



XAPP485 (v1.3) June 9, 2010

1:7 Deserialization in Spartan-3E/3A FPGAs at Speeds Up to 666 Mbps

Author: Nick Sawyer

Summary

Spartan[®]-3E and Extended Spartan-3A devices are used in a wide variety of applications requiring 1:7 deserialization at speeds up to 666 Megabits per second (Mbps). This application note targets Spartan-3E devices in applications that require 4-bit or 5-bit receive data bus widths and operate at rates up to 666 Mbps per line with a clock at 1/7th the bit rate. This type of interface is commonly used in flat panel displays and automotive applications.

Introduction

The Digital Frequency Synthesizer (DFS) block of the Digital Clock Manager (DCM) generates an internal receiver clock in the Spartan-3E/3A FPGA. Because the clock used is 3.5 times the received clock, double data rate (DDR) techniques are used to achieve a deserialization factor of seven. DDR techniques reduce the required clock rate to a reasonable speed, ensures that the clock generation falls within the range of the DFS block of the Spartan-3E/3A FPGA, and reduces overall power consumption. The maximum data rate for the Spartan-3E FPGA is 622 Mbps for the -4 speed grade and 666 Mbps for the -5 speed grade.

The design files for this application note target the Spartan-3E family. The Extended Spartan-3A family, which includes the Spartan-3A, Spartan-3AN, and Spartan-3A DSP families, supports the same design approach. The maximum data rate for the Extended Spartan-3A FPGA is 640 Mbps for the -4 speed grade and 700 Mbps for the -5 speed grade.

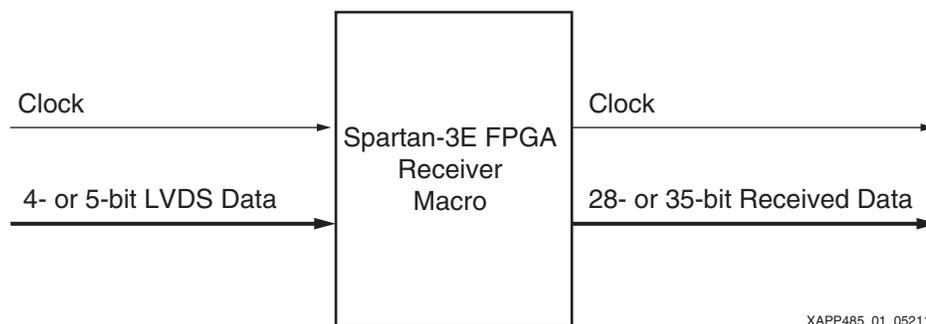


Figure 1: 1:7 Receiver Module

Input Pad Placement

The forwarded input clock for the receiver is used both as a clock and as a framing signal. That is, the position of the rising edge of the clock also indicates when the next data word starts (“Logic Description,” page 3 provides more information). The next data word starts two bit times after each rising edge. This document assumes that the incoming clock and data are exactly aligned where possible.

Figure 2 shows this relationship of clock and data along with the location of each bit to be received within the 7-bit long and 4- or 5-bit-wide frame.

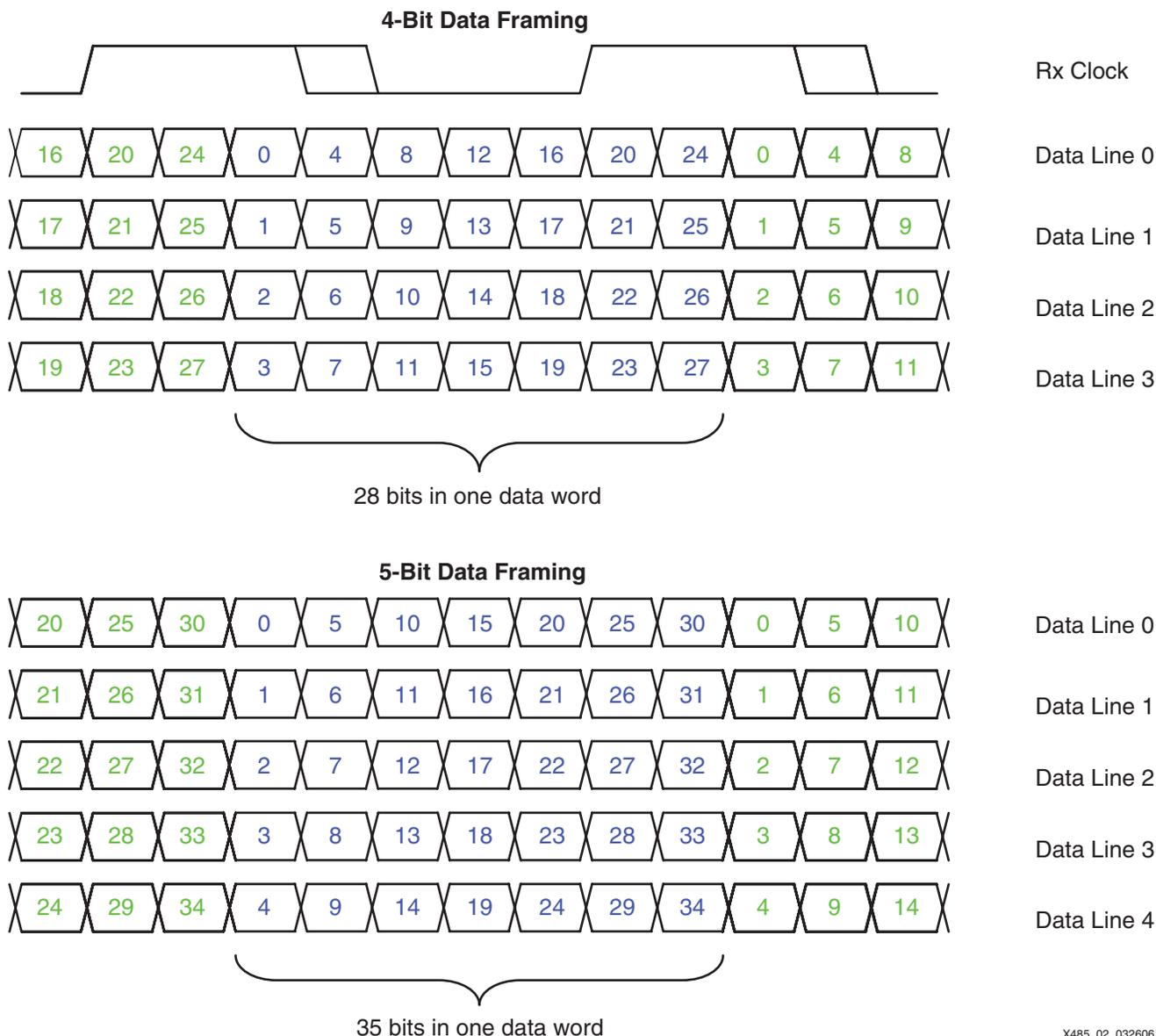


Figure 2: 4-Bit and 5-Bit Receive Data Formatting

If the application requires a different bit ordering, the descrambling is easily performed in the design code as described in [“Design Files.”](#)

The clock line is therefore used as an input to the DFS and is also sampled inside the Input/Output Block (IOB) using the DDR input flip-flops. To minimize clock skew between the clock and data, choose inputs close to the clock-capable IOBs (GCLK input pins) that can drive the DCM input directly. All the input lines should be on either the top (I/O Bank 0) or bottom (I/O Bank 2) of the Spartan-3E/3A device. Bank 0 is preferred because Bank 2 contains the I/Os that are borrowed during device configuration. The macro performs equally well in either bank.

The IOBs of Spartan-3E/3A FPGAs contain a cascade function that captures the data received on the falling clock edge and resynchronizes it to the following rising edge. This resynchronization is done by borrowing one of the flip-flops in a neighboring IOB. No loss of efficiency occurs for a differential signal because the signal requires two input pins anyway. The advantage of this new logic function is that all subsequent logic inside the part is clocked only on the rising edge, simplifying the critical path timing.

Clock Considerations

As mentioned in “Introduction,” page 1, the incoming clock is multiplied by 3.5 inside the DFS, but it also requires a slight phase shift to place the clock edge in the middle of the incoming data eye. This value (nominally 90 degrees) changes slightly with frequency, but a phase shift value of 55 when using ISE® software release 8.1 SP3 or higher provides the best results (that is, the most centered operating eye over the operating range of frequencies). The ZIP files provide details on how to add this attribute to the DFS along with example code.

Another possible mechanism to determine the required value for the phase shift is to let the FPGA work out the value at power-up. This approach gives an extremely accurate way of placing the clock in the middle of the data eye, because effectively, process (P) is being removed as a variable in this way. Temperature (T) and voltage (V) are zeroed out by the DCM in its normal operation. The operation of the automatic phase aligner is described in “Automatic Phase Alignment,” page 9.

Data capture is performed using DDR flip-flops; that is, one rising edge sample and one falling edge sample are required. This can be achieved in two different ways as detailed in the Spartan-3E/3A FPGA data sheets and [UG331](#), *Spartan-3 Generation FPGA User Guide*.

- One clock can be distributed from the DFS using one global buffer, and this clock is inverted where required, or
- Two clocks, 180 degrees apart can be distributed using two global buffers from the CLKFX and CLKFX180 outputs of the DFS.

The advantage of the second method is that any duty cycle distortion in the global clock networks becomes unimportant because only rising edges are used. For this reason, this method is recommended for high-speed interfaces.

The DFS operates at speeds up to 333 MHz (666 Mbps per line) in both Spartan-3E speed grade devices, but is limited by the speed of the internal logic and the global clock distribution buffers to a maximum rate of 622 Mbps in the -4 device. Very importantly, the DFS also operates at speeds down to 5 MHz (17.5 Mbps per line) without requiring changes to the FPGA bitstream or reprogramming the DFS. This advantage is important in systems where the incoming data stream changes frequencies, such as in multisync monitors.

Figure 3 shows the two generated clocks and their phase shifts. Because the internal clock has been multiplied by 3.5, it is not always in phase with the received clock. Sometimes (every other clock) alignment is on rising edges, and sometimes alignment is when the slow clock rises and the fast clock falls. This alignment is important to remember for the deframing logic.

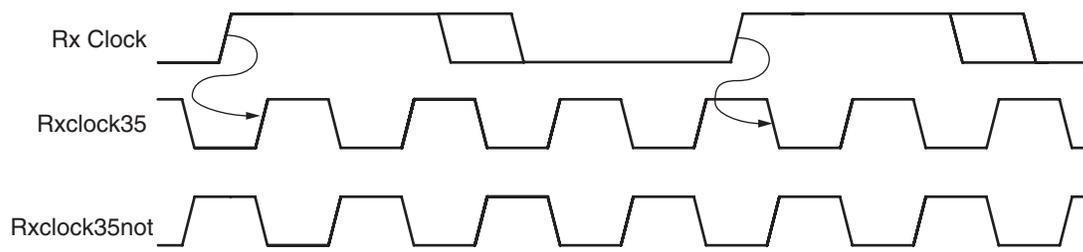


Figure 3: Clocks

Logic Description

The logic for the receiver is simplified by using the cascadable IOB DDR flip-flops. Figure 4 shows the 5-bit interface. The design of the 4-bit interface is identical, just somewhat smaller.

Two bits of data are clocked in per line, per high-speed rxclk35 clock period, thus giving 10 bits (5-bit data width) or 8 bits (4-bit data width) of data to register each high-speed clock. Because this high-speed clock is a 3.5x multiple of the incoming slow clock, the bit deserialization and deframing are slightly different under the two alignment events described in “Clock Considerations.” Basically, 35 bits (5-bit data width) or 28 bits (4-bit data width) are output each slow speed clock cycle. However, for 5-bit data width, the receiver captures 40 bits during one

slow clock cycle and 30 bits the next; for 4-bit data width, the receiver captures 32 bits per one slow clock cycle and 24 bits the next. The decision as to which bits are output and when is made by sampling the incoming clock line and treating it as another data line. In this way, the multiplexer provides correctly deframed data to the output of the macro synchronous to the low-speed clock.

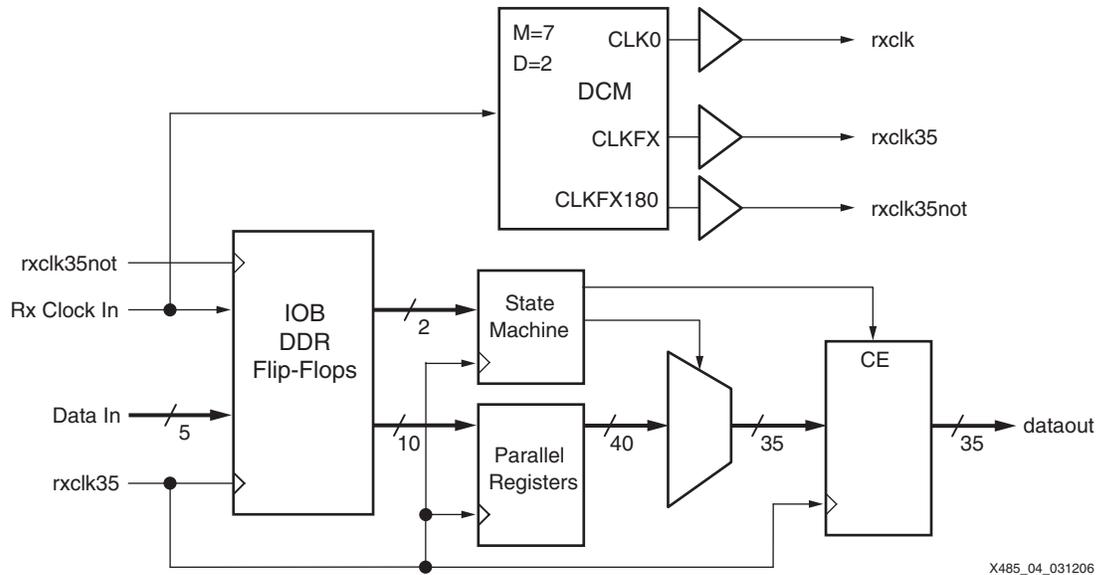


Figure 4: Spartan-3E/3A FPGA 1:7 Receiving Logic (5-Bit Module)

There is no data buffering or flow control included in the macro, because this interface is a continuous high-speed stream and requires neither.

Timing Analysis

The timing analysis for the receiver consists of subtracting the various sources of timing errors and uncertainty from the bit period in picoseconds (ps) equivalent to the bit rate. The value remaining after this analysis is the margin available to the system. A positive number indicates that the system has sufficient margin and will function properly. The parameters to be subtracted are itemized below.

A specification that often appears in data sheets for ASSPs or other devices that perform a similar deserialization function is called the receiver skew margin or RSKM. This value is generated by subtracting only the sources of uncertainty that exist inside the receiver from the bit period, and then dividing the result by 2. An illustration of RSKM is shown in Figure 5.

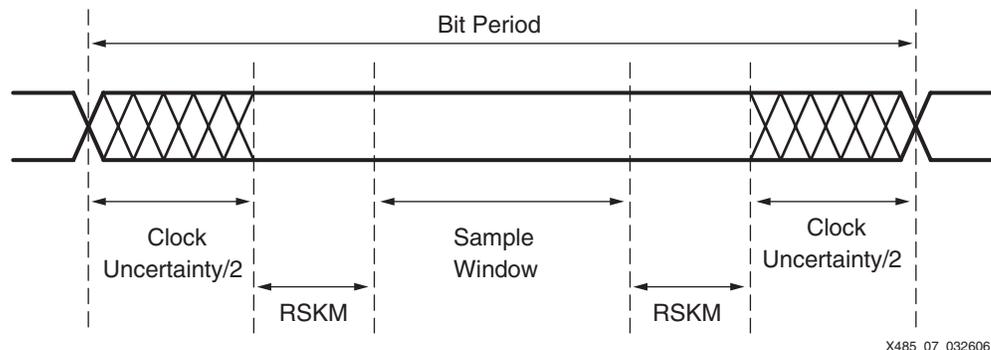


Figure 5: Receiver Skew Margin (RSKM)

For the interface described here, the sources of uncertainty are:

1. All mismatch and silicon variations are bundled into one parameter, denoted as T_{SAMP} which is guaranteed by characterization to be better than 600 ps for all Spartan-3E/3A devices with LVDS signalling. This number includes:
 - a. The setup and hold window of the device, which is the time that data must be present and valid relative to the internal synthesized clock at the IOB flip-flops.
 - b. Skew between the two global buffers distributing the high-speed clock and its complement.
 - c. The package skew among all the data and clock lines.
 - d. The internal clock skew between the IOB flip-flops in the device. This number varies with the placement of the input lines in the package. If all the Xilinx placement guidelines described herein are followed, this number is less than 50 ps.
2. Jitter and timing uncertainty is another important source that cuts into overall timing budget. This parameter is referred to in the data sheet as CLKOUT_PER_JITT_FX. Because this parameter strongly depends on the environment in which the Spartan-3E/3A FPGA is used, it is not possible to guarantee a worst-case number without knowing the environment. However, Xilinx has performed extensive characterization with various amounts of noise, and expects this jitter for all Spartan-3E/3A devices to be better than plus or minus (200 ps plus 1% of the output clock period). The chip and environmental factors that contribute to this number include (but are not limited to):
 - a. The jitter introduced by phase shifting in the DFS unit when it multiplies the incoming clock by 3.5.
 - b. Incoming clock jitter, which is obviously dependent on the system in question. While the characterization number, CLKOUT_PER_JITT_FX, includes a reasonable amount (150 ps) of input clock jitter, it is clear that increasing input clock jitter adversely affects this parameter.
 - c. Any skew between pins on the driving device and skew between traces on the PCB. This factor is also dependent on the system in question.
 - d. Excessive switching activity in the fabric of FPGA can also contribute to chip jitter and timing uncertainty. Typical fabric switching activity is 12% or less for most applications. The Xilinx characterized number is based on 25% fabric switching activity at 40 MHz.
 - e. Switching I/Os at high drive strengths and the frequency of switching contribute to the additional timing uncertainty. The Xilinx characterization result includes the noise of 40 simultaneous switching outputs (SSOs) running at 40 MHz.
 - f. The board design and chip package are also important factors. The Xilinx characterization number is based on a four-layer board and an FT256 package.

Examples of generating the system margin and RSKM are shown below. A timing calculator within an Excel spreadsheet is provided in the downloadable design files ([xapp485.zip](#)). An illustration of RSKM is shown in [Figure 5](#).

The example margin timing analysis is for a 600 Mbps design, where the DFS clock is 300 MHz. To determine the system margin:

	Bit Period	1666 ps		1/600 Mbps in ps
	– T_{SAMP}	600 ps		Xilinx number
	– CLKOUT_PER_JITT_FX	$400 + (0.02 \times (10^6 / 300))$ ps		Data sheet parameter
	– Driving Device and PCB Skew	500 ps		Estimated
		= 100 ps		System Margin

To calculate the RSKM using the same numbers:

	Bit Period	1666 ps	1/600 Mbps in ps
	- T _{SAMP}	600 ps	Xilinx number
	- CLKOUT_PER_JITT_FX	400 + (0.02 × (10 ⁶ /300)) ps	Data sheet parameter
		= 600 ps	
	Divide by 2 =	300 ps	RSKM

These calculations are valid for either the -4 or -5 speed grade of a Spartan-3E/3A device working inside its temperature and voltage limits as given in [DS312](#), *Spartan-3E FPGA Data Sheet*, and in [DS529](#), *Spartan-3A FPGA Data Sheet*. [Table 1](#) defines the T_{SAMP} parameter used for these calculations.

Table 1: T_{SAMP} Parameter Used in Calculations

Symbol	Description	Speed Grade		Units
		-5	-4	
		Max	Max	
T _{SAMP}	Setup and hold capture window of an IOB flip-flop, relative to the internally synthesized clock, using required design pinout recommendations. Includes the skew between the two global buffers distributing the clocks, the package skew among all data and clock lines, and internal clock skew between I/O flip-flops in the device.	600		ps

Design Files

The design files include both Verilog and VHDL descriptions for both 4-bit and 5-bit versions of the receiver interface, and are available from the Xilinx website ([xapp485.zip](#)). The files include source code, design examples, timing constraints (UCF files), and example pinouts for many part/package combinations. For part/package combinations not included in the ZIP file or any other questions, contact spartan3e.serdes71@xilinx.com.

The design hierarchy is very simple. A top level instantiates the required I/O and provides details on how to set up the DCM. The top level also instantiates a wrapper, which in turn instantiates the receiver module. The best place to perform any bit manipulation, if required, is within the wrapper module. Some interfaces also require that four (or five) of the data bits perform a DC-balancing function. This function requires extra logic. Send an e-mail to spartan3e.serdes71@xilinx.com for more information.

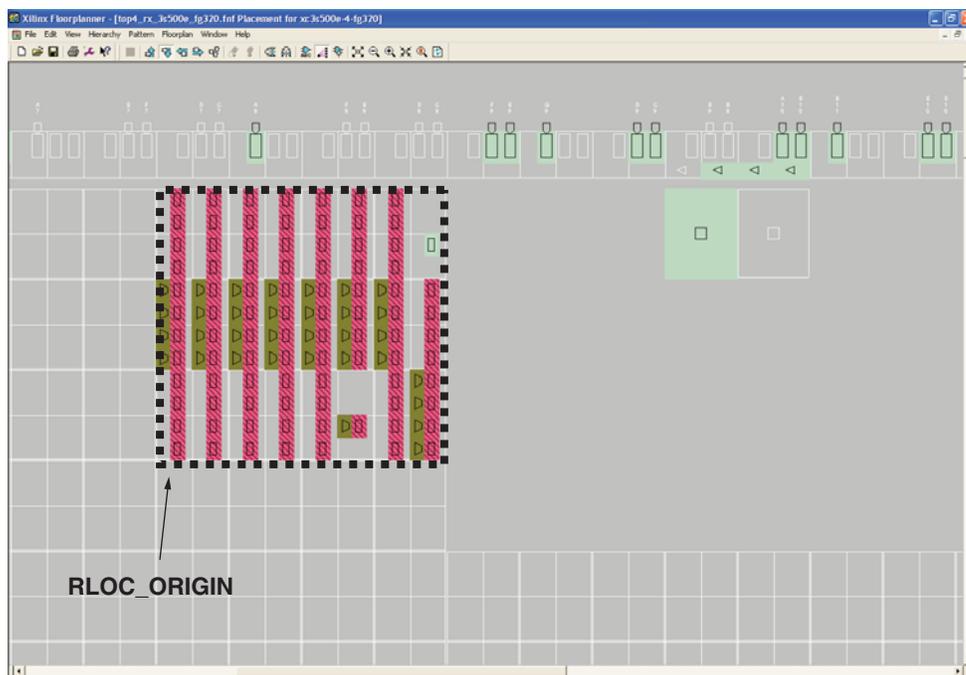
Processing the Design

The design files have been tested with ISE software 9.2 SP3 and Synplicity 8.4. For all ISE versions with either VHDL or Verilog, two modifications to the environment must be done:

1. The user must ensure that ISE software is keeping hierarchy – this is not the default (though it is for Synplicity). Right-click **Synthesize-XST** to reveal the associated **Properties** options, and make sure “Keep Hierarchy” is set to **Yes**.
2. When using ISE software (this is not required for Synplicity), the mapper must be run using the command line switch **-ignore_keep_hierarchy**. To do this, expand **Implement Design**, and then right-click on **Map** to access **Properties**. Then, for **Other Map Command Line Options**, add **-ignore_keep_hierarchy**.

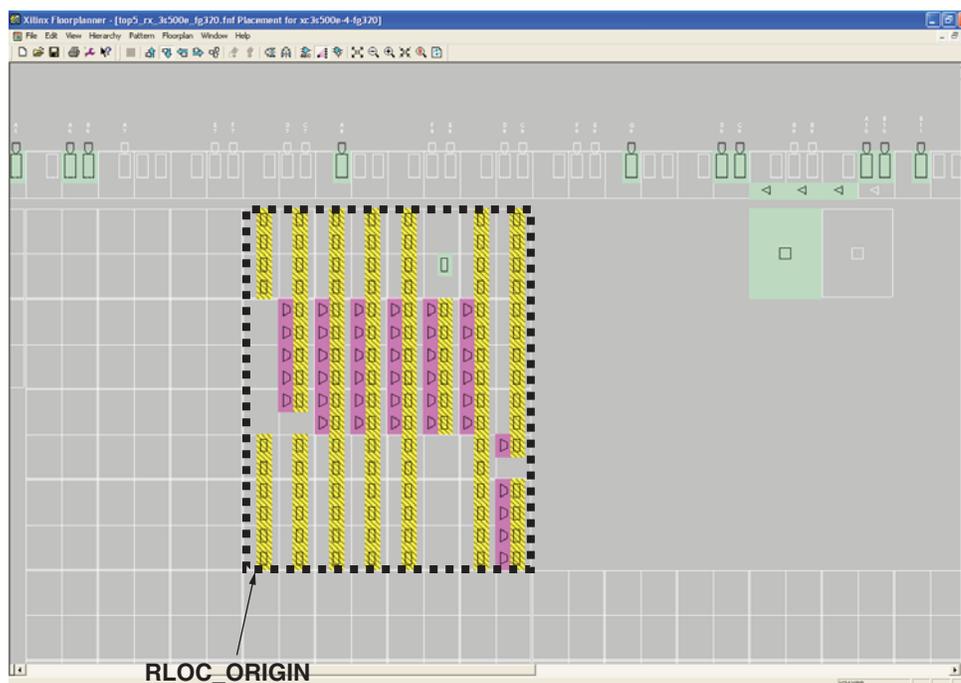
Floorplanning the Design

Place the receiver logic near the input pins located in I/O Bank 0 by using an RLOC_ORIGIN statement in the design constraint file (*.UCF). The 4-bit version receiver module is actually a 4-wide x 3-high block of CLBs (see example in [Figure 6](#)), and the 5-bit version is slightly larger at 4-wide x 4-tall (see example in [Figure 7](#)).



X485_05_051210

Figure 6: 4-Bit Spartan-3E FPGA Receiver Macro



X485_06_051210

Figure 7: 5-Bit Spartan-3E FPGA Receiver Macro

The DCM blocks have dedicated output connections to the global buffer inputs and global buffer multiplexers on the same edge of the device, either top or bottom. They are an integral part of the FPGA's global clocking infrastructure. (See [Figure 8](#).) To ensure the dedicated connection is used, constraining the DCM is required. Additional information may be found in the DCM Locations and Clock Distribution Network Interface section of [UG331, Spartan-3 Generation FPGA User Guide](#).

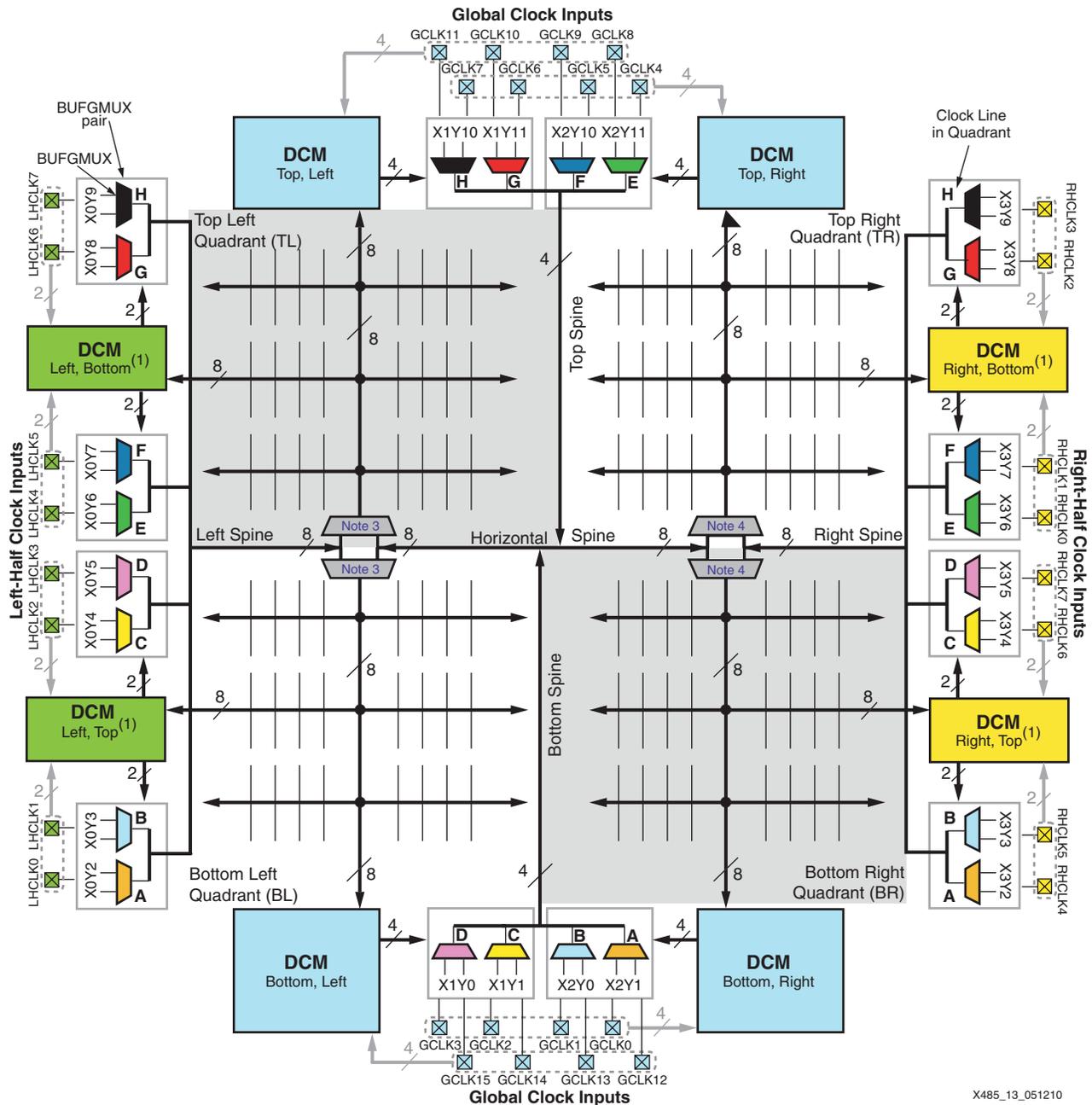


Figure 8: Spartan-3E and Extended Spartan-3A Family Internal Quadrant-Based Clock Structure

Note: This diagram also appears in the Global Clock Resources section of [UG331, Spartan-3 Generation FPGA User Guide](#). For more detailed information, refer to the notes and cross-references following the figure in the user guide.

Additionally when using the DCM to generate high speed clocks to drive the double data rate output flip-flop element, ODDR2, a specific BUFGMUX is recommended for both CLKFX and CLKFX180 to minimize period jitter. See Table 3-17 Recommended DCM/BUFG Connections in UG331.

Incorrect DCM and BUFG placement may result in incorrect phase alignment. An example of the required .ucf constraints would be:

```
inst "dcm_rxclka" LOC = "DCM_X0Y1";
inst "rxclk35_bufg" LOC = "BUFGMUX_X0Y6";
inst "rxclk35not_bufg" LOC = "BUFGMUX_X0Y9";
```

Termination Resistor Considerations

The incoming data and clock LVDS connections can be terminated either on the PCB using 100Ω resistors or inside the FPGA using the DIFF_TERM style of IOB. The internal DIFF_TERM resistors are nominally 120Ω in the Spartan-3E FPGAs, which is a bit higher than the ideal, but they do have the advantage of being located immediately at the input pins of the LVDS amplifier. The DIFF_TERM resistors are nominally 100Ω in the Spartan-3A FPGAs when V_{CCO} is 3.3V. The DIFF_TERM capability is not available on input-only pins of the Spartan-3E/3A FPGA. It is only available on pins that have bidirectional capability.

Automatic Phase Alignment

The principle of automatic phase alignment is very simple. A known input data signal (actually, the input clock) transitions from Low to High every seven bit times. As previously discussed, the DCM has generated a high-speed clock running at 3.5 times the speed of the input clock. Thus the input clock can be sampled with a flip-flop inside the IOB in exactly the same way as the incoming data lines are sampled. Figure 9 shows the functional block diagram, and Figure 10 shows the active signals.

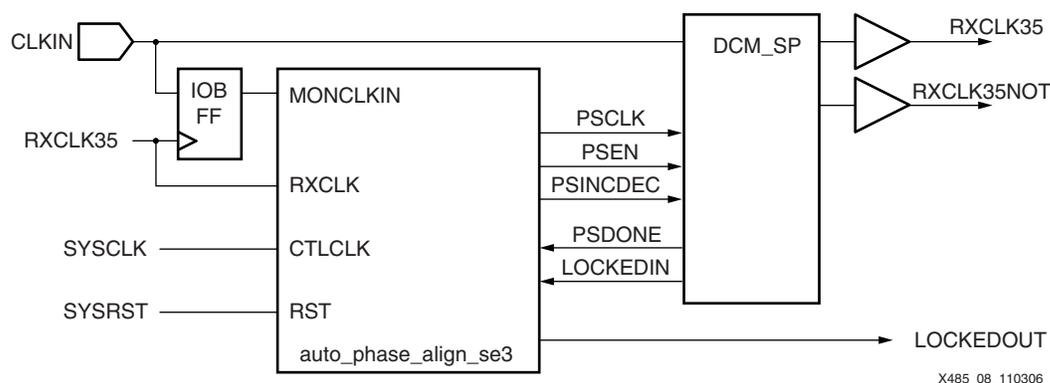


Figure 9: Topology of Automatic Phase Aligner

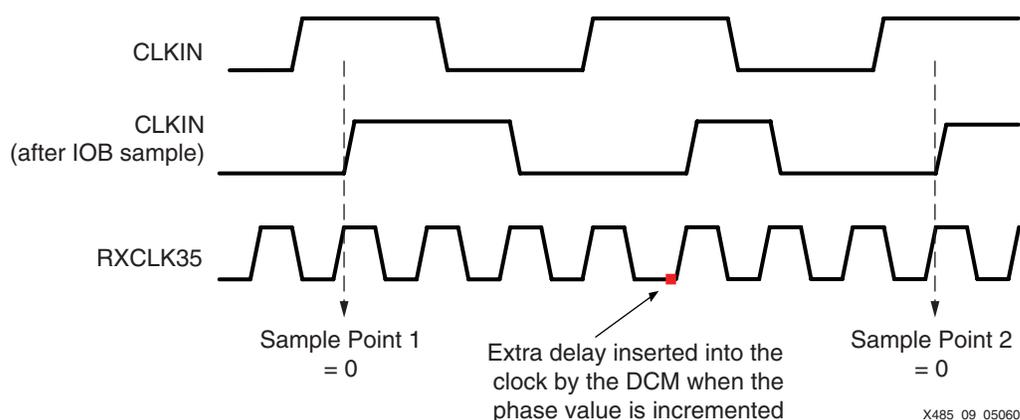


Figure 10: CLKIN Being Sampled by the High-Speed Clock (RXCLK35)

Because an expected bit pattern is known when the input clock line is sampled, the sampling clock can be moved in time (via the DCM phase shifter) to where the setup point is found. The input pattern changes as shown in Figure 11. Thus the sampling clock is moved to measure the setup time of the input flip-flop and then store a value (n_1) equal to the number of steps that the sample clock has been advanced by the DCM. It is not important whether the setup point was found using the positive or negative edge of the high-speed clock.

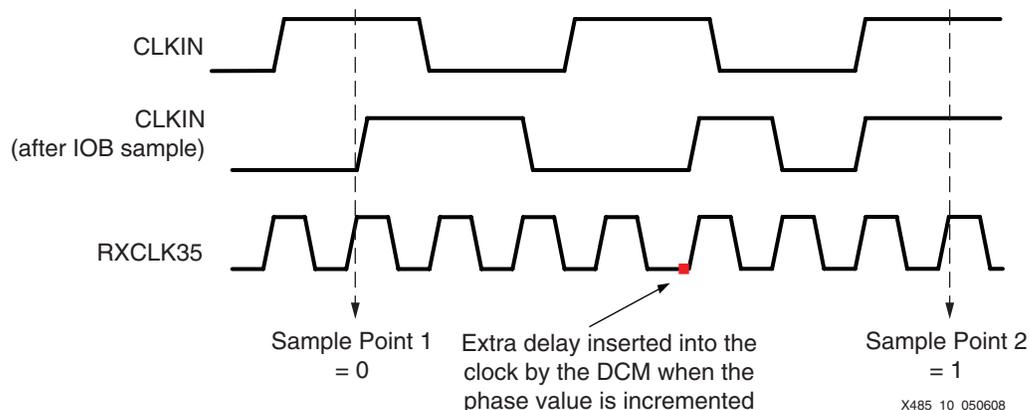


Figure 11: First Reference Point Found When Sample Changes from 0 to 1

The high-speed sample clock continues to be moved until the next edge of the sample clock again reaches the setup time of the IOB flip-flop, saving also this value (n_2) as shown in Figure 12. The difference between n_1 and n_2 corresponds directly to one bit time, no matter what the incoming frequency.

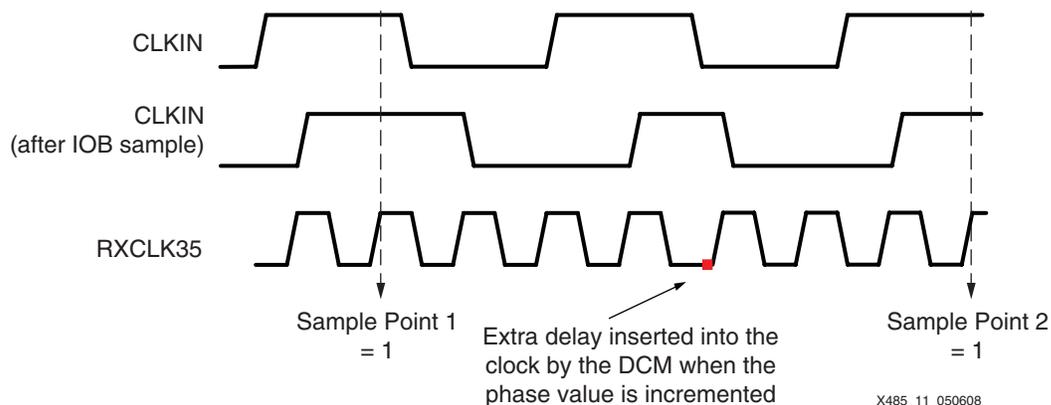


Figure 12: Second Reference Point Found When Sample Changes from 0 to 1

The Spartan-3E/3A FPGA is slightly different from the original Spartan-3 FPGA in that the variable mode shift of the phase shifter in the DCM is performed directly in time (in picoseconds). The fixed mode shift of the Spartan-3E/3A device and both modes of the Spartan-3 device are performed in phase increments. Each increment of the Spartan-3E/3A phase shifter is approximately 25 ps, but this value varies from device to device. For this application, it does not matter what that value is because it is not important to the calculations.

Taking the values n_1 and n_2 from the circuit above, a value (n_3) can be calculated that is the average of n_1 and n_2 , and is the *ideal* point for the clock to be in the middle of the data eye. The sampling clock is moved again towards this point, and the status line, LOCKEDOUT, is driven High to indicate completion of the alignment operation to the rest of the device. Figure 13 shows this entire process in a simulation waveform.

The phase aligner logic is basically a state machine that performs the described operation. Its code is included in [xapp485.zip](#) with examples of instantiation and use. Available outputs provide n_1 , n_2 , and n_3 in both BCD and binary formats, and available inputs either increment or decrement the phase value manually. With this circuit, alignment is performed every time the FPGA is powered up or reset, and the middle of the data eye is always found for maximum noise immunity. The logic does, however, consume space inside the FPGA—about 50 slices when the BCD outputs are not used.

This entire mechanism could also be implemented by a microcontroller, such as the PicoBlaze™ microcontroller, for designers who prefer that route.

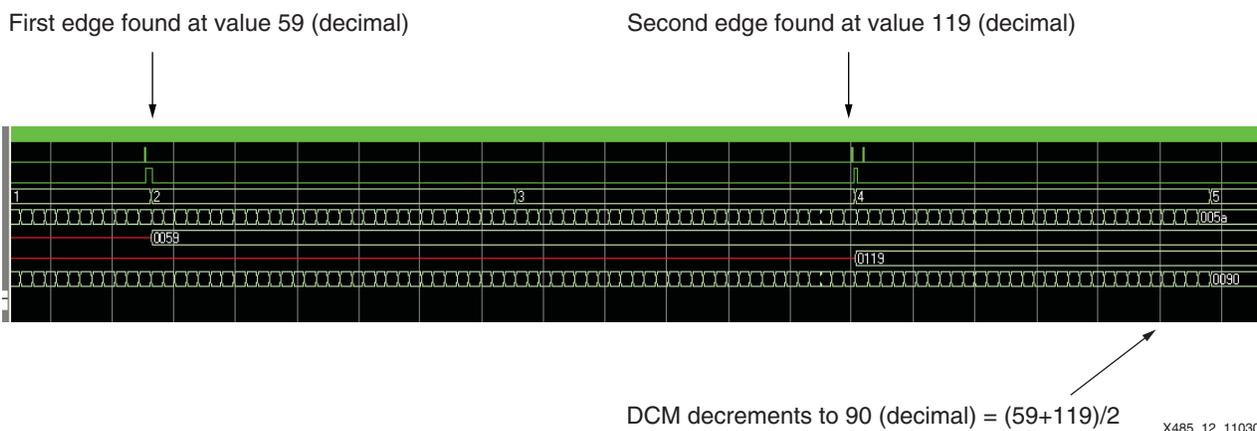


Figure 13: Simulation Waveform of the Alignment Process

Related Documents

[XAPP486](#) 7:1 Serialization in Spartan-3E FPGAs at Speeds Up to 666 Mbps

Conclusion

Spartan-3E/3A FPGAs perform in a wide variety of applications requiring 1:7 deserialization at speeds up to 700 Mbps, depending on the family and speed grade used (see the summary in [Table 2](#)).

Table 2: Speed Based on Family and Speed Grade

	Spartan-3E	Spartan-3A
-4	622 Mbps	640 Mbps
-5	666 Mbps	700 Mbps

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/03/06	1.0	Initial Xilinx release.
11/10/06	1.1	Added pin swapping logic. Added optional auto phase alignment function. Tested with release 8.2.03 of the Xilinx toolset.
05/27/08	1.2	Updated for Spartan-3A/3AN/3A DSP families. Updated jitter discussion in "Timing Analysis" by replacing special CLKOUT_PER_JITT_FX_35 (T_{J35}) with data sheet parameter CLKOUT_PER_JITT_FX. Updated sample points in Figure 10 , Figure 11 , and Figure 12 . Added reference to XAPP486 for serialization. Added reference to SCD 4103 in Table 2 . Updated links. Updated design files. Tested with release 9.2.03 of the Xilinx toolset.
06/09/10	1.3	Updated "Clock Considerations" section. Added new Figure 8 and associated text regarding use of proper DCM constraints and BUFG placement. Updated Table 2 (removed SCD4103 reference). Updated Spartan family device names.

Notice of Disclaimer

Xilinx is disclosing this Application Note to you “AS-IS” with no warranty of any kind. This Application Note is one possible implementation of this feature, application, or standard, and is subject to change without further notice from Xilinx. You are responsible for obtaining any rights you may require in connection with your use or implementation of this Application Note. XILINX MAKES NO REPRESENTATIONS OR WARRANTIES, WHETHER EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL XILINX BE LIABLE FOR ANY LOSS OF DATA, LOST PROFITS, OR FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES ARISING FROM YOUR USE OF THIS APPLICATION NOTE.