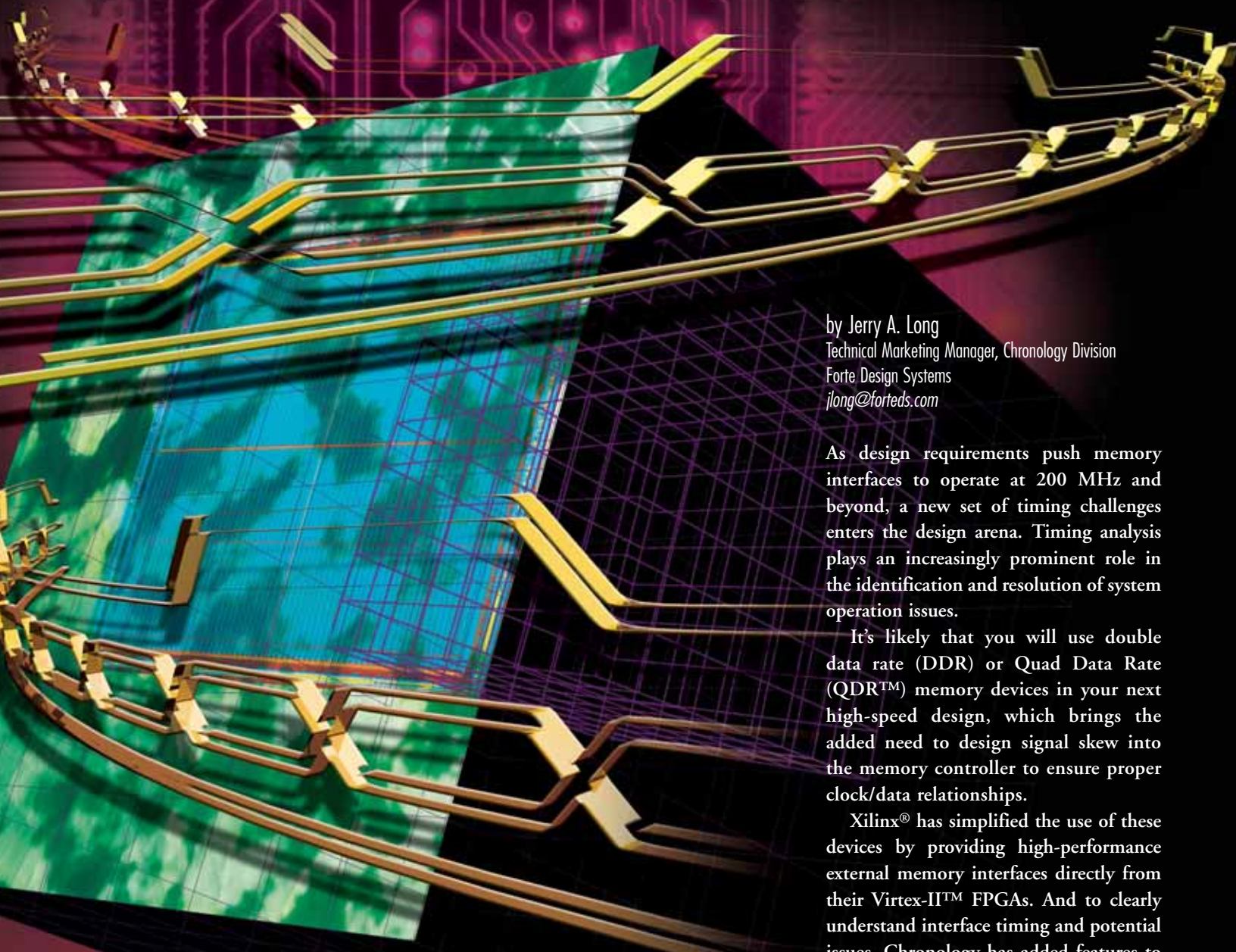
Control Your QDR Designs

A step-by-step guide to solving QDR memory data capture challenges with Virtex-II FPGAs.



by Jerry A. Long
Technical Marketing Manager, Chronology Division
Forte Design Systems
jlong@forteds.com

As design requirements push memory interfaces to operate at 200 MHz and beyond, a new set of timing challenges enters the design arena. Timing analysis plays an increasingly prominent role in the identification and resolution of system operation issues.

It's likely that you will use double data rate (DDR) or Quad Data Rate (QDR™) memory devices in your next high-speed design, which brings the added need to design signal skew into the memory controller to ensure proper clock/data relationships.

Xilinx® has simplified the use of these devices by providing high-performance external memory interfaces directly from their Virtex-II™ FPGAs. And to clearly understand interface timing and potential issues, Chronology has added features to its TimingDesigner® tool to streamline the exchange of critical timing data with the Xilinx ISE software suite.

TimingDesigner generates place and route constraints, allowing direct use of post-place-and-route timing information to verify desired FPGA interface operation. You can accurately determine the proper clock/data relationship for your DDR/QDR memory interface design, export the information as constraint data into ISE, and automatically get visual verification of design success.

For the purposes of this design guide, we'll use a QDR SRAM device to illustrate a source-synchronous interface design. Stepping through the design flow, we implemented a QDR memory controller in a Xilinx X2V50 Virtex-II Pro™ FPGA with a focus on the interface signals required for a read operation, utilizing a 133 MHz master clock. The memory device is a Micron™ 18 Mb QDR II four-word burst SRAM (MT54W1MH18J).

The features and principles outlined here are not necessarily unique to high-speed memory design. TimingDesigner's robust timing diagram-based analysis features, coupled with its Xilinx-specific import/export capabilities, allow a level of integration that can be applied to any timing-critical design interface for Xilinx FPGAs.

QDR Interface Timing Requirements

QDR SRAM devices have unique timing requirements for successful read operations. In order to guarantee accurate data capture, QDR devices require the capture clock to be center-aligned within the valid data window. For source-synchronous designs, this means shifting the optional-use echo data clock supplied by the QDR device.

Center Alignment of Capture Clock Edges

Proper timing for read operations of QDR SRAM interface designs requires you to center-align the edges of the capture clock within the valid window of the data bus for accurate data capture, as shown in Figure 1. This is because most SRAM devices typically have a positive setup and a positive hold requirement, and both are roughly the same value. So it makes sense to center the clock edge within the data valid window to maximize the safety margin for latching data. To accomplish this, the memory controller must skew the capture clock.

For source-synchronous capture of read data from a QDR device, you must use the optional echo clock signals (typically CQ and CQ#) provided from the QDR device. With the Virtex-II Pro family of FPGAs, skewing this clock is easily accomplished using one of the digital clock managers (DCM). However, the amount of phase shift necessary to achieve a safe margin is an unknown, given the external PCB skew effects on both the incoming data and its associated clock, as well as the associated routing delays encountered within the FPGA fabric. TimingDesigner will help us to accurately determine this phase shift value.

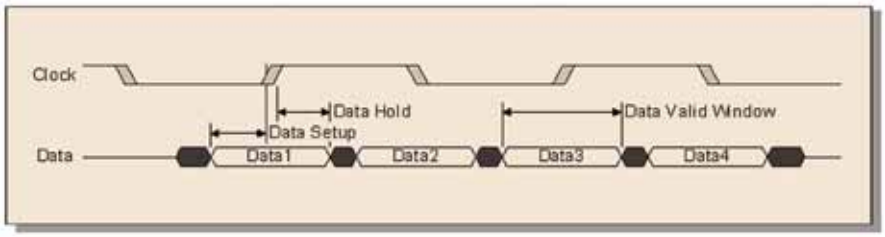


Figure 1 – Illustration of center-aligned clock/data relationship

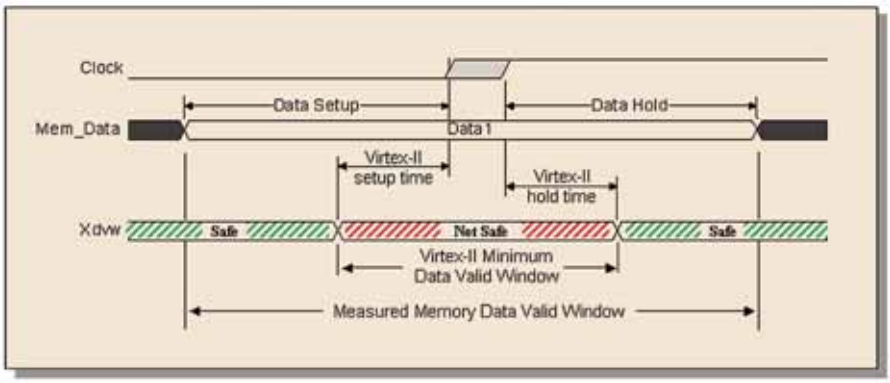


Figure 2 – Center alignment of device minimum data valid window

Capturing Read Data in Virtex-II Pro Devices

You can skew (or delay) the clock signal by using PCB trace delay techniques, or by designing clock delay into the memory controller design. Because PCB trace delay techniques aren't very flexible, it makes more sense to use the incidental PCB trace delay coupled with internal FPGA routing delay and use the Virtex-II Pro DCM element as the "adjustable" component for phase shift within the FPGA memory controller design.

Typically, memory manufacturers recommend center alignment of the clock/data relationship to ensure that various temperature effects the design may encounter (or other unforeseen influences) won't cause an excessive amount of drift in signal position during the read operation and result in a violation of the setup or hold time requirement of the receiving register. In theory, a center-aligned clock edge will maximize the setup and hold times for most devices, allowing sufficient safety margins for signal drift.

However, some devices, like the Virtex II-Pro device, have a negative hold time requirement, which simply means that data can transition to the next value before the clock edge that latches it. In effect, this characteristic places the clock edge to the right (delayed) of the data transition. So for these devices, if you center-align the capture clock within the actual data valid window, you may be satisfying the setup/hold requirements of the device, but the safety margin achieved will be greater for the hold requirement than for setup.

The ideal solution is to provide a maximum safety margin for both the setup and hold requirements of a device, which translates to "balancing" these margins. This provides equal amounts of safety for both, as illustrated in Figure 2.

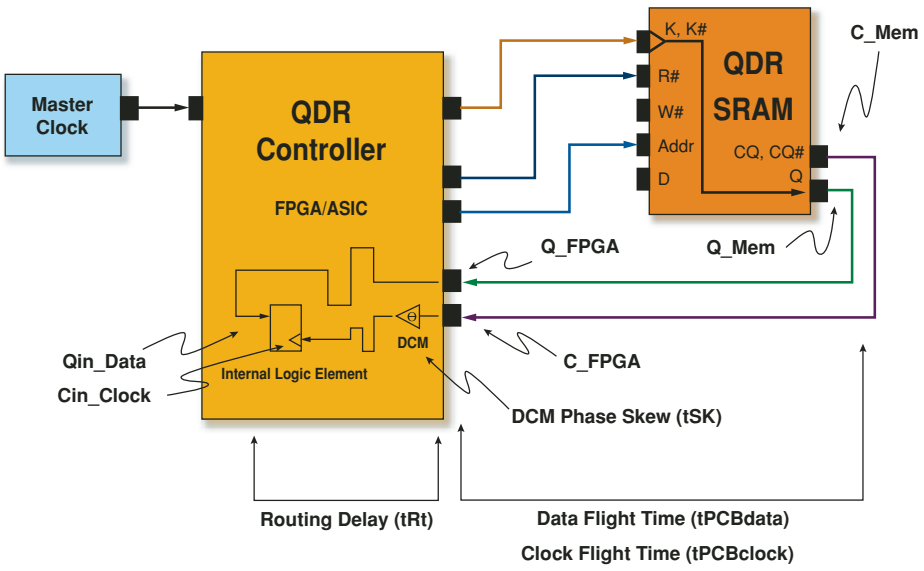


Figure 3 – Illustration of signal delay paths for QDR read operation

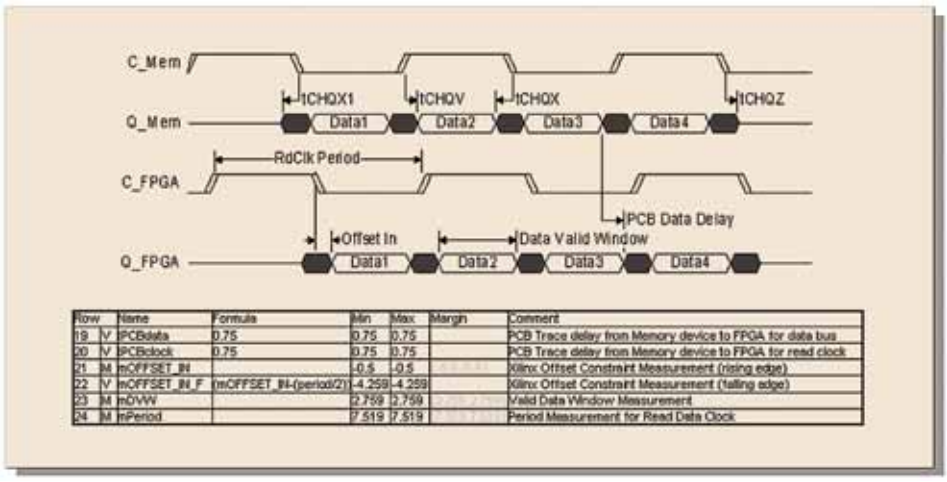
To accomplish this balance, you must determine the minimum data valid window for the receiving device, and center that window within the actual data valid window provided from the memory device, given your design parameters. Using the minimum setup and hold characteristics of the receiving device, determine a minimum "safe" data valid window with the following formula:

Min Data Valid Window = Min Setup + Min Hold

Because the placement of the resulting minimum data valid window is tied to the placement of the clock signal, skewing the clock will effectively skew the minimum data valid window. As indicated in Figure 2, as long as the data bus transitions outside of the receiver's minimum data valid window, safe data capture is ensured.

Determining the Clock/Data Relationship

To determine the required clock skew, you create a timing diagram illustrating the clock/data relationship of a read operation for the QDR device based on the actual read timing diagrams acquired directly from the memory device's data sheet. To be more descriptive of the signal relationships in the diagram, name the read data clock signal *C_Mem* and the data bus signal *Q_Mem* to reflect the signals as they appear at the pins of the memory device.



Row		Name	Formula	Min	Max	Margin	Comment
19	V	tPCBdata	0.75	0.75	0.75		PCB Trace delay from Memory device to FPGA for data bus
20	V	tPCBclock	0.75	0.75	0.75		PCB Trace delay from Memory device to FPGA for read clock
21	M	mOFFSET_IN		-0.5	-0.5		Xilinx Offset Constraint Measurement (rising edge)
22	V	mOFFSET_IN_F	(mOFFSET_IN-(period/2))	-4.259	-4.259		Xilinx Offset Constraint Measurement (falling edge)
23	M	mDVW		2.759	2.759		Valid Data Window Measurement
24	M	mPeriod		7.519	7.519		Period Measurement for Read Data Clock

Figure 4 – Clock/data timing diagram of QDR read operation

To model those same signals as they appear at the pins of the FPGA, after their accumulated flight path delay from the QDR device, create the signals *C_FPGA* and *Q_FPGA*. Figure 3 is a block diagram illustrating the signal path relationships just described.

The resulting QDR memory read diagram is displayed in Figure 4. The data bus represents the transition period necessary for all elements of the data bus between valid data words. The PCB trace delay accumulated by both data and clock is represented with separate variables: *tPCBdata* for the delay associated with the data signal and *tPCBclock* for the delay associated with the clock. This allows you to easily vary the values and see the effect on the design results.

Creating Constraints with TimingDesigner

Knowing the timing relationship of an external clock and its associated data as it arrives at the pins of the FPGA provides a unique advantage for accurate data capture and design success. The Xilinx timing constraint *OFFSET* is used along with its associated keywords to provide initial timing information to ISE, so that proper placement of data capture elements and their associated routing delays will be adequate for the design. As illustrated in Figure 5, TimingDesigner provides direct dynamic access to any measurement within a timing diagram for generation of a place and route constraint file (UCF).

For this design example, you need to measure the offset for the data (*Q_FPGA*) and clock (*C_FPGA*) signals, and the duration of the valid data window at the pins of the FPGA controller. This can be done directly from the timing diagram using TimingDesigner measurements, as indicated in Figure 4. You'll also need the read data clock period measurement.

Generating Constraint Information

TimingDesigner's Dynamic Text dialog box offers a way to reference timing diagram measurements from within a vendor's specific timing constraint syntax. Using ISE's *OFFSET* timing constraint syntax, you reference the measured offset values in

the Dynamic Text dialog box to assemble constraints for transfer into the ISE constraints file. The syntax for this design example is illustrated below:

```
OFFSET = IN %mOFFSET_IN.min VALID
%mDVW.min ns BEFORE "rd_clk" TIMEGRP
RdClkRisingFFs;
OFFSET = IN %mOFFSET_IN_F.min VALID
%mDVW.min ns BEFORE "rd_clk" TIMEGRP
RdClkFallingFFs;
```

The *VALID* keyword is for specifying the duration of the incoming data valid window, and is used for hold time calculations in ISE. The signal *rd_clk* is the pin name in the design code associated with the *C_FPGA* clock signal in the timing diagram. Notice that two offset specifications must be declared – one for the rising clock edge and one for the falling clock edge – because this is a dual-data rate read operation. Also notice the "%" syntax that indicates dynamic text access of measured values in the timing diagram.

The Dynamic Text dialog box will resolve the dynamically referenced measurements and allow direct transfer of the information into the UCF constraints file. Once the UCF file has been assembled, execute a place and route in the ISE tool set.

Determining Routing Delays

For this design example, ISE placed the read data capture registers into the I/O blocks of the Virtex-II Pro devices, because they capture data from the input pads and have a common clock and reset signal. This is a default setting for ISE's mapping process and is appropriate for this design example.

As the I/O blocks have only one routing path for input data signals, the TRACE report includes this data path delay in the setup and hold requirements of the I/O block registers. However, the clock signal has several possible paths from the input pad to the capture register, and this design example uses a path through the DCM element for phase shift control. So you must determine the routing and element delays for the clock path to achieve proper phase shifting of the clock.

A TRACE analysis report is generated in verbose mode to get the needed timing information. This report is limited to 16 paths per constraint (the data bus is 16 bits). Briefly look at the report and make note of the worst-case setup path for the data bus in preparation for import into the timing diagram.

To import the desired TRACE analysis report information (saved as a TWX file) into TimingDesigner, map the worst-case setup path identified earlier to the associated waveform in the diagram (*Q_FPGA*), and then import the information to create variables for the routing and element delays. Mapping the FPGA ports guides TimingDesigner to the desired signal information, and allows for automatic updates of existing variables if successive place and route executions are necessary.

Visualization at the Capture Register

After importing the timing results, create another clock signal (*Cin_FPGA*) to represent the clock characteristics at the data capture register. You then add the setup and hold requirements obtained in the TRACE report import as TimingDesigner constraints on the data signal relative to the register clock, as shown in Figure 6.

Notice that the indicated setup and hold margin results from the diagram match the slack values from the initial TRACE timing report, and indicate that the setup paths are failing by about 2 ns, but the hold time paths have more than enough margin. You need to shift the clock so that both the setup and hold time measurements are met with roughly the same margin.

Take Control of Your Design

After initial place and route is complete, you can generate a timing report using the TRACE timing analysis tool within ISE, and import that into the timing diagram. This will allow you to determine the actual routing delays so that you can specify the necessary phase shift for the read clock. You then enter the phase shift attribute into ISE, execute a final place and route, and generate a timing report. When complete, you then re-import the timing report to gauge how well the constraints were met.

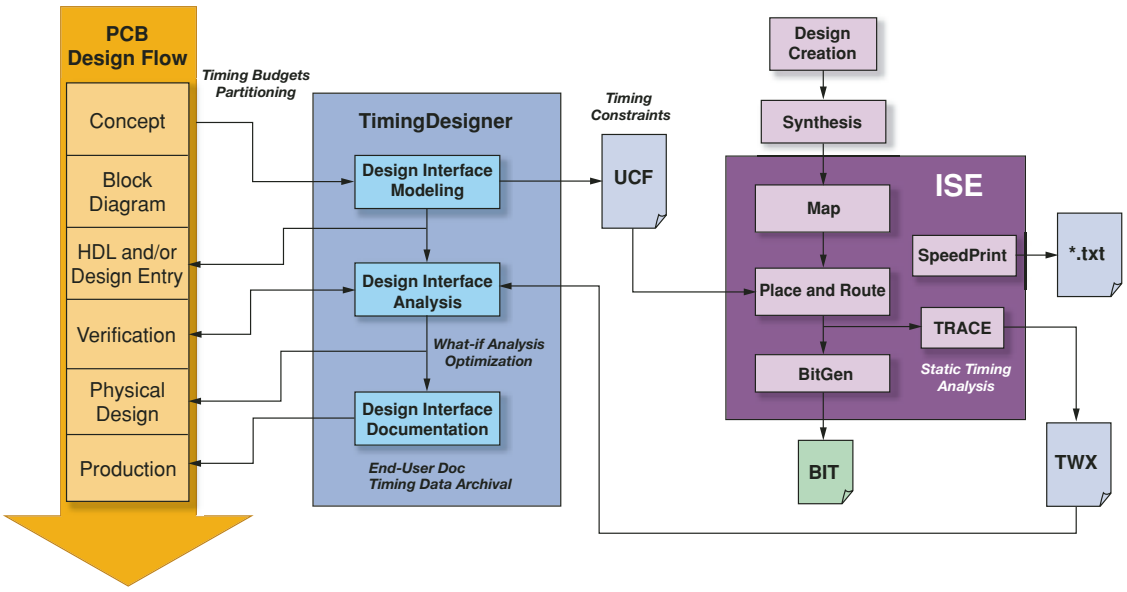


Figure 5 – TimingDesigner offers an intuitive interface to ISE's design flow.

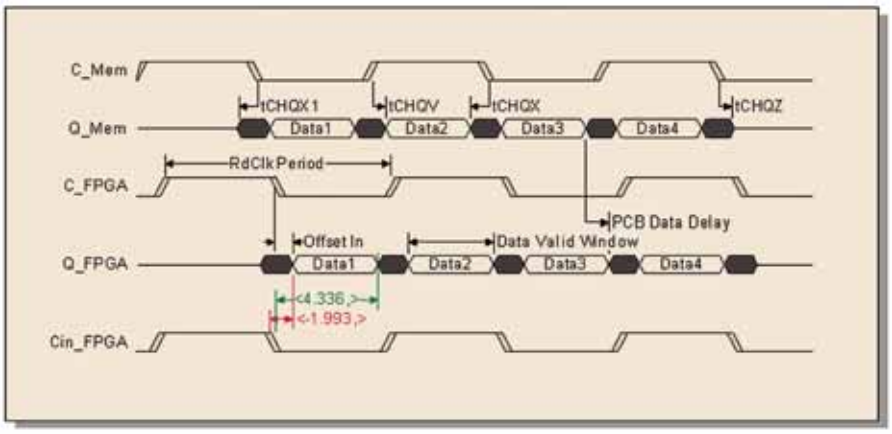


Figure 6 – Setup and hold of data capture register after initial placement

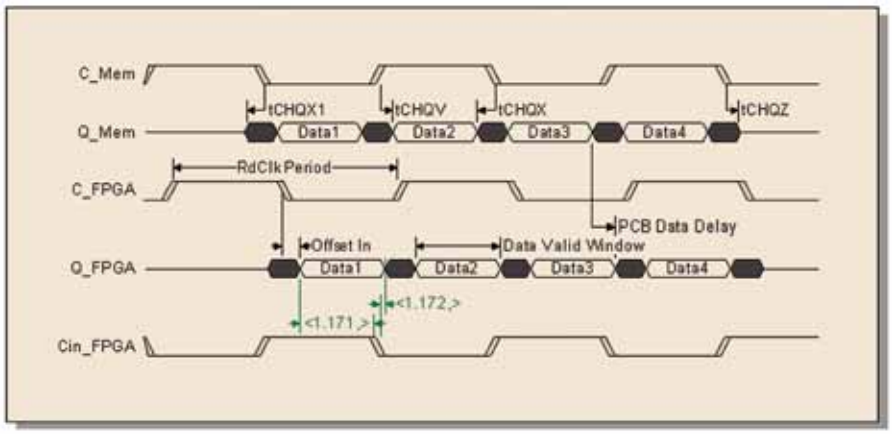


Figure 7 – Balanced setup and hold of data capture register

Making Adjustments

Using the imported information obtained from the TRACE timing analysis report and the measured values from our timing diagram, you can obtain the attribute value for the necessary DCM phase shift (see sidebar, "Balancing the Data Valid Window"). For this example, the attribute value was determined to be 108, for a shift of 3.165 ns. You then enter this value into the ISE tool flow, and execute a second place and route to shift the clock and properly balance the setup/hold margins.

Verifying the Results

Re-importing the new post-place-and-route timing report using the previous import settings will automatically update all of the necessary values, and correctly re-position the register clock signal (*Cin_FPGA*). This is illustrated in Figure 7. Examination of these timing results shows the setup and hold requirements for a read data capture in the design are now balanced with respect to margin (slack), giving the safest possible result for any unforeseen signal drift.

Conclusion

Using TimingDesigner together with Xilinx Virtex-II Pro devices allows optimal safety margins for setup and hold requirements, which are necessary for accurate data exchange with a QDR SRAM memory controller.

By following a design process that combines the use of TimingDesigner timing diagrams and ISE's post-place-and-route timing reports, you can create an accurate analysis of your design's critical timing relationships. Using timing diagrams, you can account for PCB trace delays and other external factors that will affect the signal relationship at the pins of your FPGA device.

ISE's TRACE timing analyzer provides timing reports that give you details on the route delays and constraint requirements of your FGPA device. Together, these tools allow you to accurately capture and exchange critical interface timing information, and allow visual verification that your design will perform as desired.

To learn more about TimingDesigner and the Chronology Division of Forte Design Systems, please visit *www.timingdesigner.com.*

Balancing the Data Valid Window

You can determine the amount of DCM-generated phase shift needed for the clock signal to balance the data window using the following procedure:

1. Subtract the minimum data valid window for the Virtex-II Pro device from the design's actual data valid window, and divide that result by two. This accounts for the difference between the two valid data windows (DlyDVW).

   DlyDVW = (DVWactual – DVW_VIImin) / 2

2. Subtract the offset measurement made at the pins of the FPGA device from the required setup time for the capture registers, to account for device setup requirements (DlyRelSU).

   DlyRelSU = IOBsu – OFFSET

3. Determine total clock path delay from the TRACE timing report (Xtcd).

   Xtcd = <from TRACE report>

4. Add up the necessary delays (values obtained in 1 and 2 above), and subtract the total clock path delay (value from 3 above).

   Clk_offset = (DlyDVW + DlyRelSU) – Xtcd

5. Finally, determine the correct ISE attribute value to control DCM phase shift with the formula:

   phase shift attribute =
   (Clk_offset/Clk_period)*256