block Diagram**

The functionality of the sub-blocks are discussed in the following sections. Table 4 contains the module pin descriptions.

Table 4: **RX_CLK_AND_DAT Module Pin Definitions**

I/O Type	Module Pin Name	Definition
Input	CLKINP CLKINN	Differential Receive Clock Input
	DATAINP<15:0> DATAINN<15:0>	16-bit Differential Receive Data Inputs
	IRDY	When logic High, IDELAY is ready
	USE_BITSLIP	When logic High, data recovery state machine performs the BITSLIP operation until the training pattern in TRAINING_PATTERN is found
	RST	Active High Reset – For all logic
	IRST	Active High Reset – For all ISERDES
	SCE	Active High Clock Enable
	TRAINING_PATTERN<3:0>	4-bit training pattern
	LOCKED	LOCKED signal input
	GCLKDIV	Global Clock Input – Frequency near RXCLKDIV
Output	RXCLKDIV	Received Clock divided by 4
	DATA_OUT<63:0>	64-bit Parallel Data Output
	DATA_ALIGNED	When logic High, the alignment process for one data channel is complete
	BUS_ALIGNED	When logic High, the alignment process across all data channels is complete
	SEND_CLOCK	When logic High, the alignment machine is requesting a clock signal at the data input pins

All forwarded clock and data input pins are connected to LVDSEXT_25 input buffers.

RX_CLK_AND_DAT Module Clocking Network

In this reference design, the recovered clock must be connected to input buffers at the clock capable I/Os. After the input buffer, the clock is connected to the BUFIO followed by a BUFR.

The BUFIO allows the recovered clock to access the IOCLK network. The IOCLK clock network is used for the CLK (fast or SDR clock input) of ISERDES. IOCLK can span up to three adjacent clock regions.

The BUFR is used to access the regional clock network and perform the clock divide function. The BUFR is used to provide the CLKDIV (slow or divided SDR clock input) of ISERDES. The regional clock network can span up to three adjacent clock regions. The BUFR divide function is set to four to accommodate 1:4 deserialization.

The Figure 9 illustrates the recovered clock network.

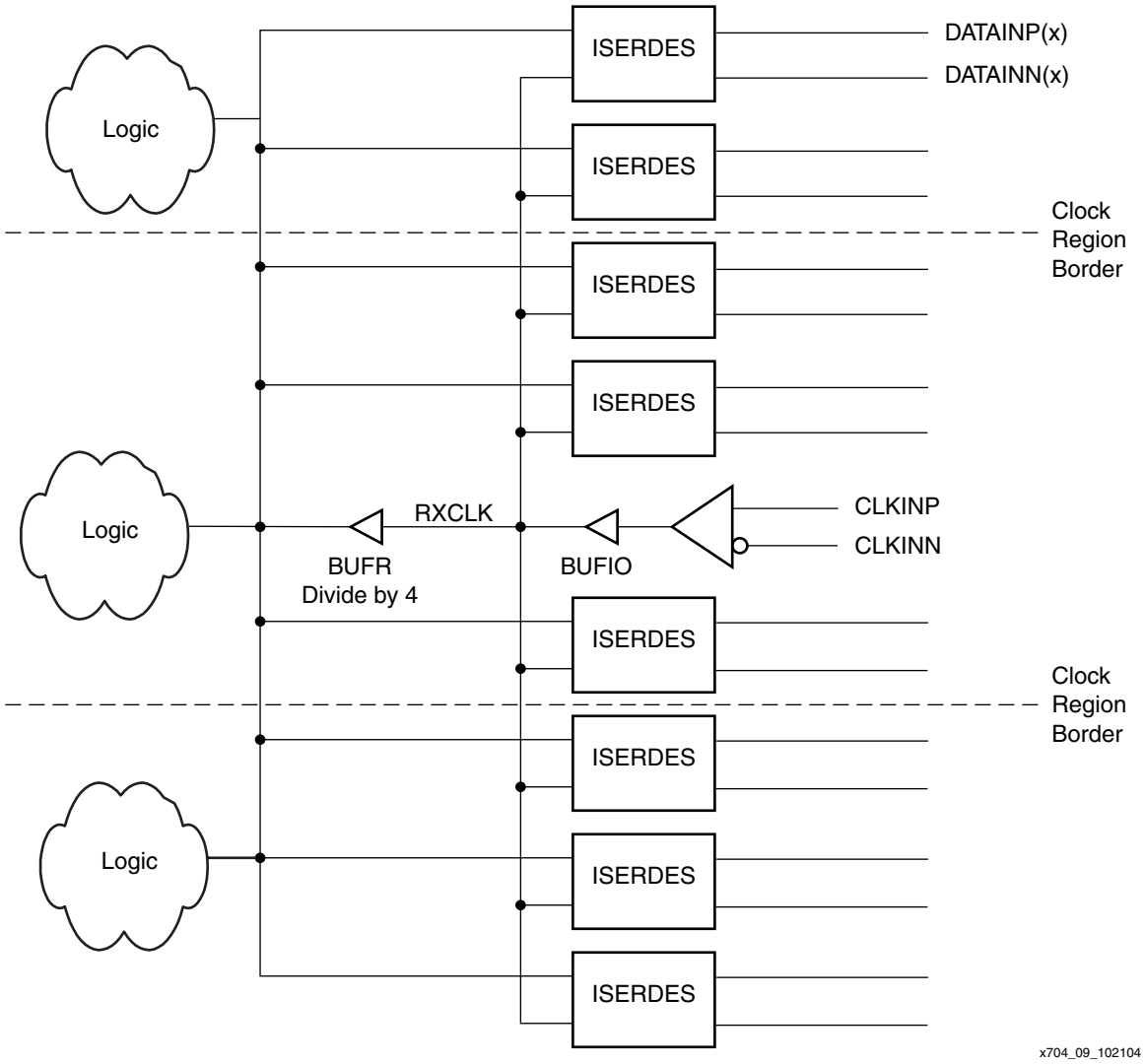

Figure 9: **RX_CLK_AND_DAT Clock Network**

ISERDES Block Characteristics

There are sixteen ISERDES blocks in this module to accommodate the 16 serial data inputs. Each ISERDES is set for 1:4 deserialization. Table 5 summarizes the data channel ISERDES settings.

Table 5: **ISERDES Settings**

Parameter Name	Parameter Value
BITSLIP_ENABLE	TRUE
DATA_RATE	SDR
DATA_WIDTH	4
INTERFACE_TYPE	NETWORKING
IOBDELAY	IFD
IOBDELAY_TYPE	VARIABLE
IOBDELAY_VALUE	0
NUM_CE	1
SERDES_MODE	MASTER

When using ISERDES, the order of the data received into fabric at every RXCLKDIV cycle is Q1 to Q4 (last in to first in). For cases larger than 4:1 serialization, the order of the data received from (last in to first in) Q1 to Q6 of MASTER ISERDES followed by Q3 to Q6 of SLAVE ISERDES. Figure 10 illustrates the order of data from ISERDES into the FPGA fabric.

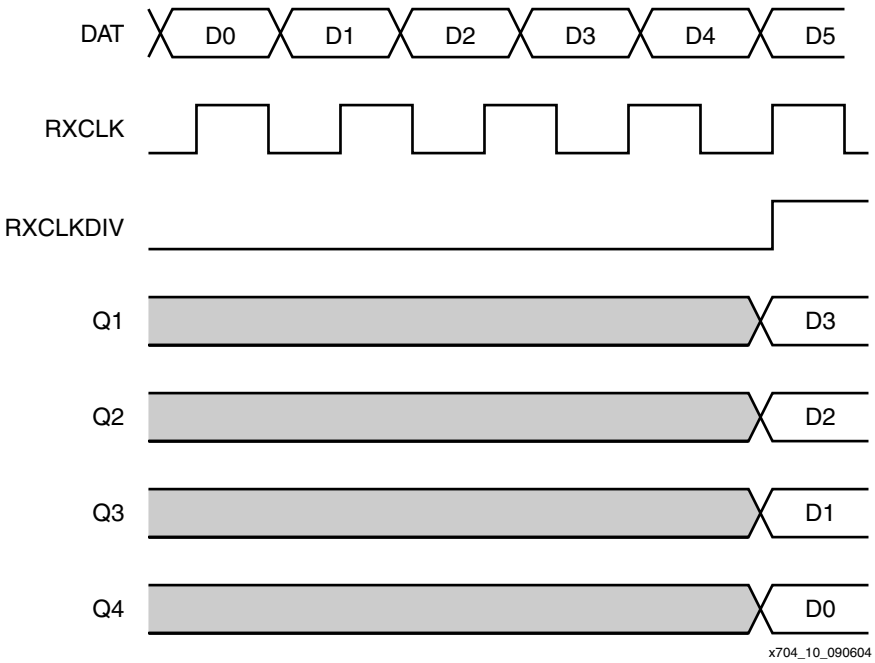

Figure 10: **Input and Output Data Relationship of ISERDES**

Since IDELAY and BITSLIP features are turned on, the BITSLIP, DLYINC, DLYCE, and DLYRST are connected to control pins.

ISERDES_ALIGNMENT_MACHINE Module

ISERDES_ALIGNMENT_MACHINE optimally centers the recovered clock to the data valid window of the incoming data using the IDELAY feature of ISERDES. In addition, when needed, this module uses the BITSLIP feature to reorder data into the desired training pattern.

Table 6 summarizes all the pins available in this module.

Table 6: **ISERDES_ALIGNMENT_MACHINE Module Pin Definitions**

I/O Type	Module Pin Name	Definition
Input	RXCLKDIV	Received clock divided by 4
	RST	Active High reset – for all logic
	SAMPLED_CLOCK<3:0>	The logic values when clock is sampled at a give number of IDELAY taps
	IRDY	When logic High, IDELAY is ready
	USE_BITSLIP	When logic High, the data recovery state machine performs a BITSLIP operation until the training pattern in TRAINING_PATTERN is found
	TRAINING_PATTERN<3:0>	4-bit training pattern
	SAP	When logic High, starts the alignment process from beginning
	RXDATA<3:0>	4-bit data recovered from ISERDES
Output	INC	Increments/Decrements the number of IDELAY taps used
	ICE	Enables/Disables change in the number of IDELAY taps used
	BITSLIP	When logic High, ISERDES uses the BITSLIP process
	DATA_ALIGNED	When logic High, the alignment process on the current data channel has been completed.
	SEND_CLOCK	When logic High, the alignment machine is requesting clock signal at data input pins. For channel alignment method, see "Appendix A".

Bus Alignment is a method of data recovery outlined in this application note. When using this method for data recovery, all data is aligned to the center of the clock. Prior to using this method, the skew between all incoming data and clock channels must be minimized. Additionally, the data transition edge is closely aligned to the positive clock of the incoming clock. This method is useful in applications where the transmitter does not provide a training pattern (i.e., SFI-4).

Using the bus alignment method, the receive clock is sampled by a 1:4 SDR SerDes (ISERDES). Four of the ISERDES outputs are used to monitor the edge transitions when IDELAY taps are applied to the registered clock input. The edge transition detection and the number of taps applied determine the data valid window width and the tap location to center align the data with respect to the clock.

Since this method requires sampling a receive clock, a slight change is made to the recovered clock network connection. Instead of directly connecting the clock input into a BUFIO, an ISERDES is inserted in between this connection.

The designer must connect the clock into the ISERDES D input. The ISERDES outputs used are the unregistered (O) output and the registered (Q) outputs. The O output is connected to the BUFIO input. IDELAY is only applied to the Q outputs. Table 7 summarizes the ISERDES settings.

Table 7: **ISERDES Settings**

Parameter Name	Parameter Value
BITSLIP_ENABLE	False
DATA_RATE	SDR
DATA_WIDTH	4
INTERFACE_TYPE	Networking
IOBDELAY	IFD
IOBDELAY_TYPE	Variable
IOBDELAY_VALUE	0
NUM_CE	1
SERDES_MODE	Master

The block diagram in Figure 11 shows the clock-to-data recovery scheme.

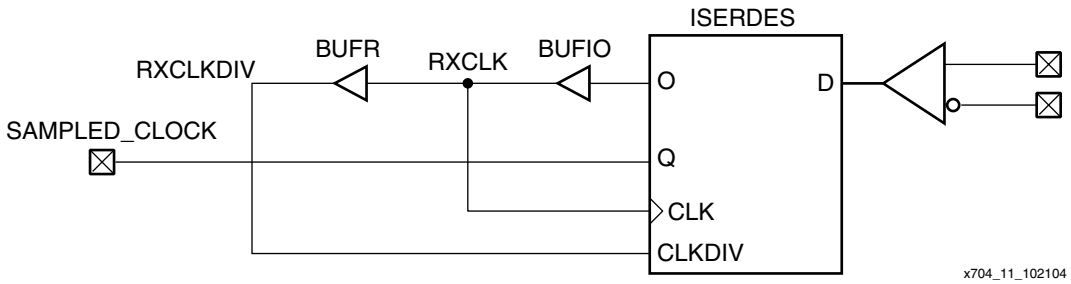

Figure 11: **Alternate Clock Data Recovery Circuit**

The process to determine the window width and to center align the data follows:

1. Increment IDELAY taps until a 1-to-0 edge transition is found. The first 1-to-0 edge transition indicates the beginning of a data valid window.
2. Begin counting the number of IDELAY taps.
3. Continue incrementing IDELAY taps until another 1-to-0 edge transition is found. When the second 1-to-0 edge is found, the data valid window width has been determined.
4. Decrement the IDELAY taps by ½ of the data valid window width. This allows the IDELAY tap at the center of the data valid window width to be used.
5. The IDELAY tap is moved for all data channels to the amount found in step 4.

This alignment scheme assumes minimized skew between all data channels and clock channel and the data transition edge is closely aligned to the positive clock of the incoming clock.

Figure 12 illustrates the relationship between the receive clock (RXCLK) and the sampled/delayed clock to show the algorithm.

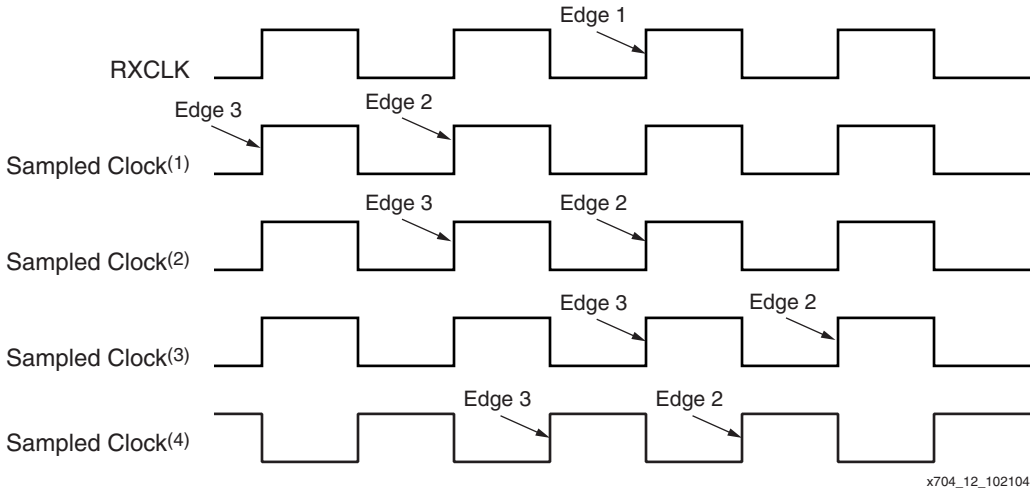

Figure 12: **Timing Relationship Between Rx Clock and Sampled Clock**

In Figure 12, the RXCLK is initially behind the clock to be sampled (edge 1 comes after edge 2). The clock to be sampled is incremented by the tap delays until a 1-to-0 transition is found at the Q outputs of the ISERDES. When this is true, edge 2 comes after edge 1. The clock to be sampled is continually incremented until another 1-to-0 transition is found. When this is true, edge 3 comes after edge 1. Finally, edge 1 is placed between edge 2 and edge 3 to center align the data with respect to the clock.

The current bus alignment module state is indicated by a combination of the DATA_ALIGNED and SEND_CLOCK pins. Table 8 summarizes the relationship between these two pins and the alignment process state of this module.

Table 8: **DATA_ALIGNED and SEND_CLOCK Relationship with Alignment State**

DATA_ALIGNED and SEND_CLOCK Value	State
00	Performing word alignment (optional)
01	Performing bit alignment
10	All alignment processes are completed
11	Don't care – Defaults to all alignment processes are completed

When the clock to data alignment process is complete, this module moves to the data reordering portion of the alignment. After asserting the USE_BITSLIP pin to a logic High and setting TRAINING_PATTERN into a desired 4-bit training pattern, the reordering portion uses the BITSLIP pattern until the desired 4-bit training pattern is found. It also requires the transmitting device to send the desired pattern.

To reduce slice utilization, remove the logic in the state machine by removing the `0101` and `0110` states and the associated control pins generated by these states. Also, remove the BITSLIP pin connections from ISERDES and set BITSLIP_ENABLE to FALSE.

When both IDELAY and BITSLIP operations are completed, the DATA_ALIGNED bit will be asserted High.

FIFO16 Modules

In this application note, a FIFO is needed to transfer the data recovered from the Regional Clock domain to the Global Clock domain. By transferring to the Global Clock domain, any logic required for data processing with the recovered data will not be limited to three clock regions. The logic can be implemented across the FPGA.

Two FIFO16s primitives are instantiated to create two 512 x 36 bit FIFOs. Since the data deserialized by ISERDES is 64-bit, the reference design uses two FIFO16.

Additional control logic is implemented for the FIFO16 to operate, with the following conditions:

1. Begin writing into FIFO from Regional Clock domain after all ISERDES have finished alignment process
2. Begin reading data into the Global Clock domain when at least 50 entries are in the FIFO
3. Stop writing data into the FIFO from Regional Clock domain when less than 50 spaces are available in the FIFO

These conditions can be changed depending on the desired conditions. Xilinx recommends a clock frequency of the write clock that is slower than or equal to the read clock. By meeting this clock frequency conditions, a FIFO overflow will not occur.

IDELAYCTRL Module

Since this design uses IDELAY, IDELAYCTRL is needed in order to guarantee proper operation of IDELAY in the Virtex-4 FPGA. IDELAYCTRL requires the following two conditions for proper operation.

- Input reference clock (REFCLK) of 200 MHz
- Minimum of 50 ns of active High reset pulse after startup

For more information on IDELAYCTRL, please refer to the Virtex-4 *User Guide*.

RST_MACHINE Module

This module is used to create a synchronous reset for all elements in given clock domain. This module is also used to create an active High reset pulse for a desired duration of time. As an example, IDELAYCTRL requires an active High reset duration of (50 ns).

To initiate the reset pulse, use an input clock and a stimulus. The reset pulse generated by this the RST_MACHINE module should be connected to all elements in the design that are clocked by the input clock.

The number of clock cycle for the active High reset is the comparator value of COUNT_VALUE in the state machine portion of this module. To shorten or lengthen the duration, change this comparator value.

Table 9 summarizes all the pins available in this module.

Table 9: **RST_MACHINE Module Pin Definitions**

I/O Type	Module Pin Name	Definition
Input	CLK_generic	The clock domain in which the reset pulse is needed
	RST_stimulus	When active High, DOMAIN_RST will be generated
	IRDY	IDELAYCTRL ready signal
Output	DOMAIN_RST	Active High reset pulse output – This should be connected to all reset pins of elements clocked by the clock connected to CLK_generic input pin

Top Level Module - SDR_LVDS_AND_LOGIC_TOP

This module uses Virtex-4 FPGAs and the ML450 development board to demonstrate the Tx and Rx loopback. The SDR_LVDS_TX_RX performs both IDELAY and BITSLIP during the alignment process. It also contains, a PRBS data generator, a FIFO to store data transmitted, and a checker using a DSP48 slice to compare incoming data and data sent. The UCF file targeted for the ML450 board is called sdr_lvds_and_logic_top.ucf. The design functions correctly when both FINAL_BUS_ALIGNED and FINAL_DATA_CHECK are asserted High.

Figure 13 illustrates a simplified block diagram of this module.

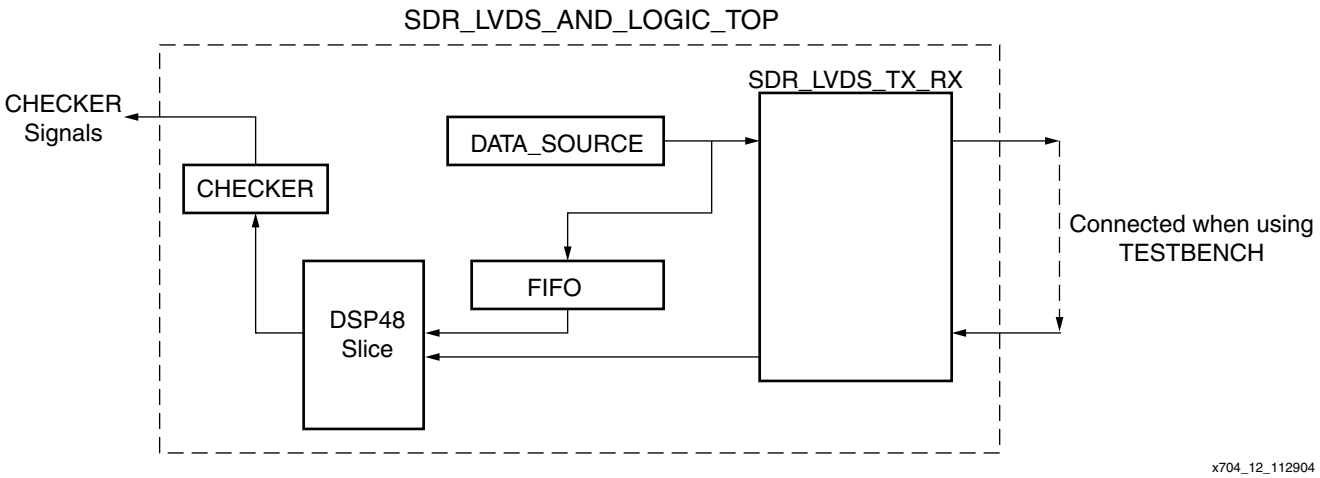

Figure 13: SDR_LVDS_AND_LOGIC Simplified Block Diagram

Simulation for SDR_LVDS_AND_LOGIC

The reference design is simulated using ModelsSim SE 5.8b. The simulation testbench is SDR_LVDS_AND_LOGIC_TOP_TESTBENCH.v, and the script is top.do. Type Run top.do to invoke the script at the Modelsim command prompt. In this simulation, the design functions correctly when both FINAL_BUS_ALIGNED and FINAL_DATA_CHECK are asserted High. Some lines in the .do file must be changed to reflect the working directory or a library location.

ISE Implementation

This design is compiled using ISE 6.3i. Files needed for this implementation are:

- SDR_LVDS_TX_RX.v
- SDR_LVDS_AND_LOGIC_TOP.v (topmost level file)
- SDR_LVDS_AND_LOGIC_TOP.ucf

The UCF file is associated with SDR_LVDS_AND_LOGIC_TOP.v

When using the ML450 development board, select XC4VLX25-11FF668 as the target device.

The following settings must be turned on or turned off:

- Synthesize - XST - Equivalent Register Removal (unchecked)
- Implement Design - MAP Properties - Trim Unconnected Signals (unchecked)

Some warnings may occur. Refer to the readme.txt file for further information on these warnings.

Table 10 summarizes the Virtex-4 device utilization on the ML450 development board.

Table 10: **SDR LVDS Device Utilization on the ML450 Development Board**

Component Name	Device Utilization
Number of External IOB	105
Number of LOCed External IOB	41
Number of External IOBM	17
Number of External IOBS	17
Number of DSP48	4
Number of FIFO16	4
Number of ISERDES	17
Number of OLOGIC	2
Number of OSERDES	16
Number of Slices	304
Number of BUFG	5
Number of BUFIO	1
Number of BUFR	1
Number of DCM	2
Number of IDELAYCTRL	16 (IDELAYCTRLs were not LOC constrained)

Design Summary

The Virtex-4 reference design assumes the following design parameters.

- Requires Flip-Chip packaged Virtex-4 FPGA
- Requires LVDSEXT at the receiver
- Current design is a 4:1 Serializer/Deserializer (SERDES)
- Place Tx and Rx pins in either left or right I/O column
- Group Tx pins as closely as possible to minimize skew (both on the board and on the device).
- Group Rx pins as closely as possible to minimize skew (both on the board and on the device) and the number of clock region used.

Table 11 summarizes the device utilization of this design (excluding the ML450 development board design utilization).

Table 11: **SDR LVDS Device Utilization in a Virtex-4 Device**

Component Name	Device Utilization
IOB	17 differential data and clock inputs 17 differential data and clock outputs 2 differential clock inputs (SDR clock and REFCLK)
FIFO16	2 for Receiver
ISERDES	17 (16 for data, one for bus alignment)
OSERDES	16
OLOGIC	1 (for Tx clock forwarding)
BUFIO	1
BUFR	1
DCM	1
IDELAYCTRL	16 (IDELAYCTRLs are not location constrained)
BUFGs	5
Slices	62

Conclusion

Virtex-4 devices can implement single data rate,16-bit, LVDS data transmission and reception at 700 MHz. This design can easily be expanded for data larger than 16-bit wide data.

Appendix A

ISERDES_ALIGNMENT_MACHINE Changes for Channel Alignment

The channel-alignment method optimally aligns each individual data channel to the center of the clock. This method differs from bus-alignment because the minimization of the skew between all data channels and the clock channel is not needed, however, a training pattern is required by the transmitter.

The channel-alignment methodology uses the training pattern to determine the data valid window at each individual data channel. The training pattern is assumed to be alternating between 1 and 0 at ½ the SDR clock frequency. This method also uses edge transitions to determine the data valid window width, and to center align data with respect to the clock.

The process to determine the window width and to center align the data follows:

1. Request the transmitter to send clock signal by asserting SEND_CLOCK pin.
2. Increment the delay until a 0-to-1 transition is found. This edge transition indicates the beginning of a data valid window.
3. Begin counting the number of IDELAY taps.
4. Continue incrementing IDELAY taps until a 1-to-0 edge transition is found. This edge transition indicates that the data valid window width is known.
5. Decrement the IDELAY taps by ½ of the data valid window width. This allows using the IDELAY tap at the center of the data valid window width.
6. The IDELAY tap of the aligned data channel is moved to the amount found in step 5.
7. Repeat this for the next data channel and until all data channel are aligned (implemented in the ROUND_ROBIN_ALIGN_CONTROL module).

Figure 14 illustrates the timing relationships between the data and clock channel during the alignment process.

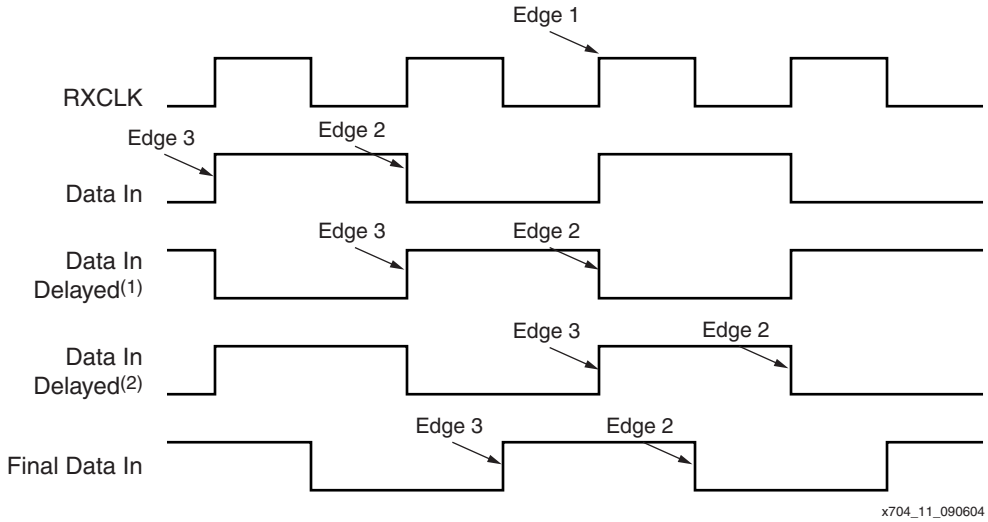

Figure 14: **Timing Relationships Between Data and Clock**

In the Figure 14 timing diagram, edge 1 of RXCLK is sampling "Data In." "Data In" delay is incremented until a positive edge (edge 2) is detected by edge 1 of RXCLK. This state is illustrated by "Data In Delayed (1)." The delay is continually incremented until a negative edge (edge 3) is detected by edge 1 of RXCLK. This state is illustrated by "Data In Delayed (2)." Finally, the delay for "Data In" is decremented by half of the increment value, to center align data with edge 1 of RXCLK. This completes the alignment process.

The current channel alignment module state is indicated by a combination of the DATA_ALIGNED and SEND_CLOCK pins.

ROUND_ROBIN_ALIGN_CONTROL Module

This module controls the alignment process for each receive data channel, when the channel alignment method is used. This module starts the alignment process from channel 1 to x, one at a time by starting at channel 1 and moving to the next channel while incrementing the value of CHANNEL_LOCKED by one when the DATA_ALIGNED pin from ISERDES_ALIGNMENT_MACHINE is asserted High. After CHANNEL_LOCKED value is updated, this module sets the SAP pin to a logic High causing the ISERDES_ALIGNMENT_MACHINE module to perform the alignment process for the new channel. All channel are aligned when CHANNEL_LOCKED is equal to the number of channels in the design (16 in this example). To change the number of channels to be aligned, change the comparator value of CHANNEL_LOCKED in the following code under RX_CLK_AND_DAT to the desired number of channels and the comparator value of CHANNEL_LOCKED.

```
assign BUS_ALIGNED = (CHANNEL_LOCKED == 5'h10) ? 1'b1 : 1'b0;
```

Also change the following code under ROUND_ROBIN_ALIGN_CONTROL to the desired number of channels minus 1.

```
else if((DATA_ALIGNED == 1'b1) && (CHANNEL_LOCKED != 5'h0F))
     begin
       COUNT_CHANNEL = 1'b1;
       NEXT_STATE = 2'b01;
     end
else if((DATA_ALIGNED == 1'b1) && (CHANNEL_LOCKED == 5'h0F))
     begin
       COUNT_CHANNEL = 1'b1;
       NEXT_STATE = 2'b10;
     End
```

When the ROUND_ROBIN_ALIGN_CONTROL is completed, the BUS_ALIGNED pin of RX_CLK_AND_DAT is asserted High.

Table 12 summarizes all the pins available in this module.

Table 12: **ROUND_ROBIN_ALIGN_CONTROL Module Pin Definitions**

I/O Type	Module Pin Name	Definition
Input	RXCLKDIV	Received Clock divided by 4
	RST	Active High Reset – For all logic
	DATA_ALIGNED	When logic High, current channel selected has finished alignment
Output	START_ALIGN_PROCESS	When logic High, start alignment process for current channel selected
	CHANNEL_LOCKED<4:0>	The number of channels locked at a given time. (5-bits)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/01/04	1.0	Initial Xilinx release.
12/10/04	1.1	Revised Figure 8 and Figure 13 ,Table 6, Table 10, and Table 11, and the “Design Summary” section.
02/08/05	1.2	Revised the section “TX_CLOCKS Module,” page 3, Figure 5, Table 10, and Table 11 to use only the DCM.